TUM School of Computation, Information and Technology
Technische Universität München

Theoretical Foundations for Exploiting Unlabelled Data in Machine Learning

Pascal Mattia Esser

Vollständiger Abdruck der von der TUM School of Computation, Information and Technology der Technischen Universität München zur Erlangung eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitz: Prof. Dr. Stephan Günnemann

Prüfende der Dissertation:

1. Prof. Debarghya Ghoshdastidar, Ph.D.
2. Prof. Dr. Reinhard Heckel

Die Dissertation wurde am 12.01.2024 bei der Technischen Universität München eingereicht und durch die TUM School of Computation, Information and Technology am 24.06.2024 angenommen.

Abstract

This thesis theoretically characterizes how to exploit the information from unlabelled data points either to obtain better representations of the data or to improve a downstream task directly. We provide theoretical guarantees for kernel methods and neural networks in different settings, from semi- and self-supervised learning over unsupervised learning to ordinal data.

Übersicht

In dieser Arbeit wird theoretisch beschrieben, wie die Informationen aus unmarkierten Datenpunkten genutzt werden können, um entweder bessere Darstellungen der Daten zu erhalten oder um eine nachgelagerte Aufgabe direkt zu verbessern. Wir bieten theoretische Garantien für Kernel-Methoden und neuronale Netze in verschiedenen Umgebungen, vom halb- und selbstüberwachten Lernen über unüberwachtes Lernen bis hin zu ordinalen Daten.

Theoretical Foundations for Exploiting Unlabelled Data in Machine Learning

Pascal Mattia Esser

November 19, 2024

Acknowledgements

I first and foremost would like to thank my advisor Debarghya Ghoshdastidar. He is an excellent researcher, professor, and supervisor. He was always very generous with his time, contributing to research projects, giving feedback on all of my papers, and working on (a lot of) teaching together. He does not only care about our work but also truly cares about the mental health of all his students. I feel very lucky to have had him as an advisor.

Thanks to Frank Nielsen for his continued support and for giving me a first insight into theoretical foundations for machine learning during my master's thesis that let me to start this Ph.D.

I want to thank my co-authors Maha, Satyaki, Max, Michaël, and Leena who were of invaluable help during each and every one of my projects. Special thanks to Maha for being the best office mate and always being there for discussing research ideas, life and keeping me motivated throughout the years. And Satyaki for being the unofficial third resident of our office who is not only a great researcher but also brought enlightenment to everyone in the group with his (mostly) philosophical discussions. Thanks to everyone else in the chair, especially Chana, who made my time at the university here much nicer during lunch and coffee breaks.

To Minnah: words are not enough to express my gratitude for your kindness and support — I could not have done it without you. When I climb alone on high and cold mountains you make the summits optional and coming home safely mandatory. Thanks to Quinterino for her frequent checkups and emotional support.

Finally, I am grateful to my parents who mean the world to me and always supported me unconditionally, and my brother who always inspired me by pursuing his own passion.

On a related but slightly less emotional note, I am thankful for the financial support I received, first through TUM and later through the Deutsche Forschungsgemeinschaft project 'SPP 2298 Theoretical Foundations of Deep Learning', that allowed me the freedom to explore a wide range of research ideas.

Extended Abstract

Learning from unlabelled data is a central paradigm in machine learning — from traditional machine learning models for representation learning to Self-Supervised learning methods, that build the backbone for the most advanced large language models. Those recent advances are however mostly engineering-driven and generally lack theoretical foundations. However, such foundations are crucial as machine learning applications are increasingly used in system-critical applications where guarantees are essential. In addition theoretical insights allow for the systematic and energy-efficient development of new models.

Throughout the thesis, the main goal is to theoretically characterize how to use information about the unlabelled data points either to obtain better representations of the data or to improve a downstream task such as clustering directly. To approach this question we consider a combination of data settings with corresponding learning objectives. Borrowing tools from mathematics and statistics we aim to precisely characterize a range of algorithms with a special focus on the influence of unlabelled data.

More specifically we analyze several data settings with gradually less information on the unlabelled data. We start by analyzing trends in Graph neural networks for the transductive node classification settings using learning theoretical measures. Secondly, removing all label information we focus on the self-supervised learning setting using (non)contrastive loss functions where we derive and analyze learning dynamics under orthogonal constraints on the parameters as well as new Kernel methods, which together build the foundation for a comprehensive analysis of such models. From there moving to the traditional unsupervised learning setups, where only features are given, we propose and analyze learning cluster-specific representations using Autoencoders as well as a new reconstruction-based Kernel method. Finally, we derive a clustering algorithm for ordinal data that is provable near-optimal in the number of required comparisons.

Contents

Chapter 1

Introduction

Machine learning is the process of establishing a general set of rules from a given set of data points. This 'process' can vary greatly depending on the given data, their representation, and type of rule we aim to obtain but the most common setup that comes to mind is known as *supervised learning*: learn a mapping from object descriptors, or features, to the labels of the objects as shown in Figure 1.1 (left). How well the mapping has been learned can then be tested by evaluating predicted labels for unseen data points. While the data setup and the overall goal is straightforward a wide range of algorithm has been developed often with the aim to optimize the prediction with regards to data-specific properties. This setting is highly relevant in real-world applications such as image- and object-recognition, customer sentiment analysis or spam detection [MMF21] and might be the most common and studied data setting in machine learning due to the wide range of applications but also due to its very precise characterization of the problem in terms of the data setting. However while it is a central focus in real-world applications it does not cover the whole range of learning problems.

On a high level, in learning one can ask the previous question more generally and aim to partition the space of objects into sets of objects that *'belong together'*. However, without going into further detail it becomes immediately clear, that 'belong together' or 'similar' are ill-defined terms without further context. In a supervised framework the objective is straightforward and intuitive: assuming $\mathcal{X}$ being the *domain or feature space* and $\mathcal{Y}$ the *label set*, the goal is to find a predictor $f : \mathcal{X} \to \mathcal{Y}$ based on n training samples $S \subset \mathcal{X} \times \mathcal{Y}$ and a loss function $\ell : \mathcal{Y} \times \mathcal{Y} \to [0, \infty)$. Therefore in this setting the notion of objects belonging together is very clearly defined through the access to the labels but what happens if we no longer have access to labels? For example, assume given just a set of images of a product from an assembly line, the goal is to find manufacturing mistakes, however, there are no labelled examples for such mistakes. How can we solve this problem without labels? What are we *'mapping to'*? Taking steps towards this direction we note that *just because we do not have access to labels, does not mean that there is no underlying structure corresponding to it.* In this case, we now reach the overall setup, that we will focus on in

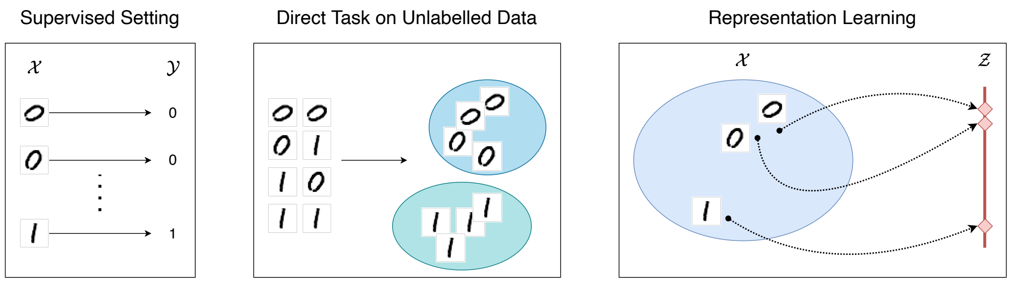


Figure 1.1: Let us consider sampled from the famous MNIST dataset [LC10b], containing images of handwritten digests from zero to nine, each having 28x28 pixels, together with the number as a label. From there we can define the following Learning objectives: **(left)** In the supervised setting we learn a mapping from the feature space $\mathcal{X}$ to the label space $\mathcal{Y}$. **(middle)** Solving the task of clustering directly on the unlabelled data: Assume a set of objects given, the goal is to group them into 'similar' clusters. **(right)** We learn a mapping from $\mathcal{X}$ to a lower dimensional representation $\mathcal{Z}$. In this setting, we do not optimize the mapping directly with regard to a downstream task.

this thesis — the setting of learning from unlabelled data. In general, this setting comes into play either if labels can not be obtained or in the large data setting where it might be possible to obtain labels, however, it is practically not feasible or too expensive.

In this thesis, we focus on this setting on a foundational level through a theoretical viewpoint and highlight how crucial a thorough understanding of the unlabelled data setting is. On a higher level we can therefore first ask:

Question 1. *What problems can we solve from unlabelled data?*

To better outline the possible tasks, which we will formalize later, let us consider the supervised setting as a starting point. From there moving towards the unlabelled setting we start by removing only some of the labels with the goal to recover them. While in those settings partial labels are present, the unlabelled objects are also considered during training and therefore differ from the supervised setting. This is often formulated in a transductive learning setting on graphs, where given a graph with partially labelled nodes the goal is to label the unlabelled nodes.

From there further moving away from the supervised learning we can consider settings without any labels[1]. While we no longer have access to the labels we might still end up with somewhat similar high-level ideas such as grouping similar objects together. Of course, since we do not have any labels we can only hope to recover the classes up to permutations of the

[1] To show how crucial this setting is in real-world applications let us outline two examples. Firstly *anomaly detection* is the task of finding extreme outliers in a dataset and is a central application in areas such as fraud detection and quality control in production [CBK09]. Since it is often not known beforehand how an out-layer can look like and therefore there do not exist any labelled examples of it, unsupervised methods allow to detect if a new example significantly differs from the reference product. Secondly we can consider *clustering* of genotypes [Ran+01] or clustering of different tissue types in three dimensional image data from Positron Emission Tomography (PET) scans [FRD10]. While in this thesis we do not focus on such specific applications, but more on theoretical foundations, it outlines how crucial the unlabelled data setting is in real-world applications.

labels[2] as illustrated in Figure 1.1 (middle). Here the most common objective is clustering where the main goal is to group similar objects into the same class in an unsupervised setting. While this problem has been extensively studied in traditional[3] machine learning and include famous algorithms such as k-means clustering [AV07], their time complexity significantly increases with high dimensional data, and therefore several methods have been developed that use deep unsupervised models to learn cluster assignments using deep neural networks [XGF16; Diz+17; Wan+16; XX15; Wan+15].

Until now we assumed that all the above tasks — supervised, unsupervised, or transductive — were performed directly on the given objects (such as clustering or labeling). In the case of MNIST, this would mean using 28x28 pixels per object (See Figure 1.1). Staying in the image setting this however, brings up the question if a pixel representation is the most optimal way to represent the objects. Could it be more beneficial to only consider a subset of the pixels or simply the mean of all pixels to reduce the noise in the data? Abstracting from the image setting this becomes the question:

Question 2. *What is a good representation of a given set of data points?*

Representation learning builds on the idea that for most data, there exists a lower dimensional embedding that still retains most of the information useful for a downstream task [BCV13]. This is intuitive by observing that one can delete a few pixels in a set of images without changing how one would group the images. While early works relied on pre-defined representations, including image descriptors such as SURF [BTVG06] or SIFT [Low99] as well as bag-of-words approaches in natural language processing, over the past decade the focus has moved to representations learned from data itself [BCV13] and proven to be more powerful than the use of hand-crafted descriptors. Therefore assuming we use a function $f_\Theta(\cdot)$ for learning the representation we perform the clustering (or any other downstream task) on $f_\Theta(X)$ instead of X [RL03; Tia+14; Wan+16]. While some properties of improved representations, such as reducing the noise in the features are intuitive, more generally it is important to note that there is no unique measure of *'goodness'* of a representation without taking a downstream task into consideration [BCV13].

To summarize we will consider two main directions of tasks on the unlabelled data: *either tasks directly on the unlabeled data as shown in Figure 1.1 (middle), such as clustering or learning improved representations of the data as shown in Figure 1.1 (right).* We will refer to the former as 'downstream tasks' and as 'representation learning' to the latter.

[2]Assume the MNIST setting: If we tell an algorithm to group the digits, it might be able to do so but does not have any inherent idea of what e.g. a *'one'* is. Therefore it can only assign labels up to permutation between the obtained clusters.

[3]Throughout the thesis we will often refer to *'traditional'* machine learning algorithms or traditional tools for theoretical analysis. While there is not a clear differentiation we will use this term to generally refer to settings that were developed before or independent of deep learning approaches.

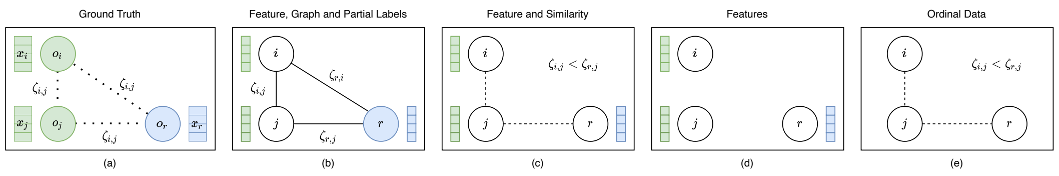


Figure 1.2: Illustration of different data settings. **(a)** There exists a set of n objects, with features $\{x_i\}_{i=1}^n$, relations between objects $\{\zeta_{i,j}\}_{i,j=1}^n$ and labels $\{y_i\}_{i=1}^n$. In the following setting, only some of the additional information are given. **(b)** Features are given as well as relations $\{\zeta_{i,j}\}$ in the form of a graph structure where each node is given by one of the objects. In addition from some objects, labels are given. **(c)** Features are given as well as semantic similarities between at least some of the objects. Framed in the Self-Supervised Learning setting this becomes a triplet setting such that objects o_j, o_i and o_r are represented by x_i, x_j^+ and x_r^-. **(d)** Only features $\{x_i\}$ are given. **(e)** Only ordinal comparisons between objects are given. Similarities between i, j given by $\zeta_{i,j}$ and between j, r given by $\zeta_{r,j}$ exist, however only $\zeta_{i,j} \lesseqgtr \zeta_{r,j}$ is observed.

Now in the previous discussion, we omitted one crucial aspect: we assumed that the unlabelled data is given in the form of only features or descriptors of the objects (e.g. pixels for an image). While this is a common setting, one might have access to different amounts of information on the unlabelled data points. Throughout the thesis, we explore a wider range of data settings by asking:

Question 3. *What information about the unlabelled data can we exploit?*

To better understand what information can be available let us start by outlining the unlabelled data setting more formally. As illustrated in Figure 1.2 (a), we assume there exist n objects $\{o_i\}_{i=1}^n$ with corresponding labels $\{y_i\}_{i=1}^n$ and d dimensional features $\{x_i\}_{i=1}^n$. In addition, we assume the existence of some type of object relation between o_i and o_j which we denote by $\zeta_{i,j}$. Throughout this work, we will consider several forms of relations such as semantic similarities, graph structures, or ordinal relations. From there, we consider several options of which of the above information on the objects is given and more specifically we investigate the following four settings.

Transductive Setting. The first setting we consider is transductive learning on graphs — a setting where we have a lot of information on the unlabelled data. As illustrated in Figure 1.2 (b) we observe a small number of the labels with the goal to predict the unlabelled ones. In addition, the relation between objects is given by a graph structure which allows us to model a set of objects and their relationships. This setting can be found in a wide range of real-world applications such as modeling social networks [Wu+18; Fan+19], particle systems [Hos17; Kip+18], knowledgeable graphs [Ham+17] or traffic networks [Cui+19]. [Zho+20] provides an extensive list of applications.

Self-Supervised Learning (SSL). Let us now assume that we have no longer access to a full underlying graph structure[4] but we observe features of the unlabelled objects as well as have access to semantic similarities between objects as shown in Figure 1.2 (c). In this context the goal is to learn better representations of the objects which improve downstream tasks and the knowledge of semantic similarities is used to determine which objects should be mapped close to each other in a learned representation. This idea is not new [Bro+93] but recently has shown great empirical success in computer vision [Che+20b; Car+21; JT19], video data [Fer+17; Ser+18], natural language tasks [MM20; Dev+19] and speech [Ste+19; Moh+22].

Unsupervised learning. While the SSL setting has become central to several state of the art models [Rad+18; Dev+19] a notion of similarity is not always available and we have to rely on approaches that only work on the features as illustrated in Figure 1.2 (d). This setting — often referred to as unsupervised learning — is popular as it includes applications such as anomaly detection [Iiv22; Sch+17; BHK23], medical imaging [RS18; Aga+18; CF20] or recommender systems [YMA21; BHK23].

Ordinal Data. In our final setup we do not have access to either explicit similarities or features labels but only ordinal comparisons of the form o_i *is more similar to* o_j *than to* o_r as shown in Figure 1.2 (e). This setup mainly stems from the psychometric and crowdsourcing literature [She62; You87; SBC05] where the importance and robustness of collecting ordinal information from human subjects have been widely discussed. In recent years, this framework has attracted an increasing amount of attention in the machine learning community and several learning paradigms have emerged. In this work, we focus on the idea of learning a similarity function and from there solve clustering objectives [KL17].

Having outlined the main tasks we are interested in solving and under what data setting we can now ask:

> **Question 4.** *What machine learning models can we use to solve learning problems in the above data settings?*

While in practice there is a large number of possible models, in this thesis we will generally rely on and analyze the following approaches.

Neural Network Approaches. Driven by rapid empirical development and practical successes, neural network-based models have become the default choice in a wide range of tasks however their development is mainly driven by engineering advances. In general a neural network can be considered as a function $\mathbb{R}^d \to \mathbb{R}^h$, comprised of learnable parameters, that perform linear transformations and fixed point-wise non-linear transformations. The

[4]For clarification we note that we can always construct a graph from given features by using some similarity metric and threshold. The distinction we consider here is that in the transduction setting a graph is given explicitly as part of the data.

parameters are then updated using backpropagation[5] of the loss. For specific settings, we will adapt this formulation to the given data setting in Chapter 2 - 6 and analyze several neural network-based approaches throughout the thesis and work towards a more complete theoretical understanding beyond the supervised setting.

Kernel Methods. In spite of the widespread use of deep learning, other models are still ubiquitous in data science. Especially Kernel methods [SSM97] are a well-established approach that relies on the pairwise similarities between datapoints, denoted by a Kernel $k(x, x')$. When the map k is positive definite, $k(x, x')$ corresponds to the inner product between (potentially infinite-dimensional) nonlinear transformations of the data and implicitly maps the data to a reproducing Kernel Hilbert space (RKHS) $\mathcal{H}$ through a feature map $\phi : \mathcal{X} \to \mathcal{H}$ that satisfies $k(x, x') = \langle \phi(x), \phi(x') \rangle$. Kernel methods are among the most successful models in machine learning, particularly due to their inherently non-linear and non-parametric nature, which nonetheless allows for a sound theoretical analysis. We take advantage of those properties mainly in Chapter 4 & 5.

Traditional Clustering Methods. Traditional clustering methods like k-means [Mac67], density-based spatial clustering of applications with noise (DBSCAN) [HPD19] or spectral clustering [NJW01] have been highly successful in practice and have been studied theoretically. In this work, we do not focus directly on the analysis of such algorithms however we consider them as a central downstream task on learned embedding in Chapter 4 - 6 to analyze and develop new models.

In recent years, due to its practical success, machine learning has been evolving at a fast pace and is mainly based on empirical improvements and heuristics but its theoretical understanding is lacking significantly behind. With this empirical success the question arises:

> **Question 5.** *If machine learning models work so well in practice, why do we need to theoretically study them?*

We argue that a theoretical study is especially important *because* of the empirical success of the machine learning models.

Development of new algorithms. Modern machine learning models are becoming increasingly more complex, deep learning models have billions of parameters and a complex combination of building blocks and while by empirical exploration advances can be achieved it is very costly in time and energy. In addition, any such analysis is often dependent on initialization and hyperparameter tuning and therefore a complete picture is hard to obtain. By theoretical analysis, we can aim to *explain* the influence of different components of the machine learning model and by using those insights improve the existing approaches. Examples

[5]While alternative approaches to optimize the network exist such as evolutionary algorithms (e.g. [ASP94; Din+13]), gradient based methods have become the standard approach in practice and also have shown to be the most promising direction with regards to theoretical characterizations.

for such results include neural network architecture search [CGW21] or determining if more data should be added to the algorithm [LZ14; Nak+21]. In this thesis we extend this general line of work by developing new Autoencoder architectures (Chapter 6 & 5), Kernel methods (Chapter 4 & 5) and clustering approaches (Chapter 7) which we all discuss in more detail later in the introduction.

Guarantees for existing algorithms. With machine learning models being applied in system-critical applications such as thread assessment in autonomous driving [Chi+22; Li+21a], medical image processing [RS18; Aga+18; CF20], or data with privacy concerns [Liu+21; RG23; JLE14] it becomes increasingly important to provide guarantees and exact characterizations of the potential and limitations of the considered machine learning models. Such an analysis can be performed in a limited capacity by empirical means however such results always depend on the specific training setting, initialization, and dataset and can not account for outliers that are not presented in the dataset. However such worst-case guarantees can be provided by theoretical means. In this thesis we provide such worst-case guarantees in Chapter 2, exact characterizations of obtained embeddings in Chapter 3, 4 & 6 and finally exactly characterize how many samples are necessary for cluster recovery for a specific clustering method in Chapter 7.

While the above answers *why* theoretical foundations are essential, answering *how* to obtain such results is more complex and nuanced. Borrowing from a broad range of tools from mathematics, statistics, and physics, in recent years a significant effort has been made to bridge this gap between empirical advances and theory. However, it is important to note that the main focus in this direction has been in the supervised setting and only very few results exist in the unlabelled data setting. We aim to work towards closing this gap by asking:

Question 6. *What theoretical analysis is feasible and provides insights in the unlabelled data setting?*

While in some traditional machine learning setting approaches for a theoretical analysis are established overall there is no unified theoretical framework to characterize and analyze machine learning models. Therefore carefully choosing theoretical tools is especially important and one has to balance the expressiveness and complexity of a given measure. In this thesis, we will mainly rely on the following two directions of analysis together with technical tools such as concentration bounds for providing theoretical insights.

Traditional machine learning theory. We can illustrate approaches for traditional machine learning models well on how the question on *generalization* of an algorithm is approached through worst-case guarantees. A fundamental problem in any learning task is to quantify how well an algorithm, trained on a given set of data points performs on unseen data.

Assume there are m labelled objects, the goal is to find a predictor $f : \mathcal{X} \to \mathcal{Y}$, that minimizes the *generalization error* (error on the unlabelled data) $\mathcal{L}_u(f)$ with a loss function $\ell : \mathcal{Y} \times \mathcal{Y} \to [0, \infty)$. In addition we define the *empirical error* (error on labelled data points) of f to be $\widehat{\mathcal{L}}_m(f)$ and we aim to obtain a bound of the form

$$\mathcal{L}_u(f) \leq \widehat{\mathcal{L}}_m(f) + \text{complexity term} \tag{1.1}$$

where the complexity term is typically characterised using learning-theoretic terms such as Vapnik–Chervonenkis (VC) Dimension [Vap82; Vap98] or Rademacher complexity [TLP16; EYP09]. While this setup has been mainly developed in the supervised setting we show that it can also be applied to the unlabelled setup. This idea provides a general framework to analyze a given algorithm and has been shown to be to be useful in contexts such as Kernel methods [SSM97]. In addition to such general approaches for analysis, we often consider specific setups such as exact recovery of clusters under planted data setting [Abb17].

Learning dynamics and loss landscape of deep learning models. Due to the complexity of neural networks worst case measures like the one outlined above have shown to be of limited use [Zha+17; Ney+17] and therefore new tools have emerged. Central concepts here include the characterization of learning dynamics where we model how the network outputs change during training with gradient descent and to what outputs they converge to. While in traditional machine learning we are often interested in the worst-case analysis, this line of work shifts the focus more towards an analysis of the expected or a highly probable outcome. Those quantities provide a more tractable expression of the problem and in the supervised setting have been shown to be an essential tool to understand the loss behavior and convergence [Fuk98; SMG14; PKK18], early stopping [Li+21b], linearised (Kernel) approximations [JGH18a; Du+19b] and, mostly importantly, generalization and inductive biases [Sou+18; Luo+19; HY21].

Overall in this thesis, we show that general ideas, developed in the supervised setup, such as the analysis of learning dynamics, can be also employed in the unlabelled data setup.

1.1 Goal of this Thesis

From the above outline it is clear that while a thorough theoretical analysis of machine learning algorithms on unlabelled data is essential, there is a vast number of settings and no unifying analysis. Each of the questions posed so far is too general to answer in a vacuum without considering the interplay with the other questions. Therefore to work towards a more complete picture of learning from unlabelled data by perusing the following approach: starting from a given data setting as outline for Q.3 we consider either a particular downstream task directly (Q.1) or learn a better representation (Q.2). Depending on the existing state of the art and the possibility of theoretical analysis we pick an appropriate machine learning model (Q.4) to achieve this goal. This combination of Q.1 to Q.4 then presents the setup to answer Q.6 and perform a thorough theoretical analysis.

Under this setup we are able to theoretically analyze several important machine learning algorithms which gives insight into the behavior and limitations with a focus on the effect of unlabelled data as well as develop several new machine learning models with sound theoretical foundations.

With the goal outlined we now discuss the main data settings we consider in the thesis in more detail together with the main questions we aim to answer and our contributions. We keep the setup high level and formalize it more in the later chapters of the thesis.

1.2 Transductive Learning

Let us start with outlining our first data setting of transductive learning. We assume access to a graph with n vertices, corresponding to the respective feature vectors $x_1, \dots, x_n$, and edge $\zeta_{i,j}$ denoting similarity of vertices i and j. The graph structure is represented by an adjacency matrix $A \in \mathbb{R}^{n \times n}$. The data structure is illustrated in Figure 1.2 (b). Without loss of generality, one may assume that the labels $y_1, \dots, y_m \in \{\pm 1\}$ are known, and the goal is to predict $y_{m+1}, \dots y_n$.

Graph Neural Network (GNN). While transductive graph learning problems have been studied in traditional machine learning for example through message passing algorithms [Bis06] in recent years graph-based neural network-based approaches have become the state of the art [GMS05; Sca+09; KW17]. In practice, the exact form of aggregation and combination steps vary across architectures to solve domain-specific tasks [KW17; Bru+14; DBV16; Vel+18; Xu+19]. In a standard feed-forward neural network each layer is defined by a learned linear transformation through the weights and a non-linear transformation. This is extended in the GNN setting by adding a graph convolution[6] for example through the adjacency matrix A, such that each layer is now given by

$$H_j := \psi\left(AH_{j-1}W_j\right)$$

where H_{j-1} is the output of the previous layer, W_j the trainable weight matrix and ψ denote the point-wise activation function. The weights are optimized by minimizing the loss between the predictions on the labelled nodes, $\hat{y}_i$, and labelled nodes.

While the understanding of GNNs is limited, there are empirical approaches to study GNNs in the transductive [Boj+18] and supervised setting [Zha+18; Yin+18]. For an extensive survey on the state of the art of GNNs see for example [Wu+20]. While empirical studies provide some insights into the behavior of machine learning models, rigorous theoretical

[6]We will later extend this to more commonly used diffusion operators [KW17; Wu+20] in Chapter 2.

analysis is the key to deep insights into a model. The focus of this part is to provide a learning-theoretic analysis of the generalization of GNNs in the transductive setting.

Generalization Error Bounds. As stated previously in the introduction, the study of generalization is a central question in machine learning tasks. The problem in the transductive setting can be formulated in the statistical learning framework which allows us to derive generalization error bounds GNNs. It differs slightly from the standard setting as in the transductive setting the features of the test points are also present during the training. Nevertheless, again a bound of the form as stated in (1.1) can be studied and we ask:

> **Question 7.** *Can we use traditional learning theoretical measures to gain insights into the generalization properties of graph neural networks in the transductive setting?*

Contributions in this thesis (based on [EVG21]). We show that, under careful construction of the complexity measure and distributional assumptions on the graph data, learning theory can provide insights into the behavior of GNNs, specifically focused on explicitly characterizing the influence of the graph information and the network architecture. The main contributions are the following: We introduce a formal setup for graph-based transductive inference, and we use this framework to show that VC Dimension [Vap82; Vap98] based generalization error bounds are typically loose, except for a few trivial cases. This observation is along the lines of existing evidence for standard supervised feed forward neural networks [Zha+17; Ney+17]. From there we extend the analysis by deriving generalization bounds based on the transductive Rademacher complexity [TLP16; EYP09]. Our results show that these bounds are more informative, suggesting that the correct choice of complexity measure is important. While some previous works on the generalization of GNNs exist [VZ19; STH18; GJJ20; LUZ21; OS20b; OS20a] (See Chapter 2 for more detailed discussion) our analysis provides more expressive and tighter bounds. We further refine the generalization error bounds under a planted model for the graph and features. Such an analysis, under random graphs, has been rarely studied in the GNN literature before [EVG21]. Our results suggest that, under distributional assumptions, learning-theoretic bounds can explain the behavior of GNNs — while not completely — significantly better. We consider GNNs with residual connections and demonstrate how the above analysis can be extended to other network architectures. We prove that residual connections have a smaller generalization gap in comparison with vanilla GNN. Finally we numerically illustrate that the trends in the derived bounds coinside with trends observed empirically and discuss limitations of the proposed approach.

The findings renewed[7] the interest in studying generalisation in neural networks in terms

[7]Follow up works to [EVG21], which builds the foundation of this chapter, considered planted models to provide further refined bounds. Examples include [Ju+23; Shi+23] and we discuss this aspect in more detail

of learning-theoretic measures however also outline that limitations of this approach prevail even under careful analysis under distributional assumptions highlighting that for a comprehensive analysis of deep learning models alternative approaches have to be considered as well.

1.3 Self-Supervised Learning (SSL)

Let us now assume that we no longer have access to partial labels and explicit underlying similarity structures such as a graph but still a notion of what makes objects similar to each other as is illustrated in Figure 1.2 (c). In this setting the main goal is generally to learn better representations of the objects, that improve downstream predictions.

(Non)Contrastive SSL Models. While there is a range of models [Gui+23] that can take advantage of this setting, (non)contrastive models have become the foundation of most state-of-the-art models [Dev+19; Rad+18]. Such models build on the idea that we only have access to features and implicit knowledge of what makes samples semantically close to others. Starting from X using this implicit knowledge of similarities is used to create inter-sample relations $\left(X, \overline{X}\right)$. Here $\overline{X}$ is often constructed through data augmentations of X known to preserve input semantics such as additive noise, rotations or horizontal flip for an image [KJC16]. This setting of learning from $\left(X, \overline{X}\right)$ is referred to as Self-Supervised representation learning and has been established in recent years as an important method between supervised and unsupervised learning. In this thesis we work towards strengthen it's theoretical foundations. Before going into the exact questions we analyze, let us formalize the (non)contrastive learning setup.

In the contrastive learning setting, we define the training samples through triplets[8] of the form $\{(x_i, x_i^+, x_i^-)\}_{i=1}^n$ where the idea is to consider an anchor object x_i, a positive sample x_i^+ generated using data augmentation techniques, as well as an independent negative sample x_i^-. We illustrate this setting in Figure 1.2 (c). The goal is to align the anchor more with the positive sample than with the independent negative sample. This idea can be implemented by learning a representation of the form $f_\Theta(x) : \mathbb{R}^d \to \mathbb{R}^h$ through optimizing specific loss functions. For a better intuition consider the simple contrastive loss [Aro+19c]

$$\min_\Theta \sum_{i=1}^{n} \underbrace{f_\Theta(x_i)^\top f_\Theta(x_i^-)}_{\text{Term I}} - \underbrace{f_\Theta(x_i)^\top f_\Theta(x_i^+)}_{\text{Term II}}. \tag{1.2}$$

where we aim to minimize *Term I*, which give the alignment between the embeddings of x and x^- and maximizes *Term II*, which give the alignment between the embeddings of x and x^+. Similarly, a non-contrastive model can be formalized by only considering *Term II* [Che+20b].

in Chapter 8.

[8]For now we consider the setting of triplets however the idea can simply be extended to e.g. multiple negative samples.

For this general setup, we investigate two parallel questions: first, we analyze the learning dynamics of neural network-based (non)contrastive SSL models, and secondly propose and analyze Kernel versions of the considered loss model.

1.3.1 Analysis of Learning Dynamics

The main focus of the theory literature on SSL has been either on providing generalization error bounds for downstream tasks on embeddings obtained by SSL [Aro+19c; Ge+23; BKB21; Lee+21; SMA21; TKH21; WXM21; BNN22; Che+22], or analysing the spectral and isoperimetric properties of data augmentation [BL22; HYZ23; Zhu+23]. While generalization theory as studied in the previous chapter remains one of the fundamental tools to characterize the statistical performance of a model, they do not provide a complete theoretical understanding. Therefore a key focus in modern deep learning theory is to understand the learning dynamics of models. The main idea is here to study how the network output or parameters evolve during training under gradient descent. This provides a more tractable expression of the problem, than traditional learning theoretical bounds.

Question 8. *Can we perform an analysis of learning dynamics in the self-supervised setting to analyze the obtained embeddings?*

Contributions in this thesis (based on [EFG23]). While we show that it is possible to analyze the learning dynamics of SSL models under contrastive and non-contrastive losses [Aro+19c; Che+20b] we also show them to be significantly different from the dynamics of supervised models. This gives a simple and precise characterization of the dynamics that can provide the foundation for future theoretical analysis of SSL models.

Constraint Neural Network Embedding Function. While the loss functions (1.2) hold for general encoding functions f_Θ we define the mapping of the data $x \in \mathbb{R}^d$ to an embedding $z \in \mathbb{R}^h$ by a one hidden layer neural network for our analysis. Consider the setting presented in Figure 1.3 where we analyze the dynamics with $f_\Theta(x)$ being a one hidden layer neural network optimized over (1.2). We can observe that without orthogonal constraints on the weights, Figure 1.3 (c), the learned representations collapse[9], whereas under orthogonal constraints in Figure 1.3 (d) we observe the learning of a meaningful representation (in the sense of no collapse). This indicates that adequate constraints or regularization is especially important in the (non)contrastive loss setup. We can therefore summarize the analyzed setup by the following embedding function:

$$\begin{aligned} f_\Theta &:= W_2^\top \psi(W_1 x) \\ \text{s.t. } W_2^\top W_2 &= \mathbb{I}_h,\ W_1^\top W_1 = \mathbb{I}_d \end{aligned} \tag{1.3}$$

[9]We consider a collapse here to be that the function output for the first dimension is equivalent to the second dimension. Therefore the second dimension does not provide any additional information.

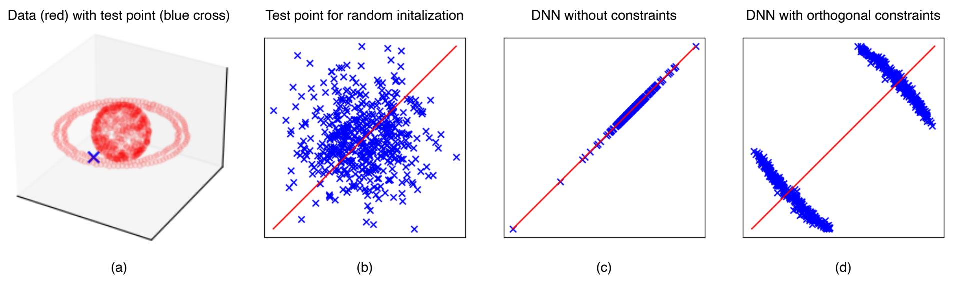


Figure 1.3: Dimension collapse without orthogonal constraints in contrastive loss. Consider the dataset as shown in **(a)** in $\mathbb{R}^3$ that is embedded into $\mathbb{R}^2$ using $f_\Theta := W_2^\top \psi(W_1 x)$ with $\psi(\cdot) = \tanh(\cdot)$. Under several random initializations of the weights, the embedding in **(b)** is observed. When optimized using (1.2) without constraints we observe a dimension collapse as illustrated in **(c)**, however when considering the orthogonal constraints ($W_2^\top W_2 = \mathbb{I}_h,\ W_1^\top W_1 = \mathbb{I}_d$) no collapse is observed in **(d)**.

where $W_2^\top W_2 = \mathbb{I}_h$ and $W_1^\top W_1 = \mathbb{I}_d$ are orthogonal constraints, that must be ensured during training to avoid dimension collapse.

Analysis. Under this simple neural network model we express the learning dynamics for both contrastive and non-contrastive learning without constraints and show that the evolution dynamics are the same across dimensions. This explains why SSL is naturally prone to dimension collapse. Assuming a 2-layer linear network ((1.3) with $\psi(x) := x$), we show that dimension collapse cannot be avoided by adding standard Frobenius norm regularisation or constraint, but by adding orthogonality or L2 norm constraints. We further show that at initialization, the dynamics of a 2-layer network with nonlinear activation are close to their linear, width-independent counterparts. We also provide empirical evidence that the evolution of the infinite width non-linear networks is close to their linear counterparts, under certain conditions on the nonlinearity. We derive the learning dynamics of SSL for linear networks, under orthogonality constraints. We further show the convergence of the learning dynamics for the one-dimensional embeddings. We numerically show, that our derived SSL learning dynamics can be solved significantly faster than training nonlinear networks, and yet provide comparable accuracy on downstream tasks.

1.3.2 Kernel Self Supervised Learning

In spite of the widespread use of deep learning — such as introduced in the last section — other models are still ubiquitous in data science however contrastive learning models, not based on neural network approaches are not common. Kernel methods are among the most successful models in machine learning, particularly due to their inherently non-linear and non-parametric nature, which nonetheless allows for a sound theoretical analysis. Kernels have been used extensively in regression [KW71; Wah90] and classification [CV95; Mik+99] but are surprisingly unexplored in the self-supervised setting. Therefore we ask:

> **Question 9.** *Can we construct non-parametric representation learning models, based on contrastive losses?*

Contributions in this thesis (based on [EFG23]) We argue that the classical representer theorems for supervised Kernel machines are not always applicable for (self-supervised) representation learning due to the difference in regularization, and present a new representer theorem, which show that the representations learned by our Kernel models can be expressed in terms of Kernel matrices.

Kernel Embedding function. We Kernelize a single hidden layer network, mapping data $x \in \mathbb{R}^d$ to an embedding $z \in \mathbb{R}^h$:

$$x \in \mathbb{R}^d \xrightarrow{\phi(\cdot)} \mathcal{H} \xrightarrow{W} z \in \mathbb{R}^h.$$

Therefore we learn a representation of the form $f_\Theta(x) = W^\top \phi(x)$ by optimizing the objective functions defined in [Aro+19c; Che+20b]. Our work takes a significant step by decoupling the representation learning paradigm from deep learning. To this end, Kernel methods are an ideal alternative since to neural networks (i) Kernel methods are suitable for small data problems that are prevalent in many scientific fields [Xu+23; TBH23; CT21]; (ii) Kernels are non-parametric, and yet considered to be quite interpretable [PM17; HH14]; and (iii) as we show, there is a natural translation from deep SSL to Kernel SSL, without compromising performance.

Analysis. The main focus is the development and analysis of Kernel methods for contrastive SSL models. More specifically: We present *Kernel variants of a single hidden layer network that minimizes two popular contrastive losses.* For a simple contrastive loss [Aro+19c], the optimization is closely related to a Kernel eigenvalue problem, while we show that the minimization of *spectral contrastive loss* [Che+20b] in the Kernel setting can be rephrased as a Kernel matrix based optimization. Moreover, the presented approaches do not reduce to traditional (unsupervised) Kernel methods.

Furthermore, we derive generalization error bounds for the proposed Kernel models which show that the prediction of the model improves with an increased number of unlabelled data. While in the neural network setting such bounds often become trivial [Zha+17; Ney+17] in the Kernel setting they are well established and considered a reliable tool. We empirically demonstrate that the proposed Kernel methods perform on par or outperform classification on the original features as well as Kernel PCA and compare them to neural network representation learning models.

Since representation learning, or finding suitable features, is a key challenge in many scientific fields, we believe there is considerable scope for developing such models in these fields building on this work.

1.4 Unsupervised Learning

While the previous SSL setup has been shown to be very successful in state-of-the-art applications, the semantic information that is needed to create meaningful augmentations is not always available. For example, a meaningful similarity is simple to construct in an image setting [BL22; HYZ23; Zhu+23] but requires significantly more insight in a categorical data setting. Therefore it is important to study approaches that only rely on given features. In this setting, unsupervised representation learning through reconstruction has been established as a central concept. Relying only on a set of features X, the high-level idea is to map the data to a lower dimensional latent space, and then back to the features. The model is then optimized by minimizing the difference between the input data and the reconstruction. While a few famous approaches exist in traditional machine learning such as (Kernel) Principal component analysis (PCA) [Pea01; SSM98], the paradigm of representation through reconstruction has built the foundation of a large number of deep learning methods.

Autoencoder (AE). Autoencoders [Kra91] use a neural network $f_\Theta(\cdot)$ for the encoding into the latent space as well as a neural network $g_\Psi(\cdot)$ for the reconstruction which results in the following mapping

$$x \in \mathbb{R}^d \xrightarrow{f_\Theta(\cdot)} z \in \mathbb{R}^h \xrightarrow{g_\Psi(\cdot)} \hat{x} \in \mathbb{R}^d$$

which is then optimized over Θ, Ψ through minimizing the reconstruction error $\|x - \hat{x}\|_2^2$.

The empirical success of autoencoders has given rise to a large body of work for a wide range of applications such as image denoising [BCM05], clustering [Yan+17] or natural language processing [Zha+22a]. However, their theoretical understanding is still limited to analyzing critical points and dynamics in shallow linear networks [Kun+19; RG22].

1.4.1 Kernel Autoencoder

AEs provide a practically well-working algorithm a theoretical analysis beyond the linear setting is complex [Kun+19]. In the previous section we outlined how Kernelized versions neural network model can provide a theoretically sound model that allow for sound theoretical analysis. Therefore:

Question 10. *Can we extend the idea of Autoencoders to Kernel methods?*

Contribution in this thesis (based on [EFG23]). We show starting from AEs, the idea can be extended to Kernel methods by replacing the encoder and decoder with Kernel machines, resulting in the mapping

$$x \in \mathbb{R}^d \xrightarrow{\phi_1(\cdot)} \mathcal{H}_1 \xrightarrow{W_1} z \in \mathbb{R}^h \xrightarrow{\phi_2(\cdot)} \mathcal{H}_2 \xrightarrow{W_2} \hat{x} \in \mathbb{R}^d$$

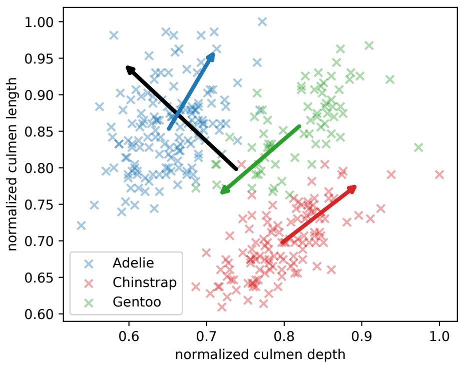


Figure 1.4: Illustration simpson's paradox in the 'penguin dataset' [GWF14]. The three clusters and their first principal component are plotted in red, blue, and green respectively, and the principal direction for the full dataset is in black.

where typically $h < d$ under careful regularization on z and optimizing over reducing the reconstruction error. While this model is interesting in and of itself for learning on small data, it can also build the foundation for a more thorough theoretical analysis of standard non-linear AEs.

Analysis. The main contributions of this part are the development and analysis of a new Kernel method for reconstruction models. We present a Kernel AE where the encoder learns a low-dimensional representation. We show that a Kernel AE can be learned by solving a Kernel matrix-based optimization problem. Furthermore, we derive generalization error bounds for the proposed Kernel models which show that the prediction of the model improves with an increased number of unlabelled data. We empirically demonstrate that Kernel AE performs on par or outperforms classification on the original features as well as Kernel PCA and compare them to neural network representation learning models.

1.4.2 Learning Cluster Specific Representations

Our proposed Kernel AE takes a step towards better understanding the representations obtained by AEs, however, it does not provide a full picture of the embeddings obtained in the non-linear setting yet. Therefore we take a step back and reconsider the question of defining a good embedding on a higher level. As previously stated in Q.2 there is not a unique 'good' representation of data. However we argue that if we consider data with inherent cluster structures, those structures should be represented in the embedding as well. While Autoencoders are widely used in practice, they do not necessarily fulfill the above condition on the embedding as they obtain a single representation of the data. Formally it has been shown that in the linear setting, the encoder recovers the principal components of the full dataset [Kun+19]. A classical example where such a representation would not capture the data structure well is the so-called *simpson's paradox* [Sim51], a known phenomenon in statistics where the trend in clusters does not align with the trend that appears in the full dataset, often observed in social-science and medical-science statistics [Wag82; Hol16].

Figure 1.4 illustrates this by showing that one representation through the principal direction, shown as the black arrow, for all the data points does not capture the structures of individual clusters well as shown by the colored arrows.

> **Question 11.** *Is it beneficial to learn cluster-specific representations in Autoencoders and if so how can we modify a standard Autoencoder to obtain this result?*

Contributions in this thesis (based on [Ess+23]). To be able to model such structures we introduce a modified AE architecture we term *Tensorized Autoencoders (TAE)* that, in the linear setting, provably recovers the principal directions of *each cluster* while jointly learning the cluster assignment. This new AE architecture considers a single AE for each cluster allowing us to learn distinct cluster representations.

We prove that this architecture with linear encoder and decoder recovers the h leading eigenvectors of the different clusters instead of the eigenvectors of the whole data set as done by standard linear AEs. Empirically we demonstrate that the general concept can be extended efficiently to the non-linear setting as well and TAE performs well in clustering and de-noising tasks on real data. Finally, we show how TAE is connected to Expectation maximization.

1.5 Learning from Ordinal Data

In our final setup we consider data where we do not have access to either explicit similarities or feature labels but only ordinal comparisons of the form object o_i is more similar to object o_j than to object o_r with the goal to recover underlying cluster structures. We propose a new similarity based on triplet comparisons to solve this problem.

Passive Comparisons. The key bottleneck in comparison-based learning is the overall number of available comparisons: given n examples, there exist $\mathcal{O}(n^3)$ different triplets. In practice, it means that, in most applications, obtaining all the comparisons is not realistic. Instead, most approaches try to use as few comparisons as possible. This problem is relatively easy when the comparisons can be actively queried and it is known that $\Omega(n \ln n)$ [HGL17; EZK18] adaptively selected comparisons are sufficient. On the other hand, this problem becomes harder when the comparisons are passively obtained. The general conclusion in most theoretical results on learning from passive ordinal comparisons is that, in the worst case, almost all the $\mathcal{O}(n^3)$ comparisons should be observed [JN11; EZK18].

> **Question 12.** *How many passive comparisons do we need to recover planted clusters exactly and what algorithm can we use to achieve this?*

While these new similarities can be used to solve any machine learning problem, we show that it is probably good for clustering under a well-known planted partitioning framework [Abb17; YSC18; XJL20]. [PEG20] shows that, when the number of clusters is constant, $\Omega(n(\ln n)^2)$ passive triplets are sufficient for exact recovery. Indeed, to cluster an example, it is necessary to observe it in a comparison at least once as, otherwise, it can only be assigned to a random cluster. Therefore this result is near-optimal as to cluster n objects, it is necessary to have access to at least $\Omega(n)$ comparisons.

Contributions in this thesis (based on [PEG20]). To obtain these results, we study an Semidefinite programming (SDP) based clustering method. We empirically validate the proposed approach and present a strategy to tune hyperparameters in the SDP; empirically validate our theoretical findings; and demonstrate the performance of the proposed approaches on real datasets. In addition, we present results for a *Quadruplet setting* of the form *objects o_i and o_j are more similar to each other than objects o_k and o_l* in Chapter 7 which we do not include here in the introduction of brevity.

1.6 Thesis Structure

The overall structure of the thesis follows the outline presentation previously in the introduction in section 1.2 — 1.5. We start in Chapter 2 with the transductive setting where we consider a graph with features and partial labels with the goal to label the unlabelled nodes using graph neural networks. We analyze the generalization properties under planted model assumptions. From there we assume we no longer have access to any labels or the underlying similarity structure but instead, we have access to features and inter-sample relations through semantic similarities. In this setting, we first analyze the learning dynamics of (non)contrastive SSL models in Chapter 3 as well as propose Kernel SSL models in Chapter 4. Further decreasing the amount of available information we reach the classical unsupervised learning setting where only features for each object are available. In Chapter 5 we propose and analyze a Kernel-based AE model and in Chapter 6 we propose and analyze a modified AE architecture, that allows for learning cluster-specific representations. Finally in Chapter 7 we consider a data setting where we only have access to ordinal relations between data. We propose a clustering approach, that relies only on a near-optimal number of passive comparisons.

1.6.1 Included works

The following papers (sorted by the order they are included in the thesis) build the foundation of this thesis. Equal contributions to the paper are marked by *.

[EVG21] *Learning Theory Can (Sometimes) Explain Generalisation in Graph Neural Networks.* **Pascal M. Esser**, Leena C. Vankadara, Debarghya Ghoshdastidar. In Conference on Neural Information Processing Systems (NeurIPS), 2021.
This work is included as Chapter 2. PME derived or contributed to all theoretical results, developed and performed the experiments, drafted the initial version of the paper and actively contributed to writing and editing.

[EMG23] *Representation Learning Dynamics of Self-Supervised Models.* **Pascal M. Esser***, Satyaki Mukherjee*, Debarghya Ghoshdastidar. Transactions on Machine Learning Research (TMLR), 2024
This work is included as Chapter 3. PME derived or contributed to all theoretical results, developed and performed the experiments, drafted the initial version of the paper and actively contributed to writing and editing.

[EFG23] *Non-Parametric Representation Learning with Kernels* **Pascal M. Esser***, Maximilian Fleissner*, Debarghya Ghoshdastidar. Accepted to AAAI Conference on Artificial Intelligence (AAAI-24), 2024.
This work is included as Chapter 4 and Chapter 5. PME derived or contributed to all theoretical results, developed and performed the experiments, drafted the initial version of the paper and actively contributed to writing and editing.

[Ess+23] *Improved Representation Learning Through Tensorized Autoencoders.* **Pascal M. Esser***, Satyaki Mukherjee*, Mahalakshmi Sabanayagam*, Debarghya Ghoshdastidar. In International Conference on Artificial Intelligence and Statistics (AISTATS), 2023.
This work is included as Chapter 6. PME derived or contributed to the theoretical results, contributed to the experiments, drafted the initial version of the paper and actively contributed to writing and editing.

[PEG20] *Near-Optimal Comparison Based Clustering.* Michaël Perrot*, **Pascal M. Esser***, Debarghya Ghoshdastidar*. In Conference on Neural Information Processing Systems (NeurIPS), 2020.
This work is included as Chapter 7. PME contributed to the algorithmic part of the paper by developing the initial version of the experiments, contributed to the experiments and contributed to writing and editing of the paper.

1.6.2 Notation

Data. Given data $x_1, \dots, x_n$ in $\mathbb{R}^d$ we collect then into a matrix $X \in \mathbb{R}^{d \times n}$. Corresponding labels are denoted by $y_1, \dots y_n$. If the data has a cluster structure, let κ be the true number of clusters. If we assume access to a graph $\mathcal{G}$ with n vertices, corresponding to the respective feature vectors $x_1, \dots, x_n$, the edge (i, j) denotes the similarity of vertices i and j. The graph structure is represented by an adjacency matrix $A \in \mathbb{R}^{n \times n}$.

Machine Learning / Neural Network Models. We denote an encoding function, parameterized by Θ as $f_\Theta(x_i)$, mapping the data to a representation $z_i \in \mathbb{R}^h$ and decoders as $g_\Psi(z_i)$ We denote trainable weights by for layer j, of width h_j as W_j. The number or layers is defined as J. ψ is used to denote our non-linear activation function and we abuse notation to also denote its co-ordinate-wise application on a vector by $\psi(\cdot)$.

Dynamics. Let the machine output is denoted by $f(\cdot)$. While f is time dependent and should be more accurately denoted as f_t we suppress the subscript where obvious. For any time dependent function, for instance f, denote $\mathring{f}$ to be its time derivative i.e. $\frac{df_t}{dt}$.

General. Let $\mathbb{I}_{\{\cdot\}}$ be the indicator function. Let $[n] := 1, \dots n$.

Matrix Calculus. Let $\mathbb{I}_n$ be an $n \times n$ identity matrix and $\mathbf{1}$ the all ones vector. For a matrix B let $\|B\|_F$ and $\|B\|_2$ be the standard frobenious norm and the $L2$-operator norm respectively. $\langle \cdot, \cdot \rangle$ is used to denote the standard dot product. Let $\mathrm{Tr}(B)$ be the trace of matrix B.

Kernel Methods. For a given Kernel $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$, we denote $\phi : \mathbb{R}^d \to \mathcal{H}$ for its canonical feature map into the associated RKHS $\mathcal{H}$. We write $\Phi := (\phi(x_1), \dots, \phi(x_n))$ and define $\mathcal{H}_X$ as the finite-dimensional subspace spanned by Φ. Recall that $\mathcal{H}$ can be decomposed as $\mathcal{H}_X \oplus \mathcal{H}_X^\perp$. We denote by $K = \Phi^\top \Phi \in \mathbb{R}^{n \times n}$ the Kernel matrix, and define $k(x', X) = \Phi^\top \phi(x')$. On a formal level, the problem could be stated within the generalised framework of matrix-valued Kernels $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^{h \times h}$, because the vector-valued RKHS $\mathcal{H}(K)$ associated with a matrix-valued Kernel K naturally contains functions W that map from $\mathbb{R}^d$ to $\mathbb{R}^h$. For the scope of this paper however, it is sufficient to assume $K(x, y) = \mathbb{I}_h \cdot k(x, y)$ for some scalar Kernel $k(x, y)$ with real-valued RKHS $\mathcal{H}$. Then, the norm of any $W \in \mathcal{H}(K)$ is simply the Hilbert-Schmidt norm that we denote as $\|W\| = \|W\|_\mathcal{H}$ (for finite-dimensional matrices, the Frobenius norm), and learning the embedding from $\mathbb{R}^d$ to $\mathbb{R}^h$ reduces to learning h individual vectors $w_1, \dots, w_h \in \mathcal{H}$. In other words, we can interpret $W \in \mathcal{H}(K)$ as a (potentially infinite-dimensional) matrix with columns $w_1, \dots, w_h \in \mathcal{H}$, sometimes writing $W = (w_1, \dots, w_h)$ for notational convenience. To underline the similarity with the deep learning framework, we denote $W^\top \phi(x) = (\langle w_t, \phi(x) \rangle)_{t=1}^h = (w_1(x), \dots, w_h(x)) \in \mathbb{R}^h$, where we invoke the reproducing property of the RKHS $\mathcal{H}$ in the last step. We denote by W^* the adjoint operator of W (in a finite-dimensional setting, W^* simply becomes the transpose $W^\top$). The constraint $W^* W = \mathbb{I}_h$ enforces orthonormality between all pairs $(w_i, w_j)_{i,j \le h}$.

Any additional, Chapter specific variables are introduced throughout the thesis.

Chapter 2

Learning Transductive Problems using Graph Neural Networks

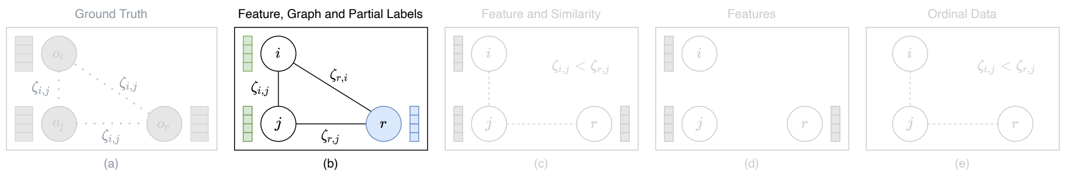


Figure 2.1: Transductive Learning. Features are given as well as $\{\zeta_{i,j}\}$ in form of a graph structure where each node is give by one of the objects. In addition from some objects labels are given.

When discussing 'learning from unlabelled data' the first notion that comes to mind is that of learning only from unlabelled data (e.g. in the context of clustering) or in the context of using them for pre-training. However, the setting of this chapter investigates the intriguing setting where we learn from labelled and unlabelled data, connected through a graph structure at the same time with the goal to recover the labels of the unlabelled objects as shown in Figure 2.1. Therefore compared to the settings we explore in the later sections of the thesis this setting contains a lot of information in relation to the unlabelled data points.

This setting can be found in a wide range of real-world applications and in recent years the main approach for solving this problem has been neural network-based [Wu+18; Fan+19; Hos17; Kip+18; Ham+17; Cui+19; Zho+20]. The rapid gain in popularity has, however, come at the cost of interpretability and reliability of complex neural network architectures. Hence, there has been an increasing interest in understanding generalization and other theoretical properties of neural networks in the theoretical machine learning community [Fel20; Aro+19a; MB17; NK19; TKM20; Gho+20]. Most of the existing theory literature focuses on the supervised learning problem, or more precisely, the setting of inductive inference. In contrast, there is a general lack of understanding of transductive problems, in particular the role of unlabeled data in training. Consequently, there has also been little progress in rigorously understanding one of the widely used tools for transductive inference — Graph neural networks (GNN).

GNNs were introduced by [GMS05; Sca+09], who used recurrent neural network architectures, for the purpose of transductive inference on graphs, that is, the task of labelling all the nodes of a graph given the graph structure, all node features and labels for few nodes. Broadly, GNNs use a combination of local aggregation of node features and non-linear transformations to predict on unlabelled nodes. In practice, the exact form of aggregation and combination steps varies across architectures to solve domain specific tasks [KW17; Bru+14; DBV16; Vel+18; Xu+19]. While some GNNs focus on the transductive setting, sometimes referred to as semi-supervised node classification,[1] GNNs have also found success in supervised learning, where the task is to label entire graphs, in contrast to labelling nodes in a graph. While the understanding of GNNs is limited, there are empirical approaches to study GNNs in the transductive [Boj+18] and supervised setting [Zha+18; Yin+18]. For an extensive survey on the state of the art of GNNs see for example [Wu+20].

2.1 Statistical Framework for Transductive Learning on GNN

For a rigorous analysis, we introduce a statistical learning framework for graph based transductive inference in Section 2.1.1. Based on this, we derive generalisation error bounds based on VC Dimension in Section 2.1.2 and demonstrate that the bounds have limited expresitivity even under strong assumptions. To overcome this problem we consider transductive Rademacher complexity in Section 2.1.3. While without further assumptions this bound also gives limited insight, the bound is more expressive and, in Section 2.2, we show that it can provide meaningful bounds under certain distributional assumptions.

2.1.1 Framework for Transductive Learning

We briefly recall the framework for supervised binary classification. Let $\mathcal{X} = \mathbb{R}^d$ be the *domain or feature space* and $\mathcal{Y} = \{\pm 1\}$ be the *label set.* The goal is to find a predictor $f : \mathcal{X} \to \mathcal{Y}$ based on m training samples $S := \{(x_i, y_i)\}_{i=1}^m \subset \mathcal{X} \times \mathcal{Y}$ and a loss function $\ell : \mathcal{Y} \times \mathcal{Y} \to [0, \infty)$. In a statistical framework, we assume that S consists independent labelled samples from a distribution $\mathcal{D} = \mathcal{D}_\mathcal{X} \times \eta$, that is, $x_i \sim \mathcal{D}_\mathcal{X}$ and $y_i \sim \eta(x_i)$, where $\eta(\cdot)$ governs the label probability for each feature. The goal of learning is to find h that minimises the *risk / generalisation error* $\mathcal{L}_\mathcal{D}(f) := \mathbb{E}_{(x,y)\sim\mathcal{D}}[\ell(f(x), y)]$. Since, $\mathcal{L}_\mathcal{D}(f)$ cannot be computed without the knowledge of $\mathcal{D}$, one minimises the *empirical risk* over the training sample S as $\mathcal{L}_S(f) := \frac{1}{m}\sum_{i=1}^m \ell\left(f\left(x_i\right), y_i\right).$

Transductive learning. In transductive inference, one restricts the domain to be $\mathcal{X} := \{x_i\}_{i=1}^n$, a finite set of features $x_i \in \mathbb{R}^d$. Without loss of generality, one may assume that the labels $y_1, \ldots, y_m \in \{\pm 1\}$ are known, and the goal is to predict $y_{m+1}, \ldots y_n$. The problem can be reformulated in the statistical learning framework as follows. We define the feature

[1]In semi-supervised learning, the learner is given a training set of labeled and unlabeled examples and the goal is to generate a hypothesis that generates predictions on the unseen examples. In transductive learning all features are available to the learner, and the goal is to transfer knowledge from the labeled to the unlabeled data points. The focus of graph-based semi-supervised learning aligns more with the latter setting.

distribution $\mathcal{D}_{\mathcal{X}}$ to be uniform over the n features, whereas $y_i \sim \eta(x_i)$ for some unknown distribution η. Hence $\mathcal{D} := \text{Unif}([n]) \times \eta$ is the joint distribution on $\mathcal{X} \times \mathcal{Y}$, and the goal is to find a predictor $h : \mathcal{X} \to \mathcal{Y}$ that minimises the *generalisation error* $\mathcal{L}_u(f) := \frac{1}{n-m}\sum_{i=m+1}^{n} \ell\left(f(x_i), y_i\right)$. In addition we define the *empirical error* of h to be $\widehat{\mathcal{L}}_m(f) := \frac{1}{m}\sum_{i=1}^{m} \ell\left(f(x_i), y_i\right)$ and the full sample error of f to be $\mathcal{L}_n(f) := \frac{1}{n}\sum_{i=1}^{n} \ell\left(f(x_i), y_i\right)$, where the latter is defined over both labelled and unlabelled instances. The purpose of this section is to derive generalisation error bound for graph based transduction of the form

$$\mathcal{L}_u(f) \leq \widehat{\mathcal{L}}_m(f) + \text{complexity term}. \tag{2.1}$$

The complexity term is typically characterised using learning-theoretic terms such as VC Dimension and Rademacher complexity. For the transductive setting see [TLP16; EYP09; TBK14].

Graph-based transductive learning. A typical view of graph information in transductive inference is as a form of a regularisation [BMN04]. In contrast, we view the graph as part of the hypothesis class and derive the impact of the graph information on the complexity term. We assume access to a graph $\mathcal{G}$ with n vertices, corresponding to the respective feature vectors $x_1, \ldots, x_n$, and edge (i,j) denoting similarity of vertices i and j. For ease of exposition, we define the matrix $X \in \mathbb{R}^{n\times d}$ as the feature matrix[2] with rows being the n feature vectors of dimension d. We also abuse notation to write a predictor as $h : \mathbb{R}^{n\times d} \to \{\pm 1\}^n$. Furthermore, typically neural networks output a soft predictor in $\mathbb{R}$, that is further transformed into labels through sign or softmax functions. Hence, much of our analysis focuses on predictors $h : \mathbb{R}^{n\times d} \to \mathbb{R}^n$, and corresponding hypothesis class

$$\mathcal{F}_{\mathcal{G}} = \left\{ f : \mathbb{R}^{n\times d} \to \mathbb{R}^n \;\; : \;\; f \text{ is parametrized by } \mathcal{G} \right\} \;\subset \mathbb{R}^{[n]}.$$

When applicable, we denote the hypothesis class of binary predictors obtained through sign function as $\text{sign} \circ \mathcal{F}_{\mathcal{G}} = \{\text{sign}(h) \mid h \in \mathcal{F}_{\mathcal{G}}\}$. Note that $\text{sign} \circ \mathcal{F}_{\mathcal{G}} \subset \mathcal{F}_{\mathcal{G}}$, and hence, VC Dimension or Rademacher complexity bounds for the latter also hold for the hypothesis class of binary predictors. We also note that the presented analysis holds for both sign and sigmoid function for binarisation.

Formal setup of GNNs. We next characterise the hypothesis class for graph neural networks. Consider graph-based neural network model with the propagation rule for layer j denoted by $f_j(H) : \mathbb{R}^{h_{j-1}} \to \mathbb{R}^{h_j}$ with layer wise input matrix $H \in \mathbb{R}^{n\times h_{j-1}}$. Consider a class of GNNs defined over J layers, with dimension of layer $j \in [J]$ being h_j and $S \in \mathbb{R}^{n\times n}$ the graph diffusion operator. Let ψ denote the point-wise activation function of the network,

[2]Remark on the notation: In the introduction and also in some of the consecutive chapters we define the features as an $d \times n$ matrix where we here define it as a $n \times d$ matrix. This is due to different convention in the areas of application. The former is commonly used in the transductive setting where as the letter is the convention in the (deep) representation learning setup.

which we assume to be a Lipschitz function with Lipschitz constant L_ψ. We assume ψ to be the same throughout the network. We define the hypothesis class over all J-layer GNNs as:

$$\mathcal{F}_{\mathcal{G}}^{\psi} := \left\{ f_{\mathcal{G}}^{\psi}(X) = g_J \circ \cdots \circ f_0 \;:\; \mathbb{R}^{n\times d} \to \{\pm 1\}^n \right\} \tag{2.2}$$

$$\text{with} \quad f_j := \psi\left(b_j + S f_{j-1}(H)\, W_j\right),\; j \in [J], \quad f_0 := X. \tag{2.3}$$

where (2.3) defines the layer wise transformation with $W_j \in \mathbb{R}^{h_{j-1}\times h_j}$ as the trainable weight matrix and $b_j \in \mathbb{R}^{h_j}$ the bias term. Here, the graph is treated as part of the hypothesis class, as indicated by the subscript in $\mathcal{F}_{\mathcal{G}}^{\psi}$. For ease of notation we drop the superscript for non-linearity where it is unambiguous. For the diffusion operator S, we consider two main formulations during discussions:

$$S_{\text{loop}} := A + \mathbb{I} \qquad \text{self loop}$$

$$S_{\text{nor}} := (D+\mathbb{I})^{-\frac{1}{2}}(A+\mathbb{I})(D+\mathbb{I})^{-\frac{1}{2}}, \qquad \text{degree normalized [KW17]}$$

where A denotes the graph adjacency matrix and D is the degree matrix. However, most results are stated for general S.

2.1.2 Generalisation Error-bound using VC Dimension

The main focus of this section is the notion of generalisation, that is, understanding how well a GNN can predict the classes of an unlabelled set given the training data. We start with one of the most fundamental learning-theoretical concepts in this context which is the Vapnik–Chervonenkis (VC) dimension of a hypothesis class, a measure of the complexity or expressive power of a space of functions learned by a binary classification algorithm. The following result bounds the VC Dimension for the hypothesis class $\mathcal{F}_{\mathcal{G}}^{\psi}$, and use it to derive a generalisation error bound with respect to the full sample error $\mathcal{L}_n$, which is close to the generalisation error for unlabelled examples $\mathcal{L}_u$ when $m \ll n$.

Definition 1 (VC-Dimension)**.** Following [VC71]. Let $\mathcal{F} \subseteq \{\pm 1\}^{\mathcal{X}}$ be a binary function class and $f \in \mathcal{F}$ a function in this class. We define $C = (x_1, \cdots x_m) \in \mathcal{X}^m$ and say that C is shattered by f if for all assignments of labels to points in C there exists a parameterization of f such that f predicts all points in C without error. From there we define the **VC-dimension** of a non-empty hypothesis class $\mathcal{F}$ as the cardinality of the largest possible subset of $\mathcal{X}$ that can be shattered by $\mathcal{F}$. If $\mathcal{F}$ can shatter arbitrarily large sets, then $\mathrm{VCdim}(\mathcal{F}) = \infty$.

Proposition 1 (Generalisation error bound for GNNs using VC Dimension)**.** *For the hypothesis class over all **linear GNNs**, that is $\psi(x) := x$, with binary outputs, the VC Di-*

mension is given by

$$\mathrm{VCdim}\big(\operatorname{sign}\circ\mathcal{F}_{\mathcal{G}}^{linear}\big) = \min\left\{d, \operatorname{rank}(S), \min_{j\in[J-1]}\{h_j\}\right\}.$$

Similarly, the VC Dimension for the hypothesis class of GNNs with ***ReLU non-linearities*** *and binary outputs, can be bounded as*

$$\mathrm{VCdim}\big(\operatorname{sign}\circ\mathcal{F}_{\mathcal{G}}^{\mathrm{ReLU}}\big) \leq \min\left\{\operatorname{rank}(S), h_{J-1}\right\}.$$

Using the above bounds, it follows that, for any $\delta \in (0,1)$, the generalisation error for any $f \in \operatorname{sign}\circ\mathcal{F}_{\mathcal{G}}$ satisfies, with probability $1-\delta$,

$$\mathcal{L}_n(h) - \widehat{\mathcal{L}}_m(h) \leq \sqrt{\frac{8}{m}\left(\min\left\{\operatorname{rank}(S), h_{J-1}\right\}\cdot\ln(em) + \ln\left(\frac{4}{\delta}\right)\right)}. \tag{2.4}$$

In the same line we can additionally note that we get similar results (of the form that in expectation $\operatorname{rank}(A) = n$) for more complex models like stochastic block models which we will discuss in further detail later, as for any matrix $A \in \mathbb{R}^{n\times n}$ there are invertible matrices arbitrarily close to A, under any norm for the $n\times n$ matrices. Motivated by those first findings we consider a different complexity measure, less reliant on combinatorial arguments, to get more insight into the role of graph information.

To interpret Proposition 1, we note that, by introducing the non-linearity, we lose the information about the hidden layers, except the last one and therefore also on the feature dimension. Nevertheless, the information on the graph information (that we are primarily interested in) is preserved. There are two situations that arise. If $h_{J-1} \leq \operatorname{rank}(S)$, then, from Proposition 1, the graph information is redundant and one could essentially train a fully connected network without diffusion on the labelled features, and use it to predict on unlabelled features. The graph information has an influence for $\operatorname{rank}(S) < h_{J-1}$. While general statements on the influence of the graph information are difficult, by considering specific assumptions on the graph we can characterise the generalisation error further.

For linear GNN on graph $\mathcal{G}$, one can bound the VC Dimension between those for empty and complete graphs, that is,

$$\mathrm{VCdim}\big(\operatorname{sign}\circ\mathcal{F}_{\mathrm{complete}}^{\mathrm{linear}}\big) \leq \mathrm{VCdim}\big(\operatorname{sign}\circ\mathcal{F}_{\mathcal{G}}^{\mathrm{linear}}\big) \leq \mathrm{VCdim}\big(\operatorname{sign}\circ\mathcal{F}_{\mathrm{empty}}^{\mathrm{linear}}\big).$$

Moreover, for disconnected graphs, $\operatorname{rank}(S)$ is related to the number of connected components. Similar observations hold for upper bounds on VC Dimension for ReLU GNNs. Based on this observation for simple settings, it holds that considering graph information in comparison to a fully connected feed forward neural network leads to a decrease in the complexity of the class, and therefore also in the generalisation error. However, the graph $\mathcal{G}$ is connected in most practical scenarios, and even under strong assumptions on the graph, for example under consideration of Erdös-Rényi graphs[3] or stochastic block models, $\mathrm{rank}(S) = O(n)$ [CV08]. Therefore, for the case $h_{J-1} > \mathrm{rank}(S) = O(n)$, Proposition 1 provides a generalisation error bound of $O\left(\sqrt{\frac{n \cdot \ln m}{m}}\right)$, which holds trivially for 0-1 loss as $n > m$. Furthermore, $\mathrm{rank}(S)$ is often similar for both self-loop S_{loop} and degree-normalised diffusion S_{nor}, and hence, the VC Dimension based error bound does not reflect the positive influence of degree normalisation—a fact that can be explained through stability based analysis [VZ19].

2.1.3 Generalisation Error-bound using Transductive Rademacher Complexity

Due to the triviality of VC Dimension based error bounds, we consider generalization error bounds based on transductive Rademacher complexity (TRC). We start by defining TRC that differs from inductive Rademacher complexity by taking the unobserved instances into consideration.

Definition 2 (Transductive Rademacher complexity [EYP09])**.** Let $\mathcal{V} \subseteq \mathbb{R}^n$, $p \in [0, 0.5]$ and m the number of labeled points. Let $\sigma = (\sigma_1, \ldots, \sigma_n)^T$ be a vector of independent and identically distributed random variables, where σ_i takes value $+1$ or -1, each with probability p, and 0 with probability $1 - 2p$. The transductive Rademacher complexity (TRC) of $\mathcal{V}$ is defined as

$$\mathfrak{R}_{m,n}(\mathcal{V}) := \left(\frac{1}{m} + \frac{1}{n-m}\right) \cdot \mathop{\mathbb{E}}_{\sigma}\left[\sup_{v \in \mathcal{V}} \sigma^\top v\right].$$

The following result derives a bound for the TRC of GNNs, defined in (2.2)–(2.3), and states the corresponding generalization error bound. The bound involves standard matrix norms, such as $\|\cdot\|_\infty$ (maximum absolute row sum) and the 'entrywise' norm, $\|\cdot\|_{2\to\infty}$ (maximum 2-norm of any column).

Theorem 1 (Generalization error bound for GNNs using TRC)**.** *Consider $\mathcal{F}_{\mathcal{G}}^{\psi,\beta,\omega} \subseteq \mathcal{F}_{\mathcal{G}}^{\psi}$ such that the trainable parameters satisfy $\|b_j\|_1 \leq \beta$ and $\|W_j\|_\infty \leq \omega$ for every $j \in [J]$. The transductive Rademacher complexity (TRC) of the restricted hypothesis class is bounded as*

$$\mathfrak{R}_{m,n}(\mathcal{F}_{\mathcal{G}}^{\psi,\beta,\omega}) \leq \frac{c_1 n^2}{m(n-m)} \left(\sum_{j=0}^{J-1} c_2^j \|S\|_\infty^j\right) + c_3 c_2^J \|S\|_\infty^J \|SX\|_{2\to\infty} \sqrt{\log(n)}\,, \tag{2.5}$$

where $c_1 := 2L_\psi \beta$, $c_2 := 2L_\psi \omega$, $c_3 := L_\psi \omega \sqrt{2/d}$ and L_ψ is Lipschitz constant for activation ψ.

The bound on TRC leads to a generalisation error bound following [EYP09]. For any $\delta \in$

[3]From [CV08] we know the following result: Let c be a constant larger then $\frac{1}{2}$, then for any $\frac{c \ln n}{n} \leq p \leq \frac{1}{2}$ for a random $\mathcal{G}$ graph sampled from a Erdös-Rényi graph has $\mathrm{rank}(A) \leq n - i(\mathcal{G})$ with probability $1 - \mathcal{O}\big((\ln \ln n)^{-\frac{1}{4}}\big)$, where $i(\mathcal{G})$ denotes the number of isolated vertices in $\mathcal{G}$.

$(0,1)$, *the generalisation error for any* $h \in \mathcal{F}_{\mathcal{G}}^{\psi,\beta,\omega}$ *satisfies*

$$\mathcal{L}_u(h) - \widehat{\mathcal{L}}_m(h) \;\leq\; \mathfrak{R}_{m,n}(\mathcal{F}_{\mathcal{G}}^{\psi,\beta,\omega}) + c_4 \frac{n\sqrt{\min\{m, n-m\}}}{m(n-m)} + c_5 \sqrt{\frac{n}{m(n-m)} \ln\left(\frac{1}{\delta}\right)} \quad (2.6)$$

with probability $1-\delta$*, where* c_4, c_5 *are absolute constants such that* $c_4 < 5.05$ *and* $c_5 < 0.8$.

The additional terms in (2.6) are $O\left(\max\left\{\frac{1}{\sqrt{m}}, \frac{1}{\sqrt{n-m}}\right\}\right)$, and hence, we may focus on the upper bound on TRC (2.5) to understand the influence of the graph diffusion S as well as its interaction with the feature matrix X. The bound depends on the choice of ω, and it suggests a natural choice of $\omega = O(1/\|S\|_\infty)$ such that the bound does not grow exponentially with network depth. The subsequent discussions focus on the dependence on $\|S\|_\infty$ and $\|SX\|_{2\to\infty}$, ignoring the role of ω. Few observations are evident from (2.5), which are also interesting in comparison to existing works.

Role of normalisation. In the case of self-loop, it is easy to see that $\|S_{\text{loop}}\|_\infty = 1 + h_{\max}$, where $h_{\max}$ denotes the maximum degree, and hence, for fixed ω, the bound grows as $O(h_{\max}^J)$. In contrast, for degree normalisation, $\|S_{\text{nor}}\|_\infty = O\left(\sqrt{\frac{h_{\max}}{h_{\min}}}\right)$, and hence, the growth is much smaller (in fact, $\|S_{\text{nor}}\|_\infty = 1$ on regular graphs). It is worth noting that, in the supervised setting, [LUZ21] derived PAC-Bayes for GNN with diffusion S_{nor}, where the bound varies as $O(h_{\max}^J)$. Theorem 1 is tighter in the sense that, for S_{nor}, the error bound has weaker dependence on $h_{\max}$, mainly through $\|SX\|_{2\to\infty}$.

From spectral radius to $\|SX\|_{2\to\infty}$. Previous analyses of GNNs in transductive setting rely on the spectral properties of S. For instance, the stability based generalisation error bound for 1-layer GNN in [VZ19] is $O(\|S\|_2^2)$, where $\|S\|_2$ is the spectral norm. In contrast, Theorem 1 shows TRC $= O(\|S\|_\infty \|SX\|_{2\to\infty})$. This is the first result that explicitly uses the relation between the graph-information and the feature information explicitly via $\|SX\|_{2\to\infty}$. One may note that without node features, that is $X = \mathbb{I}$, we have $\|S\|_{2\to\infty} \leq \|S\|_2 \leq \|S\|_\infty$ and hence, a direct comparison between (2.6) and $O(\|S\|_2^2)$ bound of [VZ19] is inconclusive. However, in presence of features X, Theorem 1 shows that the bound depends on the alignment between the feature and graph information.

In the presence of graph information we can still express Theorem 1 in terms of spectral components by considering $\|SX\|_{2\to\infty} = \max_r \|(SX)_{\cdot r}\|_2 \leq \max_r \|S\|_2 \|X_{\cdot r}\|_2 \leq \|S\|_2 \|X\|_{2\to\infty}$ and $\|SX\|_{2\to\infty}$ which can be bound as $\frac{1}{\sqrt{n}}\|S\|_\infty \leq \|S\|_2$.

Oversmoothing. While the above bound provides a weaker result than (2.5) it allows to directly connect to the oversmoothing [LHW18] effect as the diffusion operator in now only included as $\|S\|_2^j$, $j \in [J]$. Therefore with an increasing number of layers (and especially in the setting considered in [OS20a] where the number of layers goes to infinity), the information provided by the graph gets oversmoothed and therefore, a loss of information can be observed.

Before going in a refined analysis of the bound under graph assumptions let us consider three specific graph settings.

Influence of the graph information: Empty and fully-connected graph. To be able to analyse the influence of the graph information we can note that the graph information comes into play through $\|SX\|_{2\to\infty}$. We can rewrite this expression as $\|SX\|_{2\to\infty} = \max_r \sqrt{\sum_i (SX)_{ir}^2}$ and then by replacing S with the empty ($A = K_{\mathcal{G}}$) and the complete graph ($A = \mathbb{I}$) gives: $\|K_{\mathcal{G}}X\|_{2\to\infty} = \max_r \frac{1}{\sqrt{n}} \sqrt{\left(\sum_j X_{jr}\right)^2}$, and $\|\mathbb{I}X\|_{2\to\infty} = \max_j \sqrt{\sum_j X_{jr}^2}$ and since $\left(\sum_j X_{jr}\right)^2 \leq n \|X_{\cdot j}\|_2^2$ it follows that $\mathfrak{R}(\mathcal{F}^{\psi}_{K_{\mathcal{G}}}) \leq \mathfrak{R}(\mathcal{F}^{\psi}_{\mathbb{I}})$ which is consistent with the observation obtained from the VC-Dimension bound. In both cases the complexity measure of the fully connected graph is lower then the if we would not consider graph information.

Influence of the graph information: b-regular graph. Now consider a setup that incorporates a larger number of graphs. Assume $S := D^{-\frac{1}{2}}(A + \mathbb{I})D^{-\frac{1}{2}}$ and that we only consider the graph information (e.g. $X = \mathbb{I}$), then for a b-regular graph (a graph where every vertex has degree b) we can write $\|S\mathbb{I}\|_{2\to\infty} = \max_j \|S_{\cdot r}\|_2 = \sqrt{\sum_{i\sim r} \frac{1}{D_i D_r}} = \frac{1}{\sqrt{b}} < 1$. Therefore adding graph information results in $\mathfrak{R}(\mathcal{F}^{\psi}_{\mathcal{G}}) \leq \mathfrak{R}(\mathcal{F}^{\psi}_{\mathbb{I}})$ and therefore the complexity resulting in not using graph information upper bounds the complexity that results if we consider graph information.

2.2 Generalization using TRC under Planted Models

The discussion in previous section shows that TRC based generalisation error bound provides some insights into the behaviour of GNNs (example, S_{nor} is preferred over S_{loop}), but the bound is too general to give insights into the influence of the graph information on the generalisation error. The key quantity of interest is $\|SX\|_{2\to\infty}$, which characterises how the graph and feature information interact. To understand this interaction, we make specific distributional assumptions on both graph and node features. We assume that node features are sampled from a mixture of two d-dimensional isotropic Gaussians [Das99], and graph is independently generated from a two-community stochastic block model [Abb17]. Both models have been extensively studied in the context of recovering the latent classes from random observations of features matrix X or adjacency matrix A, respectively. Our interest, however, is to quantitatively analyse the influence of graph information when the latent classes in features X and graph A do not align completely. In Section 2.2.1, we present the model and derive bounds on expected TRC, where the expectation is with respect to random features and graph. We then experimentally illustrate the bounds in Section 2.4, and demonstrate that the corresponding generalisation error bounds indeed capture the trends in performance of GNN.

2.2.1 Model and Bounds on TRC

We assume that the node features are sampled latent true classes, given a $v = (z_1, \dots, z_n) \in \{\pm 1\}^n$. The node features are sampled from a Gaussian mixture model (GMM), that is, feature for node-i is sampled as $x_i \sim \mathcal{N}(z_i\mu, \sigma^2\mathbb{I})$ for some $\mu \in \mathbb{R}^d$ and $\sigma \in (0, \infty)$. We express this in terms of X as

$$X = \mathcal{X} + \epsilon \ \in \ \mathbb{R}^{n\times d}, \qquad \text{where } \mathcal{X} = v\mu^\top \ \text{ and } \epsilon = (\epsilon_{ir})_{i\in[n], r\in[d]} \overset{i.i.d.}{\sim} \mathcal{N}(0, \sigma^2). \tag{2.7}$$

We refer to above as $X \sim$ 2GMM. On the other hand, we assume that graph has two latent communities, characterised by $y \in \{\pm 1\}^n$. The graph is generated from a stochastic block model with two classes (2SBM), where edges (i, r) are added independently with probability $p \in (0, 1]$ if $y_i = y_r$, and with probability $q < [0, p)$ if $y_i \neq y_r$. In other words, we define the random adjacency $A \sim$ 2SBM as a symmetric binary matrix with $A_{ii} = 0$, and $(A_{ir})_{i<r}$ indenpendent such that

$$A_{ir} \sim \text{Bernoulli}(\mathcal{A}_{ir}), \qquad \text{where } \mathcal{A} = \frac{p+q}{2}\mathbf{1}\mathbf{1}^\top + \frac{p-q}{2}yy^\top - p\mathbb{I}. \tag{2.8}$$

The choice of two different latent classes $v, y \in \{\pm 1\}^n$ allows study of the case where the graph and feature information of do not align completely. We use $\Gamma = |y^\top v| \in [0, n]$ to quantify this alignment. Assuming y, v are both balanced, that is, $\sum_i y_i = \sum_i z_i = 0$, one can verify that

$$\|(\mathcal{A} + \mathbb{I})\mathcal{X}\|_{2\to\infty} = \|\mu\|_\infty \left(n(1-p)^2 + \tfrac{1}{4}n(p-q)^2\Gamma^2 - (p-q)(1-p)\Gamma^2\right)^{1/2}, \tag{2.9}$$

which indicates that, for dense graphs ($p, q \gg \frac{1}{n}$), the quantity $\|SX\|_{2\to\infty}$ should typically increase if the latent structure of graph and features are more aligned. This intuition is made precise in the following result that bounds the TRC, in expectation, assuming $X \sim$ 2GMM and $A \sim$ 2SBM.

Theorem 2 (Expected TRC for GNNs under SBM)**.** *Let c_1, c_2 and c_3 as defined in Theorem 1 and $\Gamma := |y^\top v|$. Let $c_6 := (1 + o(1)),\ c_7 := (1 + jo(1)),\ c_8 := (1 + Jo(1))$. Assuming $p, q \gg \frac{(\ln n)^2}{n}$ we can bound the expected TRC for A as defined in (2.8) and X as defined in (2.7) as follows:*

Case 1, Degree normalized: $S = S_{nor}$

$$\mathop{\mathbb{E}}_{X,A}\left[\mathfrak{R}_{m,n}(\mathcal{F}_{\mathcal{G}}^{\psi,\beta,\omega})\right] \le \frac{c_1 n^2}{m(n-m)}\left(\sum_{j=0}^{J-1} c_7 c_2^j \left(\frac{p}{q}\right)^{\frac{j}{2}}\right) + c_8 c_3 c_2^J \left(\frac{p}{q}\right)^{\frac{J}{2}} \sqrt{\ln(n)} \times$$
$$\left(c_6 \|\mu\|_\infty \frac{1 + \left(\frac{p-q}{2}\right)^2 \Gamma^2}{\left(\frac{p+q}{2}\right)^2} + c_6\sqrt{\frac{\ln(n)}{q}}\|\mu\|_\infty + c_6\sqrt{\frac{\sigma(1 + 2\ln(d))}{q}}\right) \tag{2.10}$$

Case 2, Self Loop: $S = S_{loop}$

$$\mathop{\mathbb{E}}_{X,A}\left[\mathfrak{R}_{m,n}(\mathcal{F}_{\mathcal{G}}^{\psi,\beta,\omega})\right] \leq \frac{c_1 n^2}{m(n-m)}\left(\sum_{j=0}^{J-1} c_7 c_2^j (np)^j\right) + c_8 c_3 c_2^J (np)^J \sqrt{\ln(n)} \times$$
$$\left(c_6 \|\mu\|_\infty n\left(1+\left(\frac{p-q}{2}\right)^2\Gamma^2\right) + n\sqrt{\frac{p+q}{2}}\|\mu\|_\infty + c_6 n\sqrt{p}\sigma\sqrt{1+2\ln(d)}\right) \tag{2.11}$$

We note that although the above bounds are stated in expectation, they can be translated into high probability bounds. Furthermore the non-triviality of the proof of Theorem 2 stems from bounds on the expectations of matrix norms, which is more complex than the computation in (2.9). Theorem 2 can be also translated into bounds on the generalisation gap $\mathcal{L}_u(f) - \widehat{\mathcal{L}}_m(f)$. By considering a planted model we can now further extend the observations of Section 2.1.2 and 2.1.3.

Role of normalisation. In the following, we can show that by normalising, the generalisation gap grows slower with increasing graph size. First we compare $\mathbb{E}\left[\|S_{\text{loop}}\|_\infty^j\right] = c_7(np)^j$ with $\mathbb{E}\left[\|S_{\text{nor}}\|_\infty^j\right] = c_7\,(p/q)^{j/2}$ and observe that by normalising we lose the n term. In addition we can consider $\mathbb{E}\left[\|SX\|_{2\to\infty}\right]$ which is bound by the second line in (2.10)–(2.11). Again in the first, deterministic, term we observe that the self loop version contains an additional dependency on n. For the two noise terms we can characterize the behaviour in terms of the density of the graph. Let $\rho = \mathcal{O}(p), \mathcal{O}(q)$ and $\rho \gg \frac{1}{n}$ then we can characterise the *dense setting* as $\rho \asymp \Omega(1)$ and the *sparse setting* as $\rho \asymp \mathcal{O}\left(\frac{\ln(n)}{n}\right)$ and observe that in both case the normalised case grows slower with n:

$$\text{Dense:}\quad \mathbb{E}\left[\|S_{\text{loop}}X\|_{2\to\infty}\right] = \mathcal{O}(n), \qquad \mathbb{E}\left[\|S_{\text{nor}}X\|_{2\to\infty}\right] = \mathcal{O}\left(\sqrt{\ln(n)}\right) \tag{2.12}$$
$$\text{Sparse:}\quad \mathbb{E}\left[\|S_{\text{loop}}X\|_{2\to\infty}\right] = \mathcal{O}\left(\sqrt{n\ln(n)}\right), \quad \mathbb{E}\left[\|S_{\text{nor}}X\|_{2\to\infty}\right] = \mathcal{O}\left(\sqrt{n}\right) \tag{2.13}$$

Influence of the graph information. We consider the idea from Section 2.1.2, to analyse the influence of graph information by comparing the TRC between the case where no graph information is considered, $S = \mathbb{I}$ and S_{nor}. We define the corresponding hypothesis classes as $\mathcal{F}_{\mathbb{I}}^{\psi,\beta,\omega}$ and $\mathcal{F}_{\text{nor}}^{\psi,\beta,\omega}$. Considering the deterministic case $(S = \mathcal{S}, X = \mathcal{X})$ we can observe $\mathfrak{R}_{m,n}(\mathcal{F}_{\mathbb{I}}^{\psi,\beta,\omega}) > \mathfrak{R}_{m,n}(\mathcal{F}_{\text{nor}}^{\psi,\beta,\omega})$ if $\Gamma > \mathcal{O}\left(\frac{n}{\sqrt{n\rho+n}}\right)$. Therefore the random graph setting allows us to more precisely characterize under what conditions adding graph information helps.

2.3 Influence of Depth and Residual Connections on the Generalisation Error

While for standard neural networks increasing the depth is a common approach for increasing the performance, this idea becomes more complex in the context of GNNs as each layer contains a left multiplication of the diffusion operator and we can therefore observe an oversmoothing effect [LHW18] — the repeated multiplication of the diffusion operator in each layer spreads the feature information such that it converges to be constant over all nodes. To overcome this problem, empirical works suggest the use of residual connections [KW17; Che+20a], such that by adding connections from previous layers the network retains some feature information. In this section we investigate this approach in the TRC setting. In Section 2.3.1 we provide the TRC bound for GNN with skip connections and show that it improves the generalisation error compared to vanilla GNNs.

2.3.1 Model and Bounds on TRC for GNN with Residual Connections

While there is a wide range of residual connections, introduced in recent years we follow the idea presented in [Che+20a] where a GNN as defined in (2.3) is extended by an interpolation over parameter α with the features. This setup is especially interesting as it captures the idea of preserving the influence of the feature information more than residual definition that only connect to the previous layer. Formally we can now write the layer wise propagation rule as

$$f_{j+1} := \psi\left((1-\alpha)\left(b_j + Sf_j\left(H\right)W_j\right) + \alpha f_0\left(H\right)\right), \qquad \text{with } \alpha \in (0,1). \tag{2.14}$$

We can now derive a generalization error bound similar to Theorem 1 for the Residual network.

Theorem 3 (TRC for Residual GNNs)**.** *Consider a Residual network as defined in* (2.14) *and $\mathcal{F}_{\mathcal{G}}^{\psi,\beta,\omega} \subset \mathcal{F}_{\mathcal{G}}^{\psi}$ such that the trainable parameters satisfy $\|b_j\|_1 \leq \beta$ and $\|W_j\|_\infty \leq \omega$ for every $j \in [K]$. Then with $\alpha \in (0,1)$ and $c_1 := 2L_\psi\beta$, $c_2 := 2L_\psi\omega$, $c_3 := L_\psi\omega\sqrt{2/d}$ the TRC of the restricted class or Residual GNNs is bounded as*

$$\begin{aligned}\mathfrak{R}_{m,n}(\mathcal{F}_{\mathcal{G}}^{\psi,\beta,\omega}) \;\leq\; & \frac{((1-\alpha)c_1 + \alpha 2L_\psi\|X\|_\infty)n^2}{m(n-m)}\left(\sum_{j=0}^{J-1}(1-\alpha)c_2^j\|S\|_\infty^j\right) \\ & + \alpha 2L_\psi\|X\|_\infty + (1-\alpha)c_3c_2^J\|S\|_\infty^J\|SX\|_{2\to\infty}\sqrt{\log(n)}\end{aligned} \tag{2.15}$$

However observing the bound isolated does not provide new insights beyond Theorem 2 into the behaviour of the generalisation error and therefore we focus on the comparison between GNNs with and without residual connections.

For readability assume $\beta = \|X\|_\infty$. Under this setup we can note that the generalisation error-bound increases with decreased alpha and in extension it follows that the generalisation error-bound for a GNN with skip connection is lower then the one without. This implication is in line with the general notion that residual connections improve the performance of networks [Che+20a; KW17]. Our general intuition behind this behavior is that with increasing α, the network architecture is closer to the one of an one hidden layer network. Having good performance in shallow networks is something that is observed in our experiments as well as in previous work (e.g., [KW17]). Therefore it appears that using the skip connection to obtain a deep network that resembles a shallow one leads to the performance increase.

We conclude this chapter with first discussing how derived bounds correspond to empirical observations and then outline the limitations from the considered approach in more detail.

2.4 Comparison of Theoretical Bounds and Empirical Observations

While we focus on the theoretical analysis of GNNs, in this section we illustrate that the bounds described in Theorem 2 follow the trends observed for the empirical generalization error. The bounds in Section 2.2.1 are derived for binary SBMs so we, therefore, focus on this setting but in addition also show that those observations extend to real-world, multi-class data on the example of the *Cora* dataset [RA15] and *Citeseer* [GBL98]. The results are presented in Figure 2.2.

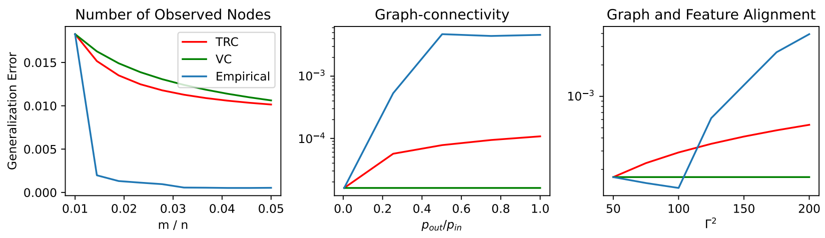
Setup. For the SBM we consider a graph with $n = 500, m = 100, p = 0.1, q = 0.01$ as default. We plot the mean over 10 random initialization. When comparing the empirical performance to the derived theoretical bounds it is important to note that learning theoretical bounds of the form (2.1) can be loose in the context of deep learning models as the absolute value is out of the $(0, 1)$ range[4]. While this does not allow us to numerically show how tight the bound is in practice, we can still make statements about the influence of the change of parameters by analyzing trends, where the experiments validate the correlation between theory and empirical observations. Therefore we plot the bounds in Figure 2.2 normalized on the empirical generalization error for the lowest parameter in the analyzed range[5]. Let us now consider several settings with regards to the influence of *data-set parameterizations, network changes, and real world data.*

Data-set parameterization. First, we can analyze how changes in the data, especially with regard to the graph influence the generalization. This allows us to model how the information, provided in addition to the unlabelled objects can help in learning.

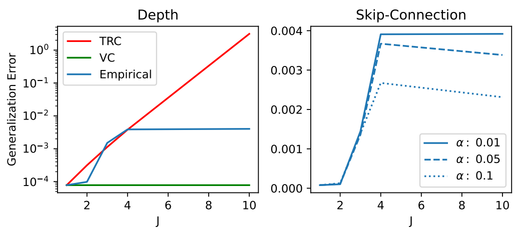
Number of observed points. Let us consider the number of observed points in Figure 2.2a (left) and a realistic setting of $m \ll n - m$ where we see a sharp decline in the setting of few observed points but then the generalization error converges which corresponds to the influence of m as described in (2.10). We see a similar trend for real data in Figure 2.2c

[4]Generalisation error bounds, even for simple machine learning models, can exceed 1 due to absolute constants that cannot be precisely estimated. Hence, the point of interest is the dependence of key parameters; for instance, in a supervised setting, the bounds are $O(1/\sqrt{m})$ and typically exceed 1 for moderate m. This problem is inherent to the bound given in [EYP09] that we base our TRC bounds on, as the slack terms can already exceed 1, and therefore further research on general TRC generalization gaps is necessary to characterize the absolute gap between theory and experiments.

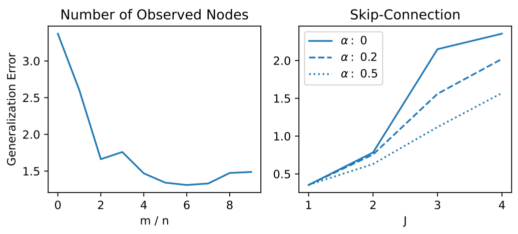
[5]More precisely assume we analyze the change of generalization over a parameter $\{a_0, \dots, a_i\}$ and let $\Delta\mathcal{L}_{a_j}^{Emp}$ be the empirical generalization error for parameterization a_j. We then plot the TRC based bound as $\Delta\mathcal{L}_{a_0}^{Emp} \frac{\Delta\mathcal{L}_{a_j}^{TRC}}{\Delta\mathcal{L}_{a_0}^{TRC}}$ $\forall j \in [i]$ and analogous for the VC based bound.



(a) Numerical evaluation of the derived bounds in the context of changes in the data set. **(left)** change of the number of observed points m in relation to the full graph size (n). **(middle)** Change of the connection probability in and between classes. Fix p_{in} and vary p_{out} relatively to p_{in} . **(right)** Change of the graph and feature allignment Γ.



(b) Numerical evaluation of the derived bounds in the context of changes in the model. **(left)** Change of depth (J). **(right)** Change of depth (J) for different α in the skip-connection model.



(c) Real data illustration. Note that the range for Cora exceeds $(0, 1)$ as the dataset is multi-class and we consider a negative log-likelihood loss. **(left)** Change in number of observed data points on *citeseer dataset* [GBL98] **(right)** Change of depth (J) for different α in the skip-connection model on the *cora dataset* [RA15].

Figure 2.2: Numerical illustration of the derived bounds in the SBM Setting. Plotted is the average empirical performance over 10 runs as well as the corresponding TRC-based generalization error bound (Theorem 1) and VC-based bound (Proposition 1). **(a)** shows the change in generalization error with the change in properties of the data set. **(b)** shows the change in generalization error with changes in the model architecture. **(c)** gives an example of how the observations from the SBM setting also translate to real-world data-sets.

(left). Practically such an observation can be useful as labeling data can be expensive and such results could be useful to determine a necessary and sufficient number of labeled data to obtain a given level of accuracy.

While the above setting can be modeled with the simple VC-based bound as well as with the TRC bound without the assumption of graph models, let us now assume two settings where the change in parameters can be precisely modeled by the TRC bound under the SBM setting.

Graph-Connectivity. Firstly we can analyze how well the graph is connected within and between communities. Intuitively we would assume that if the difference between the in and in-between cluster connectivity is high, it is easier for the network to predict the unknown nodes. This is also reflected in the TRC-based bound as shown in Figure 2.2a (middle).

Feature and graph alignment. Finally, we look at the feature and graph alignment as characterized through Γ^2 in the TRC-based bound (2.9)–(2.10) and observe in Figure 2.2a (right) that with an increase in the latent structure the generalization error increases. While this seems to be counterintuitive a possible explanation could be that reduced alignment helps to prevent overfitting and we observe that the slope matches the empirical results. Again we note that the VC dimension bound (2.4) does not allow us to model this dependency.

Changes in the Network. The second main aspect that is modeled in the derived generalization error bounds is with regard to the considered network architecture.

Depth and oversmoothing. As noted in the previous section the TRC bound predicts an exponential dependency on network depth J which can only partially be observed empirically for the first three layers as shown in Figure 2.2b (left) while for deeper networks the generalization error does no longer change. We observe a similar picture for real data in Figure 2.2c (right). Therefore a study without relying on a recursive proof[6] structure will be necessary to refine and more accurately this dependency on J.

Skip-connections. The observation in Section 2.3 suggests that including residual connections is beneficial with increasing depth which is consistent with the initial reason for introducing residual connections [Che+20a; KW17]. We further illustrate this in the context of the trend shown in (2.15). Extending the analysis of depth we now consider the *residual connections* as defined in (2.14). By (2.15) we can still observe the exponential dependency on J and therefore focus on two main aspects: i) Theoretically the generalization error for the Resnet is upper bound by GNN, which empirically is observed for both the SBM as well as for Cora (Figure 2.2c (right)). ii) Focusing on the Resnets, Theorem 3 predicts an ordering in the generalization error given by α which is again observed for both the SBM as well as for Cora. Therefore while there seems to be a deviation in the exponential behavior of J as given in Theorem 3, the ordering of the generalization error-bound described by α is

[6]However this would need a substantial deviation from the current structure as such dependencies on depth are also found in related PAC bounds [LUZ21].

observed empirically. While this does not give us a complete picture we can note that the remarks on over smoothing suggest that shallower networks are preferable and we again note that the VC dimension bound (2.4) does not provide any useful insights to the influence of depth.

2.5 Discussion

We conclude this section by first discussing additional related works with regard to learning theoretical analysis as well as parallel works that analyze GCNs using alternative tools. Finally, we discuss the implications of the derived results.

Related Work. While empirical studies provide some insights into the behavior of machine learning models, rigorous theoretical analysis is the key to deep insights into a model. The focus of this section is to provide a learning-theoretic analysis of the generalization of GNNs in the transductive setting. Vapnik first studied the problem of transductive inference and provided generalization bounds for empirical risk minimization [Vap82; Vap98]. Subsequent works further analyzed this setting in transductive regression [CM07], and derive VC Dimension and Rademacher complexity for transductive classification [TLP16; EYP09]. Generalization error bounds for 1-layer GNNs have been derived in a transductive setting based on algorithmic stability [VZ19]. In contrast, the focus of the current chapter is on learning-theoretic measures, which have been previously used to analyze GNNs in a supervised setting. In [STH18], VC Dimension is derived for a specific class of GNNs and a generalization error bound is given using node representations. However, their approach of subsuming the graph convolutions under Pfaffian functions does not allow for an explicit representation in terms of the diffusion operator which is important to our presented analysis. [GJJ20] derives the Rademacher complexity for GNN in a supervised setting with a focus of the equivariant structures of the input graphs and does not allow for an explicit inclusion and analysis of the graph information. [LUZ21] provides PAC-Bayes bounds for GNNs that are tighter than the bounds in [GJJ20].

In the context of this work, especially relevant is [OS20b; OS20a]. [OS20a] describes the effect of over-smoothing with an increasing number of layers. [OS20b] analyzes GNNs in the transductive setting. However, they consider a multiscale GCN, and therefore, the analysis is based on a weak-learning/boosting framework where the focus is mostly on exploring the weak learning component, whereas this section focuses on the specific analysis of the generalization bound and the influence of its individual components. In addition, we provide a detailed analysis of its dependence on the graph and feature information and provide a more expressive bound by considering generalization under planted models.

Conclusion. Statistical learning theory has proven to be a successful tool for a complete and rigorous analysis of traditional learning algorithms. At the same time, research suggests that when applied to deep learning models these methods become non-informative. However, on the example of GNNs, we demonstrate that classical statistical learning theory can be used under consideration of the right complexity measure and distributional assumptions on the data to provide insight into trends of deep models. Our analysis provides the first fundamental results on the influence of different parameters on generalization and opens up different lines of follow-up work. Considering the current setup we can also extend the theoretical analysis to more advanced architectures such as dropout or batch normalization. Finally, while our analysis focuses on generalization we suggest that the idea of analyzing GNNs under planted models can be extended to other learning-theoretical measures such as stability or model selection as well as the supervised (graph-classification) setting.

While in the following chapters, we shift to different data settings we will rely on the insights gained from this first chapter. This is done by using the following main insights: a) while in the neural network setting the considered analysis can provide insights into trends for a complete analysis other approaches have to be considered. b) refining the analysis using data assumptions provides more detailed insights through more expressive bounds. This is a very common idea in traditional machine learning for refining a given analysis, this also applies to the deep learning setup.

Chapter 3

Representation Learning Dynamics of Deep Self-Supervised Models

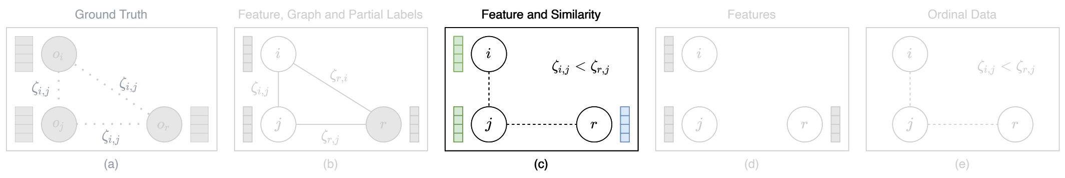


Figure 3.1: The Self-Supervised Setup. Features are given as well as semantic similarities between at least some of the objects. Framed in the Self-Supervised Learning setting this becomes a triplet setting such that objects o_j, o_i and o_r are represented by x_i, x_j^+ and x_r^-.

Let us now assume that we no longer have access to partial labels and the full underlying similarity structure such as a graph but still a notion of what makes objects similar to each other. Therefore we now enter the domain of Self-Supervised Learning (SSL) where we take advantage of our knowledge of semantic similarities to construct similar and dissimilar samples for learning with the goal to map similar objects close to each other in the embedding.

SSL is an important paradigm for learning representations from unlabelled data, and SSL with neural networks has been highly successful in practice but current theoretical analysis of SSL is mostly restricted to generalization error bounds. However, as we observed in Chapter 2 the analysis of neural network-based approaches through traditional learning theoretical measures has significant limitations. In contrast, learning dynamics often provide a precise characterization of the behavior of neural network based models but, so far, are mainly known in supervised settings. In this section, we study the learning dynamics of SSL models, specifically representations obtained by minimizing contrastive and non-contrastive losses. We show in Section 3.1 that a näive extension of the dynamics of multivariate regression to SSL leads to learning trivial scalar representations that demonstrate dimension collapse in SSL. Consequently in Section 3.2, 3.3 we formulate SSL objectives with orthogonality constraints on the weights, and derive the exact (network width independent) learning dynamics of the SSL models trained using gradient descent on the Grassmannian manifold. We also argue that the infinite width approximation of SSL models significantly deviates from the neural tangent kernel approximations of supervised models. We numerically illustrate the validity of our theoretical findings and discuss how the presented results provide a framework for further theoretical analysis of contrastive and non-contrastive SSL in Section 3.4.

Let us start with recalling the (non)contrastive setting we outlined in Chapter 1 more formally.

Contrastive Learning. Contrastive SSL has its roots in the work of [HCL06]. Recent deep learning based contrastive SSL show great empirical success in computer vision [Che+20b; Car+21; JT19], video data [Fer+17; Ser+18], natural language tasks [MM20] and speech [Ste+19; Moh+22]. In general a contrastive loss is defined by considering an anchor image, $x \in \mathbb{R}^d$, positive samples $\{x^+\} \subset \mathbb{R}^d$ generated using data augmentation techniques as well as independent negative samples $\{x^-\} \subset \mathbb{R}^d$. The heuristic goal is to align the anchor more with the positive samples than the negative ones, which is rooted in the idea of maximizing mutual information between similar samples of the data. In this work, we consider a simple contrastive loss minimisation problem along the lines of [Aro+19c], assuming exactly one positive sample x_i^+ and one negative sample x_i^- for each anchor x_i,[1]

$$\min_{\Theta} \sum_{i=1}^{n} f(x_i)^\top \left(f(x_i^-) - f(x_i^+) \right), \tag{3.1}$$

where $f = [f_1(\cdot, \Theta) \ldots f_h(\cdot, \Theta)]^\top : \mathbb{R}^d \to \mathbb{R}^h$ is the embedding function, parameterized by Θ, the learnable parameters.

Non-Contrastive Learning Non-contrastive losses emerged from the observation that negative samples (or pairs) in contrastive SSL are not necessary in practice, and it suffices to maximise only alignment between positve pairs [CH21; Che+20b; Gri+20]. Considering a simplified version of the setup in [Che+20b] one learns a representation by minimising the loss [2]

$$\min_{\Theta} \sum_{i=1}^{n} -f(x_i)^\top f(x_i^+). \tag{3.2}$$

The embedding $f = [f_1(\cdot, \Theta) \ldots f_h(\cdot, \Theta)]^\top : \mathbb{R}^d \to \mathbb{R}^h$, parametrised by Θ, typically comprises of a base encoder network and a projection head in practice [Che+20b].

3.1 Learning Dynamics of Regression and its Näive Extension to SSL

In the context of regression, [JGH18a] show that the evolution dynamics of (infinite width) neural networks, trained using gradient descent under a squared loss, is equivalent to that of specific kernel machines, known as the neural tangent kernels (NTK). The analysis has been

[1]It is straightforward to extend our analysis to multiple positive and negative samples, but the expressions become cumbersome, without providing additional insights.

[2]We simplify [Che+20b] by replacing the cosine similarity with the standard dot product and also by replacing an additional positive sample x_i^{++} by anchor x_i for convenience.

extended to a wide range of models, including convolutional networks [Aro+19b], recurrent networks [Ale+21], overparametrised autoencoders [NWH21], graph neural networks [Du+19b; SEG22] among others. However, these works are mostly restricted to squared losses, with few results for margin loss [Che+21], but derivation of such kernel machines are still open for contrastive or non-contrastive losses (3.1)–(3.2), or broadly, in the context of SSL. To illustrate the differences between regression and SSL, we outline the learning dynamics of multivariate regression with squared loss, and discuss how a näive extension to SSL is inadequate.

3.1.1 Learning Dynamics of Multivariate Regression

Given a training feature matrix $X := [x_1, \cdots, x_n]^\top \in \mathbb{R}^{n\times d}$ and corresponding z-dimensional labels $Y := [y_1, \cdots, y_n]^\top \in \mathbb{R}^{n\times h}$, consider the regression problem of learning a neural network function $f(x) = [f_1(x,\Theta) \dots f_h(x,\Theta)]^\top$, parameterized by Θ, by minimising the squared loss function

$$\mathcal{L}(\Theta) := \frac{1}{2}\sum_{i=1}^{n} \|f(x_i) - y_i\|^2.$$

Under gradient flow, the evolution dynamics of the parameter during training is $\mathring{\Theta} = -\nabla_\Theta \mathcal{L}$ and, consequently, the evolution of the l-th component of network output $(\mathbf{x})$, for any input x, follows the differential equation

$$\begin{aligned}\mathring{f}_l(x) &= \left\langle \nabla_\Theta f_l(x), \mathring{\Theta} \right\rangle \\ &= -\sum_{i=1}^{n}\sum_{j=1}^{h} \langle \nabla_\Theta f_l(x), \nabla_\Theta f_j(x_i)\rangle \left(f_j(x_i) - y_{i,j}\right). \end{aligned} \tag{3.3}$$

While the above dynamics apparently involve interaction between the different dimensions of the output $f(x)$, through $\langle \nabla_\Theta f_l(x), \nabla_\Theta f_j(x_i)\rangle$, it is easy to observe that this interaction does not contribute to the dynamics of linear or kernel models. We formalise this in the following lemma.

Lemma 1 (**No interaction across output dimensions**). *Let $f : \mathbb{R}^d \to \mathbb{R}^h$ be either a linear model $f(x) = \Theta x$, or a kernel machine $f(x) = \Theta\phi(x)$, where ϕ corresponds to the implicit feature map of a kernel k, that is, $k(x,x') = \langle \phi(x), \phi(x')\rangle$.*
Then in the infinite width limit ($h \to \infty$) the inner products between the gradients are given by

$$\langle \nabla_\Theta f_l(x), \nabla_\Theta f_j(x')\rangle = \begin{cases} 0 & \text{if } l \neq j, \\ x^\top x' & \text{if } l = j \text{ (linear case)}, \\ k(x,x') & \text{if } l = j \text{ (kernel case)}. \end{cases}$$

For infinite width neural networks, whose weights are randomly initialised with appropriate scaling, [JGH18a] show that at, initialisation, Lemma 1 holds with k being the neural tangent kernel. Approximations for wide neural networks further imply the kernel remains same during training [LZB20], and so Lemma 1 continues to hold through training.

Remark 1 (Multivariate regression = independent univariate regressions). A consequence of Lemma 1 is that the learning dynamics (3.3) simplifies to

$$\mathring{f}_l(x) = -\sum_{i=1}^{n} \langle \nabla_\Theta f_l(x), \nabla_\Theta f_l(x_i) \rangle \, (f_l(x_i) - y_{i,l}),$$

that is, each component of the output f_l evolves independently from other $f_j, j \neq l$. Hence, one may solve a z-variate squared regression problem as z independent univariate problems. We discuss below that a similar phenomenon is true in SSL dynamics with disastrous consequences.

3.1.2 Dynamics of näive SSL has Trivial Solution

We now present the learning dynamics of SSL with contrastive and non-contrastive losses in (3.1)–(3.2). For convenience, we first discuss the non-contrastive case. Assuming that the network function $f : \mathbb{R}^d \to \mathbb{R}^h$ is parametrised by Θ, the gradient of the loss $\mathcal{L}(\Theta) = \sum\limits_{i=1}^{n} -f(x_i)^\top f(x_i^+)$ is

$$\nabla_\Theta \mathcal{L}(\Theta) = -\sum_{i=1}^{n}\sum_{j=1}^{h} f_j(x_i) \cdot \nabla_\Theta f_j(x_i^+) + f_j(x_i^+) \cdot \nabla_\Theta f_j(x_i)$$

Hence, under gradient descent $\mathring{\Theta} = -\nabla_\Theta \mathcal{L}$, the evolution of each component of $f(x)$, given by $\mathring{f}_l(x) = \left\langle \nabla_\Theta f_l(x), \mathring{\Theta} \right\rangle$ is

$$\mathring{f}_l(x) = \sum_{i=1}^{n}\sum_{j=1}^{h} \langle \nabla_\Theta f_l(x), \nabla_\Theta f_j(x_i) \rangle \, f_j(x_i^+) + \left\langle \nabla_\Theta f_l(x), \nabla_\Theta f_j(x_i^+) \right\rangle f_j(x_i). \tag{3.4}$$

Similarly, in the case of contrastive loss (3.1), the learning dynamics of $f(x)$, for any input x, is similarly expressed by

$$\begin{aligned}\mathring{f}_l(x) = \sum_{i=1}^{n}\sum_{j=1}^{h} &\langle \nabla_\Theta f_l(x), \nabla_\Theta f_j(x_i) \rangle \, f_j(x_i^+) + \left\langle \nabla_\Theta f_l(x), \nabla_\Theta f_j(x_i^+) \right\rangle f_j(x_i) \\ &- \langle \nabla_\Theta f_l(x), \nabla_\Theta f_j(x_i) \rangle \, f_j(x_i^-) - \left\langle \nabla_\Theta f_l(x), \nabla_\Theta f_j(x_i^-) \right\rangle f_j(x_i). \end{aligned} \tag{3.5}$$

We note Lemma 1 depends only on the model and not the loss function, and hence, it is applicable for the SSL dynamics in (3.4)–(3.5). However, there are no multivariate training labels $y \in \mathbb{R}^h$ in SSL (i.e. $y = 0$) that can drive the dynamics of the different components

$f_1, \dots, f_h$ in different directions, which leads to dimension collapse.

Proposition 2 (**Dimension collapse in SSL dynamics**). *Under the conditions of Lemma 1, every component of the network output $u : \mathbb{R}^d \to \mathbb{R}^h$ has identical dynamics. As a consequence, the output collapses to one dimension at convergence.*
For linear model, $f(x) = \Theta x$, the dynamics of $f(x)$ is given by

$$\mathring{f}_l(x) = \sum_{i=1}^{n} (x^\top x_i) f_l(x_i^+) + (x^\top x_i^+) f_l(x_i)$$

for the non-contrastive case, and

$$\mathring{f}_l(x) = \sum_{i=1}^{n} (x^\top x_i)\big(f_l(x_i^+) - f_l(x_i^-)\big) + (x^\top x_i^+ - x^\top x_i^-) f_l(x_i)$$

for the contrastive case. For kernel models, the dynamics is similarly obtained by replacing each $x^\top x'$ by $k(x, x')$.

By the extension of Lemma 1 to neural network and NTK dynamics, one can conclude that Proposition 2 and dimension collapse also happen for wide neural networks, when trained for the SSL losses in (3.1)–(3.2).

Remark 2 (**SSL dynamics for other losses**). One may argue that the above dimension collapse is a consequence of loss definitions in (3.1)–(3.2), and may not exist for other losses. We note that [Liu+23] analyse contrastive learning with linear model under InfoNCE, and the simplified loss closely resembles (3.1), which implies decoupling of output dimensions (and hence, dimension collapse) would also happen for InfoNCE. The same argument also holds for non-constrastive loss in [Che+20b]. However, for the spectral contrastive loss of [Hao+21], the output dimensions remain coupled in the SSL dynamics due to existing interactions $f(x_i)^\top f(x_i^-)$ on the training data.

Remark 3 (**Projections cannot overcome dimension collapse**). [Jin+22] propose to project the representation learned by a SSL model into a much smaller dimension, and show that fixed (non trainable) projectors may suffice. For a linear model, this implies $f(x) = A\Theta x$, where $A \in \mathbb{R}^{r \times h}, r \ll h$ is fixed. It is straightforward to adapt the dynamics and Proposition 2 to this case, and observe that for any $r > 1$, all the r components of $f(x)$ have identical learning dynamics, and hence, collapse at convergence.

3.2 SSL with (Orthogonality) Constraints

For the remainder of the section, we assume that the SSL model $u : \mathbb{R}^d \to \mathbb{R}^h$ corresponds to a 2-layer neural network of the form

$$x \in \mathbb{R}^d \xrightarrow{W_1} \mathbb{R}^{h'} \xrightarrow{\psi(\cdot)} \mathbb{R}^{h'} \xrightarrow{W_2^\top} f(x) = W_2^\top \psi(W_1 x) \in \mathbb{R}^h,$$

where h is the size of the hidden layer and $\Theta = (W_1, W_2^\top)$ are trainable matrices. Whenever needed, we use f^ψ for the output to emphasize the nonlinear activation ψ, and contrast it with a 2-layer linear network $f^{\mathbb{I}}(x) = W_2^\top W_1 x$.

Based on the discussion in the previous section, it is natural to ask how can the SSL problem be rephrased to avoid dimension collapse. An obvious approach is to add regularisation or constraints [BPL22; Erm+21; Car+20]. The most obvious regularisation or constraint on W_1, W_2 is entry-wise, such as on Frobenius norm. While there has been little study on various regularisations in SSL literature, a plethora of variants for Frobenius norm regularisations can be found for autoencoders, such as sum-regularsiation, $\|W_1\|_F^2 + \|W_2\|_F^2$, or product regularisation $\|W_2^\top W_1\|_F^2$ [Kun+19].

It is known in the optimisation literature that regularised loss minimisation can be equivalently expressed as constrained optimisation problems. In this section, we use the latter formulation for convenience of the subsequent analysis. The following result shows that Frobenius norm constraints do not prevent the output dimensions from decoupling, and hence, it is still prone to dimension collapse.

Proposition 3 (**Frobenius norm constraint does not prevent dimension collapse**). *Consider a linear SSL model $f^\psi(x) = W_2^\top \psi(W_1 x)$. The optimisation problem*

$$\min_{W_1, W_2} \mathcal{L}(W_1, W_2) \quad s.t. \quad \|W_1\|_F \leq c_1, \|W_2\|_F \leq c_2,$$

where the loss $\mathcal{L}$ is given by (3.1) or (3.2), has a global solution $f(x) = [a(x)\ 0 \ldots 0]^\top \in \mathbb{R}^h$.

The above result precisely shows dimension collapse for linear networks $f^{\mathbb{I}}$ even with Frobenius norm constraints. An alternative to Frobenius norm constraint can be to constrain the $L2$-operator norm. To this end, the following result shows that, for linear networks, the operator norm constraint can be realised in multiple equivalent ways.

Proposition 4 (**Equivalence of operator norm and orthonognality constraints**). *Consider a linear SSL model $f^{\mathbb{I}}(x) = W_2^\top W_1 x$, and let the loss $\mathcal{L}(W_1, W_2)$ be given by either (3.1) or (3.2) whose general form is $\mathcal{L}(W_1, W_2) = \left\|W_2^\top W_1 C W_1^\top W_2\right\|_2^2$, where C has at least one negative eigenvalue. Then the following optimisation problems are equivalent:*

1. $\displaystyle\min_{W_1, W_2} \frac{\mathcal{L}(W_1, W_2)}{\|W_2\|_2^2 \|W_1\|_2^2};$
2. $\displaystyle\min_{W_1, W_2} \mathcal{L}(W_1, W_2) \quad s.t. \quad \|W_2\|_2 \leq 1, \ \|W_1\|_2 \leq 1;$
3. $\displaystyle\min_{W_1, W_2} \mathcal{L}(W_1, W_2) \quad s.t. \quad \|W_2^\top W_1\|_2 \leq 1;$
4. $\displaystyle\min_{W_1, W_2} \mathcal{L}(W_1, W_2) \quad s.t. \quad W_2^\top W_2 = \mathbb{I}_h, \ W_1^\top W_1 = \mathbb{I}_d.$

Additionally this regularization avoids dimension collapse.

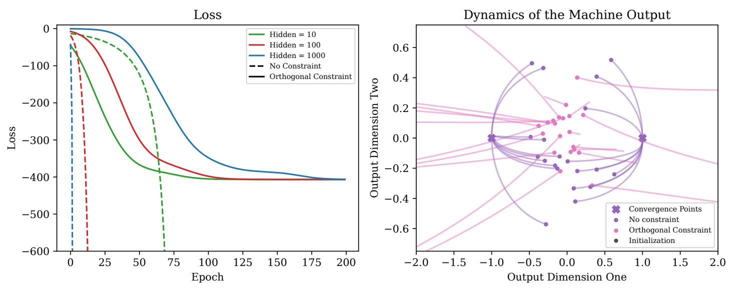


Figure 3.2: Comparison of gradient decent optimization with different regularisers **(left)** comparison of the loss function **(right)** comparison of the evolution of the outputs for the different considered constraints.

Avoidance of dimensional collapse is also heuristically evident in the orthogonality constraint $W_2^\top W_2 = \mathbb{I}_h,\ W_1^\top W_1 = \mathbb{I}_d$, which we focus on in the subsequent sections. In particular we observe from the proof of Proposition 4 that this regularization extracts the eigenvectors of C corresponding to its "most-negative" eigenvalues

Example 1 (**SSL dynamics on half moons**). We numerically illustrate the importance of constraints in SSL. We consider a contrastive setting (loss in (3.1)) for the *half moon* dataset [Ped+11], where x^- is an independent sample from the dataset and $x^+ = x + \varepsilon$ where $\varepsilon \sim \mathcal{N}(0, 0.1\mathbb{I})$. Let us now compare the dynamics of $\mathcal{L}$ (no constraints) and $\mathcal{L}_{orth}$, the scaling loss that corresponds to orthogonality constraints, and present the results in Figure 3.2. We observe that under orthogonal constraints, independent of the initialization the function converges to fixed points (which we theoretically show in Theorem 6). On the other hand the dynamics for unconstrained loss $\mathcal{L}$ diverge.

3.2.1 Non-Linear SSL Models are Almost Linear

While the above discussion pertains to only linear models, we now show that the network, with nonlinear activation ψ and orthogonality constraints,

$$\begin{aligned} &f^{\psi}_{(t)}(x) = W_2^\top \psi(W_1 x) \\ &\text{s.t. } W_2^\top W_2 = \mathbb{I}_h,\ W_1^\top W_1 = \mathbb{I}_d, \end{aligned}$$

is almost linear. For this discussion, we explicitly mention the time dependence as a subscript $f^{\psi}_{(t)}$. We begin by arguing theoretically that in the infinite width limit at initialization there is very little difference between the output of the non-linear machine $f^{\psi}_{(0)}$ and that of its linear counterpart $f^{\mathbb{I}}_{(0)}$.

Theorem 4 (**Comparison of Linear and Non-linear Network**). *Recall that $f_{(t)}$ provides the output of the machine at time t and therefore consider the linear and non-linear* *setting at initialization as*

$$\begin{aligned} f^{\mathbb{I}}_{(0)} &= W_2^\top W_1 x && \text{s.t. } W_2^\top W_2 = \mathbb{I}_h,\ W_1^\top W_1 = \mathbb{I}_d; \\ f^{\psi}_{(0)} &= W_2^\top \psi\left(W_1 x\right) && \text{s.t. } W_2^\top W_2 = \mathbb{I}_h,\ W_1^\top W_1 = \mathbb{I}_d. \end{aligned} \tag{3.6}$$

Let $\psi(\cdot)$ be an activation function, such that $\psi(0) = 0$, $\psi'(0) = 1$, and $|\psi''(\cdot)| \leq c$. [3] *Then at initialization as uniformly random orthogonal matrices*

$$\left\| f^{\psi}_{(0)} - f^{\mathbb{I}}_{(0)} \right\| \ \leq\ Kc \left\|x\right\|^2 d \sqrt{\frac{\log^4 h'}{h'}}$$

where K is an universal constant ψ, d is the feature dimension and h the width of the hidden layer.

We furthermore conjecture that the same behaviour holds during evolution.

Conjecture 1 (**Evolution of Non-linear Networks**)**.** *Consider the setup of Theorem 4 with the linear $\left(f^{\mathbb{I}}_{(t)}\right)$ and non-linear machine $\left(f^{\psi}_{(t)}\right)$ as defined in (3.6) and an optimization of the general*

$$\begin{aligned} &\min_{W_2 W_1} \operatorname{Tr}\left(f^\top_{(t)} f_{(t)}\right) \\ &\text{s.t. } W_2^\top W_2 = \mathbb{I}_h \textit{ and } W_1^\top W_1 = \mathbb{I}_d. \end{aligned}$$

Again assume ψ is an activation function, such that $\psi(0) = 0$ and $\psi'(0) = 1$. Then

$$\left\| f^{\psi}_{(t)} - f^{\mathbb{I}}_{(t)} \right\| \to 0 \quad \forall t > 0 \textit{ as } h' \to \infty.$$

Numerical justification of the above conjecture is presented in the following section.

3.2.2 Numerical Evaluation.

We now illustrate the findings of of Theorem 4 and Conjecture 1 numerically. For evaluation we use the following experimental setup: We train a network with contrastive loss as defined in (3.1) using gradient descent with learning rate 0.01 for 500 epochs and hidden layer size from 10 to 2000. We consider the following three loss functions: (1) sigmoid, (2) ReLU ($\psi(x) = \max\{x, 0\}$) and (3) tanh. The results are shown in Figure 3.3 where the plot shows the average over 10 initializations. We note that *tanh* fulfills the conditions on ψ and we see that with increasing layer size the difference between linear and non-linear goes to zero. While *ReLU* only fulfills $\psi(0) = 0$ the overall picture still is consistent with *tanh* but with slower convergence. Finally the results on *sigmoid* (which has a linear drift consistent with its value at 0) indicate that the conditions on ψ are necessary as we observe the opposite

[3]This last assumption can also be weakened to say that ψ'' is continuous at 0. See the proof of the theorem for details.

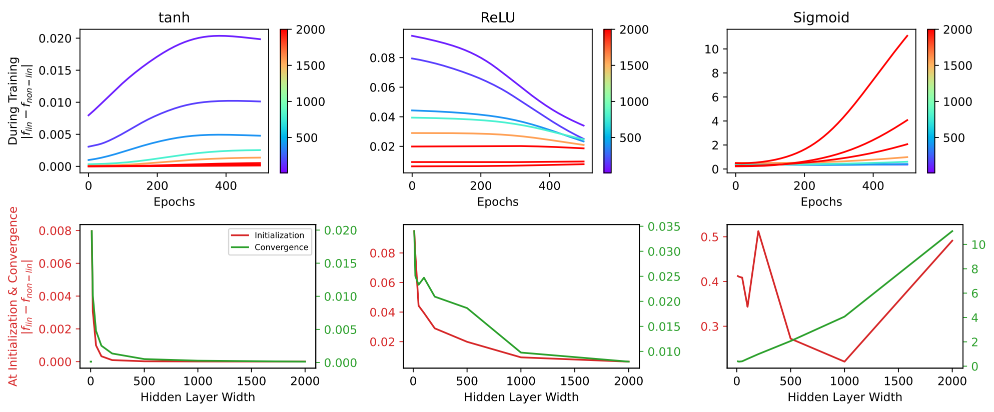


Figure 3.3: Difference between the non-linear output and the linear output under various conditions on the activation function. **Row 1.** Change of the difference while training for hidden layer size 10 to 2000 (indicated by color bar). **Row 2.** Difference at initialization and epoch 500.

picture: with increased layer width the difference between linear and non-linear increases.

3.3 Learning Dynamics of Linear SSL Models

Having showed that the non-linear dynamics are close to the linear ones we now analyze the linear dynamics. We do so by first showing that the two SSL settings discussed in the introduction can be phrased as a more general trace minimization problem. From there we derive the learning dynamics and discuss the evolution of the differential equation. Furthermore we numerically evaluate the theoretical results and show that the dynamics coincide with learning the general loss function under gradient decent.

We can define a simple linear embedding function u as: $f(x) = W_2^\top W_1 x$ where the feature dimension is d for n data points. The hidden layer dimension is h and embedding dimension z, such that the weights are given by $W_2 \in \mathbb{R}^{h' \times h}, W_1 \in \mathbb{R}^{h' \times d}$. Therefore we can write our loss function as

$$\begin{aligned}\mathcal{L} &= \sum_{i=n}^{n} \operatorname{Tr}\left(W_2^\top W_1 x_i \left(x_i^- - x_i^+\right)^\top W_1^\top W_2\right) \\ &= \operatorname{Tr}\left(W_2^\top W_1 \widetilde{C} W_1^\top W_2\right) \\ &= \operatorname{Tr}\left(W_2^\top W_1 C W_1^\top W_2\right)\end{aligned}$$

with

$$C = \frac{\widetilde{C} + \widetilde{C}^\top}{2} \quad \text{and} \quad \widetilde{C} = \sum_i^n x_i \left(x_i^- - x_i^+\right)^\top . \tag{3.7}$$

Furthermore (3.1) can easily be extended to the p positive and q negative sample setting where we then obtain $\widetilde{C} = \sum_i^n \left(\sum_j^q x_i \left(x_j^-\right)^\top - \sum_j^p x_i \left(x_j^+\right)^\top\right)$. In addition we can also frame the previously considered non-contrastive model in (3.2) in the simple linear setting by considering the general loss function with $\widetilde{C} = \sum_i^n x_i \left(x_i^+\right)^\top$. We can now consider the learning dynamics of models, that minimize objects of the form

Definition 3 (General Loss Function)**.** Consider the following loss function

$$\begin{aligned} \mathcal{L}_{W_2 W_1} &:= \operatorname{Tr}\left(W_2^\top W_1 C W_1^\top W_2\right) \\ \text{s.t.} \;\; & W_2^\top W_2 = \mathbb{I}_h \text{ and } W_1^\top W_1 = \mathbb{I}_d. \end{aligned} \tag{3.8}$$

where $W_1 \in \mathbb{R}^{h' \times d}$ and $W_2 \in \mathbb{R}^{h' \times h}$ are the trainable weight matrices. $C \in \mathbb{R}^{d \times d}$ is a symmetric, data dependent matrix.

With the general optimization problem set up we can analyze (3.8) by deriving the dynamics under orthogonality constraints on the weights, which constitutes gradient descent on the Grassmannian manifold. While orthogonality constraints are easy to initialize the main mathematical complexity arises from ensuring that the constraint is preserved over time. Following [LLY20], we do so by ensuring that the gradients lie in the tangent bundle of orthogonal matrices.

3.3.1 Theoretical Analysis

In the following we present the dynamics in Theorem 5, followed by the analysis of the evolution of the dynamics in Theorem 6.

Theorem 5 (**Learning Dynamics in the Linear Setting**)**.** *Let us recall the the general linear trace minimization problem stated in* (3.8)*:*

$$\begin{aligned} \min_{W_2 W_1} \; & \operatorname{Tr}\left(W_2^\top W_1 C W_1^\top W_2\right) \\ \text{s.t.} \;\; & W_2^\top W_2 = \mathbb{I}_h \textit{ and } W_1^\top W_1 = \mathbb{I}_d. \end{aligned}$$

where $W_1 \in \mathbb{R}^{h' \times d}$ and $W_2 \in \mathbb{R}^{h' \times h}$ are the trainable weight matrices and $C \in \mathbb{R}^{d \times d}$ a symmetric, data dependent matrices, such that $C = V \Lambda V^\top$ with $V := [v_1, \dots, v_d]$. Then with $q := \left[f^{\mathbb{I}}(v_1), \cdots, f^{\mathbb{I}}(v_d)\right]^\top$, where f represents the machine function i.e. $f^{\mathbb{I}}(x) = W_2^\top W_1 x$, the learning dynamics of q, the machine outputs are given by

$$\mathring{q} = -2\left[2\Lambda q - \Lambda q q^\top q - q q^\top \Lambda q\right]. \tag{3.9}$$

Similar differential equations to (3.9) have been analysed in [YHM94] and [Fuk98]. The typical way to find stable solutions to such equations involve converting it to a differential equation on $qq^\top$. This gives us a matrix riccati type equation. For brevity's sake we write below a complete solution when $z = 1$.

Evolution of the differential equation. While the above differential equation doesn't seem to have a simple closed form, a few critical observations can still be made about it - particularly about what this differential equation converges to. As observed in Figure 3.4 (right), independent of initialisation we converge to either of two points. In the following we formalise this observation.

Theorem 6 (Evolution of learning dynamics in (3.9) for $h = 1$)**.** *Let $h = 1$ then our update rule simplifies to*

$$\frac{\mathring{q}}{2} = -(1 - q^\top q)\Lambda q - (\mathbb{I} - qq^\top)\Lambda q. \tag{3.10}$$

We can distinguish two cases:

- *Assume all the eigenvalues of Λ are strictly positive then q converges to 0.*
- *Assume there is at least one negative eigenvalue of Λ, then q becomes the smallest eigenvector, e_1.*

The requirement of negative eigenvalues of C for a non-trivial convergence might be surprising however we can observe this when considering C in expectation. Let us assume C is constructed by (3.7) and note that $\mathbb{E}[\widetilde{C}] = \mathbb{E}\big[\sum_i^n x_i \left(x_i^- - x_i^+\right)^\top\big]$. While this already gives a heuristic of what is going on, for some more precise mathematical calculations, we can specialise to the situation where x^- is given by an independent sample and x^+ is given by adding a noise value ϵ sampled from $N(0, \sigma\mathbb{I})$, i.e. $x^+ = x + \epsilon$. Then

$$\mathbb{E}[\widetilde{C}] = \sum_{i=1}^n \mathbb{E}[x_i]\mathbb{E}[x_i^{-\top}] - \mathbb{E}[x_i x_i^{+\top}] = -n\mathbb{E}[xx^\top].$$

Thus $\mathbb{E}[C]$ is in fact negative definite.

New Datapoint. While the above dynamics provide the setting during training we can furthermore investigate what happens if we input a new datapoint or a testpoint to the machine. Because u is a linear function and because $v_1, ..., v_d$ is a basis this is quite trivial. So if $\hat{x}$ is a new point, let $\alpha = (\alpha_1, ..., \alpha_d)^\top$ be the co-ordinates of $\hat{x}$, i.e. $\hat{x} = \sum_i^d \alpha_i v_i$ or $\alpha = V^\top \hat{x}$. Then

$$f_t(\hat{x}) = f_t\left(\sum_i^d \alpha_i v_i\right) = \sum_i^d \alpha_i f_t(v_i) = q_t^\top \alpha = q_t^\top V^\top \hat{x}.$$

3.3.2 Numerical Evaluation

We can now further illustrate the above derived theoretical results empirically.

Leaning dynamics (Theorem 5) and new Datapoint. We can now illustrate that the derived dynamics in (3.9) do indeed behave similar to learning (3.8) using gradient decent

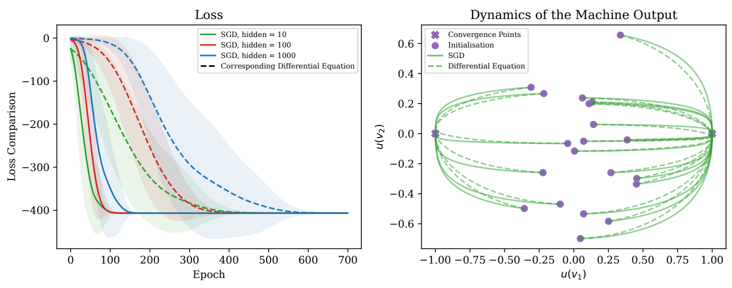


Figure 3.4: Comparison of gradient decent optimization and differential equation. **(left)** comparison of the loss function **(right)** comparison of the outputs.

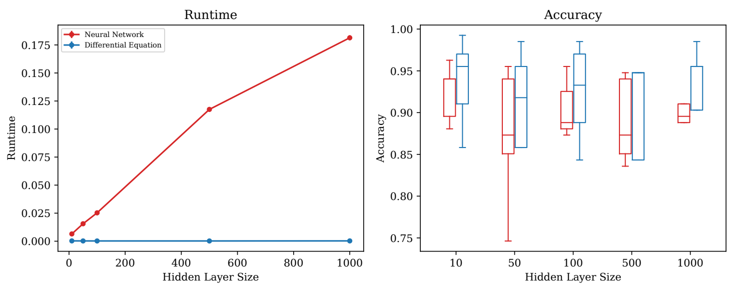


Figure 3.5: **(left)** Run-time comparison between running differential equation and SGD iteration for different hidden layer width. **(right)** Downstream task: Accuracy comparison for SVM on embedding obtained by SGD optimization and running the differential equation.

updates. To analyze the learning dynamics we consider the gradient decent update of (3.8):

$$W_{1,2}^{(t+1)} = W_{1,2}^{(t)} + \eta \nabla \mathcal{L}_{W_2^{(t)}, W_1^{(t)}} \tag{3.11}$$

where $W_1^{(t)}, W_2^{(t)}$ are the weights at time step t and η is the learning rate as a reference. Practically the constraints in (3.8) are enforced by projecting the weights back onto $W_2^\top W_2 = \mathbb{I}_h$ and $W_1^\top W_1 = \mathbb{I}_d$ after each gradient step. Secondly we consider a discretized version of (3.9)

$$q_{t+1} = q_t - 2\eta\big[2\Lambda q_t - \Lambda q_t q_t^\top q_t - q_t q_t^\top \Lambda q_t\big]. \tag{3.12}$$

where q_t is the machine outputs at time step t. We now illustrate the comparison through in Figure 3.4 where we consider different width of the network ($h \in \{10, 100, 1000\}$) and $\eta = 0.01$. We can firstly observe on the left, that the loss function of the trained network and the dynamics and observe while the decay is slightly slower in the dynamics setting both converge to the same final loss value. Secondly we can compare the function outputs during training in Figure 3.4 (right): We initialize the NN randomly and use this initial machine output as q_0. We observe that during the evolution using (3.11) & (3.12) for a given initialization the are stay close to each other and converge to the same final outputs.

Runtime and downstram task. Before going into the illustration of the dynamics we furthermore note that an update step using (3.12) is significantly faster then a SGD step using (3.11). For this illustration we now consider two classes with 200 datapoints each from the MNIST dataset [Den12]. This is illustrated in Figure 3.5 (left) where we compare the runtime over different layer width (of which (3.12) is independent of). Expectantly (3.11) scales linearly with h and overall (3.12) has a shorter runtime per timestep. While throughout the section we focus on the obtained embeddings we can furthermore consider the performance of downstream tasks on top of the embeddings. We illustrate this in the setting above where we apply a linear SVM on top of the embeddings. The results are shown in Figure 3.5 (right) where we observe that overall the performance of the downstream task for both the SGD optimization and the differential equation coincide.

Numerical Evaluation of Theorem 6. We can again illustrate that the behaviour stated in Theorem 6 can indeed be observed empirically. This is shown in Figure 3.4 (right), a setting where C has negative eigenvalues. We observe that eventually the machine outputs converge to the smallest eigenvector.

3.4 Discussion

We conclude with outlining related work on analyzing SSL models and discuss the findings of this section. Finally we outline how the results of this section build the foundation to derive more precise characterization of SSL models.

Related works. Our focus is on the evolution of the learned representations, and hence, considerably different from the main SSL literature focus on generalisation theory and spectral analysis of SSL. From an optimisation perspective, [Liu+23] derive the loss landscape of contrastive SSL with linear models, $f(x) = Wx$, under InfoNCE loss [OLV18]. Although the contrastive loss in (3.1) seems simpler than InfoNCE, they are structurally similar under linear models [Liu+23, see Eqns. 4–6]. Training dynamics for contrastive SSL with deep linear models have been partially investigated by [Tia22], who show an equivalence with principal component analysis, and by [Jin+22], who establish that dimension collapse occurs for over-parametrised linear contrastive models. Theorem 5 provides a more precise characterisation and convergence criterion of the evolution dynamics than previous works. Furthermore, none of prior works consider non-linear models or orthogonality constraints as studied in this work.

We also distinguish our contributions (and discussions on neural tangent kernel connections) with the kernel equivalents of SSL studied in [Sha+22; Cab+23]. While [Sha+22; Cab+23] specifically pose SSL objectives using kernel models, [Kia+22b] show that contrastive SSL objectives induce specific kernels. Importantly, these works neither study the learning dynamics nor consider the neural tangent kernel regime.

Discussion. The study of learning dynamics of (infinite-width) neural networks has led to important results for the supervised setting. However, there is little understanding of SSL dynamics. Our initial steps towards analysing SSL dynamics encounters a hurdle: standard SSL training has drastic dimension collapse (Proposition 2), unless there are suitable constraints. We consider a general formulation of linear SSL under orthogonality constraints (3.8), and derive its learning dynamics (Theorem 5). We also show that the derived dynamics can approximate the SSL dynamics using wide neural networks (Theorem 4) under some conditions on activation ψ. We not only provide a framework for analysis of SSL dynamics, but also shows how the analysis can critically differ from the supervised setting. As we numerically demonstrate, our derived dynamics can be used an efficient computational tool to approximate SSL models. In particular, the equivalence in Proposition 4 ensures that the orthogonality constraints can be equivalently imposed using a scaled loss, which is easy to implement in practice. We conclude with a limitation and open problem.

Chapter 4

Kernel Self-Supervised Learning

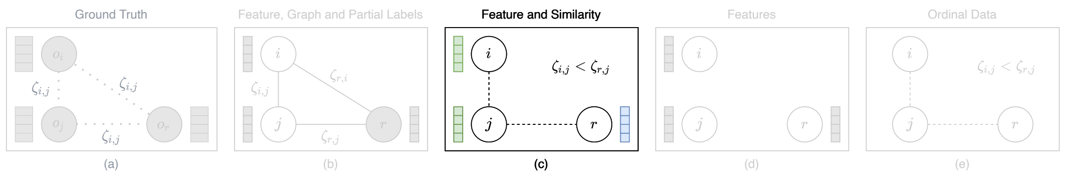


Figure 4.1: Self Supervised Setup. Self Supervised Setup. Features are given as well as semantic similarities between at least some of the objects. Framed in the Self-Supervised Learning setting this becomes a triplet setting such that objects o_j, o_i and o_r are represented by x_i, x_j^+ and x_r^-.

In the previous chapter we relied on a linear approximation of wide networks to derive dynamics, but a more precise characterisation in terms of a kernel approximation [JGH18b; LZB20] may be possible. This could better explain the dynamics of complex deep SSL models. However, integrating orthogonality or operator norm constraints in the NTK regime remains an open question. An important building block for such an exact characterization through kernel approximations of neural networks in the supervised setting such as [JGH18b] is the solution of Kernel methods in the regression setting. In this section we build an equivalent foundation for the contrastive Self-supervised learning setting and in the process show that the derived Kernel methods in and of itself provide a useful tool in the small data setup. But before going into into detail let us shortly recap the supervised setting to better outline differences.

A recap on Kernel methods. Assume we have n data points in $\mathbb{R}^d$, collected in the feature matrix $X = [X_1, \dots, X_n] \in \mathbb{R}^{d\times n}$, with associated label vector $y \in \mathbb{R}^n$. For a pre-specified kernel $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$, let $\mathcal{H}$ be the RKHS and $\phi : \mathbb{R}^d \to \mathcal{H}$ be the (canonical) feature map associated with k. The representer theorem allows us to perform all computations in the sample space ($\mathbb{R}^d$) instead of the Hilbert space.

Theorem 7 (Representer theorem in the supervised setting [KW71; SHS01]). *Let* $g : [0, \infty) \to \mathbb{R}$ *a strictly monotonic increasing function and* $\mathcal{L} : (\mathbb{R}^d \times \mathbb{R} \times \mathbb{R})^n \to \mathbb{R} \cup \{\infty\}$ *an* *arbitrary loss function. Then each minimizer $f \in \mathcal{H}$ of the regularised functional*

$$\mathcal{L}\left(\left(X_1, y_1, f\left(X_1\right)\right), \ldots, \left(X_n, y_n, f\left(X_n\right)\right)\right) + g\left(\|f\|_{\mathcal{H}}^2\right)$$

admits a representation of the form

$$f(x) = \sum_{i=1}^{n} \alpha_i k\left(X_i, X\right).$$

The implication of the above theorem can be easily explained through *kernel ridge regression*, where one learns a model $X^* \mapsto y^*$ of the form $y^* = w^T \phi(X^*)$ by solving the optimisation problem

$$\min_{w} \left\|y - w^T \phi\left(X_i\right)\right\|_F^2 + \lambda \|w\|^2. \tag{4.1}$$

Using Theorem 7 (the kernel trick), one can show that the inference *does not need access to the feature map*. For a new observation X^* it holds that

$$y^* = w_{opt}^T \phi(X^*) = y\left(K + \lambda \mathbb{I}_n\right)^{-1} k(X, X^*). \tag{4.2}$$

The important takeaway here is that we only need to evaluate the kernel k, and not ϕ.

Kernel SSL. Now recall from the previous chapter, we observed that orthogonal constraints are essential in the SSL setting, however, this is not included in the formulation in Theorem 7, therefore the first question that comes up is if Kernel methods can directly be extended to the contrastive loss setting. Secondly the closed form solution in (4.2) is directly derived from (4.1) and therefore an equivalent to the losses from this section is needed to obtain Kernel approximation results for neural networks, extending results from [JGH18b; LZB20] to deep SSL models. In the following section, we therefore address both aspects and investigate the SSL setting through the lens of Kernel methods.

We now follow the main idea outlined in the discussion of the last chapter. Staying in the overall same setting — the one of self-supervised learning (Figure 4.1) — we now shift our focus to modeling contrastive SSL losses through kernel methods. This has the main goal to (i) present new SSL kernel methods that are suitable for small data problems where kernel methods are well established in the supervised setting [Xu+23; TBH23; CT21]; (ii) kernels are non-parametric, and yet considered to be quite interpretable [PM17; HH14]; and (iii) finally show, there is a natural translation from deep SSL to kernel SSL, without compromising performance, building the foundation for future more precise modeling of infinite wide deep SSL models.

Formally to obtain kernel methods we kernelise a single hidden layer, mapping data $x \in \mathbb{R}^d$

to an embedding $z \in \mathbb{R}^h$:

$$x \in \mathbb{R}^d \xrightarrow{\phi(\cdot)} \mathcal{H} \xrightarrow{W} z \in \mathbb{R}^h.$$

Therefore we learn a representation of the form $f_\Theta(x) = W^\top \phi(x)$ by optimizing the objective functions defined in [Aro+19c; Che+20b]. In principle, kernel methods minimize a loss functional $\mathcal{L}$ over the entire, possibly infinite-dimensional RKHS. It is the celebrated representer theorem [KW71; SHS01] that ensures the practical feasibility of this approach: Under mild conditions on the loss $\mathcal{L}$, the optimizer is surely contained within the finite-dimensional subspace $\mathcal{H}_X$. This result is well established in the supervised setting under regularized loss functions. However, in the SSL setup, we consider orthogonal constraints of the form

$$W^* W = \mathbb{I}_h.$$

Therefore before going into deriving kernel methods, this brings up the question: can we extend the idea of representer theorems to the SSL setting?

4.1 Representer Theorems

In this section we show that this is indeed possible but before going into the new result let us recap the example of standard kernel ridge regression where the loss functional $\mathcal{L}$ is simply the regularized empirical squared error

$$\mathcal{L}(w) = \sum_{i=1}^{n} \left(w(x_i) - y_i\right)^2 + \lambda \|w\|^2.$$

The fact that all minimisers of this problem indeed lie in $\mathcal{H}_X$ can be seen by simply decomposing $\mathcal{H} = \mathcal{H}_X \oplus \mathcal{H}_X^\perp$, observing that $w(x_i) = 0$ for all $w \in \mathcal{H}_X^\perp$, and concluding that projecting any w onto $\mathcal{H}$ can only ever decrease the functional $\mathcal{L}$. This very argument can be extended to representation learning, where regularization is important to avoid mode collapse. We formally state the following result.

Theorem 8. *(Representer Theorem for Representation Learning) Given data $x_1, \ldots, x_n$, denote by $\mathcal{L}_X(w_1, \ldots, w_h)$ a loss functional on $\mathcal{H}^h$ that does not change whenever $w_1, \ldots, w_h$ are projected onto the finite-dimensional subspace $\mathcal{H}_X$ spanned by the data. Then, any minimiser of the regularized loss functional*

$$\mathcal{L}(w_1, \ldots, w_h) = \mathcal{L}_X(w_1, \ldots, w_h) + \lambda \|W\|_{\mathcal{H}}^2$$

consists of $w_1, \ldots, w_h \in \mathcal{H}_X$.

This justifies the use of kernel methods when the norm of the embedding map is penalized. However, it does not address loss functionals $\mathcal{L}$ that instead impose an orthonormality constraint on the embedding W. It is natural to ask when a representer theorem exist for these settings as well. Below, we give a necessary and sufficient condition.

Theorem 9 (Representer theorem under orthonormality constraints)**.** *Given data X and an embedding dimension $h \in \mathbb{N}$, let $\mathcal{L} : \mathcal{H}^h \to \mathbb{R}$ be a loss function that vanishes on $\mathcal{H}_X^\perp$. Assume* $\dim(\mathcal{H}_X^\perp) \geq h$. *Consider the following constrained minimisation problem over $w_1, \dots, w_h \in \mathcal{H}$*

$$\begin{aligned} &\textit{minimise } \mathcal{L}(w_1, \dots, w_h) \\ &\textit{s.t.} \quad W^* W = \mathbb{I}_h \end{aligned} \tag{4.3}$$

Furthermore, consider the inequality-constrained problem over $\mathcal{H}_X$

$$\begin{aligned} &\textit{minimise } \mathcal{L}(w_1, \dots, w_h) \\ &\textit{s.t.} \quad W^T W \preceq \mathbb{I}_h \textit{ and } w_1, \dots, w_h \in \mathcal{H}_X \end{aligned} \tag{4.4}$$

Then, every minimiser of (4.3) is contained in $\mathcal{H}_X^h$ if and only if every minimiser of (4.4) satisfies $W^T W = \mathbb{I}_h$.

In practice, the conditions (4.4) can often be verified directly by checking the gradient of $\mathcal{L}$ on $\mathcal{H}_X$, or under orthonormalization (see Appendix). Together with the standard representer theorem, this guarantees that kernel methods can indeed be extended to representation learning — without sacrificing the appealing properties that the representer theorem provides us with.

4.2 Representation Learning with Kernels

Building on this foundation, we can now formalize the previously discussed representation learning paradigms in the kernel setting — namely SSL using contrastive loss functions, as well as unsupervised learning through reconstruction loss.

4.2.1 Simple Contrastive Loss

For convenience, we restrict ourselves to a triplet setting with training samples

$$(x_i, x_i^+, x_i^-), i = 1, \dots, n.$$

The idea is to consider an anchor image x_i, a positive sample x_i^+ generated using data augmentation techniques, as well as an independent negative sample x_i^-. The goal is to align the anchor more with the positive sample than with the independent negative sample. In the following, we consider two loss functions that implement this idea as illustrated in Figure 4.2.

In both cases, we kernelize a single hidden layer, mapping data $x \in \mathbb{R}^d$ to an embedding

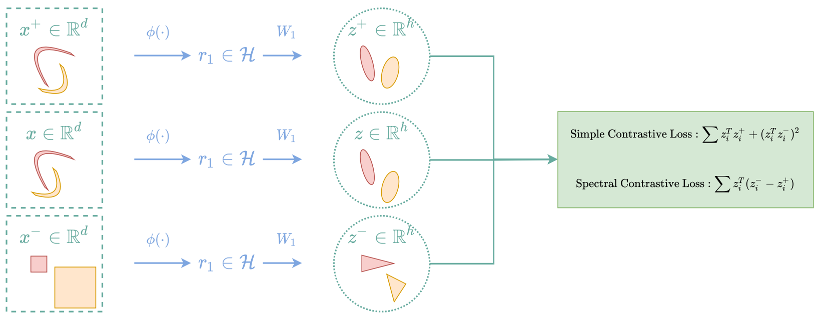


Figure 4.2: Euclidean spaces and corresponding mappings are shown in green, RKHS and corresponding mappings in blue. Considering a triplet setting, the reference (x), positive (x^+) and negative (x^-) are embedded into a latent space. The function is optimised by increasing the distance between embedding z, z^- and decreasing the distance between embedding of z, z^+. We consider two different losses, that follow this basic idea.

$z \in \mathbb{R}^h$.

$$x \in \mathbb{R}^d \xrightarrow{\phi(\cdot)} \boldsymbol{r} \in \mathcal{H} \xrightarrow{W} z \in \mathbb{R}^h. \tag{4.5}$$

We start with a simple contrastive loss inspired by [Aro+19c], with additional regularisation. Intuitively, this loss directly compares the difference in alignment between the anchor and the positive an the anchor and the negative sample. Formally, we define it as follows.

Definition 4 (Contrastive Kernel Learning)**.** We learn a representation of the form $f_W(x) = W^T\phi(x)$ (see mapping in Eq. 4.5) by optimising the objective function

$$\begin{aligned} \mathcal{L}^{\mathrm{Si}} &:= \sum_{i=1}^{n} f_W(x_i)^T \left(f_W(x_i^-) - f_W(x_i^+)\right) \\ &\text{s.t.} \quad W^*W = \mathbb{I}_h \end{aligned} \tag{4.6}$$

By verifying the conditions of Theorem 9, we reduce the problem to a finite-dimensional optimisation. Theorem 10 then provides a closed from solution to the optimisation problem in Eq. 4.6.

Theorem 10 (Closed Form Solution and Inference at Optimal parameterization)**.** *Consider the optimisation problem as stated in Definition 4. Let $X, X^+, X^- \in \mathbb{R}^{d\times n}$ denote the data corresponding to the anchors, positive and negative samples, respectively. Define the kernel matrices*

$$\begin{aligned} K &= [k(x_i, x_j)]_{i,j} & K_- &= \left[k(x_i, x_j^-)\right]_{i,j} \\ K_+ &= \left[k(x_i, x_j^+)\right]_{i,j} & K_{--} &= \left[k(x_i^-, x_j^-)\right]_{i,j} \\ K_{++} &= \left[k(x_i^+, x_j^+)\right]_{i,j} & K_{-+} &= \left[k(x_i^-, x_j^+)\right]_{i,j} \end{aligned}$$

Furthermore, define the matrices

$$K_\Delta = K_{--} + K_{++} - K_{-+} - K_{-+}^T \qquad K_1 = \begin{bmatrix} K & K_- - K_+ \\ K_3 & K_\Delta \end{bmatrix}$$

$$B = \begin{bmatrix} K_- - K_+ \\ K_\Delta \end{bmatrix} \cdot \begin{bmatrix} K & K_- - K_+ \end{bmatrix} \qquad K_2 = -\frac{1}{2}\left(B + B^T\right).$$

Let A_2 consist of the top h eigenvectors of the matrix $K_1^{-1/2} K_2 K_1^{-1/2}$, which we assume to have h non-negative eigenvalues. Let $A = K_1^{-1/2} A_2$. Then, at optimal parameterization, the embedding of any $x^* \in \mathbb{R}^d$ can be written in closed form as

$$z^* = A^T \begin{bmatrix} k(x^*, X) \\ k(x^*, X^-) - k(x^*, X^+) \end{bmatrix}$$

4.2.2 Spectral Contrastive Loss

Let us now consider a kernel contrastive learning based on an alternative, commonly used spectral contrastive loss function [Hao+21].

Definition 5 (Spectral Kernel Learning)**.** We learn a representation of the form $f_W(x) = W^T \phi(x)$ (see mapping in Eq. 4.5) by optimising the following objective function, $\mathcal{L}^{\mathrm{Sp}}$:

$$\mathcal{L} = \sum_{i=1}^{n} -2 f_W(x_i)^T f_W(x_i^+) + \left(f_W(x_i)^T f_W(x_i^-)\right)^2 + \lambda \|W\|_{\mathcal{H}}^2 .$$

While a closed from expression seems out of reach, we can directly rewrite the loss function using the kernel trick and optimise it using simple gradient descent. This allows us to state the following result, which yields an optimisation directly in terms of the embeddings $z_1, \ldots, z_n \in \mathbb{R}^h$.

Theorem 11 (Gradients and Inference at Optimal Parameterization)**.** *Consider the optimisation problem as stated in Definition 5, with K denoting the kernel matrix. Then, we can equivalently minimise the objective w.r.t. the embeddings $Z \in \mathbb{R}^{h \times 3n}$. Denoting by $z_1, \ldots, z_{3n}$ the columns of Z, the loss to be minimised becomes*

$$\min_{Z \in \mathbb{R}^{h \times 3n}} \sum_{i=1}^{n} -2 z_i^T z_{i+n} + \left(z_i^T z_{i+2n}\right)^2 + \lambda \cdot \mathrm{Tr}\left(Z K^{-1} Z^T\right)$$

The gradient of the loss function in terms of Z is therefore given by

$$2\lambda Z K^{-1} + \begin{cases} -2 z_{i+n} + 2(z_i^T z_{i+2n}) z_{i+2n} & , i \in [n] \\ -2 z_{i-n} & , i \in [n+1, 2n] \\ 2(z_i^T z_{i-2n}) z_{i-2n} & , i \in [2n+1, 3n] \end{cases}$$

For any new point $x^* \in \mathbb{R}^d$, the trained model maps it to

$$z^* := ZK^{-1}k(X, x^*).$$

4.3 Generalisation Error Bounds

Kernel methods in the supervised setting are well established and previous works offer rigorous theoretical analysis [Wah90; SS02; BM02a]. In this section, we show that the proposed kernel methods for contrastive SSL as well as for the reconstruction setting can be analysed in a similar fashion, and we provide generalisation error bounds for each of the proposed models.

4.3.1 Error Bound for Representation Learning Setting

In general we are interested in characterizing $\mathcal{L}(f) = \mathbb{E}_{X\sim\mathcal{D}}\left[l\left(f(X)\right)\right]$ where $f(X)$ is the representation function and $l(\cdot)$ is a loss function, which is either a contrastive loss or based on reconstruction. However, since we do not have access to the distribution of the data $\mathcal{D}$, we can only observe the empirical (training) error, $\widehat{\mathcal{L}}(f) = \frac{1}{n}\sum_{i=1}^{n} l\left(f(X_i)\right)$, where n is the number of unlabelled datapoints we can characterise the generalisation error as

$$\mathcal{L}(f) \leq \widehat{\mathcal{L}}(f) + \text{complexity term} + \text{slack term}$$

The exact form of the complexity and slack term depends on the embeddings and the loss. In the following, we precisely characterise them for all of the proposed models.

Theorem 12 (Error Bound for Kernel Contrastive Loss)**.** *Let*

$$\mathcal{F} := \left\{X \mapsto W^T\phi\left(X\right) : \|W^T\|_{\mathcal{H}} \leq \omega\right\}$$

be the class of embedding functions we consider in the contrastive setting. Define $\alpha := \left(\sqrt{h\,\mathrm{Tr}\left[K_X\right]} + \sqrt{h\,\mathrm{Tr}\left[K_{X^-}\right]} + \sqrt{h\,\mathrm{Tr}\left[K_{X^+}\right]}\right)$ *as well as* $\kappa := \max_{x_i' \in \{x_i, x_i^-, x_i^+\}_{i=1}^n} k(x_i', x_i')$. *We then obtain the generalisation error for the proposed losses as follows.*

1. ***Simple Contrastive Loss.*** *Let the loss be given by Definition 4. Then, for any $\delta > 0$, the following statement holds with probability at least $1-\delta$ for any $f \in \mathcal{F}$:*

$$\mathcal{L}^{\mathrm{Si}}(f) \leq \widehat{\mathcal{L}}^{\mathrm{Si}}(f) + O\left(\frac{\omega^2\sqrt{\kappa}\alpha}{n} + \omega^2\kappa\sqrt{\frac{\log\frac{1}{\delta}}{n}}\right)$$

2. ***Spectral Contrastive Loss.*** *Let the loss be given by Definition 5. Then, for any*

$\delta > 0$, the following statement holds with probability at least $1 - \delta$ for any $f \in \mathcal{F}$:

$$\mathcal{L}^{Sp}(f) \leq \widehat{\mathcal{L}}^{Sp}(f) + O\left(\lambda\omega^2 + \frac{\omega^3\kappa^{\frac{3}{2}}\alpha}{n} + \omega^4\kappa^2\sqrt{\frac{\log\frac{1}{\delta}}{n}}\right)$$

The above bounds demonstrate that with increasing number of unlabelled datapoints, the complexity term in the generalisation-error bound decreases. Thus, the proposed models follow the general SSL paradigm of increasing the number of unlabelled data to improve the model performance.

4.3.2 Error Bound for Supervised Downstream Task

While the above bounds provide us with insights on the generalisation of the representation learning setting, in most cases we are also interested in the performance on downstream tasks. Conveniently, we can use the setup presented in [Aro+19c] to bound the error of the supervised downstream tasks in terms of the unsupervised loss, providing a bound of the form

$$\mathcal{L}_{sup}(f) \leq c_1\widehat{\mathcal{L}}_{un}(f) + c_1 * \text{complexity term}$$

where c_1 and c_2 are data dependent constants.

More formally we can write this as follows. We can use the setup presented in [Aro+19c] to bound the supervised error of the downstream tasks by the unsupervised as computed above. Before we state the bound let us formally define the supervised task. We consider a two-class classification task $\mathcal{T}$ with $\{c_1, c_2\}$ distinct classes and a linear classifier on top of the learned representation. Let this function be given by $V \in \mathbb{R}^{2\times h}$. In the following let X_c be a datapoint X belonging to class c.

$$\mathcal{L}_{\text{sup}}(\mathcal{T}, f) = \inf_{V} \underset{(X,c)}{\mathbb{E}}\left[Vf(X_{c_1}) - Vf(X_{c_2}) \mid c_1 \neq c_2\right]$$

From there we can furthermore define the average supervised loss as taking the expectation over the distribution of classes. The average loss for a function f on a binary classification task tasks is defined as

$$\mathcal{L}_{\text{sup}}(f) := \underset{\{c_1,c_2\}\sim\rho^2}{\mathbb{E}}\left[L_{sup}(\{c_1, c_2\}, f) \mid c_1 \neq c_2\right]$$

where the latent class distribution is given by ρ. From there we can now bound the supervised loss by the corresponding unsupervised one.

Corollary 1 (Error Bound on Downstream Tasks)**.** *Let $t = \underset{c,c'\sim\rho^2}{\mathbb{E}} \mathbf{1}\{c = c'\}$ and $\tau := \frac{1}{(1-t)}$ be the probability that two classes sampled independently from ρ are the same. Again define*

$\alpha := \left(\sqrt{h \operatorname{Tr}[K_X]} + \sqrt{h \operatorname{Tr}[K_{X^-}]} + \sqrt{h \operatorname{Tr}[K_{X^+}]}\right)$. *In the following let* $\mathcal{L}_{sup}$ *be the loss of the* supervised downstream task.

1. ***Simple Contrastive Loss.*** *Let* $\widehat{\mathcal{L}}^{Si}$ *be the simple contrastive loss as defined in Definition 4. Then for any* $\delta > 0$*, the following statement holds with probability at least* $1-\delta$ *for any* $f \in \mathcal{F}$*:*

$$\mathcal{L}^{\mathrm{Si}}_{sup} \leq \tau \left(\widehat{\mathcal{L}}^{\mathrm{Si}}_{un} - t\right) + \tau O\left(\frac{\omega^2\sqrt{\kappa}\alpha}{n} + \omega^2\kappa\sqrt{\frac{\log\frac{1}{\delta}}{n}}\right)$$

2. ***Spectral Contrastive Loss.*** *Let* $\widehat{\mathcal{L}}^{\mathrm{Sp}}$ *be the spectral contrastive loss as defined in Definition 5. For any* $\delta > 0$*, the following statement holds with probability at least* $1-\delta$ *for any* $f \in \mathcal{F}$*:*

$$\mathcal{L}^{\mathrm{Sp}}_{sup} \leq \tau \left(\widehat{\mathcal{L}}^{\mathrm{Sp}}_{un} - t\right) + \tau O\left(\frac{\omega^3\kappa^{\frac{3}{2}}\alpha}{n} + \omega^6\kappa^3\sqrt{\frac{\log\frac{1}{\delta}}{n}}\right)$$

This highlights that a better representation (as given by a smaller loss of the unsupervised task) also improves the performance of the supervised downstream task.

4.4 Experimental Evaluation

We empirically demonstrate that the two proposed kernel methods perform on par or outperform classification on the original features as well as Kernel PCA and compare them to neural network representation learning models. We show this, together with another newly introduced kernel method, in more detail in Section 5.2 in the following chapter.

4.5 Discussion

In this chapter, we show that new variants of representer theorem allows one to rephrase SSL optimisation problems or the learned representations in terms of kernel functions. The resulting kernel SSL models provide natural tools for theoretical analysis. *We believe that presented theory and method provide both scope for precise analysis of SSL and can also be extended to other SSL principles, such as other pretext tasks or joint embedding methods* [Aro+19c; BPL22; Gri+20; CH21]. We conclude with some additional discussions.

Related Work. [JHM22] show that minimising certain contrastive losses can be interpreted as learning kernel functions that approximate a fixed positive-pair kernel, and hence, propose an approach of combining deep SSL with Kernel PCA. Closer to our work appears to be [Kia+22a], where the neural network is replaced by a function learned on the RKHS of a kernel. However, their loss functions are quite different from ours. Moreover, by generalising the representer theorem, we can also enforce orthonormality on the embedding maps from the RKHS itself. [Zha+23] studies the role of augmentations in SSL through the lense of the RKHS induced by an augmentation. [Sha+22] present a margin maximisation approach for contrastive learning that can be solved using kernel support vector machines. Their approach is close to our simple contrastive loss method (Definition 4), but not the same as we obtain a kernel eigenvalue problem. While [JHM22; Sha+22] consider specific contrastive losses, we present a wider range of kernel SSL models and provide generalisation error bounds for all proposed models.

Computational limitations and small dataset setting. Exactly computing kernel matrices is not scalable, however random feature (RF) approximations of kernel methods are well suited for large data [RR07; CRR18]. While one may construct scalable kernel representation learning methods using RF, it should be noted that RF models are lazy-trained networks [Gho+19]. So fully-trained deep representation learning models may be more suitable in such scenarios. However representation learning is relevant in all problems with availability of partially labelled data. This does not only apply to the big data regime where deep learning approaches are predominantly used, but also to *small data settings* where kernel methods are traditionally an important tool [FD+14]. *The practical significance of developing kernel approaches is to broaden the scope of the representation learning paradigm beyond the deep learning community.*

Kernel SSL vs. SSL with infinite-width neural networks. While it is known that regression with infinite-width networks is equivalent to kernel regression with neural tangent kernel (NTK) [JGH18b], similar results are not known for SSL. We believe that a study of the learning dynamics of neural network based SSL would show their equivalence with our kernel contrastive models with NTK and therefore the previous two chapters build the foundation to a more precise future characterization of SSL models.

Chapter 5

Representation Learning using Kernel Autoencoders

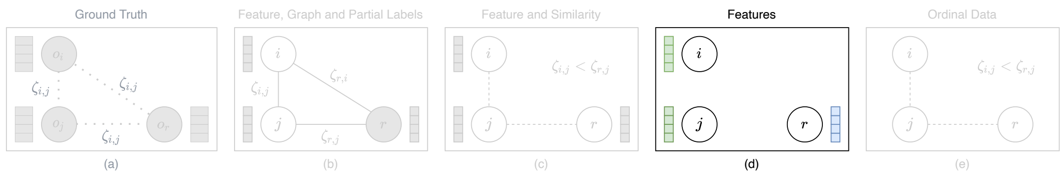


Figure 5.1: Unsupervised Setup. Given are n samples with features of dimension d.

From the SSL setting of the last two chapters we now move on to the classical *unsupervised learning setting* as shown in Figure 5.1. While previously we had access to features and a notion of what makes objects similar to construct positive and negative samples we now only have access to the feature information. But could we not just construct similarities from the features? One might naturally ask where the line can be drawn between the unsupervised and SSL settings.

Unsupervised setting $\longleftrightarrow$ SSL setting[1]**.** Let us illustrate this by showing how one can move from one setting to the other. *Unsupervised setting $\longleftarrow$ SSL setting.* This direction can be directly be obtained by disregarding the additional similarity information. *Unsupervised setting $\longrightarrow$ SSL setting.* One can observe that we can always define a similarity metric on the features to define how similar two samples are. This would indicate that we can move from the unsupervised to the SSL setup, however in practice the similarity considered in the SSL setup is generally defined on a more high-level idea (such as flips or rotations in images) and we therefore consider it as an *additional* similarity information that goes beyond one simply obtained from features. Therefore we consider the unsupervised setup to have less information on the unlabelled data in comparison to the SSL setting.

Let us now move towards outlining the main approach we consider for unsupervised representation learning. We consider representation learning, where the high-level idea is to map the data to a lower dimensional latent space, and then back to the features. The model is

[1]Note that this is not a universal distinction. Both unsupervised and SSL settings are largely driven by empirical advances through a vast number of application areas and therefore both terms are not uniquely defined in the literature. However, outlines how we interpret the fundamental idea of both settings and use them in this thesis.

optimized by minimizing the difference between the input data and the reconstruction. This has been formalized through principal component analysis (PCA) [Pea01] and its nonlinear extension Kernel PCA [SSM98]. While few approaches exist in traditional machine learning, the paradigm of representation through reconstruction has built the foundation of a large number of deep learning methods. Autoencoders (AE) [Kra91] use a neural network for both the embedding into the latent space as well as for the reconstruction. The empirical success of autoencoders has given rise to a large body of work, developed for task-specific regularization (e.g. [Yan+17]), as well as for a wide range of applications such as image denoising [BCM05], clustering [Yan+17] or natural language processing [Zha+22a].

However, the current theoretical understanding of even simple AEs is still limited to formalizing the optimal parameterization of linear AE[2] depending on the considered regularization [BH89; Kun+19; Bao+20; PKK18]. Working towards modeling and understanding non-linear models we consider an approach similar to the one proposed in Chapter 4 by replacing mappings, traditionally implemented through neural networks with Kernel machines. In the AE context this means, both the encoder and decoder correspond now to kernel machines, resulting in the mapping

$$x \in \mathbb{R}^d \xrightarrow{\phi_1(\cdot)} r_1 \in \mathcal{H}_1 \xrightarrow{W_1} z \in \mathbb{R}^h \xrightarrow{\phi_2(\cdot)} r_2 \in \mathcal{H}_2 \xrightarrow{W_2} x \in \mathbb{R}^d$$

where typically $h < d$. Let us now formalize how we can optimize the above mapping.

5.1 Define Kernel Autoencoders

While several materializations of the above high-level idea come to mind, we define the Kernel AE as follows and also illustrate it in Figure 5.2.

Definition 6 (Kernel AE)**.** Given data $X \in \mathbb{R}^{d \times n}$ and a regularization parameter $\lambda > 0$, define the loss functional

$$\mathcal{L}^{AE}(W_1, W_2) := \left\| X - W_2^T \phi_2 \left(W_1^T \phi_1 \left(X \right) \right) \right\|_{\mathcal{H}}^2 + \lambda \left(\|W_1\|_{\mathcal{H}}^2 + \|W_2\|_{\mathcal{H}}^2 \right)$$

The Kernel AE corresponds to the optimisation problem

$$\begin{aligned} \min_{W_1, W_2} \ & \mathcal{L}^{AE}(W_1, W_2) \\ \text{s.t.} \quad & \|W_1^T \phi(x_i)\|^2 = 1 \ \forall \ i \in [n] \end{aligned} \tag{5.1}$$

Let us justify our choice of architecture briefly. Firstly, we include norm regularizations on

[2][BH89] showed that linear AE without regularization finds solutions in the principal component spanning subspace, but the individual components and corresponding eigenvalues cannot be recovered. [Kun+19] show that l_2 regularization reduces the symmetry solutions to the group of orthogonal transformations. Finally [Bao+20] show that non-uniform l_2 regularization allows linear AE to recover ordered, axis-aligned principal components.

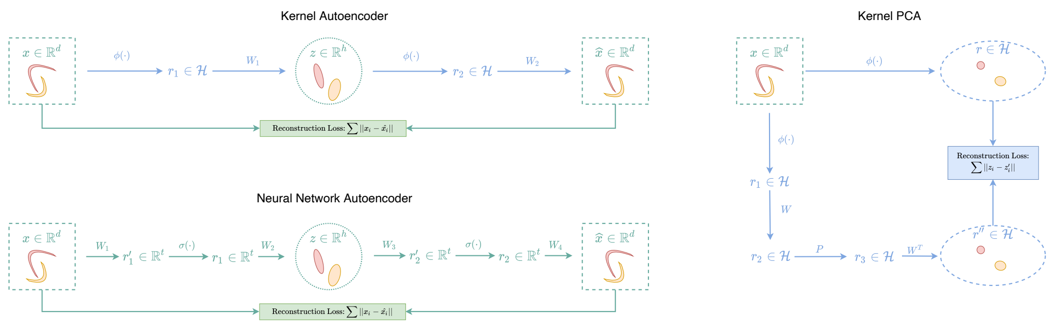


Figure 5.2: Euclidean spaces and corresponding mappings are shown in green, RKHS and corresponding mappings in blue. Overview of reconstruction-based approaches: traditional Kernel PCA, neural network based AE, and Kernel AE (proposed new method). **(left)** AE models: top shows the Kernel versions and bottom the traditional neural network formulation. The loss is computed between the original input x and the reconstruction $\widehat{x}$. **(right)** Kernel PCA: Kernel PCA essentially performs PCA in the RKHS $\mathcal{H}$ [SSM98] and therefore computes distances in the feature space. We present an alternative reconstruction-based formulation — Kernel Autoencoders (KAE) — that evaluates the reconstruction in the sample space instead of computing distances in the feature space. In comparison to Kernel PCA, this approach also provides a simple inference step for unseen data points.

both the encoder as well as the decoder. This is motivated by the following observation: When the feature map ϕ_2 maps to the RKHS of a universal kernel, **any** choice of n distinct points $z_1, \dots, z_n$ in the bottleneck allows for perfect reconstruction. We therefore encourage the Kernel AE to learn smooth maps by penalizing the norm in the RKHS. In addition, we include the constraint $\|W_1^T \phi(x_i)\|^2 = 1 \ \forall \ i \in [n]$ to prevent the Kernel AE from simply pushing the points $z_1, \dots, z_n$ to zero. This happens whenever the impact of rescaling z_i affects the norm of the encoder W_1 differently from the decoder W_2 (as is the case for commonly used kernels such as Gaussian and Laplacian). Nonetheless, we stress that other choices of regularization are also possible, and we explore some of them in the Appendix.

While a closed form solution of Definition 6 is difficult to obtain, we show that the optimisation can be rewritten in terms of kernel matrices.

Theorem 13 (Kernel formulation and inference at optimal parameterization)**.** *For any bottleneck $Z \in \mathbb{R}^{h \times n}$, define the reconstruction*

$$Q(Z) = X(K_Z + \lambda \mathbb{I}_n)^{-1} K_Z$$

Learning the Kernel AE from Definition 6 is then equivalent to minimising the following expression over all possible embeddings $Z \in \mathbb{R}^{h \times n}$:

$$\begin{aligned} &\|Q(Z) - X\|^2 + \lambda \operatorname{Tr}\left(Z K_X^{-1} Z^T + Q K_Z^{-1} Q^T\right) \\ s.t. \quad &\|z_i\|^2 = 1 \ \forall i \in [n] \end{aligned}$$

Given Z, any new $x^* \in \mathbb{R}^d$ is embedded in the bottleneck as

$$z^* = Z K_X^{-1} k(x^*, X)$$

and reconstructed as

$$\hat{x}^* = X \left(K_Z + \lambda \boldsymbol{I}_n\right)^{-1} k(z^*, Z)$$

Remark 4 (Connection to Kernel PCA)**.** In light of the known connections between linear autoencoders and standard PCA, it is natural to wonder how above Kernel AE relates to Kernel PCA [SSM98]. The latter performs PCA in the RKHS $\mathcal{H}$, and is hence equivalent to minimising the reconstruction error over all orthogonal basis transformations W in $\mathcal{H}$

$$\mathcal{L}(W) = \sum_{i=1}^{n} \left\|\phi(x_i) - W^T P_h W \phi(x_i)\right\|^2 \tag{5.2}$$

where P_h denotes the projection onto the first h canonical basis vectors, and we assume that the features $\phi(x_i)$ are centered. How does the Kernel AE $W_2^T \phi_2(W_1^T \phi_1(x))$ relate to this if we replace the regularisation terms on W_1, W_2 by an orthogonality constraint on both? For simplicity, let us assume $h = 1$. The optimisation problem then essentially becomes

$$\mathcal{L} = \sum_{i=1}^{n} \|x_i - W_2(W_1(x_i))\|^2 \tag{5.3}$$

where $W_1 : \mathbb{R}^d \to \mathbb{R}$ is a function from the RKHS over $\mathbb{R}^d$ (with unit norm), and $W_2 : \mathbb{R} \to \mathbb{R}^d$ consists of d orthonormal functions from the RKHS over $\mathbb{R}$. Clearly, Eq. 5.3 evaluates the reconstruction error in the sample space, much in contrast to the loss function in Eq. 5.2 which computes distances in the RKHS. Additionally, the map W^T learned in Eq. 5.2 from the bottleneck back to $\mathcal{H}$ is given by the basis transformation W in Kernel PCA, whereas it is fixed as the feature map ϕ over $\mathbb{R}^h$ in the AE setting. Kernel PCA can be viewed as an AE architecture that maps solely within $\mathcal{H}$, via

$$\phi(x) \to W\phi(x) \to P_h W \phi(x) \to W^T P_h W \phi(x).$$

Notably, the results of Kernel PCA usually do not translate back to the sample space easily. Given a point $x \in \mathbb{R}^d$, the projection of $\phi(x)$ onto the subspace spanned by Kernel PCA is not guaranteed to have a pre-image in $\mathbb{R}^d$, and a direct interpretation of the learned representations can therefore be difficult. In contrast, our method is quite interpretable, as it also provides an explicit formula for the reconstruction $\hat{x}^*$ of unseen data points — not just their projection onto a subspace in an abstract Hilbert space. In particular, by choosing an appropriate kernel[3] and tuning the regularization parameter λ, a practitioner

[3]The choice of kernel could be influenced by the type of functions that are considered interpretable in

may directly control the complexity of both decoder as well as the encoder.

Remark 5 (De-noising Kernel AE)**.** In this section, we considered the standard setting where the model learns the reconstruction of the input data. A common extension is the *de-nosing* setting (e.g. [BCM05; Vin+10]), which formally moves the model from a reconstruction to a SSL setting, where we replace the input with a noisy version of the data. The goal is now to learn a function that removes the noise and, in the process, learns latent representations. More formally, the mapping becomes

$$\overline{x} \in \mathbb{R}^d \xrightarrow{\phi_1(\cdot)} r_1 \in \mathcal{H}_1 \xrightarrow{W_1} z \in \mathbb{R}^h \xrightarrow{\phi_2(\cdot)} r_2 \in \mathcal{H}_2 \xrightarrow{W_2} x \in \mathbb{R}^d.$$

where $\overline{x}$ is given by $\overline{x} := x + \varepsilon$ with ε being the noise term. We again note that the simple extension to this setting further distinguishes our approach from Kernel PCA, where such augmentations are not as easily possible.

Theorem 14 (Error Bound for Kernel AE)**.** *Assume the optimisation be given by Definition 6 and define the class of encoders/decoders as:* $\mathcal{F} := \left\{X \mapsto W_2^T \phi_2 \left(W_1^T \phi_1 (X)\right) \text{ s.t. } \|W_1^T \phi(x_i)\|^2 = 1 \ \forall \ i \in [n] \ : \ \|W_1^T\|_{\mathcal{H}} \leq \omega_1, \|W_2^T\|_{\mathcal{H}} \leq \omega_2\right\}$*. Let* $r := \lambda(\omega_1^2 + \omega_2^2)$ *and* $\gamma = \max_{s \in \mathbb{R}^h \text{s.t.} \|s\|^2=1} \{k(s,s)\}$*, then for any* $\delta > 0$*, the following statement holds with probability at least* $1 - \delta$ *for any* $f \in \mathcal{F}$*:*

$$\mathcal{L}^{AE}(f) \leq \widehat{\mathcal{L}}^{AE}(f) + O\left(r + \frac{\omega_2 \sqrt{d\gamma}}{\sqrt{n}} + \sqrt{\frac{\log \frac{1}{\delta}}{n}}\right)$$

The above bounds demonstrate that with increasing number of unlabelled datapoints, the complexity term in the generalisation-error bound decreases. Thus, the proposed models follow the general SSL paradigm of increasing the number of unlabelled data to improve the model performance. Again we can also show that this improvement in the representation translates to an improvement in the downstream task.

Corollary 2 (Error Bound on Downstream Tasks)**.** *Let* $t = \underset{c,c' \sim \rho^2}{\mathbb{E}} \mathbf{1}\{c = c'\}$ *and* $\tau := \frac{1}{(1-t)}$ *be the probability that two classes sampled independently from* ρ *are the same. Again define* $\alpha := \left(\sqrt{h \operatorname{Tr}[K_X]} + \sqrt{h \operatorname{Tr}[K_{X^-}]} + \sqrt{h \operatorname{Tr}[K_{X^+}]}\right)$*. In the following let* $\mathcal{L}_{sup}$ *be the loss of the* supervised *downstream task. Consider the embedding function from the function class* $\mathcal{F} := \left\{X \mapsto W^T \phi(X) : \|W^T\|_{\mathcal{H}} \leq \omega\right\}$ *and let be* $\widehat{\mathcal{L}}_{un}^{AE+}$ *the loss on the embedding for* X^+ *and* $\widehat{\mathcal{L}}_{un}^{AE-}$ *the loss on the embedding for* X^-*, standing in for two classes*[4]*. Furthermore let* $\operatorname{Tr}[K_{X^+}], \operatorname{Tr}[K_{X^-}] \leq \beta$ *For any* $\delta > 0$*, the following statement holds with probability at least*

the domain of application.

[4]Remark: while it seems surprising that positive and negative samples suddenly appear in the AE setup we note that in the contrastive setting this allows to naturally account for mappings to different classes. Therefore in the AE, introducing this setting allows for class differentiation in the embedding.

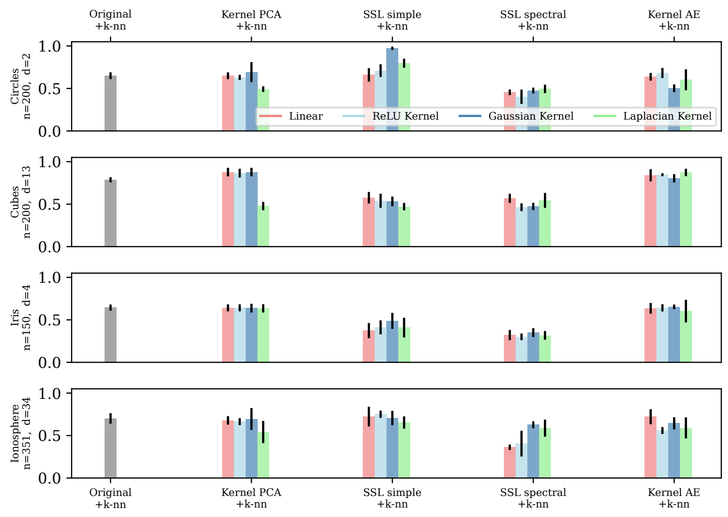


Figure 5.3: From left to right: we first consider k-nn on the original features followed by k-nn on embeddings obtained by Kernel PCA, and the proposed methods.

$1 - \delta$ *for any* $f \in \mathcal{F}$*:*

$$\mathcal{L}_{sup}^{AE} \leq \tau\left(\left|\widehat{\mathcal{L}}_{un}^{AE+} - \widehat{\mathcal{L}}_{un}^{AE-}\right| - t\right) + \tau O\left(\frac{\omega\sqrt{h\beta}}{\sqrt{n}} + \sqrt{\frac{\log\frac{1}{\delta}}{n}}\right)$$

With the formal foundations outlined we now illustrate how the proposed models can be used in practice.

5.2 Experimental Evaluation

In this section we illustrate the empirical performance of the kernel-based representation learning models introduced in this and the previous chapter (Chapter 4 & 5). As discussed in the introduction, there is a wide range of representation learning models, that are often quite specific to the given task. We mainly consider classification in a setting with only partially labelled data at our disposal, as well as image de-noising using the Kernel AE. We state the main setup and results in the following, and provide all further details (as well as experiments on additional datasets) in the supplementary material.

5.2.1 Classification on Embedding

Data. In this section, we consider the following four datasets: *concentric circles*, *cubes* [Ped+11], *Iris* [Fis36a] and *Ionosphere* [Sig+89]. We fix the following data split: *unlabelled* $=$ 50%, *labelled* $= 5\%$ and *test* $= 45\%$, and consider $h = 2$ as the embedding dimension.

Classification task using k nearest neighbours (k-nn) using embedding as features. We investigate classification as an example of a supervised downstream task. The setting is the following: We have access to $X_{unlab.}$ and $X_{lab.}$ datapoints, which we use to train the representation learning model without access to labels. Then, as the downstream classification model, we consider a k-nn model (with $k = 3$) learned on the embedding of $X_{lab.}$, with corresponding labels $Y_{lab.}$. We test on X_{test}, Y_{test}. As a benchmark, we compare to k-nn both on the original features as well as on the embeddings obtained by standard Kernel PCA.

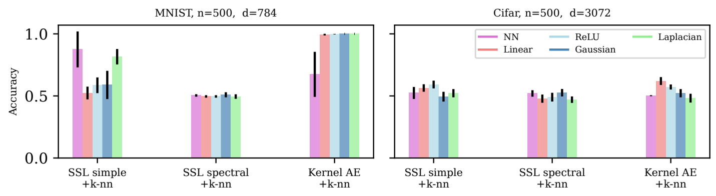
Choice of kernel and their parameterization. For the proposed kernel methods as well as for Kernel PCA we consider three standard kernels, *Gaussian, Laplacian and linear kernels* as well as a 1-layer ReLU Kernel [BB21]. For Gaussian and Laplacian kernel we choose the bandwidth using a grid search over 15 steps spaced logarithmically between 0.01 and 100. We perform leave-one-out validation on $X_{lab.}$ to pick the bandwidth of the method applied to the test set. The classification experiments on the above listed datasets are present in Figure 5.3. All results show the mean and standard deviation over five splits of each dataset. It is apparent throughout the experiments that the choice of kernel plays a significant role in the overall performance of the model. This dependency is not surprising, as the performance of a specific kernel directly links to the underlying data-structure, and the choice of kernel is an essential part of the model design. This is in accordance with existing kernel methods — and an important future direction is to analyze what kernel characteristics are beneficial in a representation learning setting.

Comparison of supervised and representation learning. As stated in the introduction (and supported theoretically in the previous section), the main motivation for representation learning is to take advantage of unlabelled data by learning embeddings that outperform the original features on downstream tasks. To evaluate this empirically for the kernel representation learning models analyzed in this paper, we compare k-nn on the original data to k-nn on the embeddings as shown in Figure 5.3. We observe that for *Circles, Cubes, Iris and Ionosphere* there always exists an embedding that outperforms k-nn on the original data. In addition, we observe in Figure 4 that increasing the number of unlabelled datapoints overall increases the accuracy for the downstream task as shown on the example of Linear and ReLU Kernel for the Circle dataset.

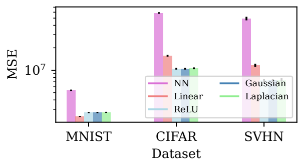
Comparing different embedding methods. Having observed that learning a representation before classification is beneficial, we now focus on the different embedding approaches. While the performed experiments do not reveal clear trends between different methods, we do note that the proposed methods overall perform on par or outperform Kernel PCA, underlining their relevance for kernel SSL.

5.2.2 Comparison to Neural Networks for Classification and De-noising

Representation learning has mainly been established in the context of deep neural networks. In this paper, we make a step towards decoupling the representation learning paradigm from the widely used deep learning models. Nonetheless, we can still compare the proposed kernel



(a) Comparison of kernel methods and neural network models for classification on the MNIST and Cifar dataset.



(b) De-noising using neural network with and Kernel AEs on MNIST, Cifar and SVHN.

Figure 5.4: Comparison to neural networks for classification and de-noising.

methods to neural networks. We construct the *corresponding NN model* by replacing the linear function in the reproducing kernel Hilbert space, $W^\top \phi(x)$ by an one-hidden layer neural network $W_2\sigma(W_1 x)$, where $\sigma(\cdot)$ is a non-linear activation function (and we still minimise a similar loss function).

Classification. We compare the performance of both representation learning approaches in Figure 5.4a for datasets *CIFAR-10* [KH+09], as well as a subset of the first two classes of *MNIST* [Den12] (i.e. $n = 500$). We observe that the kernel methods perform on par with, or even outperform the neural networks. This indicates that there is not one dominant approach but one has to choose depending on the given task.

De-Noising. As a second task, we consider de-noising using (Kernel) AE. Data is sampled from the first five classes of *MNIST*, *CIFAR-10* and *SVHN* [Net+11] with $n = 225$ and the noisy version are generated by $\overline{x} := x + \varepsilon,\ \varepsilon_i \sim \mathcal{N}(0, 0.1)$. We compare the performance of kernel-based approaches with the neural network reconstructions in Figure 5.4b by plotting the mean square error on the test set between the AE output and the clean data. Kernel AE outperforms the neural network AE in all considered settings. Moreover, there is little variation among the different kernels. This indicates that at least in the presented settings, the proposed kernel methods pose a viable alternative to traditional neural network based representation learning.

5.3 Discussion

Most remarks for this section are along the lines of the ones from the last section on SSL as the underlying idea — model an existing neural network approach through kernel methods is the same, however with a change in the objective function and data setting. Therefore again the practical application is in the small data setting as we showed in the previous Section 5.2.

From a theoretical perspective, while it is known that regression with infinite-width networks is equivalent to kernel regression with neural tangent kernel (NTK) [JGH18b; Aro+19b], similar results are not known for SSL and this brings up the question: Is kernel SSL equivalent to SSL with infinitely-wide neural networks? It is possible to show that single-layer Kernel AE with NTK is the infinite-width limit of over-parametrized AE [NWH21; RBU20]. We believe that the same equivalence also holds for kernel contrastive learning (Definition 4) with NTK, but leave this as an open problem. We do not know if Definition 6 with NTK is the limit for bottleneck deep learning AE since, as we noted earlier, there is no unique formulation for Kernel AE.

Furthermore, we note that several non-parametric generalizations of PCA, including functional PCA, kernel PCA, principal curves etc., have been studied over decades and could be compared to Kernel AEs. However, unlike kernel SSL, embedding methods are typically not inductive. As shown previously, the inductive representation learning by Kernel AE and contrastive learning make them suitable for downstream supervised tasks.

Chapter 6

Unsupervised Cluster Specific Representation Learning

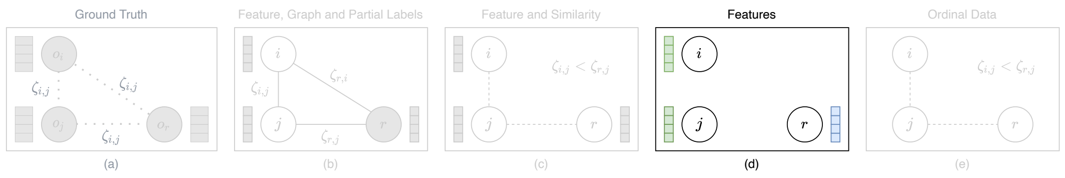


Figure 6.1: Unsupervised Setup. Given are n samples with features of dimension d.

In the last chapter, we considered the problem of learning representations from unlabelled data through optimizing reconstructions from the perspective of kernel methods. We obtained a practically helpful new method that allows for a thorough theoretical analysis due to the Kernel-based approach. This places Kernel AE in the traditional learning regime and the theoretical results we obtain are with regards to generalization. While this provides a step towards a better theoretical foundation of AE models, we now take a step back and again analyze the setting of AEs, however now in the deep learning setting and with regards to the learned representation instead of generalization.

With the increasing use of very high dimensional data, an important task is to find a good lower dimensional representation either to reduce noise in the data or to overcome the curse of dimensionality. However, as outlined in the introduction:

Question 2. *What is a good representation?*

And how can we guarantee that a given algorithm obtains it? Conceptually a latent representation should preserve certain structures of the data while removing noise dimensions. Naturally, there are different perspectives on what could constitute an important structure of the given data, such as an underlying topology or clustering structure. In this work, we consider the latter from a statistical perspective and more specifically from the perspective of its covariance structure and latent representation. Suppose our dataset consists of κ clusters that each have some inherent structure. Then we would want those separate structures to be again represented in the latent space.

The Autoencoder architecture we considered in the last section was a simple one hidden layer

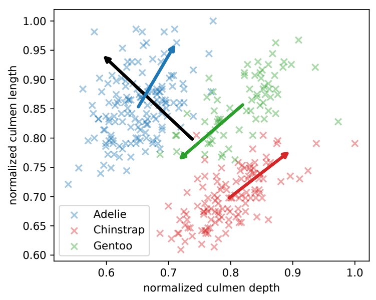


Figure 6.2: Illustration simpson's paradox in the 'penguin dataset' [GWF14]. The three clusters and their first principal component are plotted in red, blue and green respectively and the principal direction for the full dataset in black.

encoding and decoding. However in more general we can define the optimization problem as follows

$$\min_{\Phi,\Psi} \|X - g_\Psi\left(f_\Theta\left(X\right)\right)\| - \lambda * \text{penalty}, \tag{6.1}$$

where $X \in \mathbb{R}^{d\times n}$ is the *centred* data input matrix consisting of n samples of dimension d and $\widehat{X} = g_\Psi\left(f_\Theta\left(X\right)\right)$ the reconstruction. Let $g_\Psi(\cdot)$ be the *decoder function*, parameterized by Ψ mapping from the data dimension d onto the latent dimension h and $f_\Theta(\cdot)$ the *encoder function* parameterized by Θ mapping back to d. λ determines the strength of the penalty, that is chosen depending on the task or desired properties of the parameters.

It has been shown that in the linear setting the encoder recovers the principal components of the full dataset [Kun+19]. However this one representation does not necessarily capture the underlying cluster structure well as illustrated in Figure 6.2. A classical example of this is the so called *simpson's paradox* [Sim51], a known phenomenon in statistics where the trend in clusters does not align with the trend that appears in the full dataset, often observed in social-science and medical-science statistics [Wag82; Hol16]. Figure 6.2 shows that one representation for all the data points (shown in black) does not capture the structures of individual clusters well (shown by the colored arrows). To be able to model such structures we therefore introduce a modified AE architecture we term *Tensorized Autoencoders (TAE)* that, in the linear setting, provably recovers the principal directions of *each cluster* while jointly learning the cluster assignment. This new AE architecture considers a single AE for each cluster allowing us to learn distinct cluster representations and is illustrated in Figure 6.3. Important to note that while this increases the number of parameters, the representation still remains the same dimension as before. In particular this still experimentally performs better than a single autoencoder with $d \times \kappa$ encoding dimension. Formally we change (6.1) to the following

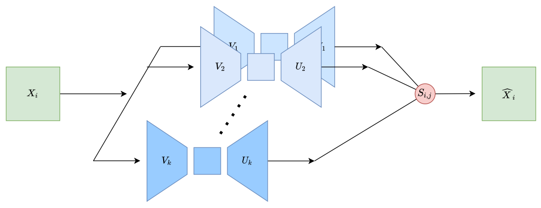


Figure 6.3: Illustration of the tensorized AE. An input X_i, is passed through κ AEs, with k-means penalty latent space that reconstruct X_i. $T_{i,j}$ weight the outputs of the AEs by class assignment.

$$\min_{\{\Phi_j,\Psi_j\}_{j=1}^{\kappa},T} \sum_{i=1}^{n}\sum_{j=1}^{\kappa} T_{j,i}\left[\left\|(X_i - C_j) - g_{\Psi_j}\left(f_{\Theta_j}\left(X_i - C_j\right)\right)\right\|^2 - \lambda * \text{penalty}_j\right], \tag{6.2}$$

where T is a $\kappa \times n$ matrix, such that $T_{j,i}$ is the probability that data point i belongs to class j, $g_{\Psi_j}(\cdot)$ and $f_{\Theta_j}(\cdot)$ are the *decoder* and the *encoder* functions respectively specific to points in class j, and C_j is a parameter centering the data passed to each autoencoder specific to class j. Specifically a k-means [Mac67] penalty is considered to enforce a cluster friendly structure in the latent space (similar to the one proposed in [Yan+17]).

To further illustrate the importance of the new formulation (6.2), that provides cluster specific representations, we look at two important representations learning downstream tasks.

Clustering. The main goal of clustering is to group similar objects into the same class in an unsupervised setting. While this problem has been extensively studied in traditional machine learning the time complexity significantly increases with high dimensional data and therefore existing works focus on projecting data into low-dimensional spaces and then cluster the embedded representations [RL03; Tia+14; Wan+16]. Several methods have been developed that use deep unsupervised models to learn representations with a clustering focus that simultaneously learns feature representations and cluster assignments using deep neural networks [XGF16; Diz+17; Wan+16; XX15; Wan+15]. However all of these algorithms learn a single representation for the full dataset. Assuming we use an AE for learning the representation we formally consider (6.1) and perform the clustering on $f_\Theta(X)$ instead of X. Now the question is, is one representation of the data sufficient? We investigate this question by considering different planted datasets as shown in Figure 6.4, where we compare the clustering obtained from k-means ++ [AV07] on the original features, the embedding obtained by a standard AE and the proposed TAE. In addition we also consider a simple variational autoencoder (VAE) [KW13] and Deep Clustering (DC) [Yan+17] and observe that in this setting they perform very similar to standard AE. Therefore for the later experiments we only focus on the comparison to k-means ++ and standard AE. The advantage of TAE is that it captures the *directions* of the clusters whereas simple AE misclassifies some points in clusters that are close in euclidean space. For those datasets the distance function is inherently linked to the shape of the clusters. We extend this analysis

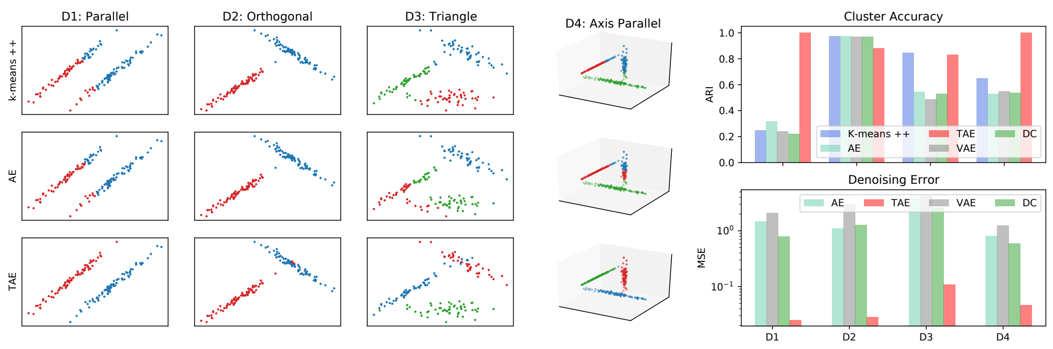


Figure 6.4: Illustration for cases where separate embedding for clusters are beneficial. **Left grid.** Plotted is the true data in $\mathbb{R}^2$. Top row: Clustering obtained from k-means ++ on the original features. Middle row: k-means ++ on the embedding obtained by a standard AE. Bottom row: k-means ++ on the proposed TAE. **Right plots.** Top: alignment of clustering with true labels measured by ARI [Ran71]. Bottom MSE for denosing on the same datasets.

to real world data and more complex AE architectures in Section 6.2.

De-noising. Consider an image corrupted by noise, the task is to remove the noise from the image. In this setting a good representation is one that only returns the true data structure and removes the noise. Here we are not directly interested in the embedding but only in the obtained reconstruction of the decoder. However we again conjecture that having separate embeddings for each cluster is beneficial as it allows to learn more cluster specific representations.

We consider a De-Noising AE (DAE), formally defined [BCM05; Cho13] by considering $X+\varepsilon$ as input in Eq 6.1 where ε is an additive noise term and the goal is to remove the noise from the data. An additional advantage in the denosing setting is that the cluster structure obtained by the TAE does not have to match the one imposed by the ground truth. This is beneficial in cases where there is not a unique way to cluster given data (e.g. when clustering cars, one might group by color or by type). We see this advantage empirically illustrated in Figure 6.4, right, where we see that the MSE of the simple AE is consistently significantly higher than for the TAE. We further validate this intuition empirically in Section 6.2.

Remark 6 (On the line between unsupervised and Self-Supervised Learning)**.** We note at this point that the above setup blurs the line between unsupervised and self-supervised learning. While clustering is purely in the unsupervised setup as we only consider the a task performed on X (or $f_\Theta(X)$), the de-noising setup is more in the self-supervised setup, as by our previous definition SSL is defined by data (X) and an augmented version ($X+\varepsilon$ in this case), however AEs in general allow to model both setups. Given the motivation of this section we mainly consider them from the unsupervised perspective through preserving cluster structures, but empirically analyze both setups in the following.

6.1 Analysis of Tensorized Linear AE with K-Means Penalty

To get a better understanding of the behaviour of the proposed architecture in (6.2), we consider a simple *linear-AE* as this allows us to analytically derive the optimal parameterization. We extend the analysis empirically to a non-linear setting in section 6.2.

6.1.1 Formal Setup

For simplicity consider the clustering setting where $\widetilde{X} := X \in \mathbb{R}^{d\times n}$, with n being the number of data-points and d the feature dimension. Note that the analysis extends directly to the de-noising setting as well. For a two layer linear AE, let $\mathcal{U} \in \mathbb{R}^{\kappa\times h\times d}$ and $\mathcal{V} \in \mathbb{R}^{\kappa\times d\times h}$ be the encoding and decoding tensor respectively. Then for each $1 \le j \le \kappa$, let $W_1^{(j)} \in \mathbb{R}^{h\times d}$, be the *encoding* matrix by taking the appropriate slice of the tensor $\mathcal{U}$. Essentially $W_2^{(j)}$ corresponds to the encoding function of the j'th cluster. We assume $W_1^{(j)}$ is a projection matrix i.e. $W_1^{(j)} W_1^{(j)\top} = \mathbb{I}_h$. Similarly define $W_2^{(j)} \in \mathbb{R}^{d\times h}$ from the *decoder* W_2 (injection matrix).Finally let C_j be the cluster centers and λ be the weight assigned to the penalty. We treat it as a hyperparameter. From there we define the loss function as

$$\mathcal{L}_\lambda(X) := \sum_{i=1}^{n}\sum_{j=1}^{\kappa} T_{j,i}\Big[\Big\|(X_i - C_j) - W_1^{(j)} W_2^{(j)}\,(X_i - C_j)\Big\|^2 \;-\; \lambda\Big\|W_2^{(j)}\,(X_i - C_j)\Big\|^2\Big],$$
$$\text{s.t.}\quad \mathbf{1}_\kappa^\top T = \mathbf{1}_n^\top,\; T_{j,i} \ge 0, \tag{6.3}$$

where we define T to be a $\kappa \times n$ matrix, such that $T_{j,i}$ is the probability that datapoint i belongs to class j. To ensure that the entries can be interpreted as probabilities we impose the above stated constraints. To further illuminate the intuition behind T consider a dataset $\{X_i\}_{i=1}^n$ where associated to each datapoint X_i there are probabilities $\{T_{j,i}\}_{j=1}^\kappa$ corresponding to how certain we are that X_i is sampled from the true distribution. These $\{T_{j,i}\}$ are latent variables that we learn from the dataset.

6.1.2 Parameterization at Optimum

For the linear setting we derive the parameterization at the optimum but for reference we first recall the optimum of a standard linear AE:

Theorem 15 (Parameterization at Optimal for Linear AE)**.** *[BH89] showed that linear AE without regurlarization finds solutions in the principal component spanning subspace, but the individual components and corresponding eigenvalues cannot be recovered. [Kun+19] show that l_2 regularization reduces the symmetry solutions to the group of orthogonal transformations. Finally [Bao+20] show that non-uniform l_2 regularization allows linear AE to recover ordered, axis-aligned principal components.*

From this we note that we only learn a single representation for the data and therefore cannot capture the underlying cluster structures. To give the intuition of this cost, before we characterize the optimal parameters of a linear TAE, lets consider a single datapoint X_i assigned to cluster C_j and its cost is,

$$T_{j,i}\left[\left\|(X_i - C_j) - W_1^{(j)}W_2^{(j)}\left(X_i - C_j\right)\right\|^2 - \lambda\left\|W_2^{(j)}\left(X_i - C_j\right)\right\|^2\right].$$

Ignoring C_j, the cost is simply the cost of X_i in AE weighted by the probability of it belonging to cluster j (i.e. $T_{j,i}$). With this intuition, we characterize the optimal parameters of a linear TAE in the following theorem.

Theorem 16 (Parameterization at Optimal for TAE)**.** *For $0 < \lambda \leq 1$, optimizing Eq. 6.3 results in the parameters at the optimum satisfying the following:*

i) ***Class Assignment.*** *While in Eq. 6.3 we define $T_{j,i}$ as the probability that X_i belongs to class j at the optimal $T_{j,i} = 1$ or 0 and therefore converges to a strict class assignment.*

ii) ***Centers.*** *C_j at optimum naturally satisfies the condition*

$$C_j = \frac{\sum_{i=1} T_{j,i}X_i}{\sum_{i=1} T_{j,i}}.$$

iii) ***Encoding / Decoding (learned weights).*** *We first show that $W_1^{(j)\top} = W_2^{(j)}$, and define*

$$\hat{\mathbf{\Sigma}}_j := \sum_{i=1}^{n} T_{j,i}\left(X_i - C_j\right)\left(X_i - C_j\right)^\top,$$

then the encoding corresponds to the top h eigenvectors of $\hat{\Sigma}_j$.

At the above values for the parameters, C_j and $\hat{\mathbf{\Sigma}}_j$ acts as estimates for the means and covariances for each specific class respectively. Thus assuming that T gives reasonable cluster assignments, $W_2^{(j)}$ and $W_1^{(j)}$ combined gives the principal components of each cluster. While points ii) and iii) follow directly by deriving the parameterization at the optimal and we give the full derivation in the supplementary material, we give a short intuition on i) here. First we note that as per our current definition $\mathcal{L}$ is linear in T. Thus the global optimum of the loss with respect to the aforementioned linear conditions on T must be at some vertex of the convex polytope defined by the conditions. Since these conditions are $\mathbf{1}_\kappa^\top T = \mathbf{1}_n^\top$ and $T_{j,i} \geq 0$, at any of the vertices of the corresponding polytope we have that $T_{j,i} = 1$ *or* 0. This combined with the fact that at global optimum C_j satisfies the condition $C_j = \frac{\sum_{i=1} T_{j,i}X_i}{\sum_{i=1} T_{j,i}}$ implies that the global optimum of the loss $\mathcal{L}$ in this expanded space is precisely same as that of the cost in the strict case we discuss in remark 7. This is important as it shows that even though we allow the optimization using gradient decent the parameterization at the optimal assigns each datapoint to one AE.

Let us compare Theorem 16 to the general notion proposed in the introduction: We would like an approach to obtain a separate meaningful representation (in the sense of recovering principal directions) for each cluster structure without having prior knowledge of which cluster a given datapoint belongs to. Theorem 16 shows that the proposed tensorized AE (Eq. 6.3) fulfills those requirements as TAE recover the top κ eigenvectors for each cluster separately.

Remark 7 (Strict cost function)**.** Since we showed in Theorem 16 that Eq. 6.3 converges to a strict class assignment we can alternatively also define the loss function directly with strict class assignments as follows: Let $\overline{X}^{T(i)}$ be the center of all data-points which X_i belongs to. We define the loss function with strict class assignments as

$$\min_{S} \min_{W_{2,T(i)},W_{1,T(i)}} \min_{C_{T(i)}} \sum_{i=1}^{n} \left\| \left(X_i - \overline{X}^{T(i)}\right) - W_{2,T(i)} W_{1,T(i)} \left(X_i - \overline{X}^{T(i)}\right) \right\|^2 \\ - \lambda \left\| W_{1,T(i)} X_i - C_{T(i)} \right\|^2$$

Similar to Theorem 16 we again characterize the parameters at the optimal and most importantly note that with

$$\hat{\Sigma}_j = \frac{\sum_{i:T(i)=j} \left(X_i - \overline{X}^j\right)\left(X_i - \overline{X}^j\right)^\top}{|\{i : T(i) = j\}|},$$

as long as $\lambda < 1$, the optimal projection $W_1^{(j)}$ for the above cost is exactly the top h eigenvectors of $\hat{\Sigma}_j$.

6.1.3 Optimization

While in the previous section we discussed the optimum that is obtained by solving the optimization problem in Eq. 6.3 we now look at how to practically train the tensorized AE. The general steps for learning the encoder and decoder are summarized as follows, where step 2 and 3 are repeated until convergence.

1. **Initialize** weights and cluster assignments according to k-means ++ [AV07][1].
2. **Update the weights** for the encoder and decoder (using e.g. a GD step).
3. **Update the class assignment** T. To do so we consider a number of different options:

   Option 1: using a GD step under constraints $\mathbf{1}_\kappa^\top S = \mathbf{1}_n^\top$, $T_{j,i} \geq 0$.

   Option 2: using an un-constrained GD step and project T back onto the constraints.

   Option 3: using a Lloyd's step[2] on a strict class-assignment.

The choice of options in step 3 mostly depends on the framework of implementation. As noted, T as defined in Eq. 6.3 allows us to perform gradient updates on the class assignments.

[1]Note on the initialization: while in this algorithm we consider a k-means++ in cases where we have access to some labels (e.g. in a semi-supervised setting) this can be replaced by considering random points with given labels for each cluster as initializations.

[2]Here the Lyod's step solves the linear problem on T assuming all other parameters are fixed.

The advantage is that since T is not defined binary, frameworks such as CVXPY [DB16] or Keras [Cho+15] can be used to perform a constrained gradient steps, however in popular frameworks such as PyTorch [Pas+19], that as of the time of writing this work do not directly support constraint optimization *Option 3* can be used.

While the general goal of the proposed setup is to learn good data representations, the exact train and test steps depend on the downstream task. In this work there are two main settings. For instance in *clustering* we directly apply the above steps and jointly learn the clustering and embedding. While in a simple AE we would have to apply a clustering algorithm onto the latent representations, in TAE the above steps directly provide the clusters. While for the linear case we prove that T_{ij} is binary, in practice, especially with more complex networks one has to compute $\operatorname{argmax}_j T_{ij}$ to determine the class assignment. On the other hand in *de-nosing* we again jointly learn the embedding and cluster assignment but only train on the train set and in a second step use the learned TAE to de-noise images in the test set. While for a simple AE we can directly pass test data, with a TAE we use the approach presented in section 6.1.4 that allows us to assign the new datapoint to the appropriate AE to process it. This same setup could be used for tasks such as super resolution or inpainting. Importantly, while the results from Theorem 16 hold only for the exact linear formulation in Eq. 6.3, the main idea and training steps can be extended to arbitrary encoding and decoding functions which we illustrate empirically in section 6.2.

6.1.4 Test on New Data

In the context of De-noising and validating clustering on an independent test dataset or other downstream tasks, we generally need a way of encoding and decoding a new or test datapoint. In contrast to AE, where one simply passes any new data through the trained network, TAE additionally decides on the latent variable T. Note then that essentially in our model the actual parameters are $C_j, W_2^{(j)}, W_1^{(j)}$ whereas T is simply an encoding of our confidence of what the latent variable or label is for each X_i. Thus to run the TAE on a new datapoint, we have to first estimate this latent variable.

Following this idea, let $t \in \mathbb{R}^\kappa$, with its j'th coordinate being t_j.[3] Then given a new datapoint $\widetilde{X}_i$ and given the trained parameters $C_j, W_2^{(j)}, W_1^{(j)}$, we first find $\hat{t}$ such that

$$\hat{t} = \operatorname*{argmin}_{\mathbf{1}_\kappa^\top t=1; t_j \geq 0} \sum_{j=1}^{\kappa} t_j \left[\left\| (\widetilde{X}_i - C_j) - W_1^{(j)} W_2^{(j)} (\widetilde{X}_i - C_j) \right\|^2 - \lambda \left\| W_2^{(j)} (\widetilde{X}_i - C_j) \right\|^2 \right].$$

We use this setting for example for a de-noising task such that the de-noised reconstruction of $\widetilde{X}_i$ would be given by $(\mathbb{I} - W_2^{(\hat{j})} W_1^{(\hat{j})})(\widetilde{X}_i - C_{\hat{j}}) + C_{\hat{j}}$.

We consider the following two cases to present a heuristic that the step in general does

[3]Note then that this is a linear problem in the variables t_j and thus its solution is at some vertex of the bounding polytope, i.e. there is some $\hat{j}$ such that $\hat{t}_{\hat{j}} = 1$. Thus the label assigned to $\widetilde{X}_i$ is this j.

not worsen the prediction. On the one hand if *clusters are well separated*, this implies that the separate AEs are quite different and therefore a wrong assignment would be significant. However well separated clusters also implies that the assignment of a new point is with high probability correct. On the other hand if the *clusters are not well separated* then the point might be assigned to the wrong cluster more easily, however similar clusters also implies similar AEs so even if we assign a new point to the wrong AE the reconstruction is still close to the one from the correct AE.

6.2 Experiments with Non-Linear and Convolutional Networks

The above linear setting allows us to perform a thorough analysis of the model and prove that the parameterization at the optimal fulfills the desired criteria of learning a meaningful representation for each cluster instead of one for the whole dataset. However in an applied setting we would like to take advantage of the expressive power of more complex, non-linear encoders and decoders as well. Therefore we first discuss how tensorized AE can be extended to a more general setting and additionally empirically validate its performance.

All further details on the implementation[4] and experiments are provided in the supplementary material.

6.2.1 Extension to Arbitrary AE Architectures

Given a single datapoint x and some specific architecture, let $\mathcal{L}_{\{\Theta_j,\Psi_j\}_{j=1}^{\kappa}}(x)$ and $f_{\Theta_j}(\cdot)$ be the corresponding loss function and encoder parameterized by Θ_j respectively. Then the corresponding tensorized autoencoder is generated by first considering κ many independent copies of the above AE $\{f_{\Theta_j}(\cdot)\}_{j=1}^{\kappa}$. Then the tensorized loss is defined by

$$\mathcal{L}(X) := \sum_{i=1}^{n}\sum_{j=1}^{\kappa} T_{j,i}\left[\mathcal{L}_{\Theta_j,\Psi_j}\left(X_i - C_j\right) - \lambda \left\| f_{\Theta_j}\left(X_i - C_j\right)\right\|^2\right],$$

where C_j is defined as $C_j = \frac{\sum_{i=1} T_{j,i} X_i}{\sum_{i=1} T_{j,i}}$. While a more involved analysis is required to prove the exact latent representation in this case, the overall idea presented in (6.2). Again for the training we follow the steps presented in Section 6.1.3.

6.2.2 Clustering on Real Data

While on the toy data in Figure 6.4 we clearly observed how the analyzed approaches deal with different datastructures, we extend the analysis to real data and also more complex AE architectures.

For comparability of the number of parameters we furthermore compare *AE 1* with latent dimension d and *AE 2* with latent dimensions $d \times k$ in Figure 6.6. We note that their

[4]The code is available at: https://github.com/mahalakshmi-sabanayagam/tensorized_autoencoder

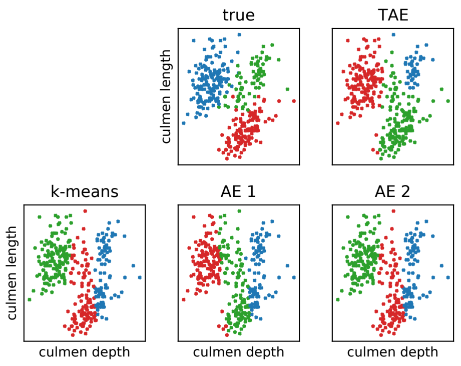


Figure 6.5: Illustration of the final clustering obtained for the *penguin dataset, two features* by the different algorithms.

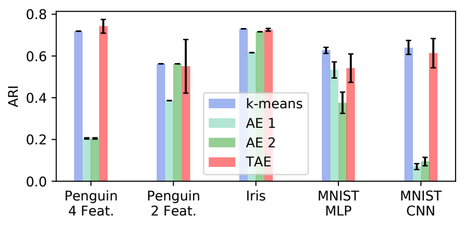


Figure 6.6: Comparison of k-means ++, standard AE with k-means ++ and TAE. AE 1 has latent dimension d while AE 2 has latent dimension $d \times k$. MNIST MLP shows the performance for a single hidden layer network, while MNIST CNN considers an encoder and decoder with two convolutional layers each. Again the latent dimension is d for both networks.

performance on the clustering task is very comparable and we therefore conclude that the performance of the TAE is not a direct result of the increased number of parameters but is achieved due to the different architecture.

To start the analysis we consider the penguin dataset presented in the introduction. Figure 6.5 (top left) shows the simpson's paradox for the penguin dataset and the other plots show the clusterings the different algorithms converge to. Notably we see that for k-means as well as for both AE architectures, the y-axis parallel decision boundaries are obtained where as for the TAE we obtain clusters that are closer to the true structures.

To further quantify the empirical performance of TAE we show the average and standard deviation over five runs for the 'penguin dataset' with two and four features as well as the 'iris dataset' [Fis36b; And36] and 'MNIST' [LC10b] (sub-sample of 50 datapoints for class $\{1, 2, 3, 4, 5\}$ each) for simple linear networks as well as two layer convolutional networks. We show this in Figure 6.6. For penguin dataset and iris we consider $d = 1$ and $d = 10$ for MNIST. We observe that the difference in the performance between AE and TAE is significant for the penguin dataset which we attribute to the above outlined clustering structures. For MNIST, we observe very similar performance of AE and TAE, which is most likely due to the fact that the underlying clustering structures have very similar properties, such there is no significant difference for the TAE to exploit.

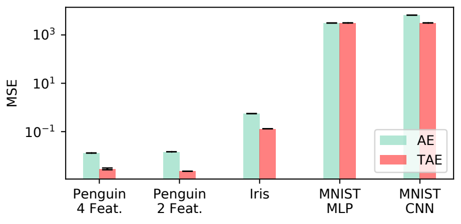


Figure 6.7: Real data de-noising for different Datasets and networks. Note that the MSE is plotted in log scale.

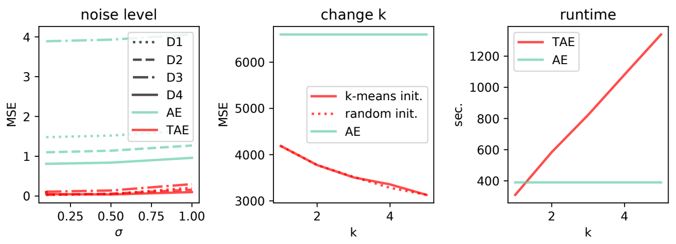


Figure 6.8: **(left)** De-nosing under change in noise level for the four toy datasets. **(middle)** De-nosing for five classes MNIST using AE and TAE with non-optimal κ. **(right)** De-nosing for five classes MNIST. Analysis of run-time with changing number of κ in TAE.

6.2.3 De-Noising on Real Data

Similar to the experiments on clustering real data we look at de-noising on various real datasets with standard AE and TAE. We use the same datasets we used in the clustering case to illustrate that TAE learns a reasonable clustering as well as a better reconstruction. The original data was corrupted by adding an isometric Gaussian to each data point. We illustrate this difference in Figure 6.7 and observe that TAE consistently performs the same or significantly better then the standard AE. For the penguin dataset we observe a better cluster recovery and a better de-noising performance. Interestingly for the Iris dataset the reconstruction obtained by TAE is significantly better the the one by the standard AE while their clustering performance is about the same. Finally for MNIST with CNN we get a markedly improved performance on using TAE, while using a single layer linear neural network as the underlying architecture doesn't gives us any significant improvement in performance. This suggests that it is important to choose a function class or underlying architecture that is capable of capturing a low dimensional representation of the data.

Let us consider the presented de-nosing setting for some further investigation of the model.

Different levels of noise. When considering de-noising an obvious question on the influence of the noise level on the performance of AE and TAE. To analyze this we go back to the toy datasets introduced previously and observe in Figure 6.8 (left) that while with increasing noise level the the MSE increases for both approaches, TAE consistently outperforms the standard AE on all datasets.

Performance of TAE when κ is not the true number of clusters. Let us consider the MNIST setting stated before with five classes. We observe in Figure 6.8 (middle) that the MSE is monotonously improved with κ getting closer to κ_{true}, all improving on standard AE. This indicates that even if the number of clusters is not known[5] TAE for de-noising tasks performs better then standard AE.

Importance of initialization of T. While we propose k-means++ as an initialization procedure for the cluster assignment, empirically the question is how important this assumption is. Figure 6.8 (middle) illustrates that in the previously considered MNIST setting there is no noticeable difference between the the k-means++ initialization and a random initialization. However we conjecture that the difference increases for more complex the dataset.

Computational demands. Finally an important point is the computational comparison between AE and TAE. By construction one can see that the computational complexity scales linearly with the number of considered clusters. We can observe that this also holds empirically as shown in Figure 6.8 (right), again for the MNIST setting considered in the previous two points.

6.3 Connection to Expectation Maximization

The astute reader would notice that the gradient descent step that we are proposing is similar to Expectation Maximization (EM) algorithm. To explain this connection more carefully (and propose a slightly modified algorithm in the case of Gaussian data), we first write down the EM algorithm itself.

Let $X_1, \dots, X_n$ be data coming from from some distribution in the set of distributions with parameter θ with some latent or unobserved variables Z. Let $L(\theta; X, Z)$ be the likelihood function of the parameters. The EM algorithm then seeks to maximise

$$Q(\theta) = \sum_{i=1}^{n} \frac{1}{n} \mathbb{E}_{Z|X_i,\theta}[\log L(\theta; X_i, Z)].$$

To make matters a little clearer let us see what this implies when the data comes from a mixture of Gaussians. Let

$$X \sim \sum_{i=1}^{\kappa} p_i N(\mu_i, \Sigma_i).$$

To imagine a latent variable imagine the data being generated as follows. First sample a random variable Z such that $p(Z = i) = p_i$. Then sample an observation from $N(\mu_z, \Sigma_Z)$. Thus in this case the latent variable Z is the assignment of the data to its respective cluster or the observed data's label. The parameters θ in this case is the vector $(p_1, ..., p_\kappa, \mu_1, ..., \mu_\kappa, \Sigma_1, ..., \Sigma_\kappa)$. Then given data X_i, the probability the corresponding Z is

[5] This is a general problem in any clustering based tasks, and also present in deep clustering [Yan+17] works. A possible future approach would be along the lines of the semidefinite programming relaxation in [YSC17], however this comes with computational overhead and the extension is not trivial.

j is given by (using Bayes rule) :

$$\begin{aligned} T_{j,i} &= \frac{p_j f(X_i;\mu_j,\Sigma_j)}{\sum_t p_t f(X_i;\mu_t,\Sigma_t)} \\ &= \frac{p_j \sqrt{\det(2\pi\Sigma_j)} \exp\left(-\frac{1}{2}(X_i-\mu_j)\Sigma_j^{-1}(X_i-\mu_j)\right)}{\sum_t p_t \sqrt{\det(2\pi\Sigma_t)} \exp\left(-\frac{1}{2}(X_i-\mu_t)\Sigma_t^{-1}(X_i-\mu_t)\right)}. \end{aligned}$$

Then the expectation of the log likelihood $\mathbb{E}_Z[L(\theta;X,Z)]$ is :

$$\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{\kappa} T_{j,i}\left(\frac{\log\det(2\pi\Sigma_j)}{2} - \frac{||\Sigma_j^{-1/2}(X_i-\mu_j)||^2}{2}\right).$$

Now note that ignoring the determinant term ($\log\det(2\pi\Sigma_j)$) maximizing this above quantity is exactly minimizing

$$\sum_{i=1}^{n}\sum_{j=1}^{\kappa} T_{j,i}\left\|\Sigma_j^{-1/2}(X_i-\mu_j)\right\|^2.$$

Note then that this is very similar to the term we are optimizing if we allow T to be independent of Σ_j and set

$$\Sigma_j^{-1} = (\mathbb{I} - W_1^{(j)}W_2^{(j)})^\top(\mathbb{I} - W_1^{(j)}W_2^{(j)}) - \lambda W_2^{(j)\top}W_2^{(j)}, \quad \text{and} \quad \mu_j = C_j.$$

In other words we are fixing the covariance matrix to have numerically low rank, i.e. it has only h (few) of its eigenvalues are $1/\lambda$ (very large) whereas rest of them are 1 (small).

The connection between TAE and the EM algorithm is especially interesting as there is a vast literature on the theoretical properties of EM and therefore opens up future research directions for a more fine-grained analysis of TAEs.

6.4 Related Work and Future Applications

Theoretical analysis of AE. The theoretical understanding of simple AE is still limited and mainly summarized in Theorem 15 and formalizes the optimal parameterization of linear AE depending on the considered regularization [BH89; Kun+19; Bao+20; PKK18]. While the considered proof techniques differ in our work we derive a similar result of the TAE. An important future direction for theoretical analysis for both simple AE and TAE is the extension to the non-linear setting.

Before going into the related work on clustering and de-noising we would like to note here that the focus of this work is to show the difference in learning a single representation for the data and a representation for each cluster in the data. The general literature of possible, task specific, AE models is vast and would exceed the limits of this related work section. Therefore we focus on the most relevant related work, which is the basic setup for clustering on embedding and de-noising with simple AEs.

Clustering. The main goal of clustering is to group similar objects into the same class in an unsupervised setting. While this problem has been extensively studied in traditional machine learning in terms of feature selection [BDM09; ATL13], distance functions [Xin+02; XNZ08] and group methods [Mac67; Lux07; LMO04] (for a more comprehensive overview see [AR14]) the time complexity significantly increases with high dimensional data, previous work focuses on projecting data into low-dimensional spaces and then cluster the embedded representations [RL03; Tia+14; Wan+16]. For there there several methods have been developed that use deep unsupervised models to learn representations with a clustering focus that simultaneously learns feature representations and cluster assignments using deep neural networks [XGF16; Diz+17; Wan+16; XX15; Wan+15].

De-noising. We consider de-noising with AEs [BCM05; Cho13]. While there are several extensions to more complex AE models and task specific setups (see e.g. [Zha+22b]) in this work we focus on the question if learning cluster specific representations is beneficial for reducing the reconstruction error, which to the best of our Knowledge has been considered so far.

Possible future applications. We note that while in this chapter we focus on *clustering and de-noising* the general concept can be extended to other AE based downstream tasks such as anomaly detection [MFB19; SY14; ZP17], image compression [The+17; BLS17], super resolution [Zen+17; Son+17] and machine translation [Cho+14; SVL14].

Such extensions are especially of interest for future applications as Figure 6.4 and Section 6.2.3 show that TAE outperform standard AE especially when measured in the reconstruction quality. This indicates that tasks such as image compression can benefit from using TAE, as for such tasks the matching to the true clustering is not required.

6.5 Discussion

This chapter presents a meta-algorithm that can be used to better adapt any existing AE architecture to datasets in which one might anticipate cluster structures. By jointly learning the cluster structure and low dimensional representations of clusters, the proposed tensor auto-encoder (TAE) directly improves upon k-means, applied to a dataset or to an encoding of the same generated by an AE. More importantly, in the context of de-noising or downstream learning tasks, while it is trivial to note that mathematically a TAE can never have a worse MSE than a corresponding AE, we verify the same experimentally. On the surface, the difference in performance might simply seem to be a matter of more parameters. However, we show experimentally (see experiment on real data clustering, Figure 6.6) that even with the same total number of parameters TAE clusters better and hence gives a more accurate reconstruction than an AE. An open question in this regard would be to show such a result mathematically under certain data assumptions. Another open implementation problem would be to design a more efficient gradient step for the TAE.

Social impact. As this chapter has a stronger focus on developing new algorithms, that can directly be used in practice we conclude by discussing how this change in existing AE approaches can have implications in a societal context. Usage of traditional autoencoders on large diverse communities often favours creating a single latent monolithic representation mostly representing a single majority. This might create situations where the interests of various smaller communities are ignored. An instance of this might be to use autoencoders to find a couple of parameters that are the most significant markers of a particular disease. Using tensorized autoencoders might be of interest in these situations as this might create multiple representations for each community. On the other hand in cases where recovering more explicit clustering structures has negative implications, TAE might not be favorable. A possible example here could be differential privacy. However, to better understand this setting the interplay between properties of latent representations in AE and privacy will have to be more thoroughly investigated in the future.

Chapter 7

Comparison Based Clustering

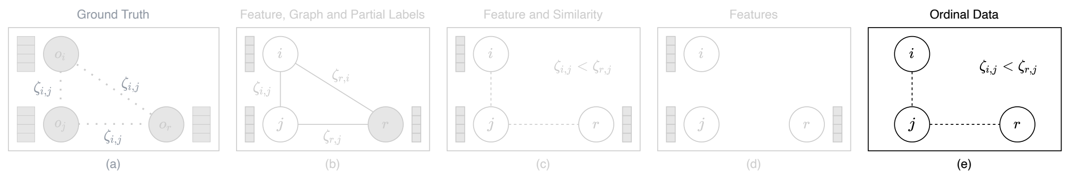


Figure 7.1: Ordinal Setup. Only ordinal comparisons between objects are given. Similarities between i, j given by $\zeta_{i,j}$ and between j, r given by $\zeta_{r,j}$ exist, however only $\zeta_{i,j} \lessgtr \zeta_{r,j}$ is observed.

All previous data-settings relied on having explicit feature descriptions of each object and in some cases relations between objects. For our final setting we consider *no feature information and only ordinal relations between data.* This can therefore be seen as the data-setting with the least information about the unlabelled objects[1].

In this setting we consider the task of clustering, where the objective is to group together objects that share the same semantic meaning, that are similar to each other, into κ disjoint partitions. In comparison to the previous chapter we however now aim to solve the clustering task directly instead of focusing on learning representations first. This problem has been extensively studied in the literature when a measure of similarity between the objects is readily available, for example when the examples have a Euclidean representation or a graph structure [SM00; AV07; von07]. However, it has attracted less attention when the objects are difficult to represent in a standard way, for example cars or food. A recent trend to tackle this problem is to use comparison based learning [Ukk17; EZK18] where, instead of similarities, one only observes comparisons between the examples:

1. **Triplet comparison:** Object x_i is more similar to object x_j than to object x_k as illustrated in Figure 7.1;
2. **Quadruplet comparison:** Objects x_i and x_j are more similar to each other than objects x_k and x_l.

[1]One could go from the unsupervised setting in the previous chapter to the one in this setting by constructing ordinal comparisons using a similarity metric on the features.

There are two ways to obtain these comparisons. On the one hand, one can adaptively query them from an oracle, for example a crowd. This is the active setting. On the other hand, they can be directly given, with no way to make new queries. This is the passive setting. In this chapter, we study comparison based learning for clustering using passively obtained triplets and quadruplets.

Motivation of this work. A key bottleneck in comparison based learning is the overall number of available comparisons: given n examples, there exist $\mathcal{O}\left(n^3\right)$ different triplets and $\mathcal{O}\left(n^4\right)$ different quadruplets. In practice, it means that, in most applications, obtaining all the comparisons is not realistic. Instead, most approaches try to use as few comparisons as possible. This problem is relatively easy when the comparisons can be actively queried and it is known that $\Omega\left(n \ln n\right)$ adaptively selected comparisons are sufficient for various learning problems [HGL17; EZK18; GPL19]. On the other hand, this problem becomes harder when the comparisons are passively obtained. The general conclusion in most theoretical results on learning from passive ordinal comparisons is that, in the worst case, almost all the $\mathcal{O}\left(n^3\right)$ or $\mathcal{O}\left(n^4\right)$ comparisons should be observed [JN11; EZK18]. The focus of this work is to show that, by carefully handling the passively obtained comparisons, it is possible to design comparison based approaches that use almost as few comparisons as active approaches for planted clustering problems.

Near-optimal guarantees for clustering with passive comparisons. In hierarchical clustering, [EZK18] showed that constructing a hierarchy that satisfies all comparisons in a top-down fashion requires $\Omega\left(n^3\right)$ passively obtained triplets in the worst case. Similarly, [GPL19] considered a planted model and showed that $\Omega\left(n^{3.5} \ln n\right)$ passive quadruplets suffice to recover the true hierarchy in the data using a bottom-up approach. Since the main difficulty lies in recovering the small clusters at the bottom of the tree, we believe that this latter result also holds for standard clustering. In this chapter, we consider a planted model for standard clustering and we show that, when the number of clusters k is constant, $\Omega\left(n(\ln n)^2\right)$ passive triplets or quadruplets are sufficient for exact recovery.[2] This result is comparable to the sufficient number of active comparisons in most problems, that is $\Omega\left(n \ln n\right)$ [HGL17; EZK18]. Furthermore, it is near-optimal. Indeed, to cluster an example, it is necessary to observe it in a comparison at least once as, otherwise, it can only be assigned to a random cluster. Thus, to cluster n objects, it is necessary to have access to at least $\Omega\left(n\right)$ comparisons. Finally, to obtain these results, we study a semi-definite programming (SDP) based clustering method and our analysis could be of significant interest beyond the comparison based framework.

General noise model for comparison based learning. In comparison based learning,

[2]When we write that $\Omega\left(n(\ln n)^2\right)$ comparisons are sufficient, we express that any number of comparisons greater than $Cn(\ln n)^2$ with C a constant is sufficient to solve the problem. In other words, it means that having exactly $Cn(\ln n)^2$ comparisons is sufficient but also that having more comparisons is not detrimental. This notation is used in statistics and information theory [FRG09] and is equivalent to $\gtrsim$.

there are two main sources of noise. First, the observed comparisons can be noisy, that is the observed triplets and quadruplets are not in line with the underlying similarities. This noise stems, for example, from the randomness of the answers gathered from a crowd. It is typically modelled by assuming that each observed comparison is randomly (and independently) flipped [JJN16; EZK18]. This is mitigated in the active setting by repeatedly querying each comparison, but may have a significant impact in the passive setting where a single instance of each comparison is often observed. Apart from the aforementioned observation errors, the underlying similarities may also have intrinsic noise. For instance, the food data set by [WKB14] contains triplet comparisons in terms of which items taste more similar, and it is possible that the taste of a dessert is closer to a main dish than to another dessert. This noise has been considered in [GPL19] by assuming that every pair of items possesses a latent random similarity, which affects the responses to comparisons. In this chapter, we propose, to the best of our knowledge, the first analysis that considers and shows the impact of both types of noise on the number of passive comparisons.

Scalable comparison based similarity functions. Several similarity and kernel functions have been proposed in the literature [KL17; GPL19]. However, computing these similarities is usually expensive as they require up to $\mathcal{O}(n)$ passes over the set of available comparisons. In this chapter, we propose new similarity functions whose construction is much more efficient than previous kernels. Indeed, they can be obtained with a single pass over the set of available comparisons. It means that our similarity functions can be computed in an online fashion where the comparisons are obtained one at a time from a stream. The main drawback compared to existing approaches is that we lose the positive semi-definiteness of the similarity matrix, but our theoretical results show that this is not an issue in the context of clustering. We also demonstrate this empirically as our similarities obtain results that are comparable with state of the art methods.

7.1 Background and Theoretical Framework

In this section, we present the comparison based framework and our planted clustering model, under which we later show that a small number of passive comparisons suffices for learning. We consider the following setup. There are n items, denoted by $[n] = \{1, 2, \dots, n\}$, and we assume that, for every pair of distinct items $i, j \in [n]$, there is an implicit real-valued similarity ζ_{ij} that we cannot directly observe. Instead, we have access to

$$\begin{aligned} &\text{Triplets:} && \mathcal{T} = \left\{(i,j,r) \in [n]^3 \ : \ \zeta_{ij} > \zeta_{ir}, \ i,j,r \text{ distinct}\right\}, \qquad \text{or} \\ &\text{Quadruplets:} && \mathcal{Q} = \left\{(i,j,r,s) \in [n]^4 \ : \ \zeta_{ij} > \zeta_{rs}, \ i \neq j, r \neq s, (i,j) \neq (r,s)\right\}. \end{aligned} \tag{7.1}$$

There are $\mathcal{O}\left(n^4\right)$ possible quadruplets and $\mathcal{O}\left(n^3\right)$ possible triplets, and it is expensive to collect such a large number of comparisons via crowdsourcing. In practice, $\mathcal{T}$ or $\mathcal{Q}$ only contain a small fraction of all possible comparisons. We note that if a triple $i, j, r \in [n]$ is observed with i as reference item, then either $(i, j, r) \in \mathcal{T}$ or $(i, r, j) \in \mathcal{T}$ depending on whether i is more similar to j or to r. Similarly, when tuples (i, j) and (r, s) are compared, we have either $(i, j, r, s) \in \mathcal{Q}$ or $(r, s, i, j) \in \mathcal{Q}$.

Sampling and noise in comparisons. This chapter focuses on passive observation of comparisons. To model this, we assume that the comparisons are obtained via uniform sampling, and every comparison is equally likely to be observed. Let $p \in (0, 1]$ denote a sampling rate that depends on n. We assume that every comparison (triplet or quadruplet) is independently observed with probability p. In expectation, $|\mathcal{Q}| = \mathcal{O}\left(pn^4\right)$ and $|\mathcal{T}| = \mathcal{O}\left(pn^3\right)$, and we can control the sampling rate p to study the effect of the number of observations, $|\mathcal{Q}|$ or $|\mathcal{T}|$, on the performance of an algorithm.

As noted in the introduction, the observed comparisons are typically noisy due to random flipping of answers by the crowd workers and inherent noise in the similarities. To model the external (crowd) noise we follow the work of [JJN16] and, given a parameter $\epsilon \in (0, 1]$, we assume that any observed comparison is correct with probability $\frac{1}{2}(1 + \epsilon)$ and flipped with probability $\frac{1}{2}(1 - \epsilon)$. To be precise, for observed triple $i, j, r \in [n]$ such that $\zeta_{ij} > \zeta_{ir}$,

$$\mathbb{P}\big((i, j, r) \in \mathcal{T} \mid \zeta_{ij} > \zeta_{ir}\big) = \frac{1 + \epsilon}{2}, \quad \text{whereas} \quad \mathbb{P}\big((i, r, j) \in \mathcal{T} \mid \zeta_{ij} > \zeta_{ir}\big) = \frac{1 - \epsilon}{2}. \tag{7.2}$$

The probabilities for flipping quadruplets can be similarly expressed. We model the inherent noise by assuming ζ_{ij} to be random, and present a model for the similarities under planted clustering.

Planted clustering model. We now present a theoretical model for the inherent noise in the similarities that reflects a clustered structure of the items. The following model is a variant of the popular stochastic block model, studied in the context of graph clustering [Abb17], and is related to the non-parametric weighted stochastic block model [XJL20].

We assume that the item set $[n]$ is partitioned into κ clusters $\mathcal{C}_1, \ldots, \mathcal{C}_\kappa$ of sizes $n_1, \ldots, n_\kappa$, respectively, but **the number of clusters κ as well as the clusters $\mathcal{C}_1, \ldots, \mathcal{C}_\kappa$ are unknown to the algorithm.** Let F_{in} and F_{out} be two distributions defined on $\mathbb{R}$. We assume that the inherent (and unobserved) similarities $\{\zeta_{ij} : i < j\}$ are random and mutually independent, and

$$\zeta_{ij} \sim F_{in} \quad \text{if } i, j \in \mathcal{C}_\ell \text{ for some } \ell, \qquad \text{and} \qquad \zeta_{ij} \sim F_{out} \quad \text{otherwise.}$$

We further assume that ζ_{ii} is undefined, $\zeta_{ji} = \zeta_{ij}$, and that for ζ, ζ' independent,

$$\begin{aligned} &\mathbb{P}_{\zeta,\zeta' \sim F_{in}}(\zeta > \zeta') = \mathbb{P}_{\zeta,\zeta' \sim F_{out}}(\zeta > \zeta') = 1/2, \quad \text{and} \\ &\mathbb{P}_{\zeta \sim F_{in}, \zeta' \sim F_{out}}(\zeta > \zeta') = (1 + \delta)/2 \quad \text{for some } \delta \in (0, 1]. \end{aligned} \tag{7.3}$$

The first condition in (7.3) requires that F_{in}, F_{out} do not have point masses, and is assumed for analytical convenience. The second condition ensures that within cluster similarities are larger than inter-cluster similarities—a natural requirement. [GPL19] used a special case of the above model, where F_{in}, F_{out} are assumed to be Gaussian with identical variances σ^2, and means satisfy $\mu_{in} > \mu_{out}$. In this case, $\delta = 2\Phi\big((\mu_{in} - \mu_{out})/\sqrt{2}\sigma\big) - 1$ where Φ is the cumulative distribution function of the standard normal distribution.

7.2 A Theoretical Analysis of Similarity Based Clustering

Before presenting our new comparison based similarity functions, we describe the SDP approach for clustering from similarity matrices that we use throughout the chapter [YSC18; CY20].

Similarity based clustering is widely used in machine learning, and there exist a range of popular approaches including spectral methods [von07], semi-definite relaxations [YS16], or linkage algorithms [Das16] among others. We consider the following SDP for similarity based clustering. Let $H \in \mathbb{R}^{n\times n}$ be a symmetric similarity matrix among n items, and $Z \in \{0,1\}^{n\times\kappa}$ be the cluster assignment matrix that we wish to estimate. For unknown number of clusters κ, it is difficult to directly determine Z, and hence, we estimate the *normalised clustering matrix* $N \in \mathbb{R}^{n\times n}$ such that $N_{ij} = \frac{1}{|\mathcal{C}|}$ if i, j co-occur in estimated cluster $\mathcal{C}$, and $N_{ij} = 0$ otherwise. Note that $\operatorname{Tr}(N) = \kappa$. The following SDP was proposed and analysed by [YSC18] under the stochastic block model for graphs, and can also be applied in the more general context of data clustering [CY20]. This SDP is agnostic to the number of clusters, but penalises large values of $\operatorname{Tr}(X)$ to restrict the number of estimated clusters:

$$\begin{aligned} &\max_{N} \ \operatorname{Tr}(HN) - \lambda \operatorname{Tr}(N) \\ &\text{s.t. } N \geq 0, \quad N \succeq 0, \quad N\mathbf{1} = \mathbf{1}. \end{aligned} \tag{SDP-λ}$$

Here, λ is a tuning parameter and $\mathbf{1}$ denotes the vector of all ones. The constraints $N \geq 0$ and $N \succeq 0$ restricts the optimisation to non-negative, positive semi-definite matrices.

[PEG20] presents a general theoretical result for SDP-λ. Assume that the data has an implicit partition into κ clusters $\mathcal{C}_1, \ldots, \mathcal{C}_\kappa$ of sizes $n_1, \ldots, n_\kappa$ and with cluster assignment matrix Z, and suppose that the similarity H is close to an *ideal similarity matrix* $\widetilde{H}$ that has a $\kappa \times \kappa$ block structure $\widetilde{H} = Z\Sigma Z^T$. The matrix $\Sigma \in \mathbb{R}^{\kappa\times\kappa}$ is such that $\Sigma_{\ell\ell'}$ represents the ideal pairwise similarity between items from clusters $\mathcal{C}_\ell$ and $\mathcal{C}_{\ell'}$. Typically, under a random planted model, $\widetilde{H}$ is the same as $\mathbb{E}[H]$ up to possible differences in the diagonal terms. For $H = \widetilde{H}$ and certain values of λ, the unique optimal solution of SDP-λ is a block diagonal matrix $N^* = Z\Lambda^{-1}Z^T$, where $\Lambda \in \mathbb{R}^{\kappa\times\kappa}$ is diagonal with entries $n_1, \ldots, n_\kappa$. Thus, in the *ideal case*, solving the SDP provides the desired normalised clustering matrix from which one can recover the partition $\mathcal{C}_1, \ldots, \mathcal{C}_\kappa$. The following result shows that N^* is also the unique optimal solution of SDP-λ if H is sufficiently close to $\widetilde{H}$.

Proposition 5 (Recovery of planted clusters using SDP-λ [PEG20])**.** *Let $Z \in \{0,1\}^{n\times\kappa}$ be the assignments for a planted κ-way clustering, $\widetilde{H} = Z\Sigma Z^T$, and $N^* = Z\Lambda^{-1}Z^T$ as defined above. Define*

$$\Delta_1 = \min_{\ell\neq\ell'}\left(\frac{\Sigma_{\ell\ell}+\Sigma_{\ell'\ell'}}{2} - \Sigma_{\ell\ell'}\right), \quad \textit{and} \quad \Delta_2 = \max_{i\in[n]}\max_{\ell\in[k]}\left|\frac{1}{|\mathcal{C}_\ell|}\sum_{j\in\mathcal{C}_\ell}\left(H_{ij}-\widetilde{H}_{ij}\right)\right|.$$

N^* is the unique optimal solution of SDP-λ for any choice of λ in the interval

$$\left\|H-\widetilde{H}\right\|_2 < \lambda < \min_\ell n_\ell \cdot \min\left\{\frac{\Delta_1}{2}, \Delta_1 - 6\Delta_2\right\}.$$

The term Δ_1 quantifies the separation between the ideal within and inter-cluster similarities, and is similar in spirit to the weak assortativity criterion for stochastic block models [YSC18]. On the other hand, the matrix spectral norm $\|H-\widetilde{H}\|_2$ and the term Δ_2 both quantify the deviation of the similarities H from their ideal values $\widetilde{H}$. Note that the number of clusters can be computed as $\kappa = \mathrm{Tr}\,(X)$ and cluster assignment Z is obtained by clustering the rows of N^* using κ-means or spectral clustering for example. In the experiments (Section 7.5), we present a data-dependent approach to tune λ and find κ.

We conclude this section by noting that most of the previous analyses of SDP clustering either assume sub-Gaussian data [YS16] or consider similarity matrices with independence assumptions [CX14; YSC18] that might not hold in general, and do not hold for our AddS-3 and AddS-4 similarities described in the next section. In contrast, the deterministic criteria stated in Proposition 5 make the result applicable in more general settings.

7.3 Similarities from Passive Comparisons

We present two new similarity functions computed from passive comparisons (AddS-3 and AddS-4) and guarantees for recovering planted clusters using SDP-λ in conjunction with these similarities. [KL17] introduced pairwise similarities computed from triplets. A quadruplets variant was proposed by [GPL19]. These similarities, detailed in Appendix 7.4, are positive-definite kernels and have multiplicative forms. In contrast, we compute the similarity between items i, j by simply adding binary responses to comparisons involving i and j.

Similarity from quadruplets. We construct the additive similarity for quadruplets, referred to as AddS-4, in the following way. Recall the definition of $\mathcal{Q}$ in Equation (7.1) and for every $i \neq j$, define

$$H_{ij} = \sum_{r\neq s}\left(\mathbb{I}_{\{(i,j,r,s)\in\mathcal{Q}\}} - \mathbb{I}_{\{(r,s,i,j)\in\mathcal{Q}\}}\right), \tag{AddS-4}$$

where $\mathbb{I}_{\{\cdot\}}$ is the indicator function. The intuition is that if i, j are similar (ζ_{ij} is large), then for every observed tuple i, j, r, s, $\zeta_{ij} > \zeta_{rs}$ is more likely to be observed. Thus, (i, j, r, s) appears in $\mathcal{Q}$ more often than (r, s, i, j), and H_{ij} is a (possibly large) positive term. On the other hand, smaller ζ_{ij} leads to a negative value of H_{ij}. Under the aforementioned planted model with clusters of size $n_1, \ldots, n_\kappa$, one can verify that H_{ij} indeed reveals the planted clusters in expectation since if i, j belong to the same planted cluster, then $\mathbb{E}[H_{ij}] = p\epsilon\delta \sum_{\ell \in [k]} \frac{n_\ell(n - n_\ell)}{2}$, and $\mathbb{E}[H_{ij}] = -p\epsilon\delta \sum_{\ell \in [k]} \binom{n_\ell}{2}$ otherwise. Thus, in expectation, the within cluster similarity exceeds the inter-cluster similarity by $p\epsilon\delta\binom{n}{2}$.

Similarity from triplets. The additive similarity based on passive triplets AddS-3 is given by

$$H_{ij} = \sum_{r \neq i,j} \left(\mathbb{I}_{\{(i,j,r)\in\mathcal{T}\}} - \mathbb{I}_{\{(i,r,j)\in\mathcal{T}\}}\right) + \left(\mathbb{I}_{\{(j,i,r)\in\mathcal{T}\}} - \mathbb{I}_{\{(j,r,i)\in\mathcal{T}\}}\right) \tag{AddS-3}$$

for every $i \neq j$. The AddS-3 similarity H_{ij} aggregates all the comparisons that involve both i and j, with either i or j as the reference item. Similar to the case of AddS-4, H_{ij} tends to be positive when ζ_{ij} is large, and negative for small ζ_{ij}. One can also verify that, under a planted model, the expected within cluster AddS-3 similarity exceeds the inter-cluster similarity by $p\epsilon\delta(n-2)$.

A significant advantage of AddS-3 and AddS-4 over existing similarities is in terms of computational time for constructing H. Unlike existing kernels, both similarities can be computed from a single pass over $\mathcal{T}$ or $\mathcal{Q}$. In addition, the following result shows that the proposed similarities can exactly recover planted clusters using only a few (near optimal) number of passive comparisons.

Theorem 17 (Cluster recovery using AddS-3 and AddS-4 [PEG20])**.** *Let N^* denote the normalised clustering matrix corresponding to the true partition, and $n_{\min}$ be the size of the smallest planted cluster. Given the triplet or the quadruplet setting, there exist absolute constants $c_1, c_2, c_3, c_4 > 0$ such that, with probability at least $1 - \frac{1}{n}$, N^* is the unique optimal solution of SDP-λ if δ satisfies $c_1 \frac{\sqrt{n \ln n}}{n_{\min}} < \delta \leq 1$, and one of the following two conditions hold:*

- ***(triplet setting)*** *H is given by AddS-3, and the number of triplets $|\mathcal{T}|$ and the parameter λ satisfy*

$$|\mathcal{T}| > c_2 \frac{n^3 (\ln n)^2}{\epsilon^2 \delta^2 n_{\min}^2}$$
$$\text{and} \quad c_3 \max\left\{\sqrt{|\mathcal{T}| \frac{\ln n}{n}}, |\mathcal{T}|\epsilon\sqrt{\frac{\ln n}{n^3}}, (\ln n)^2\right\} < \lambda < c_4 |\mathcal{T}| \frac{\epsilon\delta n_{\min}}{n^2} \ ;$$

- ***(quadruplet setting)*** *H is given by AddS-4, and the number of quadruplets $|\mathcal{Q}|$ and*

λ satisfy

$$|\mathcal{Q}| > c_2 \frac{n^3 (\ln n)^2}{\epsilon^2 \delta^2 n_{\min}^2}$$
$$\text{and} \quad c_3 \max \left\{ \sqrt{|\mathcal{Q}| \frac{\ln n}{n}}, |\mathcal{Q}| \epsilon \sqrt{\frac{\ln n}{n^3}}, (\ln n)^2 \right\} < \lambda < c_4 |\mathcal{Q}| \frac{\epsilon \delta n_{\min}}{n^2} .$$

The condition on δ and the number of comparisons ensure that the interval for λ is non-empty.

This result shows that given a sufficient number of comparisons, one can exactly recover the planted clusters using SDP-λ with an appropriate choice of λ. In particular, if there are κ planted clusters of similar sizes and δ satisfies the stated condition, then recovery of the planted clusters with zero error is possible with only $\Omega\left(\frac{\kappa^2}{\epsilon^2\delta^2} n (\ln n)^2\right)$ passively obtained triplets or quadruplets. In this particular context, we make a few important remarks about the sufficient conditions.

Remark 8 (Comparison with existing results)**.** For fixed κ and fixed $\epsilon, \delta \in (0,1]$, Theorem 17 states that $\Omega\left(n (\ln n)^2\right)$ passive comparisons (triplets or quadruplets) suffice to exactly recover the clusters. This significantly improves over the result of [GPL19] stating that $\Omega\left(n^{3.5} \ln n\right)$ passive quadruplets are sufficient in a planted setting, and the fact that $\Omega\left(n^3\right)$ triplets are necessary in the worst case [EZK18].

Remark 9 (Dependence of the number of comparisons on the noise levels ϵ, δ)**.** When one can actively obtain comparisons, [EZK18] showed that it suffices to query $\Omega\left(n \ln\left(\frac{n}{\epsilon}\right)\right)$ triplets. Compared to the $\ln\left(\frac{1}{\epsilon}\right)$ dependence in the active setting, the sufficient number of passive comparisons in Theorem 17 has a stronger dependence of $\frac{1}{\epsilon^2}$ on the crowd noise level ϵ. While we do not know whether this dependence is optimal, the stronger criterion is intuitive since, unlike the active setting, the passive setting does not provide repeated observations of the same comparisons that can easily nullify the crowd noise. The number of comparisons also depends as $\frac{1}{\delta^2}$ on the inherent noise level, which is similar to the conditions in [GPL19].

Theorem 17 states that exact recovery primarily depends on two sufficient conditions, one on δ and the other on the number of passive comparisons ($|\mathcal{T}|$ or $|\mathcal{Q}|$). The following two remarks show that both conditions are necessary, up to possible differences in logarithmic factors.

Remark 10 (Necessity of the condition on δ)**.** The condition on δ imposes the condition of $n_{\min} = \Omega\left(\sqrt{n \ln n}\right)$. This requirement on $n_{\min}$ appears naturally in planted problems. Indeed, assuming that all κ clusters are of similar sizes, the above condition is equivalent to a requirement of $\kappa = \mathcal{O}\left(\sqrt{\frac{n}{\ln n}}\right)$ and it is believed that polynomial time algorithms cannot recover $\kappa \gg \sqrt{n}$ planted clusters [CX14, Conjecture 1].

Remark 11 (Near-optimal number of comparisons)**.** To cluster n items, one needs to observe each example at least once. Hence, one trivially needs at least $\Omega(n)$ comparisons (active or passive). Similarly, existing works on actively obtained comparisons show that $\Omega(n \ln n)$ comparisons are sufficient for learning in supervised or unsupervised problems [HGL17; EZK18; GPL19]. We observe that, in the setting of Remark 8, it suffices to have $\Omega\left(n(\ln n)^2\right)$ passive comparisons which matches the necessary conditions up to logarithmic factors. However, the sufficient condition on the number of comparisons becomes $\Omega\left(\kappa^2 n(\ln n)^2\right)$ if κ grows with n while ϵ and δ are fixed. It means that the worst case of $\kappa = \mathcal{O}\left(\sqrt{\frac{n}{\ln n}}\right)$, stated in Remark 10, can only be tackled using at least $\Omega\left(n^2 \ln n\right)$ passive comparisons.

Remark 12 (No new information beyond $\Omega\left(n^2/\epsilon^2\right)$ comparisons)**.** Theorem 17 shows that for large n and $\Omega\left(n^2/\epsilon^2\right)$ number of comparisons, the condition for exact recovery of the clusters is only governed by the condition on δ as the interval for λ is always non empty. It means that, beyond a quadratic number of comparisons, no new information is gained by observing more comparisons. This explains why significantly fewer passive comparisons suffice in practice than the known worst-case requirements of $\Omega\left(n^3\right)$ passive triplets or $\Omega\left(n^4\right)$ passive quadruplets.

We conclude our theoretical discussion with a remark about recovering planted clusters when the pairwise similarities ζ_{ij} are observed. The presented methods are near optimal even in this setting.

Remark 13 (Recovering planted clusters for non-parametric F_{in}, F_{out})**.** Theoretical studies in the classic setting of clustering with observed pairwise similarities $\{\zeta_{ij} : i < j\}$ typically assume that the distributions F_{in} and F_{out} for the pairwise similarities are Bernoulli (in unweighted graphs), or take finitely many values (labelled graphs), or belong to exponential families [CX14; AJC15; YP16]. Hence, the applicability of such results are restrictive. Recently, [XJL20] considered non-parametric distributions for F_{in}, F_{out}, and presented a near-optimal approach based on discretisation of the similarities into finitely many bins. Our work suggests an alternative approach: compute ordinal comparisons from the original similarities and use clustering on AddS-3 or AddS-4. Theorem 17 then guarantees, for any non-parametric and continuous F_{in} and F_{out}, exact recovery of the planted clusters under a near-optimal condition on δ.

7.4 Existing Comparison Based Similarities and Kernel Functions

The literature on ordinal embedding from triplet comparisons is extensive [JN11; AC17]. In contrast, the idea of directly constructing similarity or kernel matrices from the comparisons, without embedding the data in an Euclidean space, is rather new. Such an approach is known to be significantly faster than embedding methods, and provides similar or sometimes better performances in certain learning tasks. To the best of our knowledge, there are only two works that learn kernel functions from comparisons [KL17; GPL19], while the works of [JJN16] and [MJN17] estimate a Gram (or kernel) matrix from the triplets, which is then further used for data embedding. In this section, we describe the aforementioned approaches for constructing pairwise similarities from comparisons. Through this discussion, we illustrate the fundamental difference between the proposed additive similarities, AddS-3 and AddS-4, and the existing kernels that are of multiplicative nature [KL17; GPL19].

Kernels from ordinal data were introduced by [KL17], who proposed two kernel functions (named k_1 and k_2) based on observed triplets. The kernels originated from the notion of Kendall's τ correlation between two rankings, and k_1 was empirically observed to perform slightly better. We mention this kernel function, which we refer to as a multiplicative triplet kernel (MulK-3). For any distinct $i, j \in [n]$, the MulK-3 similarity is computed as

$$H_{ij} = \frac{\sum\limits_{r<s} \left(\mathbb{I}_{\{(i,r,s)\in\mathcal{T}\}} - \mathbb{I}_{\{(i,s,r)\in\mathcal{T}\}}\right)\left(\mathbb{I}_{\{(j,r,s)\in\mathcal{T}\}} - \mathbb{I}_{\{(j,s,r)\in\mathcal{T}\}}\right)}{\sqrt{|\{(\ell,r,s)\in\mathcal{T} \;:\; \ell=i\}|}\sqrt{|\{(\ell,r,s)\in\mathcal{T} \;:\; \ell=j\}|}} \tag{MulK-3}$$

where $\mathcal{T}$ is the set of observed triplets. Note that this kernel does not consider comparisons involving ζ_{ij} but, instead, uses multiplicative terms indicating how i and j behave with respect to every pair r, s. For uniform sampling with rate $p \gg \frac{\ln n}{n^2}$, the denominators in MulK-3 are approximately $p\binom{n}{2}$ for every $i \neq j$. Hence, it suffices to focus only on the numerator. [GPL19] proposed a kernel similar to MulK-3 for the case of quadruplets, which is referred to as multiplicative quadruplet kernel (MulK-4). For $i \neq j$, it is given by

$$H_{ij} = \sum_{\ell\neq i,j}\sum_{r<s} \left(\mathbb{I}_{\{(i,\ell,r,s)\in\mathcal{Q}\}} - \mathbb{I}_{\{(r,s,i,\ell)\in\mathcal{Q}\}}\right)\left(\mathbb{I}_{\{(j,\ell,r,s)\in\mathcal{Q}\}} - \mathbb{I}_{\{(r,s,j,\ell)\in\mathcal{Q}\}}\right). \tag{MulK-4}$$

[GPL19] studied MulK-4 in the context of hierarchical clustering, and showed that it requires $\mathcal{O}\left(n^{3.5}\ln n\right)$ passive quadruplet comparisons to exactly recover a planted hierarchical structure in the data. Combining their concentration results with Proposition 5 shows that the same number of passive quadruplets suffices to recover the planted clusters considered in this work. Note that both MulK-3 and MulK-4 kernel functions have a multiplicative nature since each entry is an aggregate of products. This is essential for their positive semi-definite property. In contrast, the proposed AddS-3 and AddS-4 similarities simply aggregate comparisons involving the pairwise similarity w_{ij}, and hence, are not positive semi-definite kernels.

We also mention the work on fast ordinal triplet embedding (FORTE) [MJN17], which learns a metric from the given triplet comparisons. One can easily adapt the formulation to that of learning a kernel matrix $K \in \mathbb{R}^{n\times n}$ from triplets. Consider the squared distance in the corresponding reproducing kernel Hilbert space (RKHS), $d_K^2(i,j) = K_{ii} - 2K_{ij} + K_{jj}$. Assuming that the triplets adhere to the distance relation in the RKHS, it is easy to see that when a comparison of $t = \{i, r, s\}$ with i as pivot is available, then

$$\begin{aligned} y_t := \mathbb{I}_{\{(i,r,s)\in\mathcal{T}\}} - \mathbb{I}_{\{(i,s,r)\in\mathcal{T}\}} &= \operatorname{sign}\left(d_K^2(i,r) - d_K^2(i,s)\right) \\ &= \operatorname{sign}\left(K_{rr} - 2K_{ir} - K_{ss} + 2K_{is}\right), \end{aligned}$$

which is the sign of a linear map of K, which we can denote as $\operatorname{sign}(\langle M_t, K\rangle)$ for some $M_t \in \mathbb{R}^{n\times n}$. One can learn the optimal kernel matrix, that satisfies most triplet comparisons, by minimising the empirical loss $\frac{1}{|\mathcal{T}|}\sum_{t\in\mathcal{T}} \ell(y_t\langle M_t, K\rangle)$ with positive definiteness constraints for K, where ℓ is a loss function (log loss is suggested by [JJN16]).

7.5 Experiments

The goal of this section is three-fold: present a strategy to tune λ in SDP-λ; empirically validate our theoretical findings; and demonstrate the performance of the proposed approaches on real datasets.

Choosing λ and estimating the number of clusters based on Theorem 17. Given a similarity matrix H, the main difficulty involved in using SDP-λ is tuning the parameter λ. [YSC18] proposed the algorithm SPUR to select the best λ as

$$\lambda^* = \operatorname*{argmax}_{0\le\lambda\le\lambda_{\max}} \frac{\sum_{i\le k_\lambda}\sigma_i(N_\lambda)}{\operatorname{Tr}(N_\lambda)}$$

where N_λ is the solution of SDP-λ, κ_λ is the closest integer to $\operatorname{Tr}(N_\lambda)$ and an estimate of the number of clusters, $\sigma_i(N_\lambda)$ is the i-th largest eigenvalue of N_λ, and $\lambda_{\max}$ is a theoretically well-founded upper bound on λ. The maximum of the above objective is 1, achieved when N_λ has the same structure as N^* in Proposition 5. In our setting, Theorem 17 gives an upper bound on λ that depends on ϵ, δ and $n_{\min}$ which are not known in practice. Furthermore, it is computationally beneficial to use the theoretical lower bound for λ instead of using $\lambda \ge 0$ as suggested in SPUR.

We propose to modify SPUR based on the fact that the estimated number of clusters k monotonically decreases with λ. Given Theorem 17, we choose $\lambda_{\min} = \sqrt{c(\ln n)/n}$ and $\lambda_{\max} = c/n$, where $c = |\mathcal{Q}|$ or $|\mathcal{T}|$. The trace of the SDP-λ solution then gives two estimates of the number of clusters, $\kappa_{\lambda_{\min}}$ and $\kappa_{\lambda_{\max}}$, and we search over $\kappa \in [\kappa_{\lambda_{\max}}, \kappa_{\lambda_{\min}}]$ instead of searching over λ—in practice, it helps to search over the values $\max\{2, \kappa_{\lambda_{\max}}\} \le \kappa \le \kappa_{\lambda_{\min}} + 2$. We select κ that maximises the above SPUR objective, where N is computed using a simpler SDP [YSC18]:

$$\begin{aligned} &\max\nolimits_N \ \langle H, N\rangle && \text{(SDP-}\kappa\text{)} \\ &\text{s.t.} \quad N \ge 0, \quad N \succeq 0, \quad N\mathbf{1} = \mathbf{1}, \quad \operatorname{Tr}(N) = \kappa. \end{aligned}$$

Algorithm 1: Comparison-based SPUR

```
input  : The number of examples $n$ and the comparisons $\mathcal{T}$ or $\mathcal{Q}$.
begin
    Define $c = |\mathcal{T}|$ or $|\mathcal{Q}|$.
    Let $H$ be obtained with AddS-3 or AddS-4.
    Define $\lambda_{\min} = \sqrt{\frac{c(\ln c)}{n}}$ and $\lambda_{\max} = \frac{c}{n}$.
    $N_{\lambda_{\min}}, N_{\lambda_{max}} \leftarrow$ SDP-$\lambda_{\min}$, SDP-$\lambda_{\max}$ on $H$.
    $\kappa_{\lambda_{\min}}, \kappa_{\lambda_{\max}} \leftarrow \lfloor \mathrm{Tr}(N_{\lambda_{\min}}) \rceil, \lfloor \mathrm{Tr}(N_{\lambda_{\max}}) \rceil$.
    for $\kappa = \max\{2, \kappa_{\lambda_{\max}}\}$ to $\kappa_{\lambda_{min}} + 2$ do
        Solve SDP-$\kappa$ to obtain $N_\kappa$.
    end
    Choose $\hat{\kappa} = \underset{\kappa}{\mathrm{argmax}} \frac{\sum_{i \le \kappa} \sigma_i(N_\kappa)}{\mathrm{Tr}(N_\kappa)}$, where $\sigma_i(N_\kappa)$ denotes the $i$-th largest eigenvalue of
      $N_\kappa$.
end
output: Number of clusters $\hat{\kappa}$, $N_{\hat{\kappa}}$.
```

The overall approach is summarized in Algorithm 1.

Clustering with AddS-3 and AddS-4.[3] For the proposed similarity matrices AddS-3 and AddS-4, the above strategy provides the optimal number of clusters κ and a corresponding solution N_κ of SDP-κ. The partition is obtained by clustering the rows of N_κ using k-means. Alternative approaches, such as spectral clustering, lead to similar performances. In the last step of our approach, we use k-means to cluster the learned matrix N_κ. We experimentally demonstrate here that the partition obtained is, in fact, independent of the clustering algorithm used in this step. Hence, in Figure 7.2, we compare spectral clustering with k-means. As in the main previsou discussion, we here consider varying the number of observations, $|\mathcal{T}|, |\mathcal{Q}|$ and varying the crowd noise ϵ for both the setting where κ is estimated by SPUR and where we consider κ to be known. There is no differences between the ARI obtained when using k-means or spectral clustering.

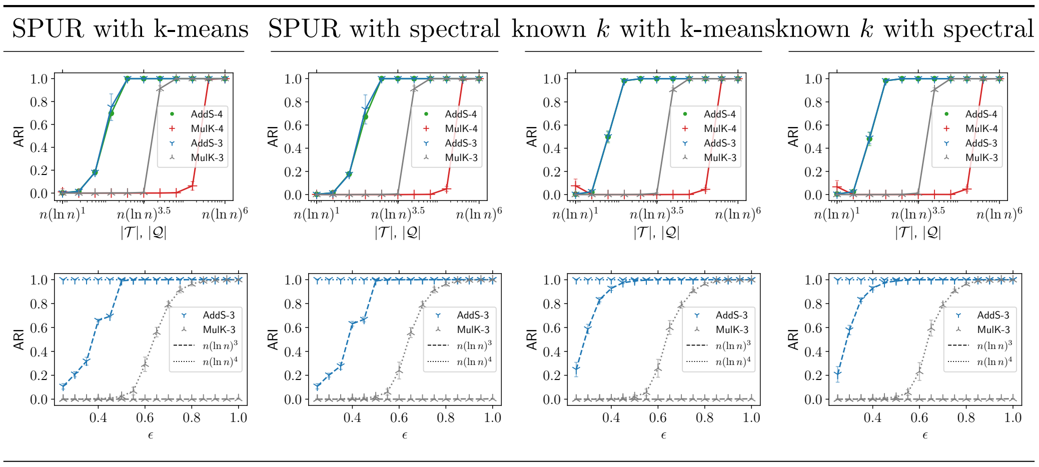
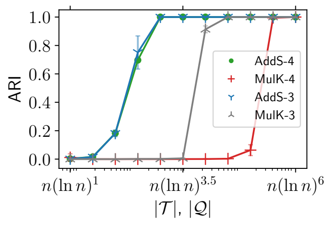


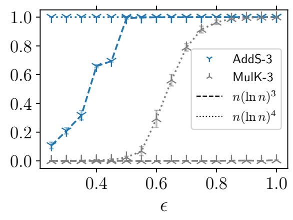
Figure 7.2: Comparing clustering algorithms to partition N in the last step. Using k-means or spectral clustering does not affect the output of our approach.

Evaluation function. We use the Adjusted Rand Index (ARI) [HA85] between the ground

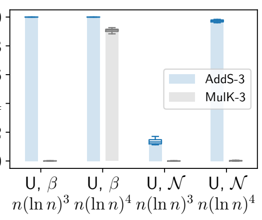
[3]We provide a Python implementation on `https://github.com/mperrot/AddS-Clustering`



(a) Vary the number of comparisons



(b) Vary the external noise level, ϵ



(c) Vary the distributions F_{in}, F_{out}

Figure 7.3: ARI of various methods on the planted model (higher is better). We vary: (7.3a) the number of comparisons $|\mathcal{T}|$ and $|\mathcal{Q}|$; (7.3b) the crowd noise level ϵ; (7.3c) the distributions F_{in} and F_{out}.

truth and the predictions. The ARI takes values in $[-1, 1]$ and measures the agreement between two partitions: 1 implies identical partitions, whereas 0 implies that the predicted clustering is random. In all the experiments, we report the mean and standard deviation over 10 repetitions.

7.5.1 Simulated Data with Planted Clusters

Overview. We generate data using the planted model from Section 7.1 and verify that the learned clusters are similar to the planted ones. As default parameters we use $n = 1000$, $\kappa = 4$, $\epsilon = 0.75$, $|\mathcal{T}| = |\mathcal{Q}| = n(\ln n)^4$ and $F_{in} = \mathcal{N}\left(\sqrt{2}\sigma\Phi^{-1}\left(\frac{1+\delta}{2}\right), \sigma^2\right), F_{out} = \mathcal{N}\left(0, \sigma^2\right)$ with $\sigma = 0.1$ and $\delta = 0.5$. In each experiment, we investigate the sensitivity of our method by varying one of the parameters while keeping the others fixed. We use SPUR to estimate the number of clusters. As baselines, we use SDP-κ (using the number of clusters estimated by our approaches) followed by k-means with two comparison based multiplicative kernels: MulK-3 for triplets [KL17] and MulK-4 for quadruplets [GPL19].

In Figure 7.3a, we vary the number of sampled comparisons. Unsurprisingly, our approaches are able to exactly recover the planted clusters using as few as $n(\ln n)^3$ comparisons—extra $\ln n$ factor compared to Theorem 17 accounts for ϵ, δ and constants. MulK-3 and MulK-4 respectively need $n(\ln n)^{4.5}$ and $n(\ln n)^{5.5}$ comparisons (both values exceed n^2 for $n = 1000$). In all our experiments, AddS-3 and AddS-4 have comparable performance while MulK-3 is significantly better than MulK-4. Thus we focus on triplets in the subsequent experiments for the sake of readability. In Figure 7.3b, we vary the external noise level ϵ. Given $n(\ln n)^4$ comparisons, AddS-3 exactly recovers the planted clusters for ϵ as small as 0.25 (high crowd noise) while, given the same number of comparisons, MulK-3 only recovers the planted clusters for $\epsilon > 0.9$. Figure 7.3c shows that AddS-3 outperforms MulK-3 even when different distributions for F_{in} and F_{out} are considered (Uniform + Beta or Uniform + Normal; details are discussed below). It also shows that the distributions affect the performances, which is not evident from Theorem 17, indicating the possibility of a refined analysis under distributional assumptions.

We now provide additional experiments on our planted model. We demonstrate that, given a sufficient number of comparisons, SPUR correctly estimates the number of clusters. We give details on the distributions used in Figure 7.3c. Finally, we consider several additional experiments where we vary the planted model parameters.

Compare SPUR with known k. An important question is how good is SPUR at estimating the true number of clusters. We illustrate this in Figure 7.4. We start by comparing the first two columns, showing how the ARI changes for various parameters of the planted model. In the setting of $|\mathcal{Q}|, |\mathcal{T}| = n(\ln n)^3$ we see that using a known number of clusters outperforms SPUR, especially in parameter ranges that are harder to cluster (e.g. small δ, ϵ or for a larger number of clusters). If we consider $|\mathcal{Q}|, |\mathcal{T}| = n(\ln n)^4$, SPUR correctly estimates the number of clusters and thus we omit the plots with known k.

Experimental details for changing F_{in}, F_{out} in the planted model In this section, we give implementation details on the different distributions considered in Figure 7.3c. In the following let ϕ be the normal pdf and Φ the normal cdf. Recall that, in all the experiments, we fix $\delta = 0.5$ as the default.

Parameters for F_{in} and F_{out} normal distributions. Let $F_{in} = \mathcal{N}(\mu_{in}, \sigma)$ and $F_{out} = \mathcal{N}(\mu_{out}, \sigma)$. We fix $\sigma = 0.1$ and $\mu_{out} = 0$. Using δ we can compute μ_{in}. Indeed, in this case, the cumulative distribution function is known and, thus, by setting it equal to $\mathbb{P}_{\zeta \sim F_{in}, \zeta' \sim F_{out}}(\zeta > \zeta') = \frac{1+\delta}{2}$ for some $\delta \in (0, 1]$ (as given in Equation (7.3)) we directly get the δ defined in Section 7.1: $\delta = 2\Phi\left((\mu_{in} - \mu_{out})/(\sqrt{2}\sigma)\right) - 1$. Then, assuming that $\mu_{out} = 0$, we get $\mu_{in} = \sqrt{2}\sigma\Phi^{-1}\left(\frac{1+\delta}{2}\right)$.

Parameters for F_{in} and F_{out} Beta distributions. Let $F_{in} = \mathrm{Beta}(\alpha, \beta), F_{out} = \mathrm{Beta}(\alpha', \beta')$. We set $\alpha' = \beta' = 1$ such that $F_{out} = \mathrm{Beta}(1, 1) = \mathrm{Unif}(0, 1)$. We can then compute

$$\mathbb{P}_{\zeta \sim \mathrm{Beta}(\alpha,\beta), \zeta' \sim \mathrm{Beta}(1,1)}(\zeta > \zeta') = \mathbb{E}_\zeta\left[\int_0^\zeta d\zeta'\right] = \mathbb{E}_\zeta\left[\zeta\right] = \frac{\alpha}{\alpha + \beta}$$

where the last line follows from the mean of the Beta distribution. Setting this equal to $\frac{1+\delta}{2}$ and solving for α gives: $\alpha = \beta\left(\frac{1+\delta}{1-\delta}\right)$. In our experiments, we fix $\beta = 2$.

Parameters for F_{in} Normal and F_{out} Uniform. Let $F_{in} = \mathcal{N}(\mu, 0), F_{out} = \mathrm{Unif}(0, 1)$. To set μ, we compute:

$$\begin{aligned}\mathbb{P}_{\zeta \sim \mathcal{N}(\mu,0), \zeta' \sim \mathrm{Unif}(0,1)}(\zeta > \zeta') &= \int_0^\infty \phi(\zeta - \mu)d\zeta\left[\int_0^{\min(w,1)} d\zeta'\right] \\ &= \int_0^1 \zeta\phi(\zeta - \mu)d\zeta + \int_1^\infty \phi(\zeta - \mu)d\zeta + \mu\left(\Phi(1-\mu) - \Phi(-\mu)\right) \\ &= 1 + \phi(-\mu) - \phi(1-\mu) + (\mu - 1)\Phi(1-\mu) - \mu\Phi(-\mu)\end{aligned}$$

Solving numerically for μ gives $\mu = \frac{1+\delta}{2}$.

Influence of different planted model parameters In this section we present additional experiments where we vary various parameters of the planted model. Recall that we consider the following parameters as default: $n = 1000$, $\kappa = 4$, $\epsilon = 0.75$, $|\mathcal{T}| = |\mathcal{Q}| = n(\ln n)^4$ and $F_{in} = \mathcal{N}\left(\sqrt{2}\sigma\Phi^{-1}\left(\frac{1+\delta}{2}\right), \sigma^2\right), F_{out} = \mathcal{N}\left(0, \sigma^2\right)$ with $\sigma = 0.1$ and $\delta = 0.5$.

Number of samples n, first row in Figure 7.4. We can first note that for $|\mathcal{Q}|, |\mathcal{T}| = n(\ln n)^3$ there is no difference in the behaviour between SPUR and known k. Both AddS-3 and AddS-4 achieve full recovery while MulK-3 and MulK-4 predictions are random. To learn somewhat meaningful partitions with MulK-3, one needs to increase the number of observations to $n(\ln n)^4$. However, even with this many comparisons, MulK-4 still learns random clusters.

Intrinsic noise δ, second row in Figure 7.4. Using $|\mathcal{Q}|, |\mathcal{T}| = n(\ln n)^3$, we see that, for both SPUR and known k, AddS-3 and AddS-4 exactly recover the clusters even when the intrinsic noise is high, that is $\delta = 0.4$. MulK-3 and MulK-4 can only make random predictions in this case. When the number of observations increases to $n(\ln n)^4$, AddS-3 and AddS-4 exactly recover the clusters even for values of δ that are as small as 0.25. In this case, MulK-4 still predicts random clusters, while MulK-3 is able to recover the clusters when the intrinsic noise is sufficiently small, that is $\delta \geq 0.6$.

Crowd noise ϵ, third row in Figure 7.4. This parameter was already analyzed previously. The plots are recalled here for the sake of completeness.

Number of clusters κ, fourth row in Figure 7.4. Finally, we vary the number of planted clusters. Here, we observe the most noticeable difference between SPUR and known κ. For $|\mathcal{Q}|, |\mathcal{T}| = n(\ln n)^3$, AddS-3 and AddS-4 with SPUR achieve perfect recovery for up to five clusters. While we notice a similar behaviour for AddS-3 and AddS-4 with known κ, the drop in ARI only starts for $\kappa > 7$ and is far less important than with SPUR. For $n(\ln n)^4$ observations AddS-3 and AddS-4 consistently recover all the clusters. On the other hand, MulK-3 only recovers clusters up to $\kappa = 3$ (here, MulK-3 uses the number of clusters estimated by AddS-3 with SPUR, that is $\kappa = 3$). Once again, MulK-4 can only make random predictions.

7.5.2 MNIST Clustering with Comparisons

We consider two datasets which are subsets of the MNIST test dataset [LC10a] that originally contains 10000 examples roughly equally distributed among the ten digits: (i) a subset of 2163 examples containing all the 1 and 7 (*MNIST 1vs.7*), two digits that are visually very similar, and (ii) a randomly selected subset of 2000 examples drawn without replacement and covering all 10 classes (*MNIST 10*). In both cases, to generate the comparisons, we use the Gaussian similarity on a 2-dimensional embedding of the entire MNIST test data constructed with t-SNE [Maa14] and normalized so that each example lies in $[-1, 1]^2$. We

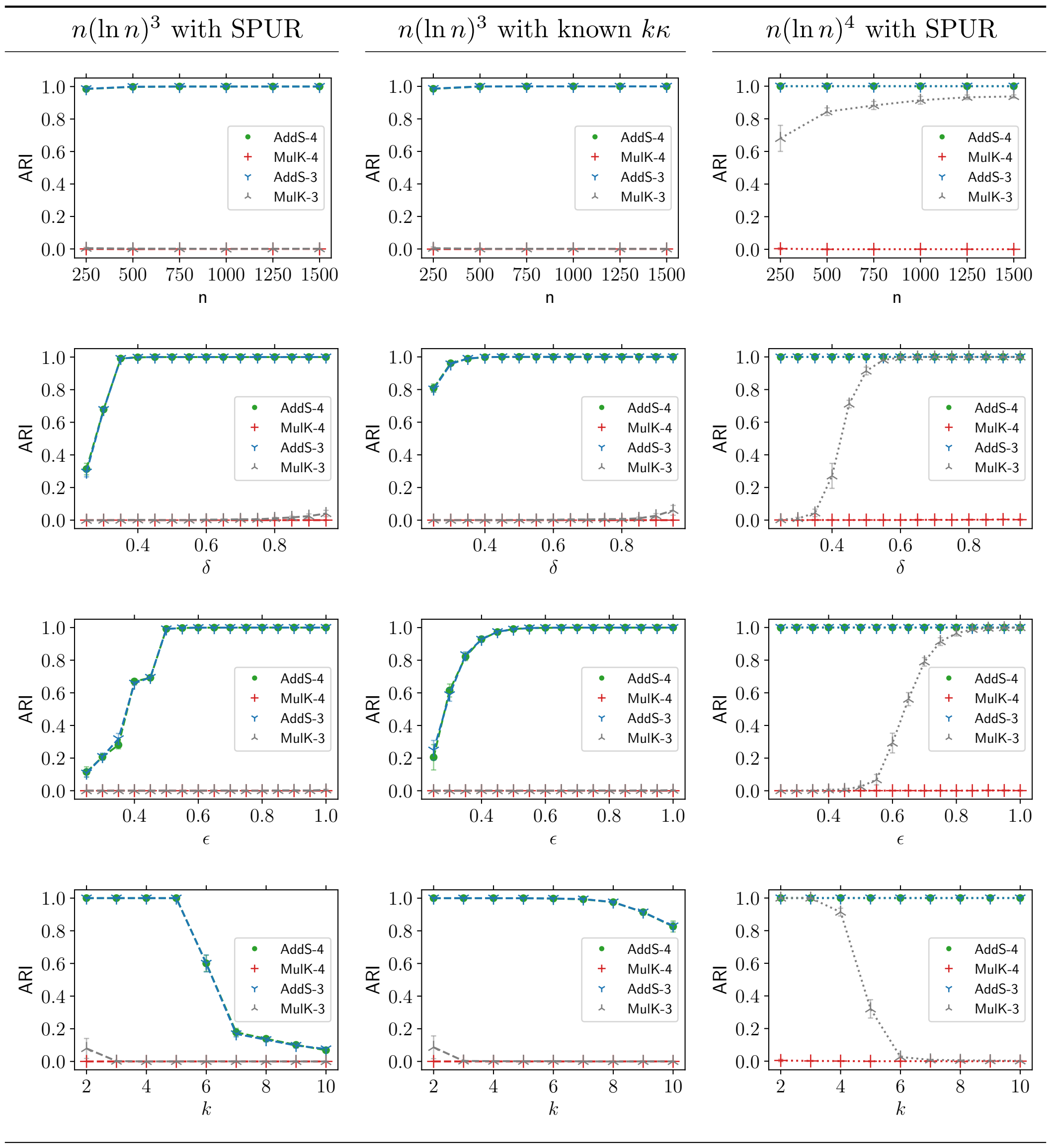
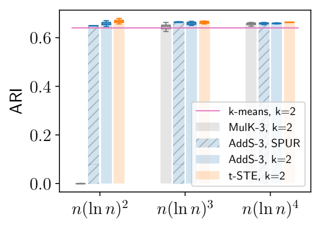
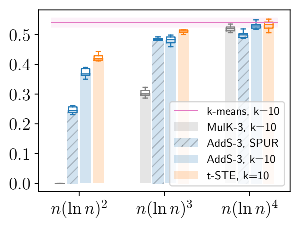


Figure 7.4: Further experiments on the planted model. On the one hand, SPUR needs sufficiently many comparisons to correctly estimate the number of underlying clusters. On the other hand, our approaches are not overly sensitive to changes in the planted model parameters and are able to exactly recover the planted clusters with $n(\ln n)^3$ comparisons even in fairly difficult cases (small δ, high κ, ...). Furthermore, given $n(\ln n)^4$ comparisons, our approaches are able to exactly recover the planted clusters in all the considered cases.



(a) MNIST 1vs.7, $n = 2163$



(b) MNIST 10, $n = 2000$

Figure 7.5: Experiments on MNIST using the cosine similarity. The absolute ARI performances are different from the Gaussian similarity. However, the overall trend is preserved and, given sufficiently many comparisons, all the ordinal baselines reach the performance of k-means on the original data.

focus on the triplet setting and we randomly and uniformly draw, without replacement, between $n(\ln n)^2$ and $n(\ln n)^4$ comparisons to be observed by the different approaches. We also consider two additional baselines. First, we use t-STE [MW12], an ordinal embedding approach, to embed the examples in 2 dimensions, and then cluster them using k-means on the embedded data. Second, we directly use k-means on the normalized data obtained with t-SNE. The latter is a baseline with access to Euclidean data instead of triplet comparisons.

We now consider additional experiments on the MNIST dataset. First, we consider a second similarity measure to generate the triplets. Then, we illustrate the partitions obtained with AddS-3 with known κ and SPUR respectively.

Gaussian similarity. We previously used the Gaussian similarity to generate the comparisons. More precisely, we compute the similarity between two examples x_i and x_j as

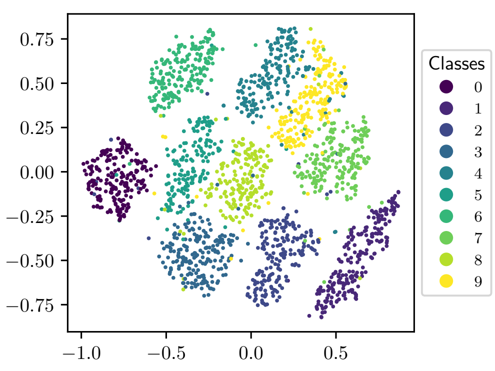
$$\zeta_{ij} = \exp\left(\frac{\|x_i - x_j\|_2^2}{\gamma^2}\right) \text{ with } \gamma = 1.$$

Cosine similarity. Instead of the Gaussian similarity, we could consider alternatives to generate the comparisons. For example, the Cosine similarity:

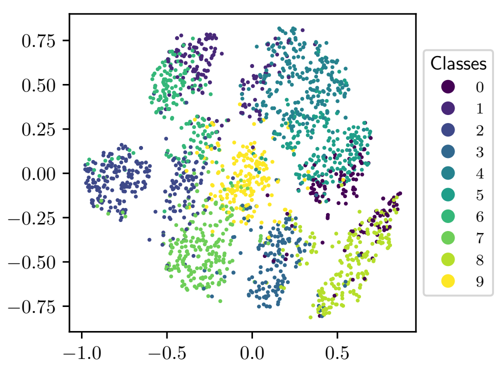
$$\zeta_{ij} = \frac{\langle x_i, x_j \rangle}{\|x_i\|_2 \|x_j\|_2}.$$

In Figure 7.5, we show that using this alternative similarity affects the absolute results of the considered approaches. However, it does not change the overall trend, that is, as the number of comparisons increases, AddS-3 converges to the baseline of k-means with access to the original similarities.

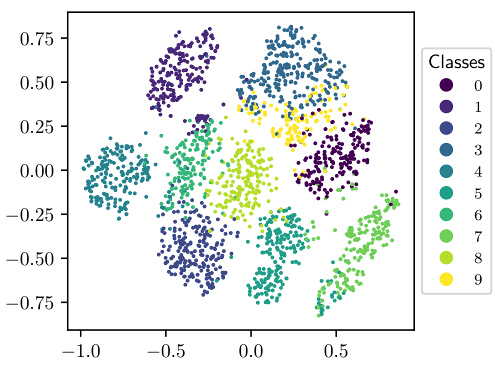
Clustering using known k. Figure 7.6a shows the t-SNE embedding of 2000 MNIST samples of all ten classes, where we see a clear separation between some classes (for example, 0 and 1) and very close embedding between others (for example, 1 and 9). Note that the classes obtained by AddS-3 are shown up to permutations and may not reflect the majority label in the different clusters. Further note that the data presented here corresponds to a single



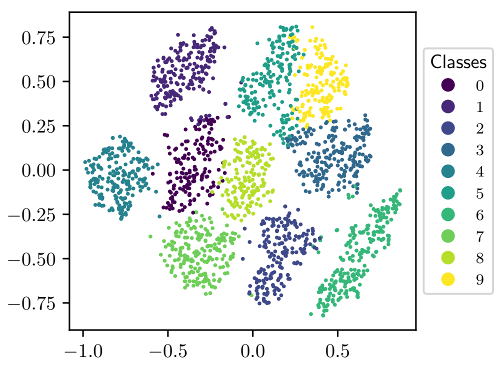
(a) MNIST embedding with true labels



(b) AddS-3 $\kappa = 10$, $|\mathcal{T}| = n(\ln n)^2$



(c) AddS-3 $\kappa = 10$, $|\mathcal{T}| = n(\ln n)^3$

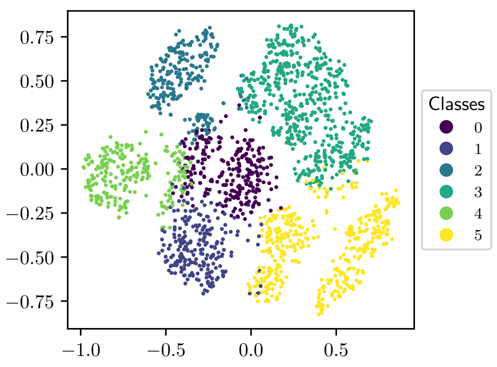


(d) AddS-3 $\kappa = 10$, $|\mathcal{T}| = n(\ln n)^4$

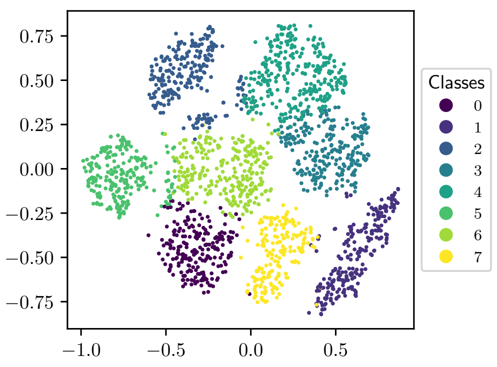
Figure 7.6: t-SNE embedding of 2000 MNIST samples with (7.6a) true labeling and (7.6d)–(7.6b) clusters obtained by AddS-3 with known $\kappa = 10$ and varying number of observations. The classes are given up to permutations and may not reflect the majority label in each cluster.

repetition out of the 10 repetitions used to compute the mean ARI (with standard deviation). In Figure 7.6d, we see that, for $|\mathcal{T}| = n(\ln n)^2$, the learned partition is not very representative of the original labels. Figure 7.6c shows that, when the number of comparisons increases to $|\mathcal{T}| = n(\ln n)^3$, the recovery ability of AddS-3 is greatly improved. However, the obtained partitions are not entirely satisfactory. Finally, Figure 7.6b shows that, when the number of comparisons further increases to $|\mathcal{T}| = n(\ln n)^4$, the clustering obtained is close to the true labeling and most clusters are correctly identified.

Clustering using SPUR. In this second set of experiments, we extend our observations from the previous paragraph to the labeling obtained by AddS-3 using SPUR. One can note that SPUR always underestimates the number of clusters. Hence, in Figure 7.7a, with $|\mathcal{T}| = n(\ln n)^3$, the number of predicted clusters is $\kappa = 6$ while, in Figure 7.7b, with $|\mathcal{T}| = n(\ln n)^4$, the number of predicted clusters is $\kappa = 8$. This explain the slightly worse behaviour of SPUR compared to known κ in Figure 7.9b. Nevertheless, the difference in average ARI is not so significant when $|\mathcal{T}| = n(\ln n)^4$, suggesting that 8 clusters is, in fact, a good estimate of the number of clusters that can reliably be distinguished by the different methods.



(a) AddS-3 SPUR, $|\mathcal{T}| = n(\ln n)^3$



(b) AddS-3 SPUR, $|\mathcal{T}| = n(\ln n)^4$

Figure 7.7: t-SNE embedding of 2000 MNIST samples with the clusters predicted by AddS-3 using SPUR and varying number of comparisons. The classes are given up to permutations and may not reflect the majority label in each cluster.

For *MNIST 1vs.7* (Figure 7.9a), $|\mathcal{T}| = n(\ln n)^2$ is sufficient for AddS-3 to reach the performance of k-means and t-STE while MulK-3 requires $n(\ln n)^3$ triplets. Furthermore, note that AddS-3 with known number of clusters performs similarly to AddS-3 using SPUR, indicating that SPUR estimates the number of clusters correctly. If we consider *MNIST 10* (Figure 7.9b) and $|\mathcal{T}| = n(\ln n)^2$, AddS-3 with known κ outperforms AddS-3 using SPUR, suggesting that the number of comparisons here is not sufficient to estimate the number of clusters accurately. Moreover, AddS-3 with known κ outperforms MulK-3 while being close to the performance of t-STE. Finally for $n(\ln n)^4$ triplets, all ordinal methods converge to the baseline of k-means with access to original data. The ARI of AddS-3 SPUR improves when the number of comparisons increases due to better estimations of the number of clusters—estimated k increases from 3 for $|\mathcal{T}| = n(\ln n)^2$ up to 9 for $|\mathcal{T}| = n(\ln n)^4$.

7.5.3 Real Comparison Based Data.

First, we consider the Food dataset [WKB14] that contains 100 examples and 190376 triplet comparisons. Unfortunately, there is no ground truth and, thus, quantitatively assessing the quality of the obtained partitions is difficult. Second, we consider the Car dataset[4] [KL16]. It contains 60 examples grouped into 3 classes (SUV, city cars, sport cars) with 4 outliers, and exhibits 12112 triplet comparisons. For this dataset, AddS-3 SPUR estimates $\kappa = 2$ instead of the correct 3 clusters. Figure 7.9c considers all ordinal methods with $\kappa = 2$ and $\kappa = 3$, and shows the pairwise agreement (ARI) between different methods and also with the true labels. While (MulK-3) with $\kappa = 3$ agrees the most with the true labels, all the clustering methods agree well for $\kappa = 2$ (top-left 3×3 block). Hence, the data may have another natural clustering with two clusters, suggesting possible discrepancies in how

[4]The Car dataset [KL16] is a comparison based dataset that contains 60 examples grouped into 3 classes (SUV, city cars, sport cars) with 4 outliers. This dataset originally comes with a set of 6056 comparisons of the form "x_i is most central in the triple x_i, x_j, x_k." Each of these comparisons corresponds to two triplets: "x_j is more similar to x_i than to x_k" and "x_k is more similar to x_i than to x_j." Hence, we have access to 12112 triplet comparisons.

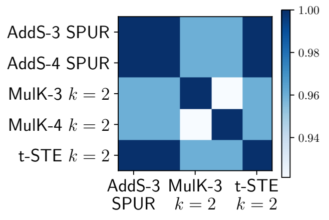
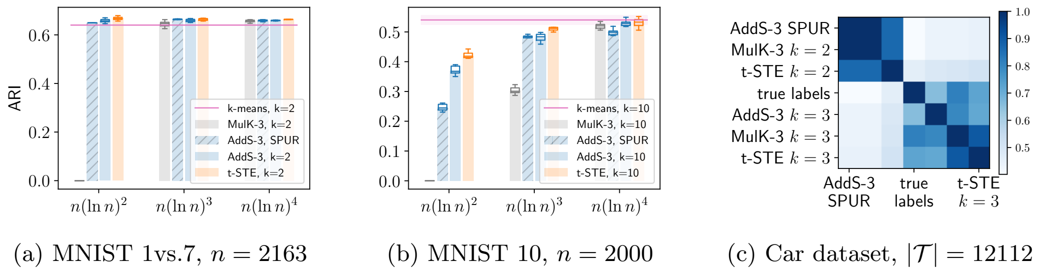


Figure 7.8: ARI between the clustering obtained by the different baselines. AddS-3 and AddS-4 with SPUR both estimate that the number of cluster is $\kappa = 2$. There is a high degree of agreement between the different approaches.



(a) MNIST 1vs.7, $n = 2163$ (b) MNIST 10, $n = 2000$ (c) Car dataset, $|\mathcal{T}| = 12112$

Figure 7.9: Experiments on real datasets. (7.9a)–(7.9b) ARI on MNIST; (7.9c) ARI similarity matrix comparing the clusters obtained by the different methods on car (darker means more agreement).

different people judge the similarities between cars (for instance, color or brand instead of the specified classes).

*Food Dataset*In addition to the Car dataset we now look at a second comparison based dataset called Food [WKB14]. It contains 100 food images and comes with 190376 triplet comparisons. Since there are no ground truth labels for the food dataset, we use the number of clusters estimated by SPUR for all methods and plot, in Figure 7.8, the similarity matrix between the different clustering approaches considered. Here, there is a high degree of agreement between all the clustering methods. Thus, most approaches predict the same clusters up to minor differences for a few data points.

7.6 Discussion

It is generally believed that a large number of passive comparisons is necessary in comparison based learning. Existing results on clustering require at least $\Omega\left(n^3\right)$ passive comparisons in the worst-case or under a planted framework. We show that, in fact, $\Omega\left(n(\ln n)^2\right)$ passive comparisons suffice for accurately recovering planted clusters. This number of comparisons is near-optimal, and almost matches the number of active comparisons typically needed for learning. Our theoretical findings are based on two simple approaches for constructing pairwise similarity matrices from passive comparisons (AddS-3 and AddS-4). The present analysis is in a restricted framework, where all within (or inter) cluster similarities are assumed to be identically distributed. Based on existing work on robustness of SDPs [MPW16], we believe that our theoretical result holds in a more general semi-random setting. Lastly, while we studied the merits of AddS-3 and AddS-4 in the context of clustering, they could be used for other problems such as semi-supervised learning, data embedding, or classification.

Broader Impact This work primarily has applications in the fields of psychophysics and crowdsourcing, and more generally, in learning from human responses. Such data and learning problems could be affected by implicit biases in human responses. However, this latter issue is beyond the scope of this work and, thus, was not formally analysed.

Chapter 8

Outlook

In this thesis, we took a journey through different data settings and algorithms and we worked towards a better understanding of the role of unlabelled data. With the number of covered settings and algorithms being so broad it is clear that a 'complete' overview of the topic would go beyond the scope of any thesis. In addition the area of learning from unlabelled data is constantly evolving with new data settings and algorithms emerging. So how broadly applicable are the findings of this work and what are its implications? While translating theoretical results from one model or data setting to another is often not directly possible, results such as the ones presented in this thesis can *build a foundation.* Consider as an example the two contrastive losses proposed in Chapter 4. While from an engineering standpoint, the two are very similar (one squares the loss term with regard to the negative sample) the theoretical analysis is very different. Nevertheless, both build on the proposed representer theorem (Theorem 9) and it is reasonable to assume that a similar analysis can be extended to other SSL losses under orthogonal constraints.

Let us therefore outline how the different parts of this thesis build such foundations. In this final chapter, we revisit the main data settings and algorithms considered in this thesis and ask what major questions or directions emerge from them. To approach this question we first consider the viewpoint of theoretical tools in Section 8.1. We start from traditional learning theoretical bounds and explore where we see their future applications and limitations. From there we highlight how an analysis through dynamics and infinite width networks can complete the picture in cases where learning theoretical bounds fail and how this work builds the foundation for such an analysis in the SSL setting. Finally we propose advances for the main proposed algorithms in Section 8.2.

8.1 From Learning Theoretical Bounds to and Infinite Width Analysis

Learning theoretical analysis of Kernel methods. In the supervised setting and in the context to traditional models such as Kernels, learning theoretical bounds still play an important role [SSM97]. We observe in Chapter 4 & 5 that this picture still applies to the unlabelled data setting when considering Kernel versions of popular neural network approaches. As we expect both chapters to also build the foundation for future developments of additional Kernel methods (e.g. under different SSL losses) this outlines one of the main areas of future applications for traditional learning theoretical measures. In those cases traditional bounds still provide precise tools that give valuable insights and guarantees for models.

Beyond learning theoretical analysis of GCNs. Moving away from the Kernel setting the application becomes more nuanced and a blind application of established learning theoretical measures does not always provide the full picture. While the analysis of GCNs using Rademacher complexity provided some good characterizations it also outlined that even under strong model assumptions such bounds will remain loose and can only model certain trends as shown in the behaviour with depth. Several follow-up works to [EVG21], underline this assessment as they mainly support our finding that to obtain more informative bounds it is crucial to pick the 'right' complexity measures. A promising candidate PAC learning [LUZ21; Ju+23] allowing for numerically tighter bounds but still suffers from some of the same limitations (e.g. with regards to depth and overparameterization). In addition, most of those approaches do not provide results that model the influence of the graph properties on the generalization behavior in detail as we could show in Theorem 2 through an analysis under planted model assumptions. However this is central if considering this problem from the perspective of unlabelled data. While it could be interesting to refine existing PAC bounds [Ju+23] by considering planted graph model settings all those settings have the limitations of traditional learning theoretical bounds in common (e.g. with regards to modeling behaviour in the overparameterized regime) and are therefore inherently limited in their explainability. Let us outline possible path that can lead beyond those limitations

Alternative Analysis Approaches. In the broader deep learning context, there has been a growing call for alternatives to standard learning theoretical bounds since they do not adequately capture the behavior of deep models [Ney+17]. To this end, different limiting case analyses have been introduced. In the context of GNNs, it is known that GNNs have a fundamental connection to belief propagation and message passing [DDS16; Gil+17] and some theoretical analyses of GNNs have been based on cavity methods and mean field approaches for supervised [ZLZ20] and transductive settings [KTO19; CBL19]. The central idea of these approaches is to show results in the limit of the number of iterations. In another limiting setting, [Du+19a] study GNNs with infinitely wide hidden layers, and derive corresponding neural tangent kernel [JGH18a; Aro+19b] that can provide generalization error bounds in the supervised setting. [KBV20] derive continuous versions of GNNs applied to large random graphs.

From Generalization error bounds to exact risk. Beyond the above approaches the modeling of generalization error bounds or more precisely the derivation of loss curves can also be approached through the lens of random matrix theory. This direction is interesting as it allows for very precise bounds and also models the overparameterized regime well. However this comes at the cost of a high complexity of analysis, making it currently only feasible for simple models such as lazy trained two-layer networks [MM22] and often under very strong assumptions on data. In addition even small changes in the considered model require significant changes in proof techniques. Since we published [EVG21], the basis of Chapter 2, follow-up works have built on the presented results and addressed several of the earlier outlined questions. Most importantly a very recent work takes an important step towards a more exact characterization, by showing that exact risk characterizations are also possible in simple GCNs under SBM assumptions [Shi+23]. While still performed for simple GCN models this work allows an exact characterization of risk bounds while taking the graph information (in the case of [Shi+23], especially the role of homophily in graphs) into consideration.

Traditional learning theoretical bounds and exact risk computation seem to outline two extremes of possible approaches to model risk and illustrate the trade-off between expressively, complexity and flexibility of the analysis. Is there a middle ground that provides helpful characterizations — maybe even beyond generalization — while being applicable to real world models?

Towards an infinite width limit analysis of SSL models through dynamics and kernel models. As discussed previously SSL models have become a central backbone of state-the-art models [Rad+18; Dev+19], and therefore deriving theoretical guarantees become increasingly more important. However existing results are mostly restricted to analyzing the generalization properties of such models [VZ19; STH18; GJJ20; LUZ21; OS20b; OS20a]. While this is an important aspect as outlined in Chapter 2 it does not completely characterize the capabilities and limitations of SSL models. A possible approach to answer questions beyond the generalization error is to build on Chapter 3 & 4 by deriving an NTK formulation for SSL models.

As outlined in Chapter 4 the first step to adapt the general idea from the supervised setting [JGH18a] is to note that the analysis of dynamics has to be performed under orthogonal constraints. We take the first step in the linear setting in Chapter 3 however an extension to the non-linear setting would make the considered models more relevant. From there as noted previously we have to rely on the kernel formulation of the SSL model to obtain the embedding for a given datapoint, which we provide in Chapter 4.

NTK formulations in the supervised setting have been shown to be a useful tool [GPK22] for theoretical analysis. We conjecture that deriving an NTK formulation in the SSL setting would allow us to provide similar results for the unlabelled data setting as well. Let us outline a few interesting directions:

1. [CGW21] showed that NTKs offer an efficient tool for architecture search in the supervised setting as the NTK provides a simple closed-form solution that can be analyzed without having to consider an expensive hyperparameter search and optimization. As those properties are not unique to the supervised setting we suggest a similar approach can be applied to the SSL setting as well.

   While this translation from the supervised to the SSL setting here is quite direct, other quantities are more ambiguous.

2. Robustness in the supervised setting has been studied through the lens of NTK [TK22] and is usually defined with regard to changes in label prediction. On a high level one can often consider robustness as the question of how much the inputs can be changed so that the prediction of the classifier changes. However if only learning of representations, as is the case in SSL, without a downstream task it is less clear what it means for the algorithm to be robust. An NTK formation could be a suitable tool to investigate this question, again due to the closed-form expression that allows for an easier analysis.

3. Beyond the above questions the representation learning setting also opens up new questions beyond the ones we consider in the supervised setting. The most obvious question here is the characterization of the learned representations as this would provide a better understanding of the model and can allow us to derive more precise guarantees for downstream tasks. We take a first step towards such a result by analyzing the linear dynamics in Theorem 6 however a similar result for non-linear models and higher dimensional embeddings would be more informative on real-world models.

4. In Chapter 3 we performed our analysis for a specific one hidden layer neural network architecture. However in the supervised setting NTK have shown to be very versatile and able to model a large number of network architectures such as deep network [JGH18a], convolutional networks [Aro+19b] or Graph neural networks [Du+19a]. Therefore an NTK formulation in the SSL setting would allow to analyze point 2 & 3 for a broad range of real world models.

Formal characterization of embeddings from non-linear AEs. While in the previous section we focused on the SSL setup we can apply the same general idea to other models in the unlabelled data setting as well. Let us therefore consider the question of an exact characterization of the embedding obtained by non-linear AEs. A first step would be to understand the learned representation in the Kernel setting by analyzing the solution learned by the Kernel AE in Chapter 5. While the derived formulation does not provide a closed form and therefore no direct characterization we still obtain iterative updated rules in terms of the Kernel in the feature space. A detailed analysis of this iterative process could provide us with a precise characterization of the learned latent representation in the Kernel AE setting. A second step could be the analysis or the infinite width limit. This comes with several technical challenges as it has been shown in [LZB20] that for networks with bottlenecks[1], such as is the case for an AE, the corresponding Kernel expression does not stay constant during training. Without the kernel staying constant, we can not directly apply the approach presented by [JGH18a] which relies on computing expectations under the distribution of the weights at initialization. However, as we are not actually interested in the full machine output but only in the learned embedding and analysis of only the encoder through an infinite width limit might offer a way to circumvent the problem of bottleneck layers and offer a way to precisely characterize embeddings obtained by non-linear AEs.

8.2 Algorithmic Advances and Extensions

The main focus of the thesis is in the development of theoretical foundation but this also goes hand in hand with the development of new algorithms. Therefore we use this final section to further outline possible future advances.

Learning cluster specific representations. We started out from a theoretical characterization of the latent representations — in this case with regards to preserving clustering structures — the resulting model also proves very effective in practice. As such more application-oriented future extensions promise further improvements.

Practically addressing the main downside of the proposed method, that it scales in time and parameters with the cluster number κ is an important aspect. A possible solution to this problem is to not tensorize the full architecture but only the mapping to and from the latent representation. The intuition behind this is that early layers learn high-level features and final layers the combination of such features, therefore one can still learn a meaningful full cluster-specific representation by only tensorizing the final mapping layer.

Secondly one can ask if the proposed idea of learning cluster-specific representations can be extended to models beyond the AE setup. The high-level idea could be extended to any embedding approach such as SSL models, where now instead of having an encoder and decoder for each cluster we consider simply an encoding function for each cluster. Both outlined directions are part of ongoing research and preliminary results show to be promising.

Near-Optimal comparison based clustering. From a theoretical perspective recall that in the ordinal setup and in the active setting $\Omega(n \ln n)$ number of comparisons is sufficient for exact recovery and our proposed algorithm shows that $\Omega(n(\ln n)^2)$ passive triplets or quadruplets are sufficient for exact recovery and therefore is near-optimal in the number of required data points. This outlines that the room for theoretical improvements is while existing, limited. At the same time on the algorithmic side, advances are still possible.

While theoretically sound, our proposed algorithm relies on computing the similarity measure from the given comparison and then solving the SDP. Therefore a possible future

[1]Bottleneck in this setting means that there exists a finite width layer *between* layers that go to infinity.

direction could be to approach the same problem but from a neural network perspective[2]. While, as we also saw in this thesis, the analysis of neural network models is notoriously complex, and therefore obtaining such precise guarantees as derived in Chapter 7 would be hard. Nevertheless a possible route would be to aim for proving approximate instead of exact recovery and trade off preciseness of recovery for faster computation.

[2]In the setting of learning embeddings from ordinal data this has been shown to be a practically useful approach. See [Van+23] for a comprehensive overview.

Appendix A

Proofs

A.1 Proofs for Chapter 2

A.1.1 Proof for Proposition 1

For the hypothesis class over all **linear GNNs**, that is $\psi(x) := x$, with binary outputs, the VC Dimension is given by

$$\mathrm{VCdim}\big(\operatorname{sign} \circ \mathcal{F}_{\mathcal{G}}^{\text{linear}}\big) = \min\left\{d, \operatorname{rank}\big(S\big), \min_{j\in[J-1]} \{h_j\}\right\}.$$

Similarly, the VC Dimension for the hypothesis class of GNNs with **ReLU non-linearities** and binary outputs, can be bounded as $\mathrm{VCdim}\big(\operatorname{sign} \circ \mathcal{F}_{\mathcal{G}}^{\text{ReLU}}\big) \leq \min\{\operatorname{rank}(S), h_{J-1}\}$.

Using the above bounds, it follows that, for any $\delta \in (0,1)$, the generalisation error for any $f \in \operatorname{sign} \circ \mathcal{F}_{\mathcal{G}}$ satisfies, with probability $1-\delta$,

$$\mathcal{L}_n(h) - \widehat{\mathcal{L}}_m(h) \leq \sqrt{\frac{8}{m}\left(\min\{\operatorname{rank}(S), h_{J-1}\} \cdot \ln(em) + \ln\left(\frac{4}{\delta}\right)\right)}. \tag{A.1}$$

Proof. For this proof we will need the following know result on the VC-dimension of linearly independent points:

Theorem 18 ([Bur98]). *Consider some set of m points in $\mathbb{R}^n$. Choose any one of the points as origin. Then the m points can be shattered by oriented hyperplanes if and only if the position vectors of the remaining points are linearly independent.*

For deriving $\mathrm{VCdim}\big(\operatorname{sign} \circ \mathcal{F}_{\mathcal{G}}^{\text{linear}}\big)$ we start with the VC-dimension of the final layer with $\mathcal{F}_B^{\text{sign}} = \left\{f_B^{\text{sign}}(x) := \operatorname{sign}(Bw) \ : \ w \in \mathbb{R}^m\right\}$ over an arbitrary matrix $B \in \mathbb{R}^{n\times m}$, where B is later substituted for the linear network. Let $\operatorname{rank}(B) = r$ then we show that there is $c \subset [n], |c| = r$ s.t. $\forall\ b \in \{\pm 1\}^r$ and $h_B^{\text{sign}}(c) = \{\pm 1\}^c$. Using SVD we decompose

$B = Um\Lambda V^\top$ and define $z_1^\top, \cdots, z_m^\top \in \mathbb{R}^k$ as the rows of U. Using this we rewrite:

$$Bw = \begin{bmatrix} z_1^\top \\ \vdots \\ z_d^\top \end{bmatrix} \underbrace{m\Lambda V^\top w}_{=a \in \mathbb{R}^d} = \begin{bmatrix} z_1^\top a \\ \vdots \\ z_d^\top a \end{bmatrix}$$

Rewrite $\mathcal{F}_B^{\text{sign}}$ as $\mathcal{F}^{\text{sign}} = \{f_a(z) = \text{sign}(a^\top z)\}$. Since $\mathcal{F}^{\text{sign}}$ lies in the class of all homogenious linear classifiers in r dimensions and from orthonormal condition on z it follows that span $(\{z_1, \cdots z_n\}) = \mathbb{R}^r$. Using this observation as well as results on the VC-dimension of linear independent pointsets [Bur98] it follows that $\text{VCdim}(\mathcal{F}_B^{\text{sign}}) = \text{VCdim}(\mathcal{F}^{\text{sign}}) = r$. Substituting B with the linear network and using that for two matrixes B' and B: $\text{rank}(B'B) = \min(\text{rank}(B'), \text{rank}(B))$ gives $\text{rank}\big(B\big) := \text{rank}\big(SH^{(p)}\big) = \text{rank}\big(S \cdots SXW^{(1)} \cdots W^{(p-1)}\big)$ as the final result.

For extending to the non-linear setting we first note that we can not make a general statement on the rank of a matrix after applying a non-linearity. That is for some matrix M and non-linearity $\text{ReLU}(\cdot)$ we have no order relation between $\text{rank}(M)$ and $\text{rank}(\text{ReLU}(M))$. This can be checked by a simple counterexample. Therefore the above presented proof does not extend to the hidden layer size but since the last layer is linear the dependency on S persists. We define the hypothesis class over all linear GNNs where all but the last activation function are linear $\psi_j(x) := x \ \forall j \in [J-1]$ and $\psi_J(x) := \text{sign}(x)$ as $\mathcal{F}_\mathcal{G}^{\text{sign},\mathbb{I}} = \left\{f_\mathcal{G}^{\text{sign},\mathbb{I}}(X)\right\}$ and recall that layer j has dimension h_j. Then the VC-Dimension is given by the minimum of the rank of the adjacency matrix, the dimension of the features and the minimum hidden layer size, that is,

$$\text{VCdim}\left(\mathcal{F}_\mathcal{G}^{\text{sign},\mathbb{I}}\right) = \min\left\{d, \text{rank}\big(S\big), \min_{j \in [J-1]} \{h_j\}\right\}. \tag{A.2}$$

Therefore consider the hypothesis class GNNs with of non-linearities $\psi_j(x) := \text{ReLU}(x) \ \forall j \in [J-1]$ and $\psi_J(x) := \text{sign}(x)$: $\mathcal{F}_\mathcal{G}^{\text{sign,ReLU}} = \left\{f_\mathcal{G}^{\text{sign,ReLU}}(X)\right\}$ and again compute the VC-Dimension, similar to the proof shown above, we can note that we lose information on the hidden layers (and therefore also on d) and the bound becomes

$$\text{VCdim}\left(\mathcal{F}_\mathcal{G}^{\text{sign,ReLU}}\right) \leq \min\left\{\text{rank}(S), h_{p-1}\right\}, \tag{A.3}$$

that is, it still depends on the rank of S but only on the last hidden layer dimension.

Following defined we use the a standard result for generalisation e.g. in [SSBD14]. For $\delta \in (0,1)$ any $h \in \mathcal{F}_\mathcal{G}$ satisfies

$$\mathcal{L}_n(h) - \widehat{\mathcal{L}}_m(h) \leq \sqrt{\frac{8\Big(\text{VCdim}\left(\mathcal{F}_\mathcal{G}\right)\ln\left(\frac{em}{\text{VCdim}(\mathcal{F}_\mathcal{G})}\right) + \ln\left(\frac{4}{\delta}\right)\Big)}{m}} \tag{A.4}$$

with probability $1-\delta$. Applying (A.2) and (A.3) to (A.4) gives the final bound. $\square$

A.1.2 Proof of Theorem 1

Consider $\mathcal{F}_{\mathcal{G}}^{\psi,\beta,\omega} \subseteq \mathcal{F}_{\mathcal{G}}^{\psi}$ such that the trainable parameters satisfy $\|b_j\|_1 \leq \beta$ and $\|W_j\|_\infty \leq \omega$ for every $j \in [J]$. The transductive Rademacher complexity (TRC) of the restricted hypothesis class is bounded as

$$\mathfrak{R}_{m,n}(\mathcal{F}_{\mathcal{G}}^{\psi,\beta,\omega}) \leq \frac{c_1 n^2}{m(n-m)} \left(\sum_{j=0}^{J-1} c_2^j \|S\|_\infty^j \right) + c_3 c_2^J \|S\|_\infty^J \|SX\|_{2\to\infty} \sqrt{\log(n)}\,, \tag{A.5}$$

where $c_1 := 2L_\psi \beta$, $c_2 := 2L_\psi \omega$, $c_3 := L_\psi \omega \sqrt{2/d}$ and L_ψ is Lipschitz constant for activation ψ.

The bound on TRC leads to a generalisation error bound following [EYP09]. For any $\delta \in (0,1)$, the generalisation error for any $h \in \mathcal{F}_{\mathcal{G}}^{\psi,\beta,\omega}$ satisfies

$$\mathcal{L}_u(h) - \widehat{\mathcal{L}}_m(h) \;\leq\; \mathfrak{R}_{m,n}(\mathcal{F}_{\mathcal{G}}^{\psi,\beta,\omega}) + c_4 \frac{n\sqrt{\min\{m, n-m\}}}{m(n-m)} + c_5 \sqrt{\frac{n}{m(n-m)} \ln\left(\frac{1}{\delta}\right)} \tag{A.6}$$

with probability $1-\delta$, where c_4, c_5 are absolute constants such that $c_4 < 5.05$ and $c_5 < 0.8$.

Proof. To derive the TRC we start with the following propositions describing the recursive TRC for a GNN neuron that is applied $J-1$ times for all but the first layer.

Proposition 6 (Recursive TRC of one GNN neuron)**.** *Consider* $g_{j+1} := \psi\left(b_j + S g_j(H) W_j\right)$, $j \in [J]$*. Now we define the function class over one neuron as*

$$\mathcal{F}_{\mathcal{G}}^{\psi} := \left\{ h_{\mathcal{G}}^{\psi}(H) = \psi\left(b_i + \sum_{l}^{h_j} W_{lj} \sum_{t}^{n} S_{it} g(H)_{lr} \right) \;\middle|\; g \in \mathcal{F}', \|b_i\|_1 \leq \beta \right\}$$

where $\mathcal{F}'$ *is the class of* $\mathbb{R}^{n\times h_j} \to \mathbb{R}$*, including the zero function. Then with* $W_{\cdot j} := \left[W_{1r}, \cdots, W_{h_j r}\right]^\top$*:*

$$\mathfrak{R}_{m,n}(\mathcal{F}_{\mathcal{G}}^{\psi}) \leq 2L_\psi \left(\beta Q(n) + \|S\|_\infty \|W_{\cdot r}\|_1 \mathfrak{R}_{m,n}(\mathcal{F}')\right)$$

After the recursive application we end up with a formulation of all layers and a dependency on the TRC of the first layer. Therefore we then use the following proposition to finish the proof.

Proposition 7 (Bound on TRC, first layer)**.** *Define the hypothesis class over the function* *of the first layer g_0 as:*

$$\mathcal{F}_{\mathcal{G}}^{\psi} := \left\{ h_{\mathcal{G}}^{\psi}(X) = \psi\left(b + SXW_1\right) \;\middle|\; \|b\|_1 \leq \beta \right\}$$

then the TRC is give by

$$\mathfrak{R}_{m,n}(\mathcal{F}_{\mathcal{G}}^{\psi}) \leq L_{\psi}\left(\beta Q(n)2 + Q\left\|W_1\right\|_{\infty}\left\|SX\right\|_{2\to\infty}\sqrt{\frac{2\log(n)}{d}}\right)$$

Then by combining the above results we obtain Theorem 1 as follows: Consider $\mathcal{F}_{\mathcal{G}}^{\psi,\beta,\omega} \subseteq \mathcal{F}_{\mathcal{G}}^{\psi}$ such that the trainable parameters satisfy $\|b_j\|_1 \leq \beta$ and $\|W_j\|_{\infty} \leq \omega$ for every $j \in [J]$. The transductive Randemacher complexity (TRC) of the restricted hypothesis class is bounded as $\mathfrak{R}_{m,n}(\mathcal{F}_{\mathcal{G}}^{\psi,\beta,\omega}) \leq \frac{c_1 n^2}{m(n-m)}\left(\sum_{k=0}^{J-1} c_2^j \|S\|_{\infty}^j\right) + c_3 c_2^J \|S\|_{\infty}^J \|SX\|_{2\to\infty}\sqrt{\log(n)}$, where $c_1 := 2L_{\psi}\beta$, $c_2 := 2L_{\psi}\omega$, $c_3 := L_{\psi}\omega\sqrt{2/d}$ and L_{ψ} is Lipschitz constant for activation ψ. $\square$

Before proofing the above propositions we proof some preliminary lemmas for TRC calculus that we will use in the later steps.

Lemma 2 (Scalar multiplication)**.** *Let $A \subseteq \mathbb{R}^n$, a scalar $c \in \mathbb{R}$ and a vector $a_0 \in \mathbb{R}^n$ then*

$$\mathfrak{R}_{m,n}\left(\{ca + a_0 : a \in A\}\right) \leq |c|\mathfrak{R}_{m,n}(A)$$

Proof. Directly by construction. $\square$

Lemma 3 (Addition)**.** *Let $A \subseteq \mathbb{R}^n, B \subseteq \mathbb{R}^n$ then*

$$\mathfrak{R}_{m,n}(A + B) = \mathfrak{R}_{m,n}(A) + \mathfrak{R}_{m,n}(B)$$

Proof. By construction and linearity of expectation. $\square$

Lemma 4 (Convex hull)**.** *Let $A \subseteq \mathbb{R}^n$*
and $A' = \left\{\sum_{j=1}^N \alpha_j a^{(j)} \;\middle|\; N \in \mathbb{N},\ \forall j,\ a^{(j)} \in A, \alpha_j \geq 0, \|\alpha\|_1 = 1\right\}$ then

$$\mathfrak{R}_{m,n}(A) = \mathfrak{R}_{m,n}(A').$$

Proof. The proof follows similar to the one for inductive Rademacher complexity (e.g. [SSBD14]). We first note that for any vector v the following holds:

$$\sup_{\alpha \geq 0 : \|\alpha\|_1 = 1} \sum_{j=1}^N \alpha_j v_j = \max_j v_j$$

Then:

$$\begin{aligned}\mathfrak{R}_{m,n}(A') &= Q\mathop{\mathbb{E}}_{\sigma}\left[\sup_{\alpha\geq 0:\|\alpha\|_1=1}\ \sup_{\{a^{(i)}\}_{i=1}^{N}}\sum_{i=1}^{n}\sigma_i\sum_{j=1}^{N}\alpha_j a_i^{(j)}\right]\\ &= Q\mathop{\mathbb{E}}_{\sigma}\left[\sup_{\alpha\geq 0:\|\alpha\|_1=1}\sum_{j=1}^{N}\alpha_j\sup_{a^{(j)}}\sum_{i=1}^{n}\sigma_i a_i^{(j)}\right]\\ &= Q\mathop{\mathbb{E}}_{\sigma}\left[\sup_{a\in A}\sum_{i=1}^{n}\sigma_i a_i\right]\\ &= \mathfrak{R}_{m,n}(A)\end{aligned}$$

which concludes the proof. □

Lemma 5 (Contraction [EYP09])**.** *Let $A \subseteq \mathbb{R}^n$ be a set of vectors. Let $f(\,\cdot\,)$ and $g(\,\cdot\,)$ be real-value functions. Let $\boldsymbol{\sigma} = \{\sigma_i\}_{i=1}^n$ be Rademacher variables as defined in Definition 2. If for all $1 \leq i \leq n$ and any $a, a' \in A$, $|f(a_i) - f(a'_i)| \leq |g(a_i) - g(a'_i)|$ then*

$$\mathop{\mathbb{E}}_{\sigma}\left[\sum_{i=1}^{n}\sigma_i f(a_i)\right] \leq \mathop{\mathbb{E}}_{\sigma}\left[\sum_{i=1}^{n}\sigma_i g(a_i)\right]$$

Extending this to Lipschitz continues functions. Let $v(\,\cdot\,)$ be a L_v-Lipschitz continues function such that $|v(f(a_i)) - v(f(a'_i))| \leq \frac{1}{L_v}|f(a_i) - f(a'_i)|$. Now let the corresponding hypothesis classes be $\mathcal{F} := \{f(\cdot)\}, \mathcal{V} := \{v(f(\cdot))\}$ then

$$\mathfrak{R}_{m,n}(\mathcal{V}) \leq \frac{1}{L_v}\mathfrak{R}_{m,n}(\mathcal{H}) \tag{A.7}$$

Lemma 6 (Cardinality of finite sets)**.** *Let $A = \{a_1, \cdots, a_n\}$ be a finite set of vectors in $\mathbb{R}^d$ and let $\overline{a} = \frac{1}{n}\sum_{i=1}^{n} a_i$ then*

$$\mathfrak{R}_{m,n}(A) \leq \max_{a\in A}\|a - \overline{a}\|_2\sqrt{\frac{2\log(n)}{d}}$$

Proof. The proof follows the general idea of the proof for *Massarts Lemma* (see e.g. [SSBD14]).

From Lemma 4 wlog. let $\overline{a} = 0$. Let $\lambda > 0$ and $A' = \{\lambda a_1, \cdots, \lambda a_n\}$. Therefore

$$\begin{aligned}
\frac{1}{Q}\mathfrak{R}_{m,n}(A') &= \mathop{\mathbb{E}}_{\sigma}\left[\max_{a\in A'}\langle\sigma, a\rangle\right] \\
&= \mathop{\mathbb{E}}_{\sigma}\left[\log\left(\max_{a\in A'}\exp\left(\langle\sigma, a\rangle\right)\right)\right] \\
&\leq \mathop{\mathbb{E}}_{\sigma}\left[\log\left(\sum_{a\in A'}\exp\left(\langle\sigma, a\rangle\right)\right)\right] && \text{Jensen inequality} \\
&\leq \log\left(\mathop{\mathbb{E}}_{\sigma}\left[\sum_{a\in A'}\exp\left(\langle\sigma, a\rangle\right)\right]\right) && \sigma_i \text{ is i.i.d.} \\
&= \log\left(\sum_{a\in A'}\prod_{i=1}\mathop{\mathbb{E}}_{\sigma_i}\left[\exp(\sigma_i a_i)\right]\right)
\end{aligned}$$

Bound $\mathop{\mathbb{E}}_{\sigma_i}\left[\exp(\sigma_i a_i)\right]$:

$$\begin{aligned}
\mathop{\mathbb{E}}_{\sigma_i}\left[\exp(\sigma_i a_i)\right] &= p\exp(1a_i) + (1-2p)\exp(0a_i) + p\exp(-1a_i) && \text{by definition of } \sigma_i \\
&= (1-2p) + p\sum_{i=0}^{\infty}\frac{(-a)^i + a^i}{i!} \\
&\leq \frac{1}{2}\sum_{i=0}^{\infty}\frac{(-a)^i + a^i}{i!} && \text{as } p \leq \tfrac{1}{2}\text{. Equality for } p = \tfrac{1}{2}. \\
&= \frac{\exp(a_i) + \exp(-a_i)}{2} \\
&\leq \exp\left(\frac{a_i^2}{2}\right)
\end{aligned}$$

Because

$$\frac{\exp(a) + \exp(-a)}{2} = \sum_{n=0}^{\infty}\frac{a^{2n}}{(2n)!} \leq \sum_{2^n n!}^{\infty} = \frac{a^{2n}}{2^n n!}\exp\left(\frac{a^2}{2}\right)$$

and $(2n)! \geq 2^n n!\ \forall n \geq 0$. Going back we now get:

$$\begin{aligned}
\frac{1}{Q}\mathfrak{R}_{m,n}(A') &\leq \log\left(\sum_{a\in A'}\prod_{i=1}\mathop{\mathbb{E}}_{\sigma_i}\left[\exp(\sigma_i a_i)\right]\right) \\
&\leq \log\left(\sum_{a\in A'}\prod_{i=1}\exp\left(\frac{a_i^2}{2}\right)\right) \\
&= \log\left(\sum_{a\in A'}\exp\left(\frac{\|a\|^2}{2}\right)\right) \\
&\leq \log\left(|A'|\max_{a\in A'}\exp\left(\frac{\|a\|^2}{2}\right)\right) \\
&= \log\left(|A'|\right) + \max_{a\in A'}\left(\frac{\|a\|^2}{2}\right)
\end{aligned}$$

By construction $\mathfrak{R}_{m,n}(A) = \frac{1}{\lambda}\mathfrak{R}_{m,n}(A')$ and therefore $\mathfrak{R}_{m,n} \leq \frac{1}{\lambda d}\left(\log(|A|) + \lambda^2\max_{a\in A'}\left(\frac{\|a\|^2}{2}\right)\right)$.

By setting $\lambda = \sqrt{\frac{2\log(|A|)}{\max_{a\in A'}\|a\|^2}}$ and rearranging:

$$\mathfrak{R}_{m,n}(A) \leq \max_{a\in A}\|a-\bar{a}\|_2\sqrt{\frac{2\log(n)}{d}}$$

which concludes the proof. □

Proof of Proposition 6

We start from the general GNN setup as defined as follows: Consider a class of GNNs defined over K layers, with dimension of layer $k\in[K]$ being d_k and $S\in\mathbb{R}^{n\times n}$ the diffusion operator. Let ϕ,ψ be L_ϕ, L_ψ-Lipschitz pointwise functions. Define:

$$\begin{aligned} g_{k+1} &:= \phi\left(b_k + Sg_k\left(H\right)W_k\right), \\ g_0 &:= X \end{aligned}$$

and the hypothesis class over all such functions as

$$\mathcal{H}_{\mathcal{G}}^{\phi,\psi} := \left\{h_{\mathcal{G}}^{\phi,\psi}(X) = \psi\left(g_K\circ\cdots\circ g_0\right)\right\}.$$

From there we derive a recursive TRC bound depending on the previous layer.

Consider $g_{k+1} := \phi\left(b_k + Sg_k\left(H\right)W_k\right)$, $k\in\{1,\cdots,K\}$. Now we define the function class over one neuron as

$$\mathcal{H}_{\mathcal{G}}^{\phi} := \left\{h_{\mathcal{G}}^{\phi}(H) = \phi\left(b_i + \sum_{l}^{d_k}W_{lj}\sum_{t}^{n}S_{it}g(H)_{lj}\right) \;\middle|\; g\in\mathcal{F}, \|b\|_1\leq\beta\right\}$$

where $\mathcal{F}$ is the class of $\mathbb{R}^{n\times d_k}\to\mathbb{R}$, including the zero function. Then with $W_{\cdot j} := [W_{1j},\cdots,W_{d_kj}]^\top$:

$$\mathfrak{R}_{m,n}(\mathcal{H}_{\mathcal{G}}^{\phi}) \leq 2L_\phi\left(\beta Q(n) + \|S\|_\infty\|W_{\cdot j}\|_1\mathfrak{R}_{m,n}(\mathcal{F})\right)$$

Proof. By Lemma 5 and Lemma 3 we get

$$\mathfrak{R}_{m,n}(\mathcal{H}_{\mathcal{G}}^{\phi}) \leq L_\phi\left(\mathfrak{R}_{m,n}(\mathcal{H}_{lin}) + \mathfrak{R}_{m,n}(\mathcal{H}_{bias})\right)$$

where

$$\begin{aligned} \mathcal{H}_{lin} &:= \left\{h_{lin}(H) = \sum_{l}^{d_k}W_{lj}\sum_{t}^{n}S_{it}g(H)_{lj} \;\middle|\; g\in\mathcal{F}, \|W_{\cdot j}\|_1\leq\omega\right\} \\ \mathcal{H}_{bias} &:= \{h_{bias}(H) = b \mid |b|\leq\beta\} \end{aligned}$$

with $W_{\cdot j} := [W_{1j},\cdots,W_{d_kj}]^\top$. We will now bound the Rademacher complexity for both the above function classes individually and start with bounding the linear term $\mathfrak{R}_{m,n}(\mathcal{H}_{lin})$. For readability we define $g_{lj} := g(H)_{lj}$ and can bound each entry of the output of one layer as follows

$$\begin{aligned}
H_{ij} &= \sum_{l}^{d_k} W_{lj} \sum_{t}^{n} S_{it} g_{lj} \\
&= \underbrace{W_{1j}S_{i1}g_{1j} + \cdots + W_{1j}S_{in}g_{1j}}_{W_{1j}g_{1j}(\sum_t^n S_{it})} + \underbrace{W_{2j}S_{i1}g_{2j} + \cdots + W_{2j}S_{in}g_{2j}}_{W_{2j}g_{2j}(\sum_t^n S_{it})} + \cdots \\
&\leq \|S\|_\infty \left(\sum_{l}^{d_k} W_{1j} g_{1j} \right)
\end{aligned}$$

where in the last step we use $\sum_t^n S_{it} \leq \|S\|_\infty$. Now we define

$$\begin{aligned}
\widetilde{\mathcal{H}}_{lin} &:= \left\{ h_{lin}(H) = \sum_{l}^{d_k} W_{lj} g(H)_{lj} \;\middle|\; g \in \mathcal{F}, \|W_{\cdot j}\|_1 \leq \omega \right\} \\
\widetilde{\mathcal{H}}'_{lin} &:= \left\{ h_{lin}(H) = \sum_{l}^{d_k} W_{lj} g(H)_{lj} \;\middle|\; g \in \mathcal{F}, \|W_{\cdot j}\|_1 = \omega \right\}
\end{aligned}$$

and since $\|S\|_\infty$ is constant we get by Lemma 2

$$\mathfrak{R}_{m,n}(\mathcal{H}_{lin}) \leq \|S\|_\infty \mathfrak{R}_{m,n}(\widetilde{\mathcal{H}}_{lin}).$$

To further bound $\mathfrak{R}_{m,n}(\widetilde{\mathcal{H}}_{lin})$ we can a similar process then for standard deep neural networks with slight deviation on the indexing of the weight matrix.

Let Hull $(\cdot)$ be a convex hull. In the first step we show that

$$\mathfrak{R}_{m,n}(\widetilde{\mathcal{H}}_{lin}) = \omega \mathfrak{R}_{m,n}(\text{Hull}\,(\mathcal{F} - \mathcal{F}))$$

where $\mathcal{F} - \mathcal{F} := \{f - f',\ f \in \mathcal{F}, f' \in \mathcal{F}\}$. Note that the maximum over all function over W_{il} with constraint $\|W_{\cdot j}\|_1 \leq \omega$ is achieved for $\|W_{\cdot j}\|_1 = \omega$ then

$$\mathfrak{R}_{m,n}(\widetilde{\mathcal{H}}_{lin}) = \mathfrak{R}_{m,n}(\widetilde{\mathcal{H}}'_{lin})$$

Let 0 be the zero function. Then for $\|W_{\cdot j}\|_1 = 1$:

$$\sum_{l} W_{lj} g_{lj} = \sum_{l: W_{lj} \geq 0} W_{lj}(g_{lj} - 0) + \sum_{l: W_{lj} < 0} |W_{lj}|(0 - g_{lj})$$

which is $\mathrm{Hull}(\mathcal{F} - \mathcal{F})$. Combining the above results we get:

$$\begin{aligned}
\mathfrak{R}_{m,n}(\mathcal{H}_{lin}) &\leq \|S\|_\infty \omega \mathfrak{R}_{m,n}\left(\widetilde{\mathcal{H}}_{lin}\right) \\
&= \|S\|_\infty \omega \mathfrak{R}_{m,n}\left(\mathrm{Hull}\left(\mathcal{F} - \mathcal{F}\right)\right) \\
&= \|S\|_\infty \omega \mathfrak{R}_{m,n}\left(\mathcal{F} - \mathcal{F}\right) \\
&= \|S\|_\infty \omega \left(\mathfrak{R}_{m,n}\left(\mathcal{F}\right) + \mathfrak{R}_{m,n}\left(-\mathcal{F}\right)\right) && \text{Lemma 3} \\
&= 2\|S\|_\infty \omega \mathfrak{R}_{m,n}\left(\mathcal{F}\right) && \text{Lemma 2}
\end{aligned}$$

which concludes this part of the proof.

Let us now bound the bias term $\mathfrak{R}_{m,n}(\mathcal{H}_{bias})$ which can be done by standard arguments on bounding Rademacher complexity terms.

$$\begin{aligned}
\mathfrak{R}_{m,n}(\mathcal{H}_{bias}) &= Q\mathop{\mathbb{E}}_{\sigma}\left[\sup_{b:|b|\leq\beta} b\sum_{i=1}^{n}\sigma_i\right] \leq Q\mathop{\mathbb{E}}_{\sigma}\left[\sup_{b:|b|\leq\beta} |b|\left|\sum_{i=1}^{n}\sigma_i]\right|\right] \\
&= \beta Q\mathop{\mathbb{E}}_{\sigma}\left[\left|\sum_{i=1}^{n}\sigma_i\right|\right] \leq \beta Q\mathop{\mathbb{E}}_{\sigma}\left[\sum_{i=1}^{n}|\sigma_i|\right] \\
&\leq \beta Q\sum_{i=1}^{n}\mathop{\mathbb{E}}_{\sigma}\left[|\sigma_i|\right] \leq \beta Q\sum_{i=1}^{n}2p \leq \beta Q(n)2p \leq \beta Q(n)2
\end{aligned}$$

This concludes this part of the proof.

Combining the two bounds gives:

$$\mathfrak{R}_{m,n}(\mathcal{H}^{\phi}_{\mathcal{G}}) \leq 2L_\phi\left(\beta Q(n) + \|S\|_\infty \|W_{\cdot j}\|_1 \mathfrak{R}_{m,n}(\mathcal{F})\right)$$

concluding the proof of Proposition 6. □

Proof of Proposition 7

Proof. First similar to Proposition 6 we use Lemma 3 and Lemma 5

$$\mathfrak{R}_{m,n}(\mathcal{H}^{\phi}_{\mathcal{G}}) \leq L_\phi\left(\mathfrak{R}_{m,n}(\mathcal{H}_{lin}) + \mathfrak{R}_{m,n}(\mathcal{H}_{bias})\right)$$

As before $\mathfrak{R}_{m,n}(\mathcal{H}_{bias}) \leq \beta Q(n)2p$. In this case we define the linear term as

$$\mathcal{H}_{lin} := \left\{h_{lin}(X) = SXW\right\}.$$

Bounding the TRC of $\mathcal{H}_{lin}$

$$\begin{aligned}\mathfrak{R}_{m,n}(\mathcal{H}_{lin}) &= Q\mathop{\mathbb{E}}_{\sigma}\left[\sup_{W:\|W\|_\infty\leq\omega} \sigma^\top SXW\right] \\ &\leq Q\left\|W\right\|_\infty \mathop{\mathbb{E}}_{\sigma}\left[\left\|\sigma^\top SX\right\|_\infty\right]\end{aligned}$$

To bound $\mathop{\mathbb{E}}_{\sigma}\left[\left\|\sigma^\top SX\right\|_\infty\right]$ we define $t_i = (x_{1j}, \dots, x_{nj})^\top$ and $T = \{t_1, \dots, t_n\}$, $T_- = \{-t_1, \dots, -t_n\}$. Therefore

$$\begin{aligned}\mathop{\mathbb{E}}_{\sigma}\left[\left\|\sigma^\top SX\right\|_\infty\right] &\leq \mathop{\mathbb{E}}_{\sigma}\left[\max_{t\in T}|\sigma^\top St|\right] \\ &= \mathop{\mathbb{E}}_{\sigma}\left[\max_{t\in T\cup T_-}\sigma^\top St\right] \\ &\leq \max_{t\in T\cup T_-}\|St\|_2\sqrt{\frac{2\log(n)}{d}} && \text{Lemma 6} \\ &= \|St\|_{2\to\infty}\sqrt{\frac{2\log(n)}{d}}\end{aligned}$$

Combining with the above results gives

$$\mathfrak{R}_{m,n}(\mathcal{H}_{lin}) \leq Q\left\|W\right\|_\infty \|St\|_{2\to\infty}\sqrt{\frac{2\log(n)}{d}}.$$

Taking the bound on the bias term into considerations gives the final bound and concludes the proof of Proposition 7. □

A.1.3 Proof of Theorem 2

Let c_1, c_2 and c_3 as defined in Theorem 1 and $\Gamma := |y^\top v|$. Let $c_6 := (1 + o(1))$, $c_7 := (1 + jo(1))$, $c_8 := (1 + Jo(1))$. Assuming $p, q \gg \frac{(lnn)^2}{n}$ we can bound the expected TRC for A as defined in (2.8) and X as defined in (2.7) as follows:

Case 1, Degree normalized: $S = S_{\mathbf{nor}}$

$$\begin{aligned}\mathop{\mathbb{E}}_{X,A}\left[\mathfrak{R}_{m,n}(\mathcal{F}_{\mathcal{G}}^{\psi,\beta,\omega})\right] \leq \frac{c_1 n^2}{m(n-m)}\left(\sum_{j=0}^{J-1} c_7 c_2^j \left(\frac{p}{q}\right)^{\frac{j}{2}}\right) + c_8 c_3 c_2^J\left(\frac{p}{q}\right)^{\frac{J}{2}}\sqrt{\ln(n)}\times \\ \left(c_6\left\|\mu\right\|_\infty \frac{1+\left(\frac{p-q}{2}\right)^2\Gamma^2}{\left(\frac{p+q}{2}\right)^2} + c_6\sqrt{\frac{\ln(n)}{q}}\left\|\mu\right\|_\infty + c_6\sqrt{\frac{\sigma(1+2\ln(d))}{q}}\right) \qquad \text{(A.8)}\end{aligned}$$

Case 2, Self Loop: $S = S_{\mathbf{loop}}$

$$\underset{X,A}{\mathbb{E}}\left[\mathfrak{R}_{m,n}(\mathcal{F}_{\mathcal{G}}^{\psi,\beta,\omega})\right] \leq \frac{c_1 n^2}{m(n-m)}\left(\sum_{j=0}^{J-1} c_7 c_2^j (np)^j\right) + c_8 c_3 c_2^J (np)^J \sqrt{\ln(n)} \times$$
$$\left(c_6 \|\mu\|_\infty n\left(1+\left(\frac{p-q}{2}\right)^2\Gamma^2\right) + n\sqrt{\frac{p+q}{2}}\|\mu\|_\infty + c_6 n\sqrt{p}\sigma\sqrt{1+2\ln(d)}\right) \tag{A.9}$$

Proof. From the above bound we can note that to derive the TRC in expectation we have to compute $\mathbb{E}\left[\|S\|_\infty^j\right]$ and $\mathbb{E}\left[\|S\|_\infty^j \|SX\|_{2\to\infty}\right]$ where we can decompose the latter as follows

$$\begin{aligned}
&\mathbb{E}\left[\|S\|_\infty^k \|SX\|_{2\to\infty}\right] \\
\leq &\mathbb{E}\left[\|S\|_\infty^j \|\mathcal{S}\mathcal{X}\|_{2\to\infty}\right] + \mathbb{E}\left[\|S\|_\infty^j \|(S-\mathcal{S})\mathcal{X}\|_{2\to\infty}\right] + \mathbb{E}\left[\|S\|_\infty^j \|S(X-\mathcal{X})\|_{2\to\infty}\right] \\
\leq &\|\mathcal{S}\mathcal{X}\|_{2\to\infty}\mathbb{E}\left[\|S\|_\infty^j\right] + \sqrt{\mathbb{E}\left[\|S\|_\infty^{2j}\right]}\sqrt{\mathbb{E}\left[\|(S-\mathcal{S})\mathcal{X}\|_{2\to\infty}^2\right]} \\
&+ \sqrt{\mathbb{E}\left[\|S\|_\infty^{2j}\right]}\sqrt{\mathbb{E}\left[\|(X-\mathcal{X})S\|_{2\to\infty}^2\right]}
\end{aligned}$$

where the second inequality follows from noting that $\|\mathcal{S}\mathcal{X}\|_{2\to\infty}$ is deterministic and does not depend on the expectation and the decomposition of the last two terms follows from using Cauchy-Schwarz inequality. Let $C = (1+o(1))$. The following table gives an overview over the bounds on the different terms, where the individual entries are derived in subsequently.

	Self Loop	Degree Normalized
A.1.3: $\|\mathcal{S}\mathcal{X}\|_{2\to\infty}$	$C\|m\mu\|_\infty n\left(1+\left(\frac{p-q}{2}\right)^2\Gamma^2\right)$	$C\|m\mu\|_\infty \frac{\left(1+\left(\frac{p-q}{2}\right)^2\Gamma^2\right)}{\left(\frac{p+q}{2}\right)}$
A.1.3: $\mathbb{E}\left[\|(S-\mathcal{S})\mathcal{X}\|_{2\to\infty}^2\right]$	$Cn^2p\|m\mu\|_\infty$	$C\frac{n\ln(n)}{1+(n-1)q}\|m\mu\|_\infty$
A.1.3: $\mathbb{E}\left[\|(X-\mathcal{X})S\|_{2\to\infty}^2\right]$	$Cn^2p\sigma^2(1+2\ln d)$	$C\frac{1}{q}$
A.1.3: $\mathbb{E}\left[\|S\|_\infty^k\right]$	$(Cnp)^k$	$\left(C\frac{p}{q}\right)^{\frac{k}{2}}$

□

Bound $\mathbb{E}\left[\left\|(S-\mathcal{S})\,\mathcal{X}\right\|_{2\to\infty}\right]$

Proof. We first note that:

$$\begin{aligned}
\left\|(S-\mathcal{S})\,\mathcal{X}\right\|_{2\to\infty} &= \left\|(S-\mathcal{S})\,z\mu^\top\right\|_{2\to\infty} && \text{by definition of } \mathcal{X}\\
&= \max_j \left\|(S-\mathcal{S})\,z\mu_j^\top\right\|_2 && \text{by definition of } \|\cdot\|_{2\to\infty}\\
&= \left\|(S-\mathcal{S})\,z\right\|_2 \left\|\mu\right\|_\infty && \text{(A.10)}
\end{aligned}$$

and we only have to compute the expectation of $\left\|(S-\mathcal{S})\,z\right\|_2$ as $\left\|\mu\right\|_\infty$ is deterministic. Taking the expectation:

$$\begin{aligned}
\mathbb{E}\left[\left\|(S-\mathcal{S})\,z\right\|_2\right] &\leq \sqrt{\mathbb{E}\left[z^\top (S-\mathcal{S})^\top (S-\mathcal{S})\,z\right]}\\
&= \left(\sum_{ij} z_i z_j \sum_k \mathbb{E}\left[(S-\mathcal{S})_{ki}\,(S-\mathcal{S})_{kj}\right]\right)^{\frac{1}{2}} && \text{(A.11)}
\end{aligned}$$

where (A.11) follows from the fact that z is deterministic. From this expression we can now consider the self loop and degree normalized case for the diffusion operator.

Case 1: Self loop.

$\sum_k \mathbb{E}\left[(S-\mathcal{S})_{ki}\,(S-\mathcal{S})_{kj}\right]$ in (A.11) now becomes $\sum_k \mathbb{E}\left[(A-\mathcal{A})_{ki}\,(A-\mathcal{A})_{kj}\right]$ where we distinguish two cases:

$$\begin{aligned}
i \neq j \qquad &\Rightarrow A_{ki} \text{ and } A_{kj} \text{ are independent} \Rightarrow \mathbb{E}\left[(A-\mathcal{A})_{ki}\,(A-\mathcal{A})_{kj}\right] = 0\\
i = j \qquad &\Rightarrow \mathbb{E}\left[(A-\mathcal{A})_{ki}\,(A-\mathcal{A})_{ki}\right] = \mathrm{Var}(A_{ki}) = \mathcal{A}_{ki}(1-\mathcal{A}_{ki})
\end{aligned}$$

Therefore (A.11) becomes

$$\begin{aligned}
\mathbb{E}\left[\left\|(S-\mathcal{S})\,z\right\|_2\right] &\leq \left(\sum_i z_i^2 \sum_k \mathcal{A}_{ki}(1-\mathcal{A}_{ki})\right)^{\frac{1}{2}}\\
&= \left(\sum_{ik} \mathcal{A}_{ki}(1-\mathcal{A}_{ki})\right)^{\frac{1}{2}} && \because z_i^2 = 1\\
&\leq \left(\sum_{ik} \mathcal{A}_{ki}\right)^{\frac{1}{2}}\\
&\leq \left(n^2 \frac{p+q}{2}\right)^{\frac{1}{2}}\\
&= n\sqrt{\frac{p+q}{2}}
\end{aligned}$$

and giving us the final bound as using the above in (A.10):

$$\mathbb{E}\left[\|(S-\mathcal{S})\,\mathcal{X}\|_{2\to\infty}\right] \leq n\sqrt{\frac{p+q}{2}}\,\|\mu\|_\infty$$

Case 2: Degree normalized.

Note that for this section we initially considered an extension of the degree normalized model where the self loop is weighted by γ. For the final version however we set $\gamma = 1$.

As before first note that:

$$\begin{aligned}\|(S-\mathcal{S})\,\mathcal{X}\|_{2\to\infty} = \left\|(S-\mathcal{S})\,z\mu^\top\right\|_{2\to\infty} &= \max_j \left\|(S-\mathcal{S})\,z\mu_j^\top\right\|_2 \\ &= \|(S-\mathcal{S})\,z\|_2\,\|\mu\|_\infty \end{aligned} \tag{A.12}$$

and we only have to compute the expectation of $\|(S-\mathcal{S})\,z\|_2$ in (A.12). To bound this term we start by defining:

$$\begin{aligned}\mathcal{S} &:= (\mathcal{D}+\gamma\mathbb{I})^{-\frac{1}{2}}(\mathcal{A}+\gamma\mathbb{I})(\mathcal{D}+\gamma\mathbb{I})^{-\frac{1}{2}} \\ S &:= (D+\gamma\mathbb{I})^{-\frac{1}{2}}(A+\gamma\mathbb{I})(D+\gamma\mathbb{I})^{-\frac{1}{2}} \\ \overline{S} &:= (\mathcal{D}+\gamma\mathbb{I})^{-\frac{1}{2}}(A+\gamma\mathbb{I})(\mathcal{D}+\gamma\mathbb{I})^{-\frac{1}{2}}\end{aligned}$$

such that we can write:

$$\|(S-\mathcal{S})\,z\|_2 \leq \left\|\left(S-\overline{S}\right)z\right\|_2 + \left\|\left(\overline{S}-\mathcal{S}\right)z\right\|_2 \tag{A.13}$$

and bound the two terms separately:

Bound first term in (A.13): $\left\|\left(\overline{S}-\mathcal{S}\right)z\right\|_2$

First we note that:

$$\left\|\left(\overline{S}-\mathcal{S}\right)z\right\|_2 \leq \left\|(\mathcal{D}+\gamma\mathbb{I})^{-\frac{1}{2}}(A-\mathcal{A})(\mathcal{D}+\gamma\mathbb{I})^{-\frac{1}{2}}z\right\|_2$$

and therefore

$$\begin{aligned}
\mathbb{E}\left[\left\|\left(\overline{S}-\mathcal{S}\right)z\right\|_2\right] &\leq \left(\mathbb{E}\left[z^\top(\mathcal{D}+\gamma\mathbb{I})^{-\frac{1}{2}}(A-\mathcal{A})(\mathcal{D}+\gamma\mathbb{I})^{-1}(A-\mathcal{A})(\mathcal{D}+\gamma\mathbb{I})^{-\frac{1}{2}}z\right]\right)^{-\frac{1}{2}} \\
&= \left(\sum_{i,j}\frac{z_iz_j}{\sqrt{(\gamma+\mathcal{D}_{ii})(\gamma+\mathcal{D}_{jj})}}\underbrace{\sum_{k\neq i,j}\frac{\mathbb{E}\left[(A-\mathcal{A})_{ki}(A-\mathcal{A})_{kj}\right]}{\gamma+\mathcal{D}_{kk}}}_{\text{term 2}}\right)^{-\frac{1}{2}} \qquad \text{(A.14)} \\
&\leq \left(\sum_i \frac{z_i^2}{\gamma+\mathcal{D}_{ii}}\cdot\frac{\mathcal{D}_{ii}}{\gamma+(n-1)q}\right)^{-\frac{1}{2}} \qquad \text{(A.15)} \\
&\leq \left(\frac{n}{\gamma+(n-1)q}\right)^{-\frac{1}{2}} \qquad \because z_i^2=1
\end{aligned}$$

Where the step form (A.14) to (A.15) follows by bounding (A.14), *term 2* as follows. For $i \neq j$ the expression is zero. Otherwise for $i = j$:

$$\begin{aligned}
\sum_{k\neq i,j}\frac{\mathbb{E}\left[(A-\mathcal{A})_{ki}(A-\mathcal{A})_{kj}\right]}{\gamma+\mathcal{D}_{kk}} &= \sum_{k\neq i}\frac{\mathrm{Var}(A_{ki})}{\gamma+\mathcal{D}_{kk}} \\
&= \sum_{k\neq i}\frac{\mathcal{A}_{ki}(1-\mathcal{A}_{ki})}{\gamma+\mathcal{D}_{kk}} \\
&\leq \sum_{k\neq i}\frac{\mathcal{A}_{ki}}{\gamma+(n-1)q} \qquad \because \mathcal{D}_{kk}\geq(n-1)q \\
&= \frac{\mathcal{D}_{ii}}{\gamma+(n-1)q}
\end{aligned}$$

Therefore

$$\mathbb{E}\left[\left\|\left(\overline{S}-\mathcal{S}\right)z\right\|_2\right] \leq \sqrt{\frac{n}{\gamma+(n-1)q}}$$

Bound second term in (A.13): $\left\|\left(S-\overline{S}\right)z\right\|_2$

Let $B := D+\gamma\mathbb{I}$ and $C := \mathcal{D}+\gamma\mathbb{I}$. We first consider the following decomposition:

$$\begin{aligned}
&B^{-\frac{1}{2}}AB^{-\frac{1}{2}} - C^{-\frac{1}{2}}AC^{-\frac{1}{2}} \\
&= B^{-\frac{1}{2}}AB^{-\frac{1}{2}} - B^{-\frac{1}{2}}AB^{-\frac{1}{2}}B^{\frac{1}{2}}C^{-\frac{1}{2}} + B^{-\frac{1}{2}}AB^{-\frac{1}{2}}B^{\frac{1}{2}}C^{-\frac{1}{2}} - \underbrace{C^{-\frac{1}{2}}B^{\frac{1}{2}}B^{-\frac{1}{2}}AB^{-\frac{1}{2}}B^{\frac{1}{2}}C^{-\frac{1}{2}}}_{\text{equal to } C^{-\frac{1}{2}}AC^{-\frac{1}{2}}} \\
&= \underbrace{B^{-\frac{1}{2}}AB^{-\frac{1}{2}}}_{S}\left(\mathbb{I}-B^{\frac{1}{2}}C^{-\frac{1}{2}}\right) + \left(\mathbb{I}-C^{-\frac{1}{2}}B^{\frac{1}{2}}\right)\underbrace{B^{-\frac{1}{2}}AB^{-\frac{1}{2}}}_{S}B^{\frac{1}{2}}C^{-\frac{1}{2}} \qquad \text{(A.16)}
\end{aligned}$$

Using (A.16) we can bound the expectation of $\left\|\left(S-\overline{S}\right)z\right\|_2$ as:

$$\begin{aligned}
&\mathbb{E}\left[\left\|\left(S-\overline{S}\right)z\right\|_2\right] \\
=&\mathbb{E}\left[\left((D+\gamma\mathbb{I})^{-\frac{1}{2}}(A+\gamma\mathbb{I})(D+\gamma\mathbb{I})^{-\frac{1}{2}}-(\mathcal{D}+\gamma\mathbb{I})^{-\frac{1}{2}}(A+\gamma\mathbb{I})(\mathcal{D}+\gamma\mathbb{I})^{-\frac{1}{2}}\right)z\right] \\
\leq&\mathbb{E}\left[\left\|S\left(\mathbb{I}-(D+\gamma\mathbb{I})^{\frac{1}{2}}(\mathcal{D}+\gamma\mathbb{I})^{-\frac{1}{2}}\right)z\right\|_2\right] \qquad (A.17)\\
&+\mathbb{E}\left[\left\|\left(\mathbb{I}-(D+\gamma\mathbb{I})^{\frac{1}{2}}(\mathcal{D}+\gamma\mathbb{I})^{-\frac{1}{2}}\right)S(D+\gamma\mathbb{I})^{\frac{1}{2}}(\mathcal{D}+\gamma\mathbb{I})^{-\frac{1}{2}}z\right\|_2\right] \qquad (A.18)
\end{aligned}$$

Bound (A.17):

$$\begin{aligned}
\mathbb{E}\left[\left\|S\left(\mathbb{I}-(D+\gamma\mathbb{I})^{\frac{1}{2}}(\mathcal{D}+\gamma\mathbb{I})^{-\frac{1}{2}}\right)z\right\|_2\right] &\leq \mathbb{E}\left[\|S\|_2\left\|\left(\mathbb{I}-(D+\gamma\mathbb{I})^{\frac{1}{2}}(\mathcal{D}+\gamma\mathbb{I})^{-\frac{1}{2}}\right)z\right\|_2\right] \\
&\leq \sqrt{\sum_i \mathbb{E}\left[\left(1-\sqrt{\frac{D_{ii}+\gamma}{\mathcal{D}_{ii}+\gamma}}\right)^2 z_i^2\right]} \quad \because \|S\|_2\leq 1 \\
&\leq \sqrt{\sum_i \mathbb{E}\left[\left(1-\sqrt{\frac{D_{ii}+\gamma}{\mathcal{D}_{ii}+\gamma}}\right)^2\right]}
\end{aligned}$$

we therefore now need to compute $\sum_i \mathbb{E}\left[\left(1-\sqrt{\frac{D_{ii}+\gamma}{\mathcal{D}_{ii}+\gamma}}\right)^2\right]$. Note that for $x\geq 0$, $|1-\sqrt{x}|\leq |1-x|$. Using this we write

$$\begin{aligned}
\sum_i \mathbb{E}\left[\left(1-\sqrt{\frac{D_{ii}+\gamma}{\mathcal{D}_{ii}+\gamma}}\right)^2\right] &\leq \sum_i \mathbb{E}\left[\left(1-\frac{D_{ii}+\gamma}{\mathcal{D}_{ii}+\gamma}\right)^2\right] \\
&=\sum_i 1-2+\frac{\mathbb{E}\left[(D_{ii}+\gamma)^2\right]}{(\mathcal{D}_{ii}+\gamma)^2} \\
&=\sum_i -1+\frac{\mathbb{E}\left[(\gamma+\sum_{k\neq i}A_{ik})^2\right]}{(\mathcal{D}_{ii}+\gamma)^2} \\
&=-n+\sum_i \frac{(\mathcal{D}_{ii}+\gamma)^2+\mathcal{D}_{ii}+\sum_{k\neq i}A_{ik}^2}{(\mathcal{D}_{ii}+\gamma)^2} \\
&=\sum_i \frac{\sum_{k\neq i}A_{ik}(1-A_{ik})}{(\mathcal{D}_{ii}+\gamma)^2} \\
&\leq \sum_i \frac{1}{\mathcal{D}_{ii}+\gamma} \\
&\leq \frac{n}{\gamma+(n-1)q} \qquad (A.19)
\end{aligned}$$

Bound (A.18):

$$\begin{aligned}
&\mathbb{E}\left[\left\|\left(\mathbb{I}-(D+\gamma\mathbb{I})^{\frac{1}{2}}(\mathcal{D}+\gamma\mathbb{I})^{-\frac{1}{2}}\right)S(D+\gamma\mathbb{I})^{\frac{1}{2}}(\mathcal{D}+\gamma\mathbb{I})^{-\frac{1}{2}}z\right\|_2\right]\\
\leq&\mathbb{E}\left[\left\|\mathbb{I}-(D+\gamma\mathbb{I})^{\frac{1}{2}}(\mathcal{D}+\gamma\mathbb{I})^{-\frac{1}{2}}\right\|_2\|S\|_2\left\|(D+\gamma\mathbb{I})^{\frac{1}{2}}(\mathcal{D}+\gamma\mathbb{I})^{-\frac{1}{2}}z\right\|_2\right]\\
\leq&\mathbb{E}\left[\max_i\left(1-\sqrt{\frac{(D+\gamma\mathbb{I})_{ii}}{(\mathcal{D}+\gamma\mathbb{I})_{ii}}}\right)\left\|(D+\gamma\mathbb{I})^{\frac{1}{2}}(\mathcal{D}+\gamma\mathbb{I})^{-\frac{1}{2}}z\right\|_2\right]\\
\leq&\left(\underbrace{\mathbb{E}\left[\max_i\left(1-\sqrt{\frac{D_{ii}+\gamma}{\mathcal{D}_{ii}+\gamma}}\right)\right]}_{term1}\underbrace{\mathbb{E}\left[\left\|(D+\gamma\mathbb{I})^{\frac{1}{2}}(\mathcal{D}+\gamma\mathbb{I})^{-\frac{1}{2}}z\right\|^2\right]}_{term2}\right)^{\frac{1}{2}}
\end{aligned}\tag{A.20}$$

where (A.20) follows from applying the Cauchy-Schwarz inequality. Then for (A.20) *term 2* we get:

$$\begin{aligned}
\mathbb{E}\left[\left\|(D+\gamma\mathbb{I})^{\frac{1}{2}}(\mathcal{D}+\gamma\mathbb{I})^{-\frac{1}{2}}z\right\|^2\right]&=\sum_i\mathbb{E}\left[\frac{D_{ii}+\gamma}{\mathcal{D}_{ii}+\gamma}z_i^2\right]\\
&=\sum_i\underbrace{\frac{\mathbb{E}\left[D_{ii}+\gamma\right]}{\mathcal{D}_{ii}+\gamma}}_{=1}\\
&=n
\end{aligned}$$

(A.20) *term 1* we again note that for $x\geq 0,\ |1-\sqrt{x}|\leq|1-x|$. Using this we write:

$$\begin{aligned}
\mathbb{E}\left[\max_i\left(1-\sqrt{\frac{D_{ii}+\gamma}{\mathcal{D}_{ii}+\gamma}}\right)^2\right]&\leq\mathbb{E}\left[\max_i\left(1-\frac{D_{ii}+\gamma}{\mathcal{D}_{ii}+\gamma}\right)^2\right]\\
&\leq\frac{1}{s}\ln\left(\exp\left(\mathbb{E}\left[s\max_i\left(1-\frac{D_{ii}+\gamma}{\mathcal{D}_{ii}+\gamma}\right)^2\right]\right)\right)\\
&\leq\frac{1}{s}\ln\left(\mathbb{E}\left[\exp\left(s\max_i\left(1-\frac{D_{ii}+\gamma}{\mathcal{D}_{ii}+\gamma}\right)^2\right)\right]\right)\\
&=\frac{1}{s}\ln\left(\mathbb{E}\left[\max_i\left(\exp s\left(\left(1-\frac{D_{ii}+\gamma}{\mathcal{D}_{ii}+\gamma}\right)^2\right)\right)\right]\right)\\
&\leq\frac{1}{s}\ln\left(\underbrace{\sum_i\mathbb{E}\left[\underbrace{\exp\left(s\underbrace{\left(1-\frac{D_{ii}+\gamma}{\mathcal{D}_{ii}+\gamma}\right)^2}_{y_i}\right)}_{term1}\right]}_{term2}\right)
\end{aligned}\tag{A.21}$$

Now to further bound (A.21) we first compute (A.21), term 1 as:

$$
\begin{aligned}
\exp(sy_i) &= 1 + sy_i + \sum_{k\geq 2} \frac{(sy_i)^k}{k!} \\
&= 1 + sy_i + (sy_i) \sum_{k\geq 2} \frac{(sy_i)^{k-1}}{k!} \\
&= 1 + sy_i + (sy_i) \sum_{k\geq 0} \frac{(sy_i)^k}{(k+1)k!} \\
&\leq 1 + sy_i + (sy_i)\exp(sy_i) \\
&\leq 1 + (\exp(s)+1)sy_i
\end{aligned}
$$

Taking the expectation over the previous line, using linearity of expectation and the expression for $\sum_i \mathbb{E}[y_i]$ from (A.19) it follows that for (A.21), term 2 we obtain

$$
\begin{aligned}
\sum_i \mathbb{E}[\exp(sy_i)] &\leq n + (\exp(s)+1)s\sum_i \mathbb{E}[y_i] \\
&= n + (\exp(s)+1)s\frac{n}{\gamma+(n-1)q}
\end{aligned}
$$

Going back to (A.21):

$$
\begin{aligned}
(\text{A.21}) &\leq \frac{1}{s}\ln\left(n + (\exp(s)+1)s\frac{n}{\gamma+(n-1)q}\right) && \forall s > 0 \\
&\leq \frac{1}{s}\ln\left(n + \exp(2s)\frac{n}{\gamma+(n-1)q}\right) && \text{Note: } s>0 \Rightarrow \ln s \leq s-1 \\
& && \Rightarrow (e^s+1)s \leq e^{2s} \\
&\leq \frac{\ln(n)}{s} + \frac{1}{s}\ln\left(1 + \frac{\exp(2s)}{\gamma+(n-1)q}\right) && \text{Let } e^{2s} \geq \gamma+(n-1)q \\
&\leq \frac{\ln(n)}{s} + \frac{1}{s}\ln\left(\frac{2\exp(2s)}{\gamma+(n-1)q}\right) \\
&\leq \frac{\ln(n)}{s} + 2 + \frac{1}{s}\ln\left(\frac{2}{\gamma+(n-1)q}\right) && \text{Take } s := \gamma+(n-1)q \geq 2 \\
&\leq C\frac{\ln(n)}{\gamma+(n-1)q}
\end{aligned}
$$

Finally combining the above results:

$$
\begin{aligned}
\mathbb{E}\left[\|(S-\overline{S})z\|_2\right] &\leq \sqrt{\frac{n}{\gamma+(n-1)q}} + \sqrt{n\frac{C\ln(n)}{\gamma+(n-1)q}} \\
&= C\sqrt{\frac{n\ln(n)}{\gamma+(n-1)q}}
\end{aligned}
$$

and

$$
\mathbb{E}\left[\|(S-\mathcal{S})\mathcal{X}\|_{2\to\infty}\right] \leq C\sqrt{\frac{n\ln n}{\gamma+(n-1)q}}\,\|\mu\|_\infty
$$

This concludes he bound of $\mathbb{E}\left[\|(S-\mathcal{S})\mathcal{X}\|_{2\to\infty}\right]$. $\square$

Bound $\mathbb{E}\left[\|(X-\mathcal{X})\,S\|_{2\to\infty}\right]$

Proof. We first note that

$$\begin{aligned}\mathbb{E}\left[\|(X-\mathcal{X})\,S\|_{2\to\infty}\right] &= \mathbb{E}\left[\max_{j\in[d]}\|S\epsilon_{\cdot j}\|_2\right]\\ &\leq \left(\mathbb{E}\left[\max_{j\in[d]}\|S\epsilon_{\cdot j}\|_2^2\right]\right)^{\frac{1}{2}}\end{aligned}$$

Let $z\sim\mathcal{N}(0,\sigma^2\mathbb{I})$ then

$$\begin{aligned}\|Sz\|_2^2 &= z^\top S^\top Sz\\ &= zV\Lambda V^\top z && \text{Eigendecompsition}\\ &= \sum_{i=1}^n \lambda_i z_i'^2 && \text{where } V^\top z = z_i' \sim \mathcal{N}(0,\sigma^2\mathbb{I})\\ &= \sum_{i=1;\lambda_i>0}^n \lambda_i\sigma^2 y_i && y_i,\cdots,y_d \overset{iid}{\sim}\mathcal{X}^2\end{aligned}$$

Where the first line follows from the eigendecomposition $S^\top S = V\Lambda V^\top$. Therefore $\|Sz\|_2^2$ is distributed as a generalised $\mathcal{X}^2$ with mean $\sigma\,\mathrm{Tr}(S^\top S)$ and variance $2\sum\lambda_i\sigma^4 = 2\sigma^4\left\|S^\top S\right\|_F^2$. Now define

$$\mathrm{MGF}_y(s) = \frac{1}{\exp\left(\frac{1}{2}\sum_{i:\lambda_i>0}\log(1-2s\lambda_i)\right)}$$

and consider $s\in\left(0,\frac{1}{2\lambda_{min}}\right)$ where λ_{min} is the smallest non-zero eigenvalue of $S^\top S$.

$$\begin{aligned}\exp\left(s\mathbb{E}\left[\max_j y_j\right]\right) &\leq \mathbb{E}\left[\exp\left(s\max(y_j)\right)\right]\\ &= \mathbb{E}\left[\max\exp\left(sy_j\right)\right]\\ &\leq \sum_j \mathbb{E}\left[\exp\left(sy_j\right)\right]\\ &= d\cdot\mathrm{MGF}_y(s)\\ &= d\exp\left(-\frac{1}{2}\sum_{i:\lambda_i>0}\log(1-2s\lambda_i)\right)\end{aligned}$$

it follows that

$$\begin{aligned}
\mathbb{E}\left[\max_j y_j\right] &\leq \frac{\ln d}{s} - \frac{1}{2s}\sum_{i:\lambda_i>0}\underbrace{\log(1-2s\lambda_i)}_{\leq -2s\lambda_i} \\
&\leq \frac{\ln d}{s} + \underbrace{\sum_{i:\lambda_i>0}\lambda_i}_{\mathrm{Tr}(S^\top S)} && \because \log(1+x)\leq x\ \forall x>-1 \\
&\leq 2\lambda_{min}\ln d + \mathrm{Tr}(S^\top S) && \because s\in\left(0,\tfrac{1}{2\lambda_{min}}\right) \text{ and min for } s=\tfrac{1}{2\lambda_{min}}
\end{aligned}$$

Using $\sigma_{min}(S) \leq \|S\|_2$ and $\|S\|_F \leq k\|S\|_2$ we can bound the last line as $\|S\|_2^2(k+2\ln d)$ in the low-rank setting. However since we consider S to be random this is not applicable (also see the remarks in the VC Dimension section). Therefore

$$\begin{aligned}
2\lambda_{min}\ln d + \mathrm{Tr}(S^\top S) &= \sigma_{min}^2(S)\ln d + \|S\|_F^2 \\
&\leq \|S\|_F^2(1+2\ln d)
\end{aligned}$$

and taking the square root gives us the final result:

$$\mathbb{E}\left[\|(X-\mathcal{X})S\|_{2\to\infty}\right] \leq \sigma\|S\|_F\sqrt{1+2\ln d}$$

Bound $\mathbb{E}\left[\|S\|_F^2\right]$.

<u>Case 1: Self loop.</u>

We first note that $\|S\|_F^2 = n +$ *number of edges* and therefore:

$$\begin{aligned}
\mathbb{E}\left[\|S\|_F^2\right] &\leq n + n^2p \\
&= (1+o(1))n^2p
\end{aligned}$$

Therefore

$$\mathbb{E}\left[\|(X-\mathcal{X})S\|_{2\to\infty}^2\right] \leq (1+o(1))n^2p\sigma^2(1+2\ln d)$$

<u>Case 2: Degree normalized.</u>

Note that we here overload the notation d such that we define the degree for node i as d_i

and similar d_{min} is the minimum degree.

$$\begin{aligned}
\mathbb{E}\left[\|S\|_F^2\right] =& \mathbb{E}\left[\|S\|_F^2 \middle| \left\{d_{min} > np - \sqrt{4cnp\ln n}\right\}\right] \mathbb{P}\left(d_{min} > np - \sqrt{4cnp\ln n}\right) \\
&+ \mathbb{E}\left[\|S\|_F^2 \middle| \left\{d_{min} < np - \sqrt{4cnp\ln n}\right\}\right] \mathbb{P}\left(d_{min} < np - \sqrt{4cnp\ln n}\right) \\
\leq& \mathbb{E}\left[\|S\|_F^2 \middle| \left\{d_{min} > np - \sqrt{4cnp\ln n}\right\}\right] \mathbb{P}\left(d_{min} > np - \sqrt{4cnp\ln n}\right) + \underbrace{n^2 \frac{1}{n^c}}_{=o(1)} \\
\leq& \sum_{i,j} \frac{A_{ij} + \mathbb{I}\{i=j\}}{(d_i+1)(d_j+1)} \\
\leq& \frac{1}{d_{min}+1} \sum_i \underbrace{\frac{\sum_j A_{ij} + \mathbb{I}\{i=j\}}{d_i+1}}_{=1} \\
\leq& \frac{n}{nq + 1 - \sqrt{4cnp\ln n}} \\
=& (1+o(1))\frac{1}{q}
\end{aligned}$$

Therefore

$$\mathbb{E}\left[\|(X-\mathcal{X})\, S\|_{2\to\infty}^2\right] \leq (1+o(1))\frac{\sigma^2(1+2\ln d)}{q}$$

This concludes the bound of $\mathbb{E}\left[\|(X-\mathcal{X})\, S\|_{2\to\infty}^2\right]$. □

Bound $\mathbb{E}\left[\|S\|_\infty^k\right]$.

Proof. In general we can note that $\|\mathcal{S}\|_\infty^k = \max_{1\leq i\leq n}\left(\sum_{j=1}^n \mathcal{S}_{ij}\right)^k$

<u>Case 1: Self loop.</u>

We first define the degree for node i as

$$d_i \sim \text{Bin}\left(\frac{n}{2}-1, p\right) + \text{Bin}\left(\frac{n}{2}, q\right)$$

then $\|\mathcal{S}\|_\infty = \max_{1\leq i\leq n}\left(\sum_{j=1}^n \mathcal{S}_{ij}\right) = 1 + \max_i d_i$ and assume $p > \frac{lnn}{n}$ and let $t = \sqrt{4np\ln n}$

$$\begin{aligned}
\mathbb{P}\left(d_i - \mathbb{E}\left[d_i\right] > t\right) \leq& \exp\left(\frac{-\frac{t^2}{2}}{np + \frac{t}{3}}\right) && \text{Bernstein inequality} \\
\leq& \exp\left(\frac{-4cnp\ln n}{4np}\right) \\
=& \frac{1}{n}c
\end{aligned}$$

and therefore

$$\mathbb{P}\left(\max_i d_i \geq np + \sqrt{4cnp\ln n}\right) \leq \frac{1}{n^{c-1}}$$
$$\mathbb{P}\left((1+\max_i d_i)^k \geq (1+np+\sqrt{4cnp\ln n})^k\right) \leq \frac{1}{n}c$$

and

$$\begin{aligned}\mathbb{E}\left[(1+\max_i d_i)^k\right] &\leq (1+np+\sqrt{4cnp\ln n})^k + \frac{1}{n^{c-i}}n^k \\ &= (1+np+\sqrt{4cnp\ln n})^k + n^{k+1-c}\end{aligned}$$

For large n and $p \gg \frac{(lnn)^2}{n}$ take $c = \ln n$:

$$\mathbb{E}\left[\|S\|_\infty^k\right] \leq ((1+o(1))np)^k$$

Case 2: Degree normalized.

$$\begin{aligned}\|S\|_\infty &= \max_i \sum_j S_{ij} \\ &= \max_i \sum_j \frac{A_{ij}}{\sqrt{d_i+1}\sqrt{d_j+1}} \\ &\leq \max_i \frac{1}{\sqrt{d_{min}+1}} \frac{\sum_j A_{ij}}{\sqrt{d_i+1}} \\ &= \max_i \sqrt{\frac{d_i+1}{d_{min}+1}} \\ &\leq \sqrt{\frac{d_min+1}{d_{min}+1}}\end{aligned}$$

Similar to above we can now note that:

$$\mathbb{P}\left(\max_i d_i \geq np + \sqrt{4cnp\ln n}\right) \leq \frac{1}{n^c}$$
$$\mathbb{P}\left(\max_i d_i \leq np + \sqrt{4cnp\ln n}\right) \leq \frac{1}{n^c}$$

and it follows

$$\mathbb{P}\left(\sqrt{\frac{d_{max}+1}{d_{min}+1}} \geq \frac{np+\sqrt{4cnp\ln n}+1}{np-\sqrt{4cnp\ln n}+1}\right) \leq \frac{2}{n^c}$$

For large n and $p, q \gg \frac{(lnn)^2}{n}$:

$$\begin{aligned}\mathbb{E}\left[\|S\|_\infty^k\right] &\leq \mathbb{E}\left[\left(\frac{d_{max}+1}{d_{min}+1}\right)^{\frac{k}{2}}\right] \\ &= \left((1+o(1))\frac{p}{q}\right)^{\frac{k}{2}}\end{aligned}$$

This concludes the bound of $\mathbb{E}\left[\|S\|_\infty^k\right]$. □

Bound $\|\mathcal{S}\mathcal{X}\|_{2\to\infty}$.

Proof.

Case 1: Self loop.

$$\begin{aligned}\mathcal{S}\mathcal{X} &= (1-p)z\mu^\top - \frac{p-q}{2}yy^\top z\mu^\top \\ &= \left((1-p)z - \left(\frac{p-q}{2}y^\top z\right)y\right)\mu^\top\end{aligned}$$

and

$$(\mathcal{S}\mathcal{X})_{ij} = \left((1-p)z_i - \underbrace{\left(\frac{p-q}{2}y^\top z\right)}_{:=\delta} y_i\right)\mu_j$$

Now using this to compute the two-infinity norm:

$$\begin{aligned}\|\mathcal{S}\mathcal{X}\|_{2\to\infty} &= \|\mu\|_\infty \sqrt{\sum_i ((1-p)z_i - \delta y_i)^2} \\ &= \|\mu\|_\infty \sqrt{\sum_i (1-p)^2 + \delta^2 - 2\delta(1-p)y_i z_i} \\ &= \|\mu\|_\infty \left(n(1-p)^2 + n(y^\top z)^2\left(\frac{p-q}{2}\right)^2 - 2(y^\top z)^2\frac{p-q}{2}(1-p)\right) \\ &= (1+o(1))\|\mu\|_\infty n\left(1 + \left(\frac{p-q}{2}\right)^2 (y^\top z)^2\right)\end{aligned}$$

Case 2: Degree normalized.

We note that the expected degree is $(1+o(1))n\frac{p+q}{2}$ and therefore similar to above we obtain

$$\|\mathcal{S}\mathcal{X}\|_{2\to\infty} = (1+o(1))\|\mu\|_\infty \frac{\left(1+\left(\frac{p-q}{2}\right)^2(y^\top z)^2\right)}{\left(\frac{p+q}{2}\right)}.$$

This concludes the bound of $\|\mathcal{S}\mathcal{X}\|_{2\to\infty}$.

□

A.1.4 Proof of Theorem 3

Consider a Residual network as defined in (2.14) and $\mathcal{F}_{\mathcal{G}}^{\psi,\beta,\omega} \subset \mathcal{F}_{\mathcal{G}}^{\psi}$ such that the trainable parameters satisfy $\|b_j\|_1 \leq \beta$ and $\|W_j\|_\infty \leq \omega$ for every $j \in [K]$. Then with $\alpha \in (0,1)$ and $c_1 := 2L_\psi\beta$, $c_2 := 2L_\psi\omega$, $c_3 := L_\psi\omega\sqrt{2/d}$ the TRC of the restricted class or Residual GNNs is bounded as

$$\begin{aligned}\mathfrak{R}_{m,n}(\mathcal{F}_{\mathcal{G}}^{\psi,\beta,\omega}) \;\leq\; & \frac{((1-\alpha)c_1 + \alpha 2L_\psi \|X\|_\infty)n^2}{m(n-m)}\left(\sum_{j=0}^{J-1}(1-\alpha)c_2^j \|S\|_\infty^j\right) \\ & + \alpha 2L_\psi \|X\|_\infty + (1-\alpha)c_3c_2^J \|S\|_\infty^J \|SX\|_{2\to\infty}\sqrt{\log(n)} \qquad \text{(A.22)}\end{aligned}$$

Proof. Recall the setup for residual connections as defined in the main paper where we can now write the layer wise propagation rule as $f_{j+1} := \psi\left((1-\alpha)\left(b_j + Sf_j\left(H\right)W_j\right) + \alpha f_0\left(H\right)\right),$ with $\alpha \in (0,1)$. We can now derive a generalization error bound similar to the one given in Theorem 1 for the Residual network. As most of the steps are the same we will only remark the main changes. Recall that for the vanilla case we considered

$$\mathfrak{R}_{m,n}(\mathcal{F}_{\mathcal{G}}^{\psi}) \leq L_\psi\left(\mathfrak{R}_{m,n}(\mathcal{F}_{lin}) + \mathfrak{R}_{m,n}(\mathcal{F}_{bias})\right)$$

and by Lemma 3 and Lemma 2 we obtain a similar bound for the Residual network as

$$\mathfrak{R}_{m,n}(\mathcal{F}_{\mathcal{G}}^{\psi}) \leq L_\psi\left((1-\alpha)\mathfrak{R}_{m,n}(\mathcal{F}_{lin}) + (1-\alpha)\mathfrak{R}_{m,n}(\mathcal{F}_{bias}) + \alpha\mathfrak{R}_{m,n}(\mathcal{F}_X)\right).$$

The bounds for $\mathfrak{R}_{m,n}(\mathcal{F}_{lin})$ and $\mathfrak{R}_{m,n}(\mathcal{F}_{bias})$ are as derived in previously. $\mathfrak{R}_{m,n}(\mathcal{F}_X)$ can be bound as

$$\mathfrak{R}_{m,n}(\mathcal{F}_X) \leq 2Q\|X\|_\infty n$$

Where the proof follows analogous to the one for the *bias term*, $\mathfrak{R}_{m,n}(\mathcal{F}_{bias}$.

Again with recursively applying the bounds for each layer and combining it with the bound on the first layer results in the full TRC bound. Consider a Residual network as defined in (2.14) and $\mathcal{F}_{\mathcal{G}}^{\psi,\beta,\omega} \subset \mathcal{F}_{\mathcal{G}}^{\psi}$ such that the trainable parameters satisfy $\|b_j\|_1 \leq \beta$ and $\|W_j\|_\infty \leq \omega$ for every $j \in [J]$. Then with $\alpha \in (0,1)$ and $c_1 := 2L_\psi\beta$, $c_2 := 2L_\psi\omega$, $c_3 := L_\psi\omega\sqrt{2/d}$ the TRC of the restricted class or Residual GNNs is bounded as (A.22), which concludes the proof. □

A.2 Proofs for Chapter 3

A.2.1 Proof of Lemma 1

Let $f : \mathbb{R}^d \to \mathbb{R}^h$ be either a linear model $f(x) = \Theta x$, or a kernel machine $f(x) = \Theta\phi(x)$, where ϕ corresponds to the implicit feature map of a kernel k, that is, $k(x, x') = \langle\phi(x), \phi(x')\rangle$. Then in the infinite width limit ($h \to \infty$) the inner products between the gradients are given by

$$\langle\nabla_\Theta f_l(x), \nabla_\Theta f_j(x')\rangle = \begin{cases} 0 & \text{if } l \neq j, \\ x^\top x' & \text{if } l = j \text{ (linear case)}, \\ k(x, x') & \text{if } l = j \text{ (kernel case)}. \end{cases}$$

Proof. Let the collumns of W_2 be denoted by $w_1, w_2, ..., w_h$. Then we note that each component of u, f_j is given by $f_j(x) = w_j^\top \psi(W_1 x)$. Thus if $l \neq j$, $f_j(x)$ has no dependence with w_l i.e. $\nabla_{w_l} f_j(x) = 0$. Thus we get that when $l \neq j$,

$$\langle\nabla_\Theta f_l(x), \nabla_\Theta f_j(x')\rangle = \langle\nabla_{W_1} f_l(x), \nabla_{W_1} f_j(x')\rangle .$$

We can now use [LZB20] (for instance its Lemma 1) which basically concludes that no training happens at the penultimate or prior layers. In limit all positive gradients arise only from the final layer. As such

$$\langle\nabla_{W_1} f_l(x), \nabla_{W_1} f_j(x')\rangle = 0.$$

By the same token, for $l = j$,

$$\begin{aligned}\langle\nabla_\Theta f_l(x), \nabla_\Theta f_j(x')\rangle &= \langle\nabla_{W_1} f_l(x), \nabla_{W_1} f_j(x')\rangle + \left\langle\nabla_{w_j} f_j(x), \nabla_{w_j} f_j(x')\right\rangle \\ &= \langle\psi(W_1 x), \psi(W_1 x')\rangle .\end{aligned}$$

Finally again using the fact that W_1 does not change in training and that W_1 is initialized from a normalized Gaussian, when ψ is the identity map, it is well known that the above converges to $x^\top x'$ (as there $\langle\psi(W_1 x), \psi(W_1 x')\rangle = x^\top(W_1^\top W_1)x \to x^\top x'$) and otherwise to a deterministic kernel k (see e.g. [LZB20], [Aro+19c]). □

A.2.2 Proof of Proposition 2

Under the conditions of Lemma 1, every component of the network output $u : \mathbb{R}^d \to \mathbb{R}^h$ has identical dynamics, and hence, identical fixed points. As a consequence, the output collapses to one dimension at convergence.

For linear model, $f(x) = \Theta x$, the dynamics of $f(x)$ is given by

$$\mathring{f}_l(x) = \sum_{i=1}^{n} (x^\top x_i) f_l(x_i^+) + (x^\top x_i^+) f_l(x_i)$$

for the non-contrastive case, and

$$\mathring{f}_l(x) = \sum_{i=1}^{n} (x^\top x_i)\big(f_l(x_i^+) - f_l(x_i^-)\big) + (x^\top x_i^+ - x^\top x_i^-) f_l(x_i)$$

for the contrastive case. For kernel models, the dynamcis is similarly obtained by replacing each $x^\top x'$ by $k(x, x')$.

A.2.3 Proof or Proposition 3

Consider a linear SSL model $f^\psi(x) = W_2^\top \psi(W_1 x)$. The optimisation problem

$$\min_{W_1, W_2} \mathcal{L}(W_1, W_2) \quad \text{s.t.} \quad \|W_1\|_F \le c_1, \|W_2\|_F \le c_2,$$

where the loss $\mathcal{L}$ is given by (3.1) or (3.2), has a global solution $f(x) = [a(x)\ 0 \ldots 0]^\top \in \mathbb{R}^h$.

Proof. For simplicity of the proof we begin by reformulating the loss function in both contrastive and noncontrastive setting to a more general form. In particular it is trivial to check that we can generalize by writing

$$\mathcal{L} = \operatorname{Tr}\left(W_2^\top f(X, W_1) W_2\right),$$

where X denotes the collection of all the relevant data (i.e. $\forall\ 1 \le i \le n\ x_i$, as well as x_i^+ and x^- where applicable), and $f(X, W_1) = \sum_{i=1}^n \psi(W_1 x_i)\left(\psi(W_1 x_i^-) - \psi(W_1 x_i^+)\right)^\top$ in the contrastive setting (3.1) while $f(X, W_1) = -\sum_{i=1}^n \psi(W_1 x_i)\psi(W_1 x_i^+)^\top$ in the non-contrastive setting (3.2).

Then decompose

$$W_2 W_2^\top = \sum_{i=1}^{k} \sigma_i^2 v_i v_i^\top.$$

Note then that $\|W_2\|_F^2 = \operatorname{Tr}\left(W_2 W_2^\top\right) = \sum_{i=1}^k \sigma_i^2$. Thus the optimization target,

$$\begin{aligned}
\mathcal{L}(W_1, W_2) &= \operatorname{Tr}\left(W_2^\top f(X, W_1) W_2\right) = \operatorname{Tr}\left(f(X, W_1) W_2 W_2^\top\right) \\
&= \operatorname{Tr}\left(f(X, W_1) \sum_{i=1}^{k} \sigma_i^2 v_i v_i^\top\right) = \sum_{i=1}^{k} \sigma_i^2 v_i^\top f(X, W_1) v_i \\
&\ge \min_{i=1\ to\ k} \{v_i^\top f(X, W_1) v_i\} \sum_{i=1}^{k} \sigma_i^2 = \|W_2\|_F^2 \min_{i=1\ to\ k} \{v_i^\top f(X, W_1) v_i\}.
\end{aligned}$$

Thus when the Frobenius norm is restricted (i.e. bounded between 0 and c), if $f(X, W_1)$ has atleast one negative eigenvalue the loss is minimized when v_1 is the eigenvector corresponding to the most negative eigenvalue of $f(X, W_1)$ with $\sigma_1 = \|W_2\|_F$, with no other non-zero singular value. On the other hand if $f(X, W_1)$ has no negative eigenvalue then the loss is minimized when $W_2 = 0$. □

A.2.4 Proof of Proposition 4

Consider a linear SSL model $f^{\mathbb{I}}(x) = W_2^\top W_1 x$, and let the loss $\mathcal{L}(W_1, W_2)$ be given by either (3.1) or (3.2) whose general form is $\mathcal{L}(W_1, W_2) = \left\|W_2^\top W_1 C W_1^\top W_2\right\|_2^2$, where C has atleast one negative eigenvalue. Then the following optimisation problems are equivalent:

1. $\displaystyle\min_{W_1, W_2} \frac{\mathcal{L}(W_1, W_2)}{\|W_2\|_2^2 \|W_1\|_2^2}$;
2. $\displaystyle\min_{W_1, W_2} \mathcal{L}(W_1, W_2) \quad \text{s.t.} \quad \|W_2\|_2 \leq 1,\ \|W_1\|_2 \leq 1$;
3. $\displaystyle\min_{W_1, W_2} \mathcal{L}(W_1, W_2) \quad \text{s.t.} \quad \|W_2^\top W_1\|_2 \leq 1$;
4. $\displaystyle\min_{W_1, W_2} \mathcal{L}(W_1, W_2) \quad \text{s.t.} \quad W_2^\top W_2 = \mathbb{I}_h,\ W_1^\top W_1 = \mathbb{I}_d$.

Additionally this regularization avoids dimension collapse.

Proof. We begin by quickly observing that (1) $\iff$ (2). This is simply done by defining $\hat{W}_i = \frac{W_i}{\|W_i\|_2}$ for $i = 1, 2$. Then we have

$$\operatorname*{argmin}_{W_1, W_2} \frac{\operatorname{Tr}\left(W_2^\top W_1 C W_1^\top W_2\right)}{\|W_1\|_2^2 \|W_2\|_2^2} = \operatorname*{argmin}_{\hat{W}_1, \hat{W}_2 : \left\|\hat{W}_1\right\|_2 = \left\|\hat{W}_1\right\|_2 = 1} \operatorname{Tr}\left(\hat{W}_2^\top \hat{W}_1 C \hat{W}_1^\top \hat{W}_2\right)$$

Using the fact that at least one eigenvalue of C is strictly negative (this rules out the case that the optimal is achieved when $W_i = 0$ as that would have prevented division by norm) then we can quickly get that

$$\operatorname*{argmin}_{\hat{W}_1, \hat{W}_2 : \left\|\hat{W}_1\right\|_2 = \left\|\hat{W}_1\right\|_2 = 1} \operatorname{Tr}\left(\hat{W}_2^\top \hat{W}_1 C \hat{W}_1^\top \hat{W}_2\right) = \operatorname*{argmin}_{\hat{W}_1, \hat{W}_2 : \left\|\hat{W}_1\right\|_2 \leq 1; \left\|\hat{W}_1\right\|_2 \leq 1} \operatorname{Tr}\left(\hat{W}_2^\top \hat{W}_1 C \hat{W}_1^\top \hat{W}_2\right).$$

For (2) $\iff$ (3), we begin by observing that by submultiplicativity of norm, any W_1, W_2 such that $\|W_1\|_2 \leq 1$ and $\|W_2\|_2 \leq 1$ automatically falls is the optimization space given by $\left\|W_1^\top W_2\right\| \leq 1$ thus giving one direction of the optimization equivalence for free. For the other side we note that given any W_1, W_2 such that $\left\|W_1^\top W_2\right\|_2^2 = \left\|W_1^\top W_2 W_2^\top W_1\right\|_2 \leq 1$, we can construct $\hat{W}_1, \hat{W}_2$ such that $\left\|\hat{W}_i\right\| \leq 1$ and $W_1^\top W_2 W_2^\top W_1 = \hat{W}_1^\top \hat{W}_2 \hat{W}_2^\top \hat{W}_1$. This follows from considering the singular values decomposition of $W_1^\top W_2$, getting $W_1^\top W_2 = U^\top \Sigma V$. As the norm of the product is smaller than 1, all the entries of the singular value matrix Σ are less than 1. Thus depending upon which among d or z is larger we consider either the matrices ΣU and V or the matrices U and ΣV to be our candidate $\hat{W}_1$ and $\hat{W}_2$ respectively.

To complete we will simply have to add zero rows to our choice i.e. say U and ΣV to match the dimensions (i.e. to get a $n \times d$ matrix from a $h \times d$ one).

Finally for (3) $\iff$ (4) we begin by defining $W = W_1^\top W_2$. Then the optimization problem in (3) becomes,

$$\min_{W:\|W\|_2 \leq 1} \operatorname{Tr}\left(W^\top C W\right) = \min_{W:\|W\|_2 \leq 1} \operatorname{Tr}\left(C W W^\top\right).$$

We then prove that we are done if we can prove the claim at optimal of (3) (i.e. the above optimization problem) all the eigenvalues of $WW^\top$ are 1 or 0. Given this claim the singular value decomposition of W becomes only $W = U^\top V$, where if $k = \mathit{rank}(W)$, U is a $k \times d$ matrix and V a $k \times h$ matrix. Additionally by property of SVD, the collumns of U and V are orthonormal. Finally as

$$k = \mathit{rank}(W) \leq \min\{\mathit{rank}(W_1), \mathit{rank}(W_2)\} \leq \min\{d, z\} \leq n,$$

we can add a bunch of zero rows to U and V to get our $n \times d$ and $n \times h$ matrices which will be our corresponding W_1 and W_2.

It remains to prove that $\operatorname{Tr}\left(CWW^\top\right)$ is minimized when all the eigenvalues of $WW^\top$ are 1 or 0. To do this simply decompose

$$WW^\top = \sum_{i=1}^{k} \sigma_i^2 v_i v_i^\top,$$

where v_i is the set of orthonormal eigenvectors of $WW^\top$ corresponding to non-zero eigenvalues of $WW^\top$ (or alternatively non-zero singular values of W) Then

$$\operatorname{Tr}\left(CWW^\top\right) = \operatorname{Tr}\left(C \sum_{i=1}^{k} \sigma_i^2 v_i v_i^\top\right) = \sum_{i=1}^{k} \sigma_i^2 \operatorname{Tr}\left(C v_i v_i^\top\right) = \sum_{i=1}^{k} \sigma_i^2 v_i^\top C v_i.$$

Thus if C has l many strictly negative eigenvalues $\lambda_1 \leq \cdots \leq \lambda_l$ with corresponding eigenvectors $c_1, \ldots, c_l$ and σ_i^2 is positive the above quantity is minimized by choosing as many of these as possible i.e. $v_1 = c_1, \ldots, v_{\min\{d,z,l\}} = c_{\min\{d,z,l\}}$ and setting the corresponding σ_i to be 1 while every setting all other eigen-values to 0.

We then also note by consequence of the above proof that we avoid dimension collapse when possible i.e. when C has multiple strictly negative eigenvalues (which is what one should expect if the data is not one dimensional as $\mathbb{E}[C] = -\mathbb{E}[xx^\top]$) $\square$

A.2.5 Proof of Theorem 4

Recall that $f_{(t)}$ provides the output of the machine at time t and therefore consider the linear and non-linear setting at initialization as

$$\begin{aligned} f^{\mathbb{I}}_{(0)} &= W_2^\top W_1 x && \text{s.t. } W_2^\top W_2 = \mathbb{I}_h,\ W_1^\top W_1 = \mathbb{I}_d; \\ f^{\psi}_{(0)} &= W_2^\top \psi\left(W_1 x\right) && \text{s.t. } W_2^\top W_2 = \mathbb{I}_h,\ W_1^\top W_1 = \mathbb{I}_d. \end{aligned} \tag{A.23}$$

Let $\psi(\cdot)$ be an activation function, such that $\psi(0) = 0$, $\psi'(0) = 1$, and $|\psi''(\cdot)| \le c$. [1] Then at initialization as uniformly random orthogonal matrices

$$\left\| f^{\psi}_{(0)} - f^{\mathbb{I}}_{(0)} \right\| \ \le\ Kc\,\|x\|^2\, d\sqrt{\frac{\log^4 h'}{h'}}$$

where K is an universal constant ψ, d is the feature dimension and h the width of the hidden layer.

Proof. Let us start by defining some properties for the non-linearity: Assume the non-linear function ψ is continuously twice differentiable near 0 and has no bias i.e. $\psi(0) = 0$. Then via scaling we can assume WLOG that $\psi'(0) = 1$. As $|\psi''(x)| \le c$, we get that [2]

$$|\psi(x) - x| \le \frac{cx^2}{2}. \tag{A.24}$$

Before contiuning we first state the following Lemma (see proof [***])

Lemma 7. *Given any $d \le p$, Let Q be a uniformly random $h' \times d$ semi-orthonormal matrix. I.e. Q is the first d columns of an uniformly random $h' \times h'$ orthonormal matrix. Then there are constants L and a sequence ϵ_p converging to 0 as h goes to infinity such that ,*

$$P\left(\max |Q_{i,j}| \ge \frac{L \log h'}{\sqrt{h'}}\right) \le \epsilon_n$$

Now recall that the mapping of the first weight matrix is given by $W_1 : \mathbb{R}^d \to \mathbb{R}^{h'}, \quad h' \gg d$ under the constraint that $W_1^\top W = \boldsymbol{I}$. Under uniformly random initialization by Lemma 7 then with probability asymptotically going to 1 we have that

$$\max\left(W_1\right)^2_{i,j} \le C\frac{\log^2 h'}{h'}$$

[1]This last assumption can also be weakened to say that ψ'' is continuous at 0. See the proof of the theorem for details.

[2]We can actually also use the weaker assumption that $\psi''(0)$ is continuous at 0. Thus there is some bounded (compact) set A containing 0 and a constant c such that $\forall x \in A$, $|\psi(x) - x| \le \frac{cx^2}{2}$

Thus the norm of each row of W_1 we get with a.w.h.p. :

$$\|\text{row}_i\,(W_1)\|^2 = \sum_{j=1}^{d}(W_1)_{i,j}^2 \leq C\frac{d\log^2 h'}{h'}$$

From there we can now write the value of each node in the layer using Cauchy-Schwarz inequality as

$$|\text{row}_i(W_1)\cdot x|^2 \leq \|\text{row}_i\,(W_1)\|^2\,\|x\|^2 \leq C\,\|x\|^2\,\frac{d\log^2 h'}{h'}. \tag{A.25}$$

We now apply the non-linearity to this quantity and denote the output of the first layer after the non-linearity as

$$v_i = \psi\left(\text{row}_i\,(W_1)\cdot x\right)$$

Define the vector $\epsilon \in \mathbb{R}^{h'}$, where

$$\epsilon_j = v_i - \text{row}_i\,(W_1)\cdot x$$

Then we have for h' large enough[3]:

$$\begin{aligned}
\|\epsilon\|^2 &= \sum_{i=1}^{h'}\epsilon_i^2 = \sum_{i=1}^{h'}(v_i - \text{row}_i\,(W_1)\cdot x)^2 \\
&\leq \sum_{i=1}^{h'}\frac{c^2}{4}\left(\text{row}_i\,(W_1)\cdot x\right)^4 && \textit{by equation A.24} \\
&\leq \sum_{i=1}^{h'}\frac{c^2}{4}\left(C\,\|x\|^2\,\frac{d\log^2 h'}{h'}\right)^2 && \textit{by equation A.25} \\
&= K^2c^2\,\|x\|^4\,\frac{h'd^2\log^4 h'}{h'^2} = K^2c^2\,\|x\|^4\,\frac{d^2\log^4 h'}{h'},
\end{aligned}$$

where K is the universal constant $\frac{C}{2}$. Combining this with the second layer we get the difference of the outputs of the two networks as

$$\begin{aligned}
\left\|f^{\psi}_{(0)} - f^{\boldsymbol{I}}_{(0)}\right\| = \left\|W_2^\top v - W_2^\top W_1 x\right\| &= \left\|W_2^\top\left(v - W_1 x\right)\right\| \\
&\leq \|W_2\|\,\|\epsilon\| = \|\epsilon\| && \textit{as } \|W_2\| = 1 \\
&\leq Kc\,\|x\|^2\,d\sqrt{\frac{\log^4 h'}{h'}} \to 0.
\end{aligned}$$

□

[3]Note that for the weaker assumption we can still use equation A.24. This is because by equation A.25,w.h.p. $\text{row}_i(W_1)\cdot x$ goes to 0 and thus $\text{row}_i(W_1)\cdot x \in A$ in limit

A.2.6 Proof of Lemma 7

Let us first restate the lemma: Given any $d \leq p$, Let Q be a uniformly random $h \times d$ semi-orthonormal matrix. I.e. Q is the first d columns of an uniformly random $h \times h$ orthonormal matrix. Then there are constants L and a sequence ϵ_p converging to 0 as h goes to infinity such that ,

$$P\left(\max |Q_{i,j}| \geq \frac{L \log h}{\sqrt{h}}\right) \leq \epsilon_n$$

Proof. We note that it is enough to prove the claim when $d = h'$, i.e. Q is uniformly random $h' \times h'$ orthonormal matrix. Then as our distribution is uniform, the density at any particular Q is same as the density at any UQ where U is some other fixed orthogonal matrix. Thus if q_1 is the first column of Q, the marginal distribution of q_1 has the property that its density at any q_1 is same as that of Uq_1 for any orthogonal matrix U. In other words the marginal distribution for any column of Q is simply that of the uniform unit sphere.

Consider then the following random variable which has the same distribution as that of a fixed column of Q i.e. uniform unit h-sphere. Let $X = (X_1, ..., X_{h'})$ be iid random variables from $\mathcal{N}(0,1)$. Then we know that $X \sim \mathcal{N}(0, \boldsymbol{I}_{h'})$. From the rotational symmetry property of standard gaussian then we have that $\frac{X}{\|X\|}$ is distributed as an uniform sample from the unit sphere in h' dimensions. By union bound then, we have

$$\begin{aligned} P\left(\max_{1\leq i\leq h'} |X_i| \geq t\log h'\right) &\leq \frac{1}{\sqrt{2\pi}} h' e^{-\frac{t^2 \log^2 h'}{2}} \\ \implies P\left(\max_{1\leq i\leq h'} |X_i| \leq t\log h'\right) &\geq 1 - \frac{1}{\sqrt{2\pi}} h' e^{-\frac{t^2 \log^2 h'}{2}}. \end{aligned}$$

As each X_i is iid normal, X_i^2 is iid Chi-square with $\mathbb{E}[X_i^2] = 1$, thus by Chernoff there exists constants C', c' such that

$$P\left(\frac{\sum_{i=1}^{h'} X_i^2}{h'} \geq 1 - s\right) \geq 1 - C' e^{-c' h' s^2}.$$

Since $\max_{1\leq i\leq h'} |X_i| \leq t\log h'$ and $\frac{\sum_{i=1}^{h'} X_i^2}{h'} \leq (1+s)$ implies that $\max_{1\leq i\leq h'} \frac{|X_i|}{\|X\|} \leq \frac{t\log h'}{\sqrt{h'(1-s)}}$, we get that

$$\begin{aligned} P\left(\max_{1\leq i\leq h'} \frac{|X_i|}{\|X\|} \leq \frac{t\log h'}{\sqrt{h'(1-s)}}\right) &\geq 1 - \frac{1}{\sqrt{2\pi}} h' e^{-\frac{t^2\log^2 h'}{2}} - C' e^{-c'h's^2} \\ \implies P\left(\max_{1\leq i\leq h'} \frac{|X_i|}{\|X\|} \geq \frac{t\log h'}{\sqrt{h'(1-s)}}\right) &\leq \frac{1}{\sqrt{2\pi}} h' e^{-\frac{t^2\log^2 h'}{2}} + C' e^{-c'h's^2} \end{aligned}$$

From the argument before that any j'th column of Q is distributed as X. Using the above and another union bound then get us

$$P\left(\max_{1\le i\le h'}\max_{1\le i\le h'}|Q_{i,j}|\ge\frac{t\log h'}{\sqrt{h'(1-s)}}\right)\le\frac{1}{\sqrt{2\pi}}h'e^{-\frac{t^2\log^2 h'}{2}}+C'e^{-c'h's^2}$$
$$\implies P\left(\max_{1\le j\le h'}\max_{1\le i\le h'}|Q_{i,j}|\ge\frac{t\log h'}{\sqrt{h'(1-s)}}\right)\le\frac{1}{\sqrt{2\pi}}{h'}^2e^{-\frac{t^2\log^2 h'}{h'}}e^{-c'h's^2}$$

We note that for any constants t, c' that as h' goes to infinity, both ${h'}^2e^{-\frac{t^2\log^2 h'}{2}}$ and $h'e^{-c'h's^2}$ goes to zero. The proof is then finished by choosing some appropriate constants $s, t \ge 0$. □

A.2.7 Proof of Theorem 5

Let us recall the the general linear trace minimization problem stated in (3.8):

$$\min_{W_2W_1}\operatorname{Tr}\left(W_2^\top W_1CW_1^\top W_2\right)\quad\text{s.t.}\quad W_2^\top W_2=\mathbb{I}_h\text{ and }W_1^\top W_1=\mathbb{I}_d.$$

where $W_1\in\mathbb{R}^{h'\times d}$ and $W_2\in\mathbb{R}^{h'\times h}$ are the trainable weight matrices and $C\in\mathbb{R}^{d\times d}$ a symmetric, data dependent matrices, such that $C=V\Lambda V^\top$ with $V:=[v_1,\dots,v_d]$. Then with $q:=\left[f^{\mathbb{I}}(v_1),\cdots,f^{\mathbb{I}}(v_d)\right]^\top$, where f represents the machine function i.e. $f^{\mathbb{I}}(x)=W_2^\top W_1x$, the learning dynamics of q, the machine outputs are given by

$$\mathring{q}=-2\left[2\Lambda q-\Lambda qq^\top q-qq^\top\Lambda q\right].\tag{A.26}$$

Proof. To simplify notation we are dropping the superscript $\mathbb{I}$ from $f^{\mathbb{I}}_{(t)}$. The u in the following proof is already presumed to be linear. For the same reason we are also dropping the symbol of time, t, from u, W_2, W_1 even though all of them are indeed time dependent. Finally for any time dependent function f, we denote $\frac{\partial f}{\partial t}$ by $\mathring{f}$.

From [EAS98], we get that the derivative of a function γ restricted to a grassmanian is derived by left-multiplying $1-\gamma\gamma^\top$ to the "free" or unrestricted derivative of γ. Using this and recalling that the loss in Eq. 3.8 is given by

$$\mathcal{L}=\operatorname{Tr}\left(W_2^\top W_1CW_1^\top W_2\right),$$

we therefore can write $\mathring{W}_1$ and $\mathring{W}_2$ as

$$\mathring{W}_2(t)=-\left(\boldsymbol{I}-W_2W_2^\top\right)\nabla_{W_2}\mathcal{L}=-2\left(\boldsymbol{I}-W_2W_2^\top\right)\left(W_1CW_1^\top W_2\right)$$
$$\mathring{W}_1(t)=-\left(\boldsymbol{I}-W_1W_1^\top\right)\nabla_{W_1}\mathcal{L}=-2\left(\boldsymbol{I}-W_1W_1^\top\right)\left(W_2W_2^\top W_1C\right).$$

Thus we obtain

$$\begin{aligned}
\frac{\partial\, f_{(t)}(x)}{\partial\, t} =& \mathring{W}_2(t)^\top W_1(t)x + W_2(t)^\top \mathring{W}_1(t)x \\
=& \left(\left(\boldsymbol{I} - W_2 W_2^\top\right)\left(-2W_1 C W_1^\top W_2\right)\right)^\top W_1(t)x \\
&+ W_2(t)^\top \left(\boldsymbol{I} - W_1 W_1^\top\right)\left(-2W_2 W_2^\top W_1 C\right) x \\
=& -2\left(W_2^\top W_1 C \cancel{W_1^\top W_1} x \;+\; \cancel{W_2^\top W_2} W_2^\top W_1 C x\right) \\
&+ 2\left(W_2^\top W_1 C W_1^\top W_2 W_2^\top W_1 x \;+\; W_2^\top W_1 W_1^\top W_2\; W_2^\top W_1 C x\right) \\
=& -2\left(2W_2^\top W_1 C x - W_2^\top W_1 C W_1^\top W_2 W_2^\top W_1 x\right. \\
&\left. - \sum_i^d W_2^\top W_1 v_i v_i^\top W_1^\top W_2\; W_2^\top W_1 C x\right),
\end{aligned}$$

where we obtain the second equality by expanding the terms, taking advantage of $W_2^\top W_2 = \boldsymbol{I}, W_1^\top W_1 = \boldsymbol{I}$ and $\boldsymbol{I}_d = \sum_i^d v_i v_i^\top$. Now setting x as v_j and using the fact that they are eigenvectors for C and using $C = \sum_i^d \lambda_i v_i v_i^\top$ gives us:

$$\mathring{f}(v_j) = -2\left(2\lambda_j f_{(t)}(v_j) - \sum_i^d \lambda_i f_{(t)}(v_i) f_{(t)}(v_i)^\top f_{(t)}(v_j) - \lambda_j \sum_i^d f_{(t)}(v_i) f_{(t)}(v_i)^\top f_{(t)}(v_j)\right)$$

Let's rewrite this in matrix notation. First define $q := [f(v_1), \ldots f(v_d)]^\top$ thus obtaining:

$$\mathring{q} = -2\left[2\Lambda q - \Lambda q q^\top q - q q^\top \Lambda q\right]$$

which concludes the proof. $\square$

A.2.8 Proof of Theorem 6

Let $h = 1$ then our update rule simplifies to

$$\frac{\mathring{q}}{2} = -(1 - q^\top q)\Lambda q - (\mathbb{I} - q q^\top)\Lambda q. \tag{A.27}$$

We can distinguish two cases:

- Assume all the eigenvalues of Λ are strictly positive then q converges to 0.
- Assume there is at least one negative eigenvalue of Λ, then q becomes the smallest eigenvector, e_1.

Proof. For instance first suppose that all the eigenvalues of Λ are strictly positive and thus $q^\top \Lambda q > 0$. Then

$$\frac{d(q^\top q)}{dt} = 2q^\top \mathring{q} = 4\left[-(1 - q^\top q) q^\top \Lambda q - q^\top (\boldsymbol{I} - q q^\top)\Lambda q\right] = -8(1 - q^\top q) q^\top \Lambda q$$

Observing now that because of orthonormality of our weight matrices, $q^\top q = \|q\|^2 \leq 1$ we get that the derivative of $\|q\|^2$ is always negative and thus q converges to 0.

Now suppose on the other hand there is atleast one negative eigenvalue. WLOG let e_1 denote the eigenvector with the smallest eigenvalue (which is negative). Then

$$\begin{aligned}\frac{d(e_1^\top q)}{dt} &= e_1^\top \mathring{q} = 2\big[-(1-q^\top q)e_1^\top \Lambda q - e_1^\top(\boldsymbol{I} - qq^\top)\Lambda q\big] \\ &= 2\big[(1-q^\top q)(-\lambda_1)e_1^\top q + (q^\top \Lambda q - \lambda_1)e_1^\top q)\big]\end{aligned}$$

We now note that $q^\top \Lambda q - \lambda_1 \geq 0$ as λ_1 is the smallest eigenvalue. Thus as $-\lambda_1$ is positive, the derivative of $e_1^\top q$ is always positive unless $1 - q^\top q = q^\top \Lambda q - \lambda_1 = 0$, which only happens at $q = e_1$. In other words, eventually q becomes the smallest eigenvector e_1. □

A.3 Proofs for Chapter 4

A.3.1 Proof of Theorem 9

Given data X and an embedding dimension $h \in \mathbb{N}$, let $\mathcal{L} : \mathcal{H}^h \to \mathbb{R}$ be a loss function that vanishes on $\mathcal{H}_X^\perp$. Assume $\dim(\mathcal{H}_X^\perp) \geq h$. Consider the following constrained minimisation problem over $w_1, \dots, w_h \in \mathcal{H}$

$$\begin{aligned} &\text{minimise } \mathcal{L}(w_1, \dots, w_h) \\ &\text{s.t.} \quad W^* W = \boldsymbol{I}_h \end{aligned} \tag{A.28}$$

Furthermore, consider the inequality-constrained problem over $\mathcal{H}_X$

$$\begin{aligned} &\text{minimise } \mathcal{L}(w_1, \dots, w_h) \\ &\text{s.t.} \quad W^T W \preceq \boldsymbol{I}_h \text{ and } w_1, \dots, w_h \in \mathcal{H}_X \end{aligned} \tag{A.29}$$

Then, every minimiser of (A.28) is contained in $\mathcal{H}_X^h$ if and only if every minimiser of (A.29) satisfies $W^T W = \boldsymbol{I}_h$.

Let us prove our characterization of loss functionals that admit representer theorems under orthonormality constraints. Denote $P_X : \mathcal{H} \to \mathcal{H}_X$ for the projection onto the finite-dimensional subspace $\mathcal{H}_X$. Recall that P_X is a bounded linear operator with operator norm $\|P_X\| = 1$, satisfying $P_X^2 = P_X$.

Proof. Let us begin by assuming that a collection of functions $w_1, \dots, w_h$ is a minimiser of

$$\begin{aligned} &\text{minimise } \mathcal{L}(w_1, \dots, w_h) \\ &\text{s.t.} \quad W^* W = \boldsymbol{I}_h \end{aligned} \tag{A.30}$$

that is not contained in $\mathcal{H}_X$. Then, $P_X w_1, \dots, P_X w_h \in \mathcal{H}_X$ achieve the same minimum, because $\mathcal{L}$ vanishes outside of $\mathcal{H}_X$. Because not all $w_1, \dots, w_h$ are contained in $\mathcal{H}_X$, $W^* W \neq \boldsymbol{I}_h$. Thus, we have found a minimiser of

$$\begin{aligned} &\text{minimise } \mathcal{L}(w_1, \dots, w_h) \\ &\text{s.t.} \quad W^T W \preceq \boldsymbol{I}_h \text{ and } w_1, \dots, w_h \in \mathcal{H}_X \end{aligned} \tag{A.31}$$

that does not satisfy the constraint with equality. For the other direction, assume that there exists a minimiser $(w_1, \dots, w_h) \in \mathcal{H}_X$ of (A.31) that does not satisfy the constraint with equality. Then, exploiting the fact that $\dim(\mathcal{H}_X^\perp) \geq h$, we can simply add $V_1, \dots, V_h \in \mathcal{H}_X^\perp$ to $w_1, \dots, w_h$ without changing $\mathcal{L}$, while at the same time ensuring $\langle w_i + V_i, w_j + V_j \rangle = 1_{i=j}$. This new minimiser of $\mathcal{L}$ is not contained in $\mathcal{H}_X$ and hence there is no representer theorem. □

Remark 14. As mentioned in the main paper, checking that optimisers W of (A.31) are indeed orthonormal can be done by analyzing the behaviour of the loss functional $\mathcal{L}$ under orthonormalization of a given solution W with $W^TW \neq \boldsymbol{I}_h$. To illustrate this briefly in a finite-dimensional setting, consider the trace loss

$$\mathcal{L}(W) = -\operatorname{Tr}(W^TAW)$$

for some $A \succeq 0$. If $W^TW \preceq \boldsymbol{I}_h$, then we can orthonormalize it via $X = WV^{-1}$, where $VV^T = W^TW$. Then, $X^TX = \boldsymbol{I}_h$ and

$$\mathcal{L}(X) = \operatorname{Tr}(W^TAW(VV^T)^{-1}) = \operatorname{Tr}(W^TAW(W^TW)^{-1}) \leq \mathcal{L}(W)$$

where the final inequality follows from the fact that $W^TW \neq \boldsymbol{I}_h$.

A.3.2 Proof of Theorem 10

Consider the optimisation problem as stated in Definition 4. Let $X, X^+, X^- \in \mathbb{R}^{d\times n}$ denote the data corresponding to the anchors, positive and negative samples, respectively. Define the kernel matrices

$$\begin{aligned}
K &= [k(x_i, x_j)]_{i,j} & K_- &= \left[k(x_i, x_j^-)\right]_{i,j} \\
K_+ &= \left[k(x_i, x_j^+)\right]_{i,j} & K_{--} &= \left[k(x_i^-, x_j^-)\right]_{i,j} \\
K_{++} &= \left[k(x_i^+, x_j^+)\right]_{i,j} & K_{-+} &= \left[k(x_i^-, x_j^+)\right]_{i,j}
\end{aligned}$$

Furthermore, define the matrices

$$K_\Delta = K_{--} + K_{++} - K_{-+} - K_{-+}^T \qquad K_1 = \begin{bmatrix} K & K_- - K_+ \\ K_3 & K_\Delta \end{bmatrix}$$

$$B = \begin{bmatrix} K_- - K_+ \\ K_\Delta \end{bmatrix} \cdot \begin{bmatrix} K & K_- - K_+ \end{bmatrix} \qquad K_2 = -\frac{1}{2}\left(B + B^T\right).$$

Let A_2 consist of the top h eigenvectors of the matrix $K_1^{-1/2}K_2K_1^{-1/2}$, which we assume to have h non-negative eigenvalues. Let $A = K_1^{-1/2}A_2$. Then, at optimal parameterization, the embedding of any $x^* \in \mathbb{R}^d$ can be written in closed form as

$$\boldsymbol{z}^* = A^T \begin{bmatrix} k(x^*, X) \\ k(x^*, X^-) - k(x^*, X^+) \end{bmatrix}$$

Proof. Define the (possibly infinite-dimensional) matrices $\Phi = [\phi(x_1), \dots, \phi(x_n)]$ and $\Delta = [\phi(x_1^-) - \phi(x_1^+), \dots, \phi(x_n^-) - \phi(x_n^+)]$, where $\phi : \mathbb{R}^d \to \mathcal{H}$ is the canonical feature map associated with the given kernel and $\mathcal{H}$ is the corresponding RKHS. We start by deriving the optimal parameterization. To do so recall the loss problem setup:

$$\begin{aligned} &\min_W \sum_{i=1}^n f(x_i)^T \left(f(x_i^-) - f(x_i^+)\right) \\ &\text{where} \quad f(x_i) = W^T\phi(x_i) \quad \text{s.t.} \quad W^*W = \boldsymbol{I}_h \end{aligned} \tag{A.32}$$

By virtue of the representer theorem under orthonormality constraints, we may reduce this to a finite-dimensional optimisation problem on the span of (Φ, Δ), change the constraints to $W^TW \preceq I_h$ for the moment, and finally verify that the optimal solution does in fact satisfy $W^TW = \boldsymbol{I}_h$. If that is the case, we know that this solution is also the minimiser of (A.32) over the entire space $\mathcal{H}$. Hence, let us assume that there exists $A \in \mathbb{R}^{2n\times h}$ such that

$$W = [\Phi, \Delta]A$$

Thus, denoting $K_1 = [\Phi, \Delta]^T[\Phi, \Delta] \in \mathbb{R}^{2n\times 2n}$, we may rewrite our optimisation problem as

$$\min_A \sum_{i=1}^n \phi(x_i)^T[\Phi, \Delta]AA^T[\Phi, \Delta]^T \left(\phi(x_i^-) - \phi(x_i^+)\right) \tag{A.33}$$

$$\text{s.t.} \quad A^TK_1A \preceq \boldsymbol{I}_h \tag{A.34}$$

This is equivalent to

$$\begin{aligned} &\min_A \operatorname{Tr}\left(A^T[\Phi, \Delta]^T\Delta\Phi^T[\Phi, \Delta]A\right) \\ &\text{s.t.} \quad A^TK_1A \preceq \boldsymbol{I}_h \end{aligned}$$

Denoting $B = [\Phi, \Delta]^T\Delta\Phi^T[\Phi, \Delta]$ and $K_2 = -\frac{1}{2}\left(B + B^T\right)$ for the negative symmetric part of B, we are left with the trace maximization

$$\begin{aligned} &\max_A \operatorname{Tr}\left(A^TK_2A\right) \\ &\text{s.t.} \quad A^TK_1A \preceq \boldsymbol{I}_h \end{aligned}$$

Writing $A_2 = K_1^{1/2}A \in \mathbb{R}^{2n\times h}$, this simplifies to

$$\begin{aligned} &\max_{A_2} \operatorname{Tr}\left(A_2^TK_1^{-1/2}K_2K_1^{-1/2}A_2\right) \\ &\text{s.t.} \quad A_2^TA_2 \preceq \boldsymbol{I}_h \end{aligned}$$

This expression is maximized when A_2 consists of the top h orthonormal eigenvectors of $K_1^{-1/2}K_2K_1^{-1/2}$, which do in fact satisfy $A_2^TA_2 = \boldsymbol{I}_h$, and hence $W = [\Phi, \Delta]A$ also satisfies $W^TW = \boldsymbol{I}_h$ with equality. Note that both K_1 and K_2 depend only on inner products in the RKHS and can hence be computed directly from the kernel k, and that there is no need for evaluating the feature map directly. Finally, $A = K_1^{-1/2}A_2$ becomes the desired minimiser of A.33. For a new point x^*, it holds that

$$\begin{aligned} f(x^*) &= W^T\phi(x^*) \\ &= A^T[\Phi, \Delta]^T\phi(x^*) \\ &= A^T\begin{bmatrix} k(x^*, X) \\ k(x^*, X^-) - k(x^*, X^+) \end{bmatrix} \end{aligned}$$

which again requires only knowledge of the kernel, and not of the (implicit) feature map. □

A.3.3 Proof of Theorem 11

Consider the optimisation problem as stated in Definition 5, with K denoting the kernel matrix. Then, we can equivalently minimise the objective w.r.t. the embeddings $Z \in \mathbb{R}^{h\times 3n}$. Denoting by $\boldsymbol{z}_1, \dots, \boldsymbol{z}_{3n}$ the columns of Z, the loss to be minimised becomes

$$\min_{Z\in\mathbb{R}^{h\times 3n}} \sum_{i=1}^{n} -2\boldsymbol{z}_i^T\boldsymbol{z}_{i+n} + \left(\boldsymbol{z}_i^T\boldsymbol{z}_{i+2n}\right)^2 + \lambda \cdot \operatorname{Tr}\left(ZK^{-1}Z^T\right)$$

The gradient of the loss function in terms of Z is therefore given by

$$2\lambda ZK^{-1} + \begin{cases} -2\boldsymbol{z}_{i+n} + 2(\boldsymbol{z}_i^T\boldsymbol{z}_{i+2n})\boldsymbol{z}_{i+2n} & , i \in [n] \\ -2\boldsymbol{z}_{i-n} & , i \in [n+1, 2n] \\ 2(\boldsymbol{z}_i^T\boldsymbol{z}_{i-2n})z_{i-2n} & , i \in [2n+1, 3n] \end{cases}$$

For any new point $x^* \in \mathbb{R}^d$, the trained model maps it to

$$\boldsymbol{z}^* := ZK^{-1}k(X, x^*).$$

Proof. Recall that in contrastive learning with the spectral contrastive loss, we learn a representation of the form $f_W(x) = W^T\phi(x) = (w_1(x), \dots, w_h(x))$ by optimising the following objective function:

$$\mathcal{L}^{\text{Sp}} := \sum_{i=1}^{n} -2f_W(x_i)^T f_W(x_i^+) + \left(f_W(x_i)^T f_W(x_i^-)\right)^2 + \lambda \left\|W\right\|_{\mathcal{H}}^2.$$

Since the term $\sum_{i=1}^n -2f_W(x_i)^T f_W(x_i^+) + \left(f_W(x_i)^T f_W(x_i^-)\right)^2$ vanishes for any choice of $w_1, \dots, w_h \in \mathcal{H}_X^{\perp}$, and we add a norm regularization to this objective function, it is clear by the representer theorem that any minimiser of $\mathcal{L}^{\text{Sp}}$ must consist of h functions from $\mathcal{H}_X$. We may hence write $w_j = \Phi\boldsymbol{a}_j$ for some $\boldsymbol{a}_j \in \mathbb{R}^n$. Denoting Z for the embeddings under the map $W^T\phi(x_i)$ and A for the matrix with columns $\boldsymbol{a}_1, \dots, \boldsymbol{a}_h$, we see that for any $j \in [h]$, it must hold that $A^TK = Z$ and hence $A = K^{-1}Z^T$. Thus, $W = \Phi K^{-1}Z^T$ and $\|W\|^2 = \operatorname{Tr}(ZK^{-1}KK^{-1}Z^T) = \operatorname{Tr}(ZK^{-1}Z^T)$. This allows us to reformulate the minimisation problem as an optimisation over the embedded points Z, yielding the gradients

$$\nabla \mathcal{L}^{\mathrm{Sp}} = 2\lambda Z K^{-1} + \begin{cases} -2\boldsymbol{z}_{i+n} + 2(\boldsymbol{z}_i^T \boldsymbol{z}_{i+2n})\boldsymbol{z}_{i+2n} & , i \in [n] \\ -2\boldsymbol{z}_{i-n} & , i \in [n+1, 2n] \\ 2(\boldsymbol{z}_i^T \boldsymbol{z}_{i-2n}) z_{i-2n} & , i \in [2n+1, 3n] \end{cases}$$

Finally, by virtue of our choice of $W = \Phi K^{-1} Z^T$, any new point $x^* \in \mathbb{R}^d$ is mapped to

$$\boldsymbol{z}^* := Z K^{-1} k(X, x^*).$$

□

A.3.4 Proof of Theorem 12

Let $\mathcal{F} := \left\{ X \mapsto W^T \phi(X) : \|W^T\|_{\mathcal{H}} \leq \omega \right\}$ be the class of embedding functions we consider in the contrastive setting. Define $\alpha := \left(\sqrt{h \operatorname{Tr}[K_X]} + \sqrt{h \operatorname{Tr}[K_{X^-}]} + \sqrt{h \operatorname{Tr}[K_{X^+}]} \right)$ as well as $\kappa := \max_{x_i' \in \{x_i, x_i^-, x_i^+\}_{i=1}^n} k(x_i', x_i')$. We then obtain the generalisation error for the proposed losses as follows.

1. **Simple Contrastive Loss.** Let the loss be given by Definition 4. Then, for any $\delta > 0$, the following statement holds with probability at least $1 - \delta$ for any $f \in \mathcal{F}$:

$$\mathcal{L}^{\mathrm{Si}}(f) \leq \widehat{\mathcal{L}}^{\mathrm{Si}}(f) + O\left(\frac{\omega^2 \sqrt{\kappa} \alpha}{n} + \omega^2 \kappa \sqrt{\frac{\log \frac{1}{\delta}}{n}} \right)$$

2. **Spectral Contrastive Loss.** Let the loss be given by Definition 5. Then, for any $\delta > 0$, the following statement holds with probability at least $1 - \delta$ for any $f \in \mathcal{F}$:

$$\mathcal{L}^{Sp}(f) \leq \widehat{\mathcal{L}}^{Sp}(f) + O\left(\lambda \omega^2 + \frac{\omega^3 \kappa^{\frac{3}{2}} \alpha}{n} + \omega^4 \kappa^2 \sqrt{\frac{\log \frac{1}{\delta}}{n}} \right)$$

Proof. **Part 1. Simple Contrastive Loss.** We start from the following Lemma, that is defined in the context of the simple contrastive loss:

Lemma 8 ([Aro+19c]). *With probability at least $1 - \delta$ over the training set, $\forall f \in \mathcal{F}$:*

$$\mathcal{L}_{un} \leq \widehat{\mathcal{L}}_{un} + O\left(R \frac{\mathfrak{R}_c(\mathcal{F}, \mathbf{x})}{n} + B \sqrt{\frac{\log \frac{1}{\delta}}{n}} \right)$$

where $\|f(\cdot)\| \leq R$ and B the bound on the loss function wich in the case of the simple contrastive loss can be given by $B = O(R^2)$.

Furthermore the the Rademacher complexity term in the above lemma is defined over the following definition.

Definition 7 (Expected Rademacher Complexity for Contrastive Setting (following [Aro+19c])). Let our dataset be consistent of triplets $\mathcal{S} = \left\{x_i, x_i^+, x_i^-\right\}_{i=1}^n$ and

$$f_{|\mathcal{S}} = \left(f_t\left(x_j\right), f_t\left(x_j^+\right), f_t\left(x_j^-\right)\right)_{j\in[n], t\in[h]} \in \mathbb{R}^{3hn}$$

be the restriction of and $f \in \mathcal{F}$ to S, then we define the empirical Rademacher Complexity as

$$\mathfrak{R}_c(\mathcal{F}, \mathbf{x}) = \underset{\sigma\sim\{\pm 1\}^{3nh}}{\mathbb{E}}\left[\sup_{f\in\mathcal{F}}\sum_{i=1}^{3dn}\sigma_i f_{|\mathcal{S}}\left(x_i\right)\right].$$

Having the setup complete we can now compute the complexity term of the in this paper considered kernel function. In the first step we pluck in our considered model and split it up by the reference, positive and negative samples:

$$\begin{aligned}
&\mathfrak{R}_c(\mathcal{F}, \mathbf{x})\\
&= \underset{\sigma\sim\{\pm 1\}^{3nd}}{\mathbb{E}}\left[\sup_{f\in\mathcal{F}}\sum_{i=1}^{3hn}\sigma_i f_{|\mathcal{S}}\left(x_i\right)\right]\\
&= \underset{\sigma\sim\{\pm 1\}^{3nd}}{\mathbb{E}}\left[\sup_{W:\|W\|_{\mathcal{H}}\le\omega}\sum_{j=1}^{h}\sum_{i=1}^{n}\sigma_i W_{\cdot,j}^T\phi\left(x_i\right) + \sigma_{i+n}W_{\cdot,j}^T\phi\left(x_i^-\right) + \sigma_{i+2n}W_{\cdot,j}^T\phi\left(x_i^+\right)\right]\\
&\le \omega\left(\mathbb{E}_\sigma\left[h\left\|\sum_{i=1}^n\sigma_i\phi\left(x_i\right)\right\|\right] + \mathbb{E}_\sigma\left[h\left\|\sum_{i=1}^n\sigma_i\phi\left(x_i^-\right)\right\|\right] + \mathbb{E}_\sigma\left[h\left\|\sum_{i=1}^n\sigma_i\phi\left(x_i^+\right)\right\|\right]\right)\\
&\hspace{12em}\text{(by linearity of expectation and Cauchy-Schwartz inequality)}\\
&= \omega\left(\mathbb{E}_\sigma\left[\sqrt{h\left\|\sum_{i=1}^n\sigma_i\phi\left(x_i\right)\right\|^2}\right] + \mathbb{E}_\sigma\left[\sqrt{h\left\|\sum_{i=1}^n\sigma_i\phi\left(x_i^-\right)\right\|^2}\right] + \mathbb{E}_\sigma\left[\sqrt{h\left\|\sum_{i=1}^n\sigma_i\phi\left(x_i^+\right)\right\|^2}\right]\right)\\
&\le \omega\left(\sqrt{h\mathbb{E}_\sigma\left[\left\|\sum_{i=1}^n\sigma_i\phi\left(x_i\right)\right\|^2\right]} + \sqrt{h\mathbb{E}_\sigma\left[\left\|\sum_{i=1}^n\sigma_i\phi\left(x_i^-\right)\right\|^2\right]} + \sqrt{h\mathbb{E}_\sigma\left[\left\|\sum_{i=1}^n\sigma_i\phi\left(x_i^+\right)\right\|^2\right]}\right)\\
&\hspace{24em}\text{(Jensen's inequality)}\\
&= \omega\left(\sqrt{h\,\mathrm{Tr}\left[K_X\right]} + \sqrt{h\,\mathrm{Tr}\left[K_{X^-}\right]} + \sqrt{h\,\mathrm{Tr}\left[K_{X^+}\right]}\right).
\end{aligned}$$

Secondly we have to bound the quantity $\|f(\cdot)\| \leq R$:

$$\begin{aligned}
\|f(\cdot)\| &= \left\|W^T\phi(x)\right\| && \text{(by definition of considered embedding function)}\\
&\leq \left\|W^T\right\|\|\phi(x)\| \\
&\leq \omega\|\phi(x)\| && \text{(by definition of function class } \left\|W^T\right\| \text{ is bound)}\\
&\leq \omega\sqrt{\langle\phi(x)^T\phi(x)\rangle} \\
&\leq \omega\sqrt{\max_{x_i}.k(x_i,x_i)} && \text{(bounding over all possible } x_i\text{)}
\end{aligned}$$

Defining $\kappa := \max_{x_i' \in \{x_i, x_i^-, x_i^+\}_{i=1}^n} k(x_i', x_i')$ to account for reference, positive and negative samples and combining all results concludes this part of the proof. $\square$

Proof. **Part 2. Spectral Contrastive Loss.**

The overall proof structure follows the one presented above for the simple contrastive loss, however Lemma 8 is define for the simple contrastive loss. Therefore we will adept the proof of [Aro+19c] Lemma A.2. to obtain the following lemma for the spectral contrastive loss.

Lemma 9. *With probability at least* $1-\delta$ *over the training set,* $\forall f \in \mathcal{F}$*:*

$$\mathcal{L}_{un} \leq \widehat{\mathcal{L}}_{un} + O\left(R^3\frac{\mathfrak{R}_c(\mathcal{F},\mathbf{x})}{n} + B\sqrt{\frac{\log\frac{1}{\delta}}{n}}\right) \tag{A.35}$$

where $\mathfrak{R}_c(\mathcal{F},\mathbf{x})$ *is the Vector Rademacher Complexity where* $\|f(\cdot)\| \leq R$ *and* $B = O(R^4)$*.*

Before proofing Lemma 9 we first recall the following Lemma:

Lemma 10 (Corollary 4 in [Mau16])**.** *Let* Z *be any set and* $\mathcal{S} = \{z_j\}_{j=1}^n \in Z^n$*. Let* $\widetilde{\mathcal{F}}$ *be a class of functions* $\tilde{f} : Z \to \mathbb{R}^d$ *and* $h : \mathbb{R}^d \to \mathbb{R}$ *be L-Lipschitz. For all* $\tilde{f} \in \widetilde{\mathcal{F}}$*, let* $g_{\tilde{f}} = h \circ \tilde{f}$*. Then*

$$\underset{\sigma\sim\{\pm1\}^n}{\mathbb{E}}\left[\sup_{\tilde{f}\in\tilde{\mathcal{F}}}\left\langle\sigma,\left(g_{\tilde{f}}\right)_{|\mathcal{S}}\right\rangle\right] \leq \sqrt{2}L \underset{\sigma\sim\{\pm1\}^{dn}}{\mathbb{E}}\left[\sup_{\tilde{f}\in\tilde{\mathcal{F}}}\left\langle\sigma,\tilde{f}_{|\mathcal{S}}\right\rangle\right] \tag{A.36}$$

where $\tilde{f}_{|\mathcal{S}} = \left(\tilde{f}_t\left(z_j\right)\right)_{t\in[d],j\in[n]}$.

We start by considering the classical Rademacher complexity based generalization error. For a real function class $\mathcal{G}$ whose functions map from a set Z to $[0,1]$ and for any $\delta > 0$, if $\mathcal{S}$ is a training set composed by n i.i.d. samples $\{z_j\}_{j=1}^n$, then with probability at least $1-\frac{2}{\delta}$,for all $g \in \mathcal{G}$

$$\mathbb{E}[g(z)] \leq \frac{1}{n}\sum_{j=1}^n g\left(z_i\right) + \frac{2\mathfrak{R}_{\mathcal{S}}(\mathcal{G})}{n} + 3\sqrt{\frac{\log\frac{4}{\delta}}{2n}}$$

where $\mathfrak{R}_{\mathcal{S}}(G)$ is the standard Rademacher complexity. We can apply this to our setting by considering $Z = \mathcal{X}^3$ and defining the function class as

$$\mathcal{G} = \left\{ g_f\left(x, x^+, x^-\right) = \frac{1}{B}\left(f(x)^T f\left(x^+\right) - \left(f\left(x_i\right)^T f\left(x_i^-\right)\right)^2 \right) \mid f \in \mathcal{F} \right\}.$$

Now to show (A.35) consider some universal constant c we have to show $\mathfrak{R}_{\mathcal{S}}(\mathcal{G}) \leq c\frac{R^3}{B}\mathfrak{R}_{\mathcal{S}}(\mathcal{G})$ or equivalently

$$\underset{\sigma \sim \{\pm 1\}^n}{\mathbb{E}}\left[\sup_{f \in \mathcal{F}} \left\langle \sigma, \left(g_f\right)_{|\mathcal{S}} \right\rangle\right] \leq c\frac{R^3}{B} \underset{\sigma \sim \{\pm 1\}^{3dn}}{\mathbb{E}}\left[\sup_{f \in \mathcal{F}} \left\langle \sigma, f_{|\mathcal{S}} \right\rangle\right] \tag{A.37}$$

where $\left(g_f\right)_{|\mathcal{S}} = \left\{ g_f\left(x_j, x_j^+, x_1^-\right)\right\}_{j=1}^n$. We can now observe by setting $Z = \mathcal{X}^3, b = 3d$ and

$$\widetilde{\mathcal{F}} = \left\{ \tilde{f}\left(x, x^+, x^-\right) = \left(f(x), f\left(x^+\right), f\left(x^-\right),\right) \mid f \in \mathcal{F} \right\}$$

and using $g_{\tilde{f}} = g_f$ that (A.36) and (A.37) exactly coincide and we need to show $L \leq \frac{c}{\sqrt{2}}\frac{R^3}{B}$ for some constant c. Now for $z = (x, x^+, x^-)$ we have $g_{\tilde{f}}(z) = \frac{1}{B}\psi(\tilde{f}(z))$ where $\psi : \mathbb{R}^{(1+2)d} \to \mathbb{R}$ with $\psi\left(\left(v_t, v_t^+, v_t^-\right)_{t \in [d]}\right) = \sum_t -v_t v_t^+ + \left(v_t v_t^-\right)^2$. We can now show that ψ is R^3 lipschitz where $\sum_t v_t^2, \sum_t \left(v_t^+\right)^2, \sum_t \left(v_t^-\right)^2 \leq R^2$ by computing its Jacobian. To do so we derive $\frac{\partial \psi_i}{\partial v_t^+} = -v_t \quad \frac{\partial \psi_i}{\partial v_t} = -v_t^+ + v_t^- v_t^- v_t$ and $\frac{\partial \psi_i}{\partial v_t^-} = v_t v_t v_t^-$ and get by triangle inequality the Frobenius norm on the Jacobian J of ψ

$$\|J\|_F \leq \sqrt{\sum_t (v_t^-)^4 (v_t)^2 + (v_t^-)^3 v_t^3 + (v_t^-)^2 v_t^4 + (v_t^+)^2 + (v_t^+) v_t + (v_t)^2} = O(R^3).$$

Finally using $\|J\|_2 \leq \|J\|_F$ bounds the lipschitzness and concludes the proof of Lemma 9.

As the function class we consider for embedding does not change

$$\left(\mathcal{F} := \left\{X \mapsto W^T\phi(X) : \|W^T\|_{\mathcal{H}} \leq \omega\right\}\right)$$

, we again obtain $\mathfrak{R}_c(\mathcal{F}, \mathbf{x}) \leq \omega\left(\sqrt{h \operatorname{Tr}\left[K_X\right]} + \sqrt{h \operatorname{Tr}\left[K_{X^-}\right]} + \sqrt{h \operatorname{Tr}\left[K_{X^+}\right]}\right)$ and $\omega\sqrt{\kappa} \leq R$, which combined with the above Lemma 9 concludes the proof. □

A.3.5 Proof Corollary 1

Let $t = \underset{c,c' \sim \rho^2}{\mathbb{E}} \mathbf{1}\left\{c = c'\right\}$ and $\tau := \frac{1}{(1-t)}$ be the probability that two classes sampled independently from ρ are the same. Again define $\alpha := \left(\sqrt{h \operatorname{Tr}\left[K_X\right]} + \sqrt{h \operatorname{Tr}\left[K_{X^-}\right]} + \sqrt{h \operatorname{Tr}\left[K_{X^+}\right]}\right)$. In the following let $\mathcal{L}_{sup}$ be the loss of the *supervised downstream task.*

1. **Simple Contrastive Loss.** Let $\widehat{\mathcal{L}}^{Si}$ be the simple contrastive loss as defined in Definition 4. Then for any $\delta > 0$, the following statement holds with probability at

least $1-\delta$ for any $f \in \mathcal{F}$:

$$\mathcal{L}^{\text{Si}}_{sup} \leq \tau \left(\widehat{\mathcal{L}}^{\text{Si}}_{un} - t \right) + \tau O \left(\frac{\omega^2 \sqrt{\kappa}\alpha}{n} + \omega^2 \kappa \sqrt{\frac{\log \frac{1}{\delta}}{n}} \right)$$

2. **Spectral Contrastive Loss.** Let $\widehat{\mathcal{L}}^{\text{Sp}}$ be the spectral contrastive loss as defined in Definition 5. For any $\delta > 0$, the following statement holds with probability at least $1-\delta$ for any $f \in \mathcal{F}$:

$$\mathcal{L}^{\text{Sp}}_{sup} \leq \tau \left(\widehat{\mathcal{L}}^{\text{Sp}}_{un} - t \right) + \tau O \left(\frac{\omega^3 \kappa^{\frac{3}{2}} \alpha}{n} + \omega^6 \kappa^3 \sqrt{\frac{\log \frac{1}{\delta}}{n}} \right)$$

Proof. Before we state the bound let us formally define the supervised task. We consider a two-class classification task $\mathcal{T}$ with $\{c_1, c_2\}$ distinct classes and a linear classifier on top of the learned representation. Let this function given by $V \in \mathbb{R}^{2\times h}$. In the following let X_c be a datapoint X belonging to class c.

$$\mathcal{L}_{\text{sup}}\left(\mathcal{T}, f\right) = \inf_{V} \mathop{\mathbb{E}}_{(X,c)} \left[Vf(X_c) - Vf(X_{c'}) \mid c_i \neq c_j \right]$$

From there we can furthermore define the average supervised loss as taking the expectation over the distribution of classes. The average loss for a function f on a binary classification task tasks is defined as

$$\mathcal{L}_{\text{sup}}\left(f\right) := \mathop{\mathbb{E}}_{\{c_1,c_2\}\sim\rho^2} \left[L_{sup}\left(\{c_1, c_2\}, f\right) \mid c_i \neq c_j \right]$$

where the latent class distribution is given by ρ.

In the following let $\mu_c = \mathop{\mathbb{E}}_{x\sim\mathcal{D}_c} f(x)$ be the mean of class c and $f(X)$ the embedding function.

1. **Simple Contrastive Loss.** This lemma is directly proven for the simple contrastive loss in [Aro+19c]. We will first restate the proof for the simple contrastive loss from completeness. We can now bound the unsupervised loss:

$$\begin{aligned}
\mathcal{L}_{un}(f) &= \underset{\substack{(X,X^+)\sim\mathcal{D}_{\text{sim}}\\ X^-\sim\mathcal{D}_{\text{neg}}}}{\mathbb{E}} \left[f(X)^T\left(f\left(X^+\right)-f\left(X^-\right)\right)\right]\\
&= \underset{\substack{c^+,c^-\sim\rho^2\\ x\sim\mathcal{D}_{c^+}}}{\mathbb{E}}\ \underset{\substack{x^+\sim\mathcal{D}_{c^+}\\ x^-\sim\mathcal{D}_{c^-}}}{\mathbb{E}} \left[f(X)^T\left(f\left(X^+\right)-f\left(X^-\right)\right)\right]\\
&\geq \underset{c^+,c^-\sim\rho^2}{\mathbb{E}}\ \underset{X\sim\mathcal{D}_{c^+}}{\mathbb{E}} \left[f(x)^T\left(\mu_{c^+}-\mu_{c^-}\right)\right]\\
&= (1-\tau)\underset{c^+,c^-\sim\rho^2}{\mathbb{E}} \left[L_{\text{sup}}^{\mu}\ \left(\left\{c^+,c^-\right\},f\right)\mid c^+\neq c^-\right]+\tau\\
&= (1-\tau)L_{sup}(f)+\tau
\end{aligned}$$

The bound then follows directly from Theorem 14 and the above results.

2. **Spectral Contrastive Loss.** We follow the same general idea as in *Case 1*, with changed loss function. Observe that we can bound the spectral by the simple contrastive loss with an additional constant. While this is a very rough bound in this setting we are only interested in bounding the unsupervised by the supervised loss and constants are observed into the big O notation. The bound then follows directly from Theorem 14 and the above results.

This concludes the proof. □

A.4 Proofs for Chapter 5

A.4.1 Proof of Theorem 13

For any bottleneck $Z \in \mathbb{R}^{h\times n}$, define the reconstruction

$$Q(Z) = X(K_Z + \lambda I_n)^{-1}K_Z$$

Learning the Kernel AE from Definition 6 is then equivalent to minimising the following expression over all possible embeddings $Z \in \mathbb{R}^{h\times n}$:

$$\begin{aligned}&\|Q(Z) - X\|^2 + \lambda \operatorname{Tr}\left(ZK_X^{-1}Z^T + QK_Z^{-1}Q^T\right)\\ &\text{s.t.} \quad \|z_i\|^2 = 1 \ \forall i \in [n]\end{aligned}$$

Given Z, any new $x^* \in \mathbb{R}^d$ is embedded in the bottleneck as

$$z^* = ZK_X^{-1}k(x^*, X)$$

and reconstructed as

$$\hat{x}^* = X\left(K_Z + \lambda \boldsymbol{I}_n\right)^{-1} k(z^*, Z)$$

Proof. Writing $Z = [z_1, \ldots, z_n] \in \mathbb{R}^{h\times n}$ for the points in the bottleneck and $Q = [q_1, \ldots, q_n] \in \mathbb{R}^{d\times n}$ for the points in the output layer, and denoting Φ_X, Φ_Z, K_X, K_Z for the respective feature maps and kernel matrices of inputs X and bottleneck Z, the representer theorem (with norm regularization) and the same argument as in the proof of Theorem 11 yields that the minimum-norm W_1 and W_2 satisfying $W_1^T\Phi_X = Z$ and $W_2^T\Phi_Z = Q$ are given by

$$\begin{aligned}W_1 &= \Phi_X K_X^{-1}Z^T\\ W_2 &= \Phi_Z K_Z^{-1}Q^T\end{aligned}$$

Their Frobenius norms (in the infinite-dimensional case, their Hilbert-Schmidt norms) are

$$\begin{aligned}\|W_1\|^2 &= \operatorname{Tr}(ZK_X^{-1}Z^T)\\ \|W_2\|^2 &= \operatorname{Tr}(QK_Z^{-1}Q^T)\end{aligned}$$

Thus, the loss function is equivalent to minimising the expression

$$\begin{aligned}&\min_{Z,Q} \|X - Q\|^2 + \lambda \cdot \operatorname{Tr}(ZK_X^{-1}Z^T + QK_Z^{-1}Q^T)\\ &\text{s.t.} \quad \|z_i\|^2 = 1 \quad \text{for all} \quad i \in [n]\end{aligned}$$

Observe that for any fixed bottleneck Z, $\operatorname{Tr}(ZK_X^{-1}Z^T)$ remains constant and the above problem reduces to a sum of d kernel ridge regressions, with labels X and observations Z. Thus, the optimal parameterization W_2 simplifies to

$$W_2 = \Phi_Z \left(K_Z + \lambda \boldsymbol{I}_n\right)^{-1} X^T$$

and directly implies the final layer

$$Q = X \left(K_Z + \lambda \boldsymbol{I}_n\right)^{-1} K_Z$$

Learning the Kernel AE from Definition 6 is hence equivalent to minimising the following expression over all possible embeddings $Z \in \mathbb{R}^{h \times n}$:

$$\begin{aligned} &\|Q(Z) - X\|^2 + \lambda \operatorname{Tr}\left(Z K_X^{-1} Z^T + Q K_Z^{-1} Q^T\right) \\ &\text{s.t.} \quad \|z_i\|^2 = 1 \ \forall i \in [n]. \end{aligned}$$

Given Z, any new $x^* \in \mathbb{R}^d$ is embedded in the bottleneck as

$$z^* = Z K_X^{-1} k(x^*, X)$$

and reconstructed as

$$\hat{x}^* = X \left(K_Z + \lambda \boldsymbol{I}_n\right)^{-1} k(z^*, Z)$$

□

A.4.2 Proof of Theorem 14

Assume the optimisation be given by Definition 6 and define the class of encoders/decoders as: $\mathcal{F} := \left\{X \mapsto W_2^T \phi_2\left(W_1^T \phi_1\left(X\right)\right) \text{s.t.} \ \|W_1^T \phi(x_i)\|^2 = 1 \ \forall \ i \in [n] \ : \ \|W_1^T\|_{\mathcal{H}} \leq \omega_1, \|W_2^T\|_{\mathcal{H}} \leq \omega_2\right\}$. Let $r := \lambda(\omega_1^2 + \omega_2^2)$ and $\gamma = \max_{s \in \mathbb{R}^h \text{s.t.} \|s\|^2 = 1} \{k(s,s)\}$, then for any $\delta > 0$, the following statement holds with probability at least $1 - \delta$ for any $f \in \mathcal{F}$:

$$\mathcal{L}^{AE}(f) \leq \widehat{\mathcal{L}}^{AE}(f) + O\left(r + \frac{\omega_2 \sqrt{d\gamma}}{\sqrt{n}} + \sqrt{\frac{\log \frac{1}{\delta}}{n}}\right)$$

Proof. Again we have to characterize the *complexity and the slack term.* We will start by following a version of the standard Rademacher complexity to account for the multi-dimensional output to bound the former.

Definition 8 (Empirical Vector Rademacher Complexity following [MP16]). Let us consider a function class $\mathcal{F} := \left\{f : \mathbb{R}^d \to \mathbb{R}^h\right\}$ and a dataset $S = \{x_i\}_{i=1}^n, x_i \in \mathbb{R}^d$. Let $I : [h] \to 2^{[n]}$ be a function which assignes to every $t \in [h]$ a subset $I_t \subset [n]$ and σ_{ij} are doubly indexed, independent Rademacher variables. Then we define the Rademacher complexity as

$$\mathfrak{R}_I(\mathcal{F}, \mathbf{x}) = \frac{1}{n}\mathbb{E}\left[\sup_{f\in\mathcal{F}}\sum_{t=1}^{h}\sum_{i\in I_t}\sigma_{ti} f_t\left(x_i\right)\right].$$

In simple terms this is a standard Rademacher approach while taking the dimension over the output dimension into account.

We first start with the overall loss function and

$$\mathcal{L}\left(X, f^{BAE}_{W_1,W_2}\left(\overline{X}\right)\right) := \left\|X - W_2^T\phi_2\left(W_1^T\phi_1\left(\overline{X}\right)\right)\right\|^2_{\mathcal{H}} + \lambda\left(\|W_1\|^2_{\mathcal{H}} + \|W_2\|^2_{\mathcal{H}}\right)$$
$$\text{s.t.} \quad \|W_1^T\phi(\overline{x}_i)\|^2 = 1 \;\forall\; i \in [n],$$

and use the additive nature of Rademacher complexity to bound the regularization terms first by ω_1, ω_2. Secondly noting that the square norm is L-Lipschitz and using the Lipschitz composition property of Rademacher complexity to bound

$$\mathfrak{R}_I(\ell\circ\mathcal{F}, \mathbf{x}) \leq L\mathfrak{R}_I(\mathcal{F}, \mathbf{x})$$

and we therefore can focus on the encoding-decoding function $\mathfrak{R}_I(\mathcal{F}, \mathbf{x})$. Starting from this general formulation we can now apply this to our setting

$$\begin{aligned}
\mathfrak{R}_I(\mathcal{F}, \mathbf{x}) &= \frac{1}{n}\mathbb{E}\left[\sup_{f\in\mathcal{F}}\sum_{t=1}^{h}\sum_{i\in I_t}\sigma_{ti} f_t\left(x_i\right)\right] && \text{(by Definition 8)}\\
&= \frac{1}{n}\mathbb{E}\left[\sup_{W_1,W_2:\|W_2\|_{\mathcal{H}}\leq\omega_2}\sum_{t}^{d}\sum_{i\in I_t}\sigma_{ti} W_{\cdot t}\phi\left(W_1\sigma(x_i)\right)\right] && \text{(model definition)}\\
&= \frac{1}{n}\mathbb{E}\left[\sup_{W_1,W_2:\|W_2\|_{\mathcal{H}}\leq\omega_2}\sum_{t}^{d}\left\langle W_{\cdot t}, \sum_{i\in I_t}\sigma_{ti}\phi\left(W_1\sigma(x_i)\right)\right\rangle\right]\\
&= \frac{\omega_2}{n}\mathbb{E}\left[\sup_{W_1}\sqrt{\sum_{t}^{d}\left\|\sum_{i\in I_t}\sigma_{ti}\phi\left(W_1\sigma(x_i)\right)\right\|^2}\right]\\
&\leq \frac{\omega_2}{n}\sqrt{\sum_{t}^{d}\mathbb{E}\left[\sup_{W_1}\left\|\sum_{i\in I_t}\sigma_{ti}\phi\left(W_1\sigma(x_i)\right)\right\|^2\right]} && \text{(Jenson inequality)}\\
&= \frac{\omega_2}{n}\sqrt{d\sum_{i}\sup_{W_1}\left\|\phi\left(W_1\sigma(x_i)\right)\right\|^2} && \left(\mathbb{E}\left[\sigma_i\sigma_j\right] = 0, i\neq j\right)
\end{aligned}$$

Now recall that by definition $\|W_1^T\phi(x_i)\|^2 = 1 \;\forall\; i \in [n]$. Therefore picking the supremum over W_1 is obtained for $\gamma = \max_{s\in\mathbb{R}^h \quad \text{s.t.} \quad \|s\|^2=1}\{k(s,s)\}$ and

$$\frac{\omega_2}{n}\sup_{W_1}\sqrt{d\sum_{i}\left\|\phi\left(W_1\sigma(x_i)\right)\right\|^2} \leq \frac{\omega_2}{n}\sqrt{dn\gamma} = \frac{\omega_2\sqrt{d\gamma}}{\sqrt{n}}$$

Combining the above with the standard generalisation error bound [BM02b] in the regression setting adds the *slack term* and concludes the proof. □

A.4.3 Proof Corollary 2

Let $t = \mathbb{E}_{c,c'\sim\rho^2} \mathbf{1}\{c = c'\}$ and $\tau := \frac{1}{(1-t)}$ be the probability that two classes sampled independently from ρ are the same. Again define $\alpha := \left(\sqrt{h \operatorname{Tr}[K_X]} + \sqrt{h \operatorname{Tr}[K_{X^-}]} + \sqrt{h \operatorname{Tr}[K_{X^+}]}\right)$. In the following let $\mathcal{L}_{sup}$ be the loss of the *supervised downstream task.*

Consider the embedding function from the function class $\mathcal{F} := \{X \mapsto W^T\phi(X) : \|W^T\|_{\mathcal{H}} \leq \omega\}$ and let be $\widehat{\mathcal{L}}^{AE+}_{un}$ the loss on the embedding for X^+ and $\widehat{\mathcal{L}}^{AE-}_{un}$ the loss on the embedding for X^-, standing in for two classes[4]. Furthermore let $\operatorname{Tr}[K_{X^+}], \operatorname{Tr}[K_{X^-}] \leq \beta$ For any $\delta > 0$, the following statement holds with probability at least $1 - \delta$ for any $f \in \mathcal{F}$:

$$\mathcal{L}^{AE}_{sup} \leq \tau\left(\left|\widehat{\mathcal{L}}^{AE+}_{un} - \widehat{\mathcal{L}}^{AE-}_{un}\right| - t\right) + \tau O\left(\frac{\omega\sqrt{h\beta}}{\sqrt{n}} + \sqrt{\frac{\log\frac{1}{\delta}}{n}}\right)$$

Proof. Before we state the bound let us formally define the supervised task. We consider a two-class classification task $\mathcal{T}$ with $\{c_1, c_2\}$ distinct classes and a linear classifier on top of the learned representation. Let this function given by $V \in \mathbb{R}^{2\times h}$. In the following let X_c be a datapoint X belonging to class c.

$$\mathcal{L}_{\text{sup}}(\mathcal{T}, f) = \inf_V \mathbb{E}_{(X,c)}\left[Vf(X_c) - Vf(X_{c'}) \mid c_i \neq c_j\right]$$

From there we can furthermore define the average supervised loss as taking the expectation over the distribution of classes. The average loss for a function f on a binary classification task tasks is defined as

$$\mathcal{L}_{\text{sup}}(f) := \mathbb{E}_{\{c_1,c_2\}\sim\rho^2}\left[L_{sup}(\{c_1, c_2\}, f) \mid c_i \neq c_j\right]$$

where the latent class distribution is given by ρ.

In the following let $\mu_c = \mathbb{E}_{x\sim\mathcal{D}_c} f(x)$ be the mean of class c and $f(X)$ the embedding function.

We can now observe that while the general idea is the same, there is an important difference between the two contrastive approaches above and the kernel AE approach. While in Case 1 and 2 $f(X)$ directly gives the embedding function and also a difference between mappings to different classes using positive and negative samples. In the AE case the loss is computed on the reconstruction and not directly on the embedding. In the following we therefore consider $f(X)$ as the embedding function. And let us consider the embedding of positive

[4]Remark: while it seems surprising that positive and negative samples suddenly appear in the AE setup we note that in the contrastive setting this allows to naturally account for mappings to different classes. Therefore in the AE, introducing this setting allows for class differentiation in the embedding.

and negative samples as stand ins for the classes.

$$\begin{aligned}\left|\mathcal{L}_{un}(f(X)) - \mathcal{L}_{un}(f(X^-))\right| &\geq \mathop{\mathbb{E}}_{\substack{X\sim\mathcal{D}_{\text{sim}}\\ X^-\sim\mathcal{D}_{\text{neg}}}} [f(X) - f(X_{c^-})] \\ &= \mathop{\mathbb{E}}_{c,c^-\sim\rho^2} \mathop{\mathbb{E}}_{X\sim\mathcal{D}_c} [f(X) - f(X_{c^-})] \\ &= (1-\tau) \mathop{\mathbb{E}}_{c,c^-\sim\rho^2} \left[L^{\mu}_{\text{sup}}\ \left(\left\{c, c^-\right\}, f\right) \mid c \neq c^-\right] + \tau \\ &= (1-\tau)L_{sup}(f) + \tau\end{aligned}$$

Again observe that $f(X)$ is now only the embedding function, over the class

$$\mathcal{F} := \left\{X \mapsto W^T\phi(X) : \|W^T\|_{\mathcal{H}} \leq \omega\right\}.$$

Similarly to the proof of Theorem 14 we directly get the complexity term as:

$$\mathfrak{R}_c(\mathcal{F}, \mathbf{x}) \leq \omega\sqrt{h\,\mathrm{Tr}\,[K_X]}.$$

Considering it for both positive and negative samples and combined with the above results we obtain the bound for the Kernel AE case. This concludes the proof. ☐

A.5 Proofs for Chapter 6

A.5.1 Proof Theorem 16

For brevity of the proof define

$$U_j := W_1^{(j)}$$
$$V_j := W_2^{(j)}$$

and let us recall the proof statement: For $0 < \lambda \leq 1$, optimizing Eq. 6.3 results in the parameters at the optimum satisfying the following:

i) **Class Assignment.** While in Eq. 6.3 we define $S_{j,i}$ as the probability that X_i belongs to class j at the optimal $S_{j,i} = 1 \; or \; 0$ and therefore converges to a strict class assignment.

ii) **Centers.** C_j at optimum naturally satisfies the condition

$$C_j = \frac{\sum_{i=1} S_{j,i} X_i}{\sum_{i=1} S_{j,i}}.$$

iii) **Encoding / Decoding (learned weights).** We first show that $V_j^T = U_j$, and define

$$\hat{\mathbf{\Sigma}}_j := \sum_{i=1}^{n} S_{j,i} \left(X_i - C_j\right) \left(X_i - C_j\right)^T ,$$

then the encoding corresponds to the top h eigenvectors of $\hat{\Sigma}_j$.

Proof. (i) This part is trivial once we note that for a fixed C_j, U_j, V_j, the loss is linear in $S_{j,i}$. Thus the optimal must occur at the extreme points of the constraints

$$\mathbf{1}_k^T S = \mathbf{1}_n^T$$
$$S_{j,i} \geq 0.$$

Let us first prove part (iii). To get conditions on U_j, V_j let us fix S, C_j and get conditions of optimal in terms of the fixed quantities. Define

$$\hat{\Sigma}_j := \sum_{i=1}^{n} S_{j,i}(X_i - C_j)(X_i - C_j)^T$$

Let us now optimize V_j, U_j for each fixed j.

$$
\begin{aligned}
&\min_{V_j,U_j} \sum_{i=1}^{n} S_{j,i} \left(\|(X_i - C_j) - V_j U_j (X_i - C_j)\|^2 - \lambda \|U_j(X_i - C_j)\|^2 \right) \\
&= \min_{V_j,U_j} \sum_{i=1}^{n} S_{j,i} \left(\|(\mathbb{I} - V_j U_j)(X_i - C_j)\|^2 - \lambda \|U_j(X_i - C_j)\|^2 \right) \\
&= \min_{V_j,U_j} \sum_{i=1}^{n} S_{j,i} \left((X_i - C_j)^T (\mathbb{I} - V_j U_j)^T (\mathbb{I} - V_j U_j)(X_i - C_j) \right. \\
&\quad \left. - \lambda (X_i - C_j)^T U_j^T U_j (X_i - C_j) \right) \\
&= \min_{V_j,U_j} \sum_{i=1}^{n} S_{j,i} \operatorname{Tr}\left[(X_i - C_j)^T (\mathbb{I} - V_j U_j)^T (\mathbb{I} - V_j U_j)(X_i - C_j) \right] \\
&\quad - \lambda \boldsymbol{e}_j^T \boldsymbol{S} \boldsymbol{e}_i \operatorname{Tr}\left[(X_i - C_j)^T U_j^T U_j (X_i - C_j) \right] \\
&= \min_{V_j,U_j} \sum_{i=1}^{n} S_{j,i} \operatorname{Tr}\left[(\mathbb{I} - V_j U_j)^T (\mathbb{I} - V_j U_j)(X_i - C_j)(X_i - C_j)^T \right] \\
&\quad - \lambda \boldsymbol{e}_j^T \boldsymbol{S} \boldsymbol{e}_i \operatorname{Tr}\left[U_j^T U_j (X_i - C_j)(X_i - C_j)^T \right] \\
&= \min_{V_j,U_j} \operatorname{Tr}\left[(\mathbb{I} - V_j U_j)^T (\mathbb{I} - V_j U_j) \hat{\Sigma}_j \right] - \lambda \operatorname{Tr}\left[U_j^T U_j \hat{\Sigma}_j \right] \\
&= \min_{V_j,U_j} \operatorname{Tr}\left[((\mathbb{I} - V_j U_j)^T (\mathbb{I} - V_j U_j) - \lambda U_j^T U_j) \hat{\Sigma}_j \right] \\
&= \min_{V_j,U_j} \operatorname{Tr}\left[(\mathbb{I} - V_j U_j - U_j^T V_j^T + (1-\lambda) U_j^T U_j) \hat{\Sigma}_j \right].
\end{aligned}
$$

We can now bound the above as follows

$$
\begin{aligned}
&\min_{V_j,U_j} \operatorname{Tr}\left[(\mathbb{I} - V_j U_j - U_j^T V_j^T + (1-\lambda) U_j^T U_j) \hat{\Sigma}_j \right] \\
&= \min_{V_j,U_j} \operatorname{Tr}\left[(\mathbb{I} + (1-\lambda) U_j^T U_j) \hat{\Sigma}_j \right] - \operatorname{Tr}\left[(U_j^T V_j^T + V_j U_j) \hat{\Sigma}_j \right] \\
&= \min_{V_j,U_j} \operatorname{Tr}\left[(\mathbb{I} + (1-\lambda) U_j^T U_j) \hat{\Sigma}_j \right] - 2 \operatorname{Tr}\left[U_j \hat{\Sigma}_j V_j^T \right] \\
&\geq \min_{V_j,U_j} \operatorname{Tr}\left[(\mathbb{I} + (1-\lambda) U_j^T U_j) \hat{\Sigma}_j \right] - 2 \sqrt{ \operatorname{Tr}\left[U_j \hat{\Sigma}_j^{1/2} \hat{\Sigma}_j^{1/2} U_j^T \right] \operatorname{Tr}\left[V_j^T \hat{\Sigma}_j^{1/2} \hat{\Sigma}_j^{1/2} V_j \right] } && \text{(A.38)} \\
&\geq \min_{V_j,U_j} \operatorname{Tr}\left[(\mathbb{I} + (1-\lambda) U_j^T U_j) \hat{\Sigma}_j \right] - 2 \sqrt{ \operatorname{Tr}\left[U_j \hat{\Sigma}_j U_j^T \right] \sum_{i=1}^{h} \alpha_i } && \text{(A.39)} \\
&= \min_{V_j,U_j} \sum_{i=1}^{d} \alpha_i - 2 \sqrt{ \operatorname{Tr}\left[U_j \hat{\Sigma}_j U_j^T \right] \sum_{i=1}^{h} \alpha_i } + (1-\lambda) \operatorname{Tr}\left[U_j \hat{\Sigma}_j U_j^T \right],
\end{aligned}
$$

where the inequality A.38 follows from applying Cauchy Schwarz inequality and the inequality A.39 from observing that the columns of V are h orthonormal vectors to get $\operatorname{Tr}\left[V_j^T \hat{\Sigma}_j V_j \right] \leq \sum_{i=1}^{h} \alpha_i$.

Let us set $x = \sqrt{\operatorname{Tr}\left[U_j \hat{\Sigma}_j U_j^T \right]}$, $a = 1 - \lambda$, $b = 2\sqrt{\sum_{i=1}^{h} \alpha_i}$ and $c = \sum_{i=1}^{d} \alpha_i$. We can now observe that $0 \leq x = \sqrt{\operatorname{Tr}\left[U_j \hat{\Sigma}_j U_j^T \right]} \leq \sqrt{\sum_{i=1}^{h} \alpha_i} = \frac{b}{2}$ and the minimization problem becomes

$$ax^2 - bx + c$$

which is a quadratic and thus decreasing when $x \leq \frac{b}{2a}$. Therefore as $\frac{b}{2} \leq \frac{b}{2a}$ the minimum is achieved at $x = \frac{b}{2}$ thus we get

$$\begin{aligned}\min_{V_j,U_j} \operatorname{Tr}\left[(\mathbb{I} - V_jU_j - U_j^TV_j^T + (1-\lambda)U_j^TU_j)\hat{\Sigma}_j\right] &\geq \sum_{i=1}^{d}\alpha_i - 2\sqrt{\sum_{i=1}^{h}\alpha_i\sum_{i=1}^{h}\alpha_i} + (1+\lambda)\sum_{i=1}^{h}\alpha_i \\ &= \sum_{i=1}^{d}\alpha_i - (1+\lambda)\sum_{i=1}^{h}\alpha_i\end{aligned}$$

Reviewing the proof above we note that for the inequality to be achieved, we must satisfy the conditions :

- $\operatorname{Tr}\left[U_j\hat{\Sigma}_jU_j^\top\right] = \sum_{i=1}^{h}\alpha_i$ (from optimizing the quadratic above),
- $V_j^T\hat{\Sigma}_j^{1/2} = U_j\hat{\Sigma}_j^{1/2}$ (from equalizing C.S. used to get inequality A.38),
- $\operatorname{Tr}\left[V_j^\top\hat{\Sigma}_jV_j\right] = \sum_{i=1}^{h}\alpha_i$ (from equalizing inequality A.39).

Luckily the above conditions are only *all* satisfied for the following unique choice of orthonormal $\hat{U}_j$ and $\hat{V}_j$. If α_i and $\boldsymbol{w}_i$ are the eigenvalues (arranged in descending order) and eigenvectors of $\hat{\Sigma}_j$ respectively define

$$\hat{U}_j = \begin{bmatrix}\boldsymbol{w}_1^T \\ \vdots \\ \boldsymbol{w}_h^T\end{bmatrix}, \quad \hat{V}_j = [\boldsymbol{w}_1, \dots, \boldsymbol{w}_h].$$

We then obtain,

$$\begin{aligned}\operatorname{Tr}\left[(\mathbb{I} - \hat{V}_j\hat{U}_j - \hat{U}_j^T\hat{V}_j^T + (1-\lambda)\hat{U}_j^T\hat{U}_j)\hat{\Sigma}_j\right] &= \sum_{i=1}^{d}\alpha_i - (1+\lambda)\sum_{j}^{h}\alpha_j \\ &= -\lambda\sum_{i=1}^{h}\alpha_i + \sum_{j=h+1}^{d}\alpha_j.\end{aligned}$$

Let us finally derive (ii).

Define

$$A := (\mathbb{I} - V_jU_j - U_j^TV_j^T + (1-\lambda)U_j^TU_j).$$

Then,

$$\begin{aligned}
&\min_{C_j} \operatorname{Tr}\left[\left(\mathbb{I} - V_jU_j - U_j^TV_j^T + (1-\lambda)U_j^TU_j\right)\left(\sum_{i=1}^n S_{j,i}(X_i - C_j)(X_i - C_j)^T\right)\right] \\
=& \min_{C_j} \operatorname{Tr}\left[A\left(\sum_{i=1}^n S_{j,i}(X_i - C_j)(X_i - C_j)^T\right)\right] \\
=& \min_{C_j} \operatorname{Tr}\left[A\left(\sum_{i=1}^n S_{j,i}(X_iX_i^T - X_iC_j^T + C_jC_j^T)\right)\right] \\
=& \min_{C_j} \operatorname{Tr}\left[A\left(\left(\sum_{i=1}^n S_{j,i}X_iX_i^T\right) - \left(\sum_{i=1}^n S_{j,i}X_i\right)C_j^T + \left(\sum_{i=1}^n S_{j,i}\right)C_jC_j^T\right)\right].
\end{aligned}$$

As U_j, V_j are variables varying in a space independent of C_j, at optimality of the above, the partial derivative with respect to C_j must be 0. Thus we have,

$$\frac{\partial}{\partial C_j} = -A\left(\sum_{i=1}^n S_{j,i}X_i\right) + A\left(\sum_{i=1}^n S_{j,i}\right)C_j = 0 \tag{A.40}$$

On the other hand from the derivation of (iii), we also have the condition that at optimality $V_j = U_j^\top = \hat{U}_j^\top$. Thus at optimality

$$A = \left(\mathbb{I} - (1+\lambda)\hat{U}_j^\top \hat{U}_j\right).$$

In particular when $\lambda > 0$, A (at optimality) is invertible (as its eigenvalues are 0 and $-\lambda$). Using this in equation A.40 we get

$$\hat{C}_j = \frac{\sum_{i=1}^n S_{j,i}X_i}{\sum_{i=1}^n S_{j,i}}$$

which concluded the proof. □

Appendix B

Additional Experimental Details

B.1 Experimental Details for Chapter 5

B.1.1 Kernel Definitions

For completeness we include the definitions of the kernels considered in this paper below.

Definition 9 (Radial Basis Function (RBF) Kernel)**.** Let x and y be vectors, the the *RBF kernel* is defined as:

$$k(x,y) = \exp\left(-\gamma\|x-y\|^2\right).$$

In the case $\gamma = \sigma^{-2}$ this becomes the Gaussian kernel of variance σ^2.

Definition 10 (Laplacian Kernel)**.** Similar to the *RBF kernel*, let x and y be vectors, then the *Laplacian kernel* is defined as:

$$k(x,y) = \exp\left(-\gamma\|x-y\|_1\right).$$

Definition 11 (Linear Kernel)**.** Let x and y be vectors, then the *linear kernel* is defined as:

$$k(x,y) = x^\top y.$$

In addition to the above definitions that we consider in this paper the following definition illustrate that NTK inspired kernels such as the *ReLU Kernel* are also viable options to be considered as a 'pluck in option' to the proposed models.

Definition 12 (ReLU Kernel [BB21])**.** For a ReLU network with L layers with inputs on the sphere, taking appropriate limits on the widths, one can show: $k_{\text{NTK}}(x,y) = \kappa^L_{\text{NTK}}\left(x^\top y\right)$,

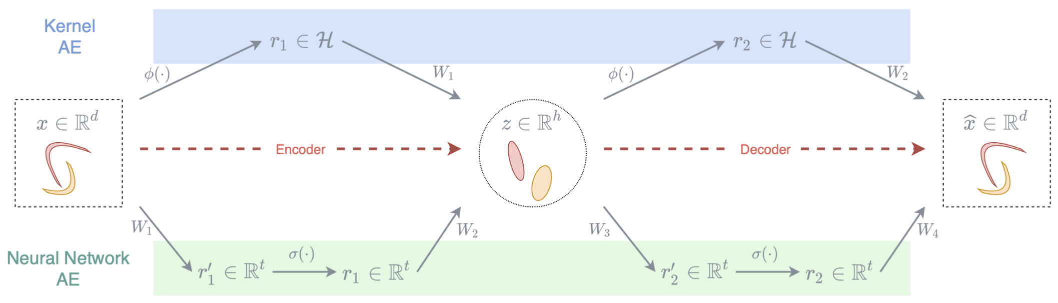


Figure B.1: Data x is mapped to the latent space $\boldsymbol{z}$ using the *encoder* and then reconstructed as $\widehat{x}$ using the *decoder*. From AE to Kernel AE. *Bottom mapping / green:* Standard fully trained deep leaning approach where hidden layers are mapped into a high dimensional euclidean space. *Top mapping / blue:* proposed kernel version where hidden layers are mapped into a Hilbert space.

with $\kappa^1_{\text{NTK}}(u) = \kappa^1(u) = u$ and for $\ell = 2, \dots, L$

$$\kappa^\ell(u) = \kappa_1\left(\kappa^{\ell-1}(u)\right)$$
$$\kappa^\ell_{\text{NTK}}(u) = \kappa^{\ell-1}_{\text{NTK}}(u)\kappa_0\left(\kappa^{\ell-1}(u)\right) + \kappa^\ell(u)$$

where

$$\kappa_0(u) = \frac{1}{\pi}(\pi - \arccos(u))$$
$$\kappa_1(u) = \frac{1}{\pi}\left(u \cdot (\pi - \arccos(u)) + \sqrt{1-u^2}\right).$$

In this final section we provide the experimental details for the comparison to neural network methods, referenced in the final section of the main paper as well as the experiments with SVM in addition to k-nn as a downstream task. In addition we provide additional experiments on further datasets.

B.1.2 Further discussion and experiments comparing neural networks and kernel approaches

As discussed in the introduction, representation learning has become established mainly in the contest of deep learning models. In this paper, we decouple the representation learning paradigm from the widely used deep learning models. While the paper focuses on the specific examples of kernel autoencoders and kernel contrastive learning, our constructions follow a general principle: instead of considering a (one-hidden layer) neural network $W_2\sigma(W_1x)$ we consider a linear functional in the reproducing kernel Hilbert space $W^\top\phi(x)$, but still minimise a similar loss functions (reconstruction error in AEs or contrastive losses). We further illustrate this on the example of the Kernel AE.

Comparison of deep learning AE and Kernel AE. Consider the *kernel AE* illustrated in Figure B.1. A *deep learning* model is shown in the bottom. The encoder maps the input

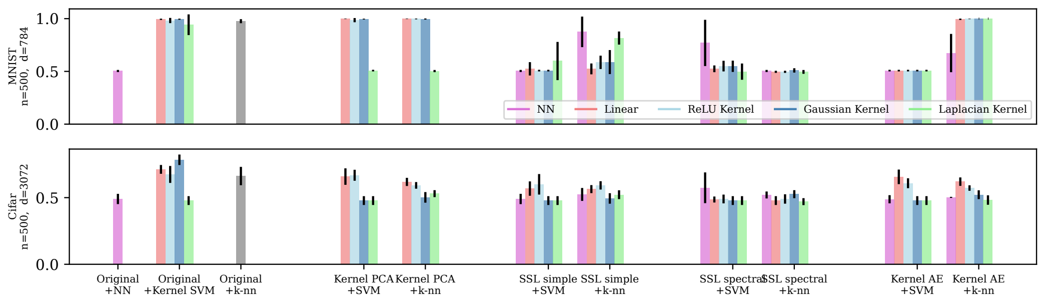


Figure B.2: Comparison of kernel methods and neural network models. Plotted is the accuracy for Kernel methods as defined in Section 6.2 with the addition of a supervised Neural network on the labelled data as well as the neural network models corresponding to the contrastive and AE kernel methods.

$x \in \mathbb{R}^d$ to a hidden layer $r_1 = \sigma(W_1 x) \in \mathbb{R}^t$ via a linear map W_1, and a non-linear activation $\sigma(\cdot)$ with $t \gg d$ and then to a latent representation $\boldsymbol{z} = W_2 r_1 \in \mathbb{R}^k$ with typically $h < d$. Similarly, the decoder maps the representation $\boldsymbol{z}$ to the output $\widehat{x} = W_4 r_2 \in \mathbb{R}^d$ via the hidden layer $r_2 = \sigma(W_3 \boldsymbol{z}) \in \mathbb{R}^t$. The weights $W_1, \ldots, W_4$ are learned through a regularized loss minimisation given training samples $x_1, \ldots, x_n$. A *non-parametric (kernel) variant* of the AE is obtained by replacing the encoder/decoder with implicit maps $\phi_1 : \mathbb{R}^d \to \mathcal{H}$, $\phi_2 : \mathbb{R}^h \to \mathcal{H}$, where $\mathcal{H}$ is the RKHS associated some positive definite kernel k. In the main part we show that, for any new point x^*, the reconstructed point $\widehat{x}^*$ can be expressed only in terms of the kernel evaluation $k(x, x')$, computed between $x^*, x_1, \ldots, x_n$, **without explicit knowledge of $\mathcal{H}$, $\phi_1(\cdot)$ or $\phi_2(\cdot)$**.

Experimental comparison. Similar to the above comparison we can also define deep learning models analogues to the contrastive SSL models . The general implementation is done in Python with the implementation in *PyTorch* [Pas+19] for fully and optimisation of trained models. The presented setup in Figure B.2 is the same as in the main paper with some additional experiments. We firstly extend the analysis by considering SVM as a downstream task as well. We can first note that learning a neural network on the small set of labelled data (most left bar) fails, most likely due to the fact that due to the high complexity of the model overfits the training data. Overall we observe that under SVM the general comparison between neural network and kernel methods are aligned with the one under k-nn.

We will conclude this section with some additional remarks on the comparison and connection between Kernel approaches and neural network methods.

Do we need deep kernel representation learning models? Although our construction considers only one a linear functional $W^\top \phi(x)$, the feature map $\phi(x)$ can also capture deeper networks. For instance, one may use L-layer ReLU NTK [BB21] in kernel SSL to model the behaviour "deep" SSL.

Comparison to generalisation error bounds for deep learning models. A common problem in the analysis of modern machine and deep learning methods is that thorough statistical approaches such as VC-dimension [Vap82; Vap98] or Rademacher complexity [TLP16] do not hold in the overparmeterized learning regime. On the other hand in the context of kernel machines those approaches are well developed [Wah90; SS02; BM02a]. The proposed extensions allow for a more thorough theoretical analysis of representation learning as well as kernel variants of unsupervised deep learning methods.

B.1.3 Further experiments

In this section we extend the experiments presented in the main section. We additionally provided the performance of linear SVM as a donwstream classifier in addition to the earlier considered k-nn classifier. Furthermore we extend the analysis by considering the following datasets.

We denote the split as $split = unlabelled\%/labelled\%/test\%$. We show the results for the following three dadatsets: *concentric circles, factor 0.6* ($n = 200, d = 2, \#classes = 2$) [Ped+11], *cubes* ($n = 200, d = 13, \#classes = 4, split = 50/10/40$) [Ped+11], *Iris* ($n = 150, d = 4, \#classes = 3, split = 50/5/45$) [Fis36a], *Ionosphere* ($n = 351, d = 34, \#classes = 2, split = 50/5/45$) [Sig+89], *blobs* ($n = 200, d = 2, \#classes = 3, split = 50/5/45$) [Ped+11], *Breast cancer* ($n = 569, d = 30, \#classes = 2, split = 50/5/45$) [Wol+95], *Heart Failure* ($n = 299, d = 13, \#classes = 2, split = 50/5/45$) [DG20], *Mushroom (sub-sample)* ($n = 200, d = 22, \#classes = 2, split = 50/5/45$) [Sch87], *Wine* ($n = 178, d = 13, \#classes = 3, split = 50/10/4$) [AF91], *Parkinson's Disease Classification from Speech* ($n = 756, d = 754, \#classes = 2, split = 50/5/45$) [Sak+19] and *Moons* ($n = 200, d = 2, \#classes = 2, split = 50/5/45$) [Ped+11].

We can conclude this section by observing that overall the main findings in the main paper stay consistent throughout the analysis of additional datasets and downstream tasks.

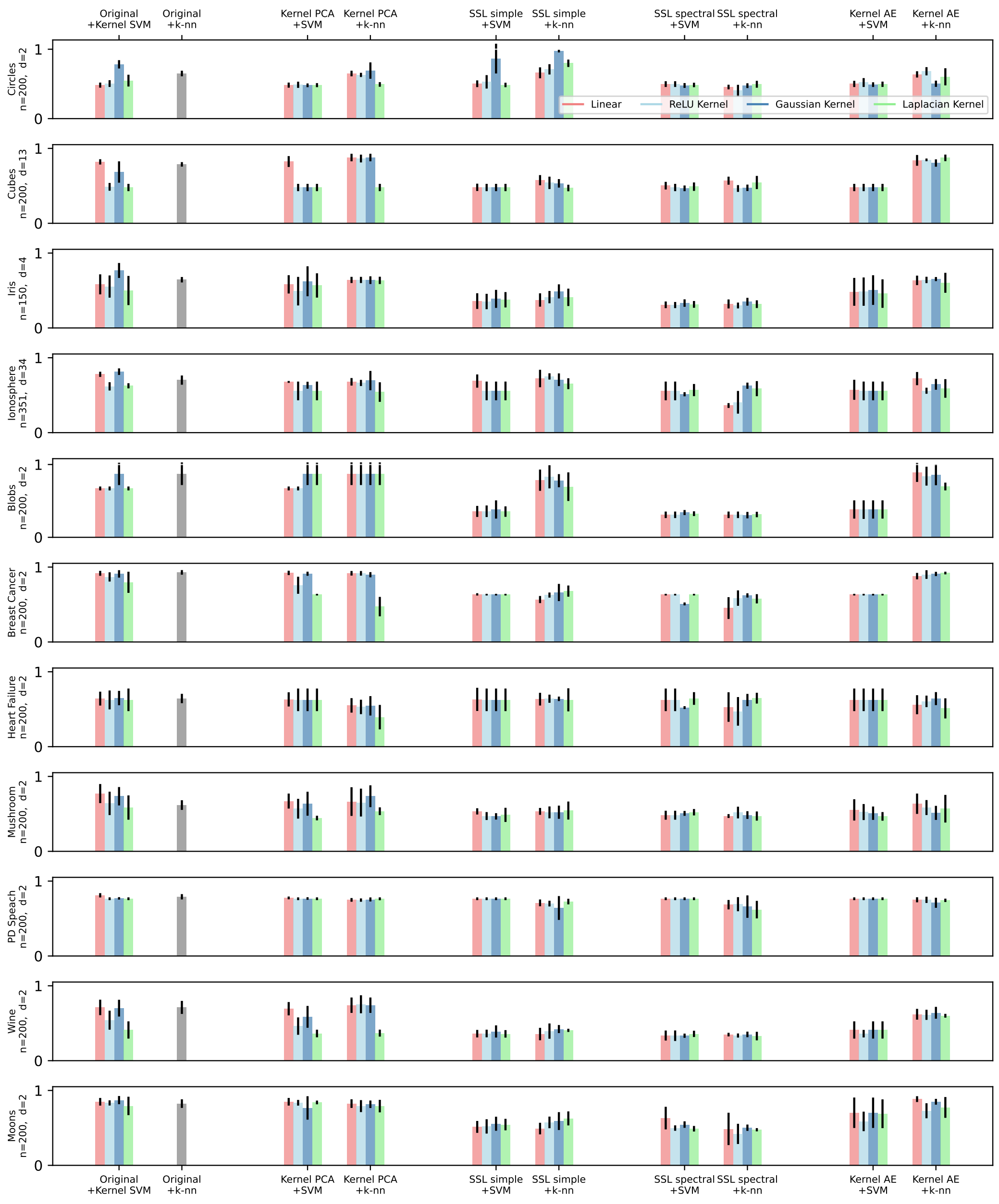


Figure B.3: Accuracy for different methods. From left to right: we first consider k-nn and Kernel SVM on the original features followed by SVM and k-nn on embeddings obtained by Kernel SVM, simple contrastive kernel method, spectral contrastive kernel method and kernel AE.

B.2 Algorithmic Details for Ordinal Clustering Algorithm

In this section, we provide details on the modified SPUR algorithm that we use to tune the parameter λ, and to select the number of clusters.

SPUR, acronym for Semidefinite Program with Unknown r (r denoting the number of clusters), was proposed by [YSC18] to tune the parameter λ of SDP-λ in the context of graph clustering (see Algorithm 2). The underlying idea of this approach is to search for the optimal λ using a grid search over the range $0 < \lambda < \lambda_{\max}$, where $\lambda_{\max}$ is derived from an exact recovery result under stochastic block model.

Algorithm 2: Semidefinite Program with Unknown k (SPUR).

```
input  : graph $A$, number of candidates $T$.
begin
    for $t = 1$ to $T$ do
        $\lambda_t = \exp\left(\frac{t}{T}\ln(1+\lambda_{\max})\right) - 1$.        ([YSC18] set $\lambda_{\max} = \|A\|_{op}$)
        Solve SDP-$\lambda$ with $\lambda = \lambda_t$ to obtain $N_t$.
        Estimate $k_t$ = integer approximation of $\mathrm{Tr}(N_t)$.
    end
    Choose $\hat{t} = \underset{t}{\operatorname{argmax}} \frac{\sum_{i \le \kappa_t} \sigma_i(N_t)}{\mathrm{Tr}(N_t)}$, where $\sigma_i(N_t)$ denotes $i$-th largest eigenvalue of $N_t$.
end
output: Number of clusters $\kappa_{\hat{t}}$, $N_{\hat{t}}$.
```

In the present setting, Theorem 17 shows that the planted clusters can be exactly recovered given a sufficient number of comparisons and an appropriate choice of λ. From Theorem 17, a candidate for $\lambda_{\max}$ can be chosen as $\frac{|\mathcal{T}|}{n}$ (for triplets) or $\frac{|\mathcal{Q}|}{n}$ (for quadruplets), which is a loose upper bound for the theoretical interval for λ, obtained by noting that $\epsilon\delta n_{\min} \le n$. Thus, following [YSC18], we could use Algorithm 2 with our choice of $\lambda_{\max}$.

Unfortunately, this approach has two main drawbacks. First, it ignores the lower bound in Theorem 17 and, second, setting T, the number of λ values that should be considered in Algorithm 2, is difficult. To address the former issue, we propose to consider Theorem 17 once more and to use $\lambda_{\min} = \sqrt{c(\ln n)/n}$ as a lower bound for λ instead of 0, as used in [YSC18]. To address the latter issue, we use the fact that the estimated number of clusters κ monotonically decreases with λ as shown in the next Lemma.

Lemma 11 (**The estimated number of clusters decreases monotonically with increasing** λ)**.** *For any $\lambda > 0$, let N_λ denote the solution of SDP-λ and $\kappa_\lambda = \lfloor Tr(N_\lambda) \rceil$ be the integer approximation of $Tr(N_\lambda)$, which is an estimate of the number of clusters. Then, κ_λ is a non-increasing function of λ, that is*

$$\lambda' \ge \lambda \Rightarrow \kappa_{\lambda'} \le \kappa_\lambda.$$

Proof. We start this proof by noting that since κ_λ is the integer approximation of $\mathrm{Tr}(N_\lambda)$, it suffices to show that $\mathrm{Tr}\,(N_\lambda)$ is a non-increasing function of λ. Then, consider distinct λ', λ and let $N_{\lambda'}$, N_λ be the solutions of SDP-λ with parameters λ', λ, respectively. We have

$$\begin{aligned}\mathrm{Tr}\,(HN_\lambda) - \lambda\mathrm{Tr}\,(N_\lambda) &\geq \mathrm{Tr}\,(S_{\lambda'}) - \lambda\mathrm{Tr}\,(N_{\lambda'})\ ,\\ \mathrm{Tr}\,(HN_\lambda) - \lambda'\mathrm{Tr}\,(N_\lambda) &\leq \mathrm{Tr}\,(HN_{\lambda'}) - \lambda'\mathrm{Tr}\,(N_{\lambda'})\ .\end{aligned}$$

Subtracting the second inequality from the first inequality implies

$$\begin{aligned}\mathrm{Tr}\,(HN_\lambda) - \lambda\mathrm{Tr}\,(N_\lambda) - \ &(\mathrm{Tr}\,(HN_\lambda) - \lambda'\mathrm{Tr}\,(N_\lambda))\\ &\geq \mathrm{Tr}\,(HN_{\lambda'}) - \lambda\mathrm{Tr}\,(N_{\lambda'}) - \mathrm{Tr}\,(SN_{\lambda'}) + \lambda'\mathrm{Tr}\,(N_{\lambda'})\end{aligned}$$

which implies

$$(\lambda' - \lambda)\mathrm{Tr}\,(N_\lambda) \geq (\lambda' - \lambda)\mathrm{Tr}\,(N_{\lambda'})$$

or equivalently, $(\lambda' - \lambda)(\mathrm{Tr}\,(N_{\lambda'}) - \mathrm{Tr}\,(N_\lambda)) \leq 0$. Thus, for $\lambda' > \lambda$, we can conclude that $\mathrm{Tr}\,(N_{\lambda'}) \leq \mathrm{Tr}\,(N_\lambda)$, which shows that $\mathrm{Tr}\,(N_\lambda)$ and k_λ are non-increasing functions of λ. □

Following this, using $\lambda_{\min}$ and $\lambda_{\max}$, we get two estimates of the number of clusters, $\kappa_{\lambda_{\min}}$ and $\kappa_{\lambda_{\max}}$. Then, we search over $\kappa \in [\kappa_{\lambda_{\max}}, \kappa_{\lambda_{\min}}]$ instead of searching over λ—in practice, it helps to search over the values $\max\{2, \kappa_{\lambda_{\max}}\} \leq \kappa \leq \kappa_{\lambda_{\min}} + 2$. We select κ that maximises the above SPUR objective, where N_κ is computed using the simpler SDP-κ [YSC18]. This approach is summarized in Algorithm 1 in the main section.

List of Figures

Bibliography

[Abb17] E. Abbe. “Community detection and stochastic block models: recent developments”. In: *The Journal of Machine Learning Research* 18.1 (2017), pp. 6446–6531.

[AC17] E. Arias-Castro. “Some theory for ordinal embedding”. In: *Bernoulli* 23.3 (2017), pp. 1663–1693.

[AF91] Stefan Aeberhard and M. Forina. *Wine*. UCI Machine Learning Repository. 1991.

[Aga+18] Iman Aganj, Mukesh G Harisinghani, Ralph Weissleder, and Bruce Fischl. “Unsupervised medical image segmentation based on the local center of mass”. In: *Scientific reports* (2018).

[AJC15] C. Aicher, A. Z. Jacobs, and A. Clauset. “Learning latent block structure in weighted networks”. In: *Journal of Complex Networks* 3.2 (2015), pp. 221–248.

[Ale+21] Sina Alemohammad, Zichao Wang, Randall Balestriero, and Richard G. Baraniuk. “The Recurrent Neural Tangent Kernel”. In: *International Conference on Learning Representations*. 2021.

[And36] Edgar Anderson. “The Species Problem in Iris”. In: *Annals of the Missouri Botanical Garden* (1936).

[AR14] Charu C. Aggarwal and Chandan K. Reddy, eds. *Data Clustering: Algorithms and Applications*. CRC Press, 2014. ISBN: 978-1-46-655821-2.

[Aro+19a] Sanjeev Arora, Nadav Cohen, Noah Golowich, and Wei Hu. “A Convergence Analysis of Gradient Descent for Deep Linear Neural Networks”. In: *International Conference on Learning Representations*. 2019.

[Aro+19b] Sanjeev Arora, Simon S. Du, Wei Hu, Zhiyuan Li, Ruslan Salakhutdinov, and Ruosong Wang. “On Exact Computation with an Infinitely Wide Neural Net”. In: *Advances in Neural Information Processing Systems*. 2019.

[Aro+19c] Sanjeev Arora, Hrishikesh Khandeparkar, Mikhail Khodak, Orestis Plevrakis, and Nikunj Saunshi. “A Theoretical Analysis of Contrastive Unsupervised Representation Learning”. In: *International Conference on Machine Learning*. 2019.

[ASP94] Peter J Angeline, Gregory M Saunders, and Jordan B Pollack. “An evolutionary algorithm that constructs recurrent neural networks”. In: *IEEE transactions on Neural Networks* (1994).

[ATL13] Salem Alelyani, Jiliang Tang, and Huan Liu. “Feature Selection for Clustering: A Review”. In: *Data Clustering: Algorithms and Applications.* 2013.

[AV07] David Arthur and Sergei Vassilvitskii. “K-Means++: The Advantages of Careful Seeding”. In: *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms* (2007).

[Bao+20] Xuchan Bao, James Lucas, Sushant Sachdeva, and Roger B Grosse. “Regularized linear autoencoders recover the principal components, eventually”. In: *Advances in Neural Information Processing Systems.* 2020.

[BB21] Alberto Bietti and Francis Bach. “Deep Equals Shallow for ReLU Networks in Kernel Regimes”. In: *International Conference on Learning Representations.* 2021.

[BCM05] Antoni Buades, Bartomeu Coll, and Jean-Michel Morel. “A review of image denoising algorithms, with a new one”. In: *Multiscale Modeling and Simulation: A SIAM Interdisciplinary Journal* (2005).

[BCV13] Yoshua Bengio, Aaron Courville, and Pascal Vincent. “Representation learning: A review and new perspectives”. In: *IEEE transactions on pattern analysis and machine intelligence* (2013).

[BDM09] Christos Boutsidis, Petros Drineas, and Michael W Mahoney. “Unsupervised Feature Selection for the k-means Clustering Problem”. In: *Advances in Neural Information Processing Systems.* 2009.

[BH89] Pierre Baldi and Kurt Hornik. “Neural networks and principal component analysis: Learning from examples without local minima”. In: *Neural Networks* (1989).

[BHK23] Kilian Batzner, Lars Heckler, and Rebecca König. “Efficientad: Accurate visual anomaly detection at millisecond-level latencies”. In: *arXiv preprint arXiv:2303.14535* (2023).

[Bis06] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics).* Springer-Verlag, 2006.

[BKB21] Yamini Bansal, Gal Kaplun, and Boaz Barak. “For self-supervised learning, Rationality implies generalization, provably”. In: *9th International Conference on Learning Representations.* 2021.

[BL22] Randall Balestriero and Yann LeCun. “Contrastive and Non-Contrastive Self-Supervised Learning Recover Global and Local Spectral Embedding Methods”. In: *Advances in Neural Information Processing Systems.* 2022.

[BLS17] J Ballé, V Laparra, and E P Simoncelli. “End-to-end optimized image compression”. In: *International Converence on Learning Representations (ICLR).* 2017.

[BM02a] Peter L. Bartlett and Shahar Mendelson. "Rademacher and Gaussian Complexities: Risk Bounds and Structural Results". In: *Journal of Machine Learning Research* (2002).

[BM02b] Peter L Bartlett and Shahar Mendelson. "Rademacher and Gaussian complexities: Risk bounds and structural results". In: *Journal of Machine Learning Research* (2002).

[BMN04] Mikhail Belkin, Irina Matveeva, and Partha Niyogi. "Regularization and semi-supervised learning on large graphs". In: *International Conference on Computational Learning Theory*. 2004.

[BNN22] Han Bao, Yoshihiro Nagano, and Kento Nozawa. "On the Surrogate Gap between Contrastive and Supervised Losses". In: *International Conference on Machine Learning*. 2022.

[Boj+18] Aleksandar Bojchevski, Oleksandr Shchur, Daniel Zügner, and Stephan Günnemann. "NetGAN: Generating Graphs via Random Walks". In: *International Conference on Machine Learning*. 2018.

[BPL22] Adrien Bardes, Jean Ponce, and Yann LeCun. "VICReg: Variance-Invariance-Covariance Regularization for Self-Supervised Learning". In: *International Conference on Learning Representations*. 2022.

[Bro+93] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. "Signature verification using a" siamese" time delay neural network". In: *Advances in neural information processing systems* (1993).

[Bru+14] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann Lecun. "Spectral networks and locally connected networks on graphs". In: *International Conference on Learning Representations*. 2014.

[BTVG06] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. "SURF: Speeded Up Robust Features". In: *Computer Vision – ECCV 2006*. Springer Berlin Heidelberg, 2006.

[Bur98] Chris J.C. Burges. "A Tutorial on Support Vector Machines for Pattern Recognition". In: 1998.

[Cab+23] Vivien Cabannes, Bobak Toussi Kiani, Randall Balestriero, Yann LeCun, and Alberto Bietti. "The SSL Interplay: Augmentations, Inductive Bias, and Generalization". In: *CoRR* abs/2302.02774 (2023). arXiv: 2302.02774.

[Car+20] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. "Unsupervised learning of visual features by contrasting cluster assignments". In: *Advances in neural information processing systems* (2020).

[Car+21] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. "Emerging properties in self-supervised vision transformers". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021.

[CBK09] Varun Chandola, Arindam Banerjee, and Vipin Kumar. "Anomaly detection: A survey". In: *ACM computing surveys (CSUR)* (2009).

[CBL19] Zhengdao Chen, Joan Bruna, and Lisha Li. "Supervised community detection with line graph neural networks". In: *International Conference on Learning Representations*. 2019.

[CF20] Junyu Chen and Eric C Frey. "Medical image segmentation via unsupervised convolutional neural network". In: *arXiv preprint arXiv:2001.10155* (2020).

[CGW21] Wuyang Chen, Xinyu Gong, and Zhangyang Wang. "Neural architecture search on imagenet in four gpu hours: A theoretically inspired perspective". In: *arXiv preprint arXiv:2102.11535* (2021).

[CH21] Xinlei Chen and Kaiming He. "Exploring Simple Siamese Representation Learning". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2021.

[Che+20a] Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. "Simple and Deep Graph Convolutional Networks". In: *International Conference on Machine Learning*. 2020.

[Che+20b] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. "A simple framework for contrastive learning of visual representations". In: *International conference on machine learning*. PMLR. 2020.

[Che+21] Yilan Chen, Wei Huang, Lam M. Nguyen, and Tsui-Wei Weng. "On the Equivalence between Neural Network and Support Vector Machine". In: *Advances in Neural Information Processing Systems 34*. 2021.

[Che+22] Changyou Chen, Jianyi Zhang, Yi Xu, Liqun Chen, Jiali Duan, Yiran Chen, Son Tran, Belinda Zeng, and Trishul Chilimbi. "Why do We Need Large Batchsizes in Contrastive Learning? A Gradient-Bias Perspective". In: *Advances in Neural Information Processing Systems*. 2022.

[Chi+22] Wei Ming Dan Chia, Sye Loong Keoh, Cindy Goh, and Christopher Johnson. "Risk assessment methodologies for autonomous driving: A survey". In: *IEEE transactions on intelligent transportation systems* 23.10 (2022), pp. 16923–16939.

[Cho13] Kyunghyun Cho. "Simple Sparsification Improves Sparse Denoising Autoencoders in Denoising Highly Corrupted Images". In: *International Conference on Machine Learning* (2013).

[Cho+14] Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. "On the Properties of Neural Machine Translation: Encoder–Decoder Approaches". In: *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*. 2014.

[Cho+15] François Chollet et al. *Keras*. `https://keras.io`. 2015.

[CM07] Corinna Cortes and Mehryar Mohri. "On Transductive Regression". In: *Advances in Neural Information Processing Systems*. 2007.

[CRR18] Luigi Carratino, Alessandro Rudi, and Lorenzo Rosasco. "Learning with SGD and Random Features". In: *Advances in Neural Information Processing Systems 31*. 2018.

[CT21] Husanjot Chahal and Helen Toner. "'Small Data' Are Also Crucial for Machine Learning". In: *Scientific American* (2021).

[Cui+19] Zhiyong Cui, Kristian Henrickson, Ruimin Ke, and Yinhai Wang. "Traffic graph convolutional recurrent neural network: A deep learning framework for network-scale traffic learning and forecasting". In: *IEEE Transactions on Intelligent Transportation Systems* (2019).

[CV08] Kevin P. Costello and Van H. Vu. "The rank of random graphs". In: *Random Structures & Algorithms*. 2008.

[CV95] Corinna Cortes and Vladimir Vapnik. "Support-vector networks". In: *Machine learning* (1995).

[CX14] Y. Chen and J. Xu. "Statistical-computational phase transitions in planted models: The high-dimensional setting". In: *International Conference on Machine Learning*. 2014, pp. 244–252.

[CY20] X. Chen and Y. Yang. "Diffusion K-means clustering on manifolds: provable exact recovery via semidefinite relaxations". In: *Applied and Computational Harmonic Analysis* (2020).

[Das16] S. Dasgupta. "A cost function for similarity-based hierarchical clustering". In: *Symposium on Theory of Computing*. 2016, pp. 118–127.

[Das99] Sanjoy Dasgupta. "Learning mixtures of Gaussians". In: *Annual Symposium on Foundations of Computer Science*. 1999.

[DB16] Steven Diamond and Stephen Boyd. "CVXPY: A Python-embedded modeling language for convex optimization". In: *Journal of Machine Learning Research* (2016).

[DBV16] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. "Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering". In: *International Conference on Neural Information Processing Systems*. 2016.

[DDS16] Hanjun Dai, Bo Dai, and Le Song. "Discriminative Embeddings of Latent Variable Models for Structured Data". In: *International Conference on Machine Learning*. 2016.

[Den12] Li Deng. "The mnist database of handwritten digit images for machine learning research". In: *IEEE Signal Processing Magazine* (2012).

[Dev+19] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Proceedings of the 2019 Conference of the North American Chapter of the*

Association for Computational Linguistics: Human Language Technologies, Association for Computational Linguistics, 2019.

[DG20] Chicco Davide and Jurman Giuseppe. *Heart failure clinical records.* UCI Machine Learning Repository. 2020.

[Din+13] Shifei Ding, Hui Li, Chunyang Su, Junzhao Yu, and Fengxiang Jin. "Evolutionary artificial neural networks: a review". In: *Artificial Intelligence Review* (2013).

[Diz+17] Kamran Dizaji, Amirhossein Herandi, Cheng Deng, Weidong Cai, and Heng Huang. "Deep Clustering via Joint Convolutional Autoencoder Embedding and Relative Entropy Minimization". In: *International Conference on Computer Vision (ICCV)* (2017).

[Du+19a] Simon S Du, Kangcheng Hou, Russ R Salakhutdinov, Barnabas Poczos, Ruosong Wang, and Keyulu Xu. "Graph Neural Tangent Kernel: Fusing Graph Neural Networks with Graph Kernels". In: *Advances in Neural Information Processing Systems.* 2019.

[Du+19b] Simon S Du, Kangcheng Hou, Russ R Salakhutdinov, Barnabas Poczos, Ruosong Wang, and Keyulu Xu. "Graph Neural Tangent Kernel: Fusing Graph Neural Networks with Graph Kernels". In: *Advances in Neural Information Processing Systems.* Vol. 32. 2019.

[EAS98] Alan Edelman, Tomás A Arias, and Steven T Smith. "The geometry of algorithms with orthogonality constraints". In: *SIAM journal on Matrix Analysis and Applications* (1998).

[EFG23] Pascal Esser, Maximilian Fleissner, and Debarghya Ghoshdastidar. *Non-Parametric Representation Learning with Kernels.* 2023. arXiv: `2309.02028 [cs.LG]`.

[EMG23] Pascal Esser, Satyaki Mukherjee, and Debarghya Ghoshdastidar. *Representation Learning Dynamics of Self-Supervised Models.* 2023. arXiv: `2309.02011 [cs.LG]`.

[Erm+21] Aleksandr Ermolov, Aliaksandr Siarohin, Enver Sangineto, and Nicu Sebe. "Whitening for self-supervised representation learning". In: *International Conference on Machine Learning.* 2021.

[Ess+23] Pascal Esser, Satyaki Mukherjee, Mahalakshmi Sabanayagam, and Debarghya Ghoshdastidar. "Improved Representation Learning Through Tensorized Autoencoders". In: *International Conference on Artificial Intelligence and Statistics.* PMLR. 2023.

[EVG21] Pascal Esser, Leena Vankadara, and Debarghya Ghoshdastidar. "Learning theory can (sometimes) explain generalisation in graph neural networks". In: *Advances in Neural Information Processing Systems* (2021).

[EYP09] Ran El-Yaniv and Dmitry Pechyony. "Transductive Rademacher Complexity and its Applications". In: *Journal of Artificial Intelligence Research.* 2009.

[EZK18] E. Emamjomeh-Zadeh and D. Kempe. “Adaptive Hierarchical Clustering Using Ordinal Queries”. In: *Symposium on Discrete Algorithms*. 2018, pp. 415–429.

[Fan+19] Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. “Graph neural networks for social recommendation”. In: *The world wide web conference*. 2019.

[FD+14] Manuel Fernández-Delgado, Eva Cernadas, Senén Barro, and Dinani Amorim. “Do we need hundreds of classifiers to solve real world classification problems?” In: *The journal of machine learning research* (2014).

[Fel20] Vitaly Feldman. “Does Learning Require Memorization? A Short Tale about a Long Tail”. In: *Annual ACM SIGACT Symposium on Theory of Computing*. 2020.

[Fer+17] Basura Fernando, Hakan Bilen, Efstratios Gavves, and Stephen Gould. “Self-supervised video representation learning with odd-one-out networks”. In: *IEEE conference on computer vision and pattern recognition*. 2017.

[Fis36a] Ronald A Fisher. “The use of multiple measurements in taxonomic problems”. In: *Annals of eugenics* (1936).

[Fis36b] Rory A. Fisher. “THE USE OF MULTIPLE MEASUREMENTS IN TAXONOMIC PROBLEMS”. In: *Annals of Human Genetics* (1936).

[FRD10] Roman Filipovych, Susan Resnick, and Christos Davatzikos. “Semi-supervised Cluster Analysis of Imaging Data”. In: *NeuroImage* (2010).

[FRG09] Alyson K Fletcher, Sundeep Rangan, and Vivek K Goyal. “Necessary and sufficient conditions for sparsity pattern recovery”. In: *IEEE Transactions on Information Theory* 55.12 (2009), pp. 5758–5772.

[Fuk98] Kenji Fukumizu. “Dynamics of batch learning in multilayer neural networks”. In: *ICANN 98: Proceedings of the 8th International Conference on Artificial Neural Networks*. Springer. 1998.

[GBL98] C. Lee Giles, Kurt D. Bollacker, and Steve Lawrence. “CiteSeer: An Automatic Citation Indexing System”. In: *Proceedings of the Third ACM Conference on Digital Libraries*. Association for Computing Machinery, 1998.

[Ge+23] Jiawei Ge, Shange Tang, Jianqing Fan, and Chi Jin. “On the Provable Advantage of Unsupervised Pretraining”. In: *arXiv preprint* abs/2303.01566 (2023).

[Gho+19] Behrooz Ghorbani, Song Mei, Theodor Misiakiewicz, and Andrea Montanari. “Limitations of Lazy Training of Two-layers Neural Network”. In: *Advances in Neural Information Processing Systems 32*. 2019.

[Gho+20] Behrooz Ghorbani, Song Mei, Theodor Misiakiewicz, and Andrea Montanari. “When Do Neural Networks Outperform Kernel Methods?” In: *Advances in Neural Information Processing Systems*. 2020.

[Gil+17] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. “Neural Message Passing for Quantum Chemistry”. In: *International Conference on Machine Learning*. 2017.

[GJJ20] Vikas Garg, Stefanie Jegelka, and Tommi Jaakkola. “Generalization and Representational Limits of Graph Neural Networks”. In: *International Conference on Machine Learning*. 2020.

[GMS05] Maria Cristina Gori, Gabriele Monfardini, and Franco Scarselli. “A new model for learning in graph domains”. In: *IEEE International Joint Conference on Neural Networks*. 2005.

[GPK22] Eugene Golikov, Eduard Pokonechnyy, and Vladimir Korviakov. *Neural Tangent Kernel: A Survey*. 2022. arXiv: `2208.13614 [cs.LG]`.

[GPL19] D. Ghoshdastidar, M. Perrot, and U. von Luxburg. “Foundations of Comparison-Based Hierarchical Clustering”. In: *Advances in Neural Information Processing Systems*. 2019, pp. 7454–7464.

[Gri+20] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Ávila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. “Bootstrap Your Own Latent - A New Approach to Self-Supervised Learning”. In: *Advances in Neural Information Processing Systems*. 2020.

[Gui+23] Jie Gui, Tuo Chen, Qiong Cao, Zhenan Sun, Hao Luo, and Dacheng Tao. “A survey of self-supervised learning from multiple perspectives: Algorithms, theory, applications and future trends”. In: *arXiv preprint arXiv:2301.05712* (2023).

[GWF14] Kristen B. Gorman, Tony D. Williams, and William R. Fraser. “Ecological Sexual Dimorphism and Environmental Variability within a Community of Antarctic Penguins (Genus Pygoscelis)”. In: *PLOS ONE* (2014).

[HA85] L. Hubert and P. Arabie. “Comparing partitions”. In: *Journal of Classification* 2.1 (1985), pp. 193–218.

[Ham+17] Takuo Hamaguchi, Hidekazu Oiwa, Masashi Shimbo, and Yuji Matsumoto. “Knowledge transfer for out-of-knowledge-base entities: A graph neural network approach”. In: *arXiv preprint arXiv:1706.05674* (2017).

[Hao+21] Jeff Z. HaoChen, Colin Wei, Adrien Gaidon, and Tengyu Ma. “Provable Guarantees for Self-Supervised Deep Learning with Spectral Contrastive Loss”. In: *Advances in neural information processing systems*. 2021.

[HCL06] Raia Hadsell, Sumit Chopra, and Yann LeCun. “Dimensionality reduction by learning an invariant mapping”. In: *2006 IEEE computer society conference on computer vision and pattern recognition (CVPR’06)*. IEEE. 2006.

[HGL17] S. Haghiri, D. Ghoshdastidar, and U. von Luxburg. "Comparison-Based Nearest Neighbor Search". In: *International Conference on Artificial Intelligence and Statistics*. 2017, pp. 851–859.

[HH14] Jens Hainmueller and Chad Hazlett. "Kernel regularized least squares: Reducing misspecification bias with a flexible and interpretable machine learning approach". In: *Political Analysis* (2014).

[Hol16] Gerhard B. Holt. "Potential Simpson's Paradox in Multicenter Study of Intraperitoneal Chemotherapy for Ovarian Cancer". In: *Journal of Clinical Oncology* (2016).

[Hos17] Yedid Hoshen. "Vain: Attentional multi-agent predictive modeling". In: *Advances in neural information processing systems* 30 (2017).

[HPD19] Michael Hahsler, Matthew Piekenbrock, and Derek Doran. "dbscan: Fast Density-Based Clustering with R". In: *Journal of Statistical Software* (2019).

[HY21] Reinhard Heckel and Fatih Furkan Yilmaz. "Early Stopping in Deep Networks: Double Descent and How to Eliminate it". In: *International Conference on Learning Representations*. 2021.

[HYZ23] Lu Han, Han-Jia Ye, and De-Chuan Zhan. "Augmentation Component Analysis: Modeling Similarity via the Augmentation Overlaps". In: *The Eleventh International Conference on Learning Representations*. 2023.

[Iiv22] Albin Iivari. *Anomaly detection techniques for unsupervised machine learning.* 2022.

[JGH18a] Arthur Jacot, Franck Gabriel, and Clement Hongler. "Neural tangent kernel: Convergence and generalization in neural networks". In: *Advances in neural information processing systems*. 2018.

[JGH18b] Arthur Jacot, Franck Gabriel, and Clément Hongler. "Neural Tangent Kernel: Convergence and Generalization in Neural Networks". In: *International Conference on Neural Information Processing Systems*. 2018.

[JHM22] Daniel D Johnson, Ayoub El Hanchi, and Chris J Maddison. "Contrastive Learning Can Find An Optimal Basis For Approximately View-Invariant Functions". In: *arXiv preprint arXiv:2210.01883* (2022).

[Jin+22] Li Jing, Pascal Vincent, Yann LeCun, and Yuandong Tian. "Understanding Dimensional Collapse in Contrastive Self-supervised Learning". In: *The Tenth International Conference on Learning Representations*. 2022.

[JJN16] L. Jain, K. G. Jamieson, and R. Nowak. "Finite sample prediction and recovery bounds for ordinal embedding". In: *Advances in Neural Information Processing Systems*. 2016, pp. 2711–2719.

[JLE14] Zhanglong Ji, Zachary C Lipton, and Charles Elkan. "Differential privacy and machine learning: a survey and review". In: *arXiv preprint arXiv:1412.7584* (2014).

[JN11] K. G. Jamieson and R. D. Nowak. "Low-dimensional embedding using adaptively selected ordinal data". In: *Annual Allerton Conference on Communication, Control, and Computing*. 2011, pp. 1077–1084.

[JT19] Longlong Jing and Yingli Tian. "Self-Supervised Visual Feature Learning With Deep Neural Networks: A Survey". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2019).

[Ju+23] Haotian Ju, Dongyue Li, Aneesh Sharma, and Hongyang R Zhang. "Generalization in Graph Neural Networks: Improved PAC-Bayesian Bounds on Graph Diffusion". In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2023.

[KB15] Diederik P. Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization". In: *International Conference on Learning Representations*. 2015.

[KBV20] Nicolas Keriven, Alberto Bietti, and Samuel Vaiter. "Convergence and Stability of Graph Convolutional Networks on Large Random Graphs". In: *Advances in Neural Information Processing Systems*. 2020.

[KH+09] Alex Krizhevsky, Geoffrey Hinton, et al. *Learning multiple layers of features from tiny images*. 2009.

[Kia+22a] Bobak T Kiani, Randall Balestriero, Yubei Chen, Seth Lloyd, and Yann LeCun. "Joint embedding self-supervised learning in the kernel regime". In: *arXiv preprint arXiv:2209.14884* (2022).

[Kia+22b] Bobak Toussi Kiani, Randall Balestriero, Yubei Chen, Seth Lloyd, and Yann LeCun. "Joint Embedding Self-Supervised Learning in the Kernel Regime". In: *CoRR* abs/2209.14884 (2022). DOI: 10.48550/arXiv.2209.14884. arXiv: 2209.14884.

[Kip+18] Thomas Kipf, Ethan Fetaya, Kuan-Chieh Wang, Max Welling, and Richard Zemel. "Neural relational inference for interacting systems". In: *International conference on machine learning*. PMLR. 2018.

[KJC16] Angjoo Kanazawa, David W Jacobs, and Manmohan Chandraker. "Warpnet: Weakly supervised matching for single-view reconstruction". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2016.

[KL16] M. Kleindessner and U. von Luxburg. *Lens depth function and k-relative neighborhood graph: versatile tools for ordinal data analysis*. 2016. arXiv: 1602.07194 [stat.ML].

[KL17] M. Kleindessner and U. von Luxburg. "Kernel functions based on triplet similarity comparisons". In: *Advances in Neural Information Processing Systems*. 2017, pp. 6807–6817.

[Kra91] Mark A Kramer. "Nonlinear principal component analysis using autoassociative neural networks". In: *AIChE journal* (1991).

[KTO19] Tatsuro Kawamoto, Masashi Tsubaki, and Tomoyuki Obuchi. "Mean-field theory of graph neural networks in graph partitioning". In: *Journal of Statistical Mechanics: Theory and Experiment*. 2019.

[Kun+19] Daniel Kunin, Jonathan Bloom, Aleksandrina Goeva, and Cotton Seed. "Loss Landscapes of Regularized Linear Autoencoders". In: *International Conference on Machine Learning*. 2019.

[KW13] Diederik P Kingma and Max Welling. "Auto-encoding variational bayes". In: *arXiv preprint arXiv:1312.6114* (2013).

[KW17] Thomas N. Kipf and Max Welling. "Semi-Supervised Classification with Graph Convolutional Networks". In: *International Conference on Learning Representations*. 2017.

[KW71] George Kimeldorf and Grace Wahba. "Some results on Tchebycheffian spline functions". In: *Journal of Mathematical Analysis and Applications* (1971).

[LC10a] Y. LeCun and C. Cortes. *MNIST handwritten digit database*. http://yann.lecun.com/exdb/mnist/. 2010.

[LC10b] Yann LeCun and Corinna Cortes. "MNIST handwritten digit database". In: (2010). URL: http://yann.lecun.com/exdb/mnist/.

[Lee+21] Jason D. Lee, Qi Lei, Nikunj Saunshi, and Jiacheng Zhuo. "Predicting What You Already Know Helps: Provable Self-Supervised Learning". In: *Advances in Neural Information Processing Systems*. 2021.

[LHW18] Qimai Li, Zhichao Han, and Xiao ming Wu. "Deeper Insights Into Graph Convolutional Networks for Semi-Supervised Learning". In: *AAAI Conference on Artificial Intelligence*. 2018.

[Li+21a] Guofa Li, Yifan Yang, Tingru Zhang, Xingda Qu, Dongpu Cao, Bo Cheng, and Keqiang Li. "Risk assessment based collision avoidance decision-making for autonomous vehicles in multi-scenarios". In: *Transportation research part C: emerging technologies* (2021).

[Li+21b] Jiangyuan Li, Thanh V Nguyen, Chinmay Hegde, and Raymond K. W. Wong. "Implicit Sparse Regularization: The Impact of Depth and Early Stopping". In: *Advances in Neural Information Processing Systems*. 2021.

[Liu+21] Bo Liu, Ming Ding, Sina Shaham, Wenny Rahayu, Farhad Farokhi, and Zihuai Lin. "When machine learning meets privacy: A survey and outlook". In: *ACM Computing Surveys (CSUR)* (2021).

[Liu+23] Ziyin Liu, Ekdeep Singh Lubana, Masahito Ueda, and Hidenori Tanaka. "What shapes the loss landscape of self supervised learning?" In: *International Conference on Learning Representations*. 2023.

[LLY20] Zehua Lai, Lek-Heng Lim, and Ke Ye. *Simpler Grassmannian optimization*. 2020. URL: https://arxiv.org/abs/2009.13502.

[LMO04] Tao Li, Sheng Ma, and Mitsunori Ogihara. "Entropy-Based Criterion in Categorical Clustering". In: *Proceedings of the International Conference on Machine Learning* (2004).

[Low99] David G Lowe. "Object recognition from local scale-invariant features". In: *Proceedings of the seventh IEEE international conference on computer vision.* 1999.

[Luo+19] Ping Luo, Xinjiang Wang, Wenqi Shao, and Zhanglin Peng. "Towards Understanding Regularization in Batch Normalization". In: *International Conference on Learning Representations.* 2019.

[Lux07] U. von Luxburg. "A Tutorial on Spectral Clustering". In: *Statistics and Computing* (2007).

[LUZ21] Renjie Liao, Raquel Urtasun, and Richard Zemel. "A PAC-Bayesian Approach to Generalization Bounds for Graph Neural Networks". In: *International Conference on Learning Representations.* 2021.

[LZ14] Yu-Feng Li and Zhi-Hua Zhou. "Towards making unlabeled data never hurt". In: *IEEE transactions on pattern analysis and machine intelligence* (2014).

[LZB20] Chaoyue Liu, Libin Zhu, and Mikhail Belkin. "On the linearity of large non-linear models: when and why the tangent kernel is constant". In: *Advances in Neural Information Processing Systems 33.* 2020.

[Maa14] Laurens van der Maaten. "Accelerating t-SNE using tree-based algorithms". In: *The Journal of Machine Learning Research* 15.1 (2014), pp. 3221–3245.

[Mac67] J. Macqueen. "Some methods for classification and analysis of multivariate observations". In: *In 5-th Berkeley Symposium on Mathematical Statistics and Probability.* 1967, pp. 281–297.

[Mau16] Andreas Maurer. "A Vector-Contraction Inequality for Rademacher Complexities". In: *Algorithmic Learning Theory.* Ed. by Ronald Ortner, Hans Ulrich Simon, and Sandra Zilles. Springer International Publishing, 2016.

[MB17] Siyuan Ma and Mikhail Belkin. "Diving into the shallows: a computational perspective on large-scale shallow learning". In: *Advances in Neural Information Processing Systems.* 2017.

[MFB19] A. Morales-Forero and S. Bassetto. "Case Study: A Semi-Supervised Methodology for Anomaly Detection and Diagnosis". In: *International Conference on Industrial Engineering and Engineering Management.* 2019.

[Mik+99] Sebastian Mika, Gunnar Ratsch, Jason Weston, Bernhard Scholkopf, and Klaus-Robert Mullers. "Fisher discriminant analysis with kernels". In: *Neural networks for signal processing IX: Proceedings of the 1999 IEEE signal processing society workshop.* 1999.

[MJN17] B. Mason, L. Jain, and R. Nowak. "Learning low-dimensional metrics". In: *Advances in neural information processing systems.* 2017, pp. 4139–4147.

[MM20] Ishan Misra and Laurens van der Maaten. “Self-supervised learning of pretext-invariant representations”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020.

[MM22] Song Mei and Andrea Montanari. “The generalization error of random features regression: Precise asymptotics and the double descent curve”. In: *Communications on Pure and Applied Mathematics* (2022).

[MMF21] Mohammed Amine El Mrabet, Khalid El Makkaoui, and Ahmed Faize. “Supervised Machine Learning: A Survey”. In: *2021 4th International Conference on Advanced Communication Technologies and Networking (CommNet)*. 2021.

[Moh+22] Abdel-Rahman Mohamed, Hung yi Lee, Lasse Borgholt, Jakob D. Havtorn, Joakim Edin, Christian Igel, Katrin Kirchhoff, Shang-Wen Li, Karen Livescu, Lars Maaløe, Tara N. Sainath, and Shinji Watanabe. “Self-supervised speech representation learning: A review”. In: *IEEE JSTSP Special Issue on Self-Supervised Learning for Speech and Audio Processing* (2022).

[MP16] Andreas Maurer and Massimiliano Pontil. “Bounds for vector-valued function estimation”. In: *arXiv preprint arXiv:1606.01487* (2016).

[MPW16] Ankur Moitra, William Perry, and Alexander S Wein. “How robust are reconstruction thresholds for community detection?” In: *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*. 2016, pp. 828–841.

[MW12] L. van der Maaten and K. Weinberger. “Stochastic triplet embedding”. In: *IEEE International Workshop on Machine Learning for Signal Processing*. 2012, pp. 1–6.

[Nak+21] Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. “Deep double descent: Where bigger models and more data hurt”. In: *Journal of Statistical Mechanics: Theory and Experiment* (2021).

[Net+11] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. “Reading Digits in Natural Images with Unsupervised Feature Learning”. In: *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*. 2011.

[Ney+17] Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nati Srebro. “Exploring Generalization in Deep Learning”. In: *Advances in Neural Information Processing Systems*. 2017.

[NJW01] Andrew Y. Ng, Michael I. Jordan, and Yair Weiss. “On Spectral Clustering: Analysis and an Algorithm”. In: *Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic*. Cambridge, MA, USA: MIT Press, 2001.

[NK19] Vaishnavh Nagarajan and J. Zico Kolter. “Uniform convergence may be unable to explain generalization in deep learning”. In: *Advances in Neural Information Processing Systems*. 2019.

[NWH21] Thanh V. Nguyen, Raymond K. W. Wong, and Chinmay Hegde. “Benefits of Jointly Training Autoencoders: An Improved Neural Tangent Kernel Analysis”. In: *IEEE Transactions on Information Theory* (2021).

[OLV18] Aäron van den Oord, Yazhe Li, and Oriol Vinyals. “Representation Learning with Contrastive Predictive Coding”. In: *CoRR* abs/1807.03748 (2018). arXiv: 1807.03748. URL: http://arxiv.org/abs/1807.03748.

[OS20a] Kenta Oono and Taiji Suzuki. “Graph Neural Networks Exponentially Lose Expressive Power for Node Classification”. In: *International Conference on Learning Representations*. 2020.

[OS20b] Kenta Oono and Taiji Suzuki. “Optimization and Generalization Analysis of Transduction through Gradient Boosting and Application to Multi-scale Graph Neural Networks”. In: *Advances in Neural Information Processing Systems*. 2020.

[Pas+19] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. “PyTorch: An Imperative Style, High-Performance Deep Learning Library”. In: *Advances in Neural Information Processing Systems 32*. 2019.

[Pea01] Karl Pearson. “On lines and planes of closest fit to systems of points in space”. In: *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* (1901).

[Ped+11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* (2011).

[PEG20] Michaël Perrot, Pascal Esser, and Debarghya Ghoshdastidar. “Near-optimal comparison based clustering”. In: *Advances in Neural Information Processing Systems* (2020).

[PKK18] Arnu Pretorius, Steve Kroon, and Herman Kamper. “Learning Dynamics of Linear Denoising Autoencoders”. In: *International Conference on Machine Learning*. 2018.

[PM17] Pedro Ponte and Roger G Melko. “Kernel methods for interpretable machine learning of order parameters”. In: *Physical Review B* (2017).

[RA15] Ryan A. Rossi and Nesreen K. Ahmed. “The Network Data Repository with Interactive Graph Analytics and Visualization”. In: *AAAI Conference on Artificial Intelligence*. 2015.

[Rad+18] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. "Improving language understanding by generative pre-training". In: (2018).

[Ran+01] Koustubh Ranade, Mau-Song Chang, Chih-Tai Ting, Dee Pei, Chin-Fu Hsiao, Michael Olivier, Robert Pesich, Joan Hebert, Yii-Der I Chen, Victor J Dzau, et al. "High-throughput genotyping with single nucleotide polymorphisms". In: *Genome Research* (2001).

[Ran71] William M. Rand. "Objective Criteria for the Evaluation of Clustering Methods". In: *Journal of the American Statistical Association* (1971).

[RBU20] Adityanarayanan Radhakrishnan, Mikhail Belkin, and Caroline Uhler. "Overparameterized neural networks implement associative memory". In: *Proceedings of the National Academy of Sciences of the United States of America* (2020).

[RG22] Maria Refinetti and Sebastian Goldt. "The dynamics of representation learning in shallow, non-linear autoencoders". In: *International Conference on Machine Learning.* 2022.

[RG23] Maria Rigaki and Sebastian Garcia. "A survey of privacy attacks in machine learning". In: *ACM Computing Surveys* (2023).

[RL03] Volker Roth and Tilman Lange. "Feature Selection in Clustering Problems". In: *Advances in Neural Information Processing Systems.* 2003.

[RR07] Ali Rahimi and Benjamin Recht. "Random Features for Large-Scale Kernel Machines". In: *Advances in Neural Information Processing Systems 20.* 2007.

[RS18] Khalid Raza and Nripendra Kumar Singh. "A Tour of Unsupervised Deep Learning for Medical Image Analysis". In: *Current medical imaging* (2018).

[Sak+19] C. Okan Sakar, Gorkem Serbes, Aysegul Gunduz, Hunkar C. Tunc, Hatice Nizam, Betul Erdogdu Sakar, Melih Tutuncu, Tarkan Aydin, M. Erdem Isenkul, and Hulya Apaydin. "A comparative analysis of speech signal processing algorithms for Parkinson's disease classification and the use of the tunable Q-factor wavelet transform". In: *Applied Soft Computing* (2019).

[SBC05] N. Stewart, G. D. A. Brown, and N. Chater. "Absolute identification by relative judgment." In: *Psychological review* 112.4 (2005), p. 881.

[Sca+09] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. "The Graph Neural Network Model". In: *IEEE Transactions on Neural Networks.* 2009.

[Sch+17] Thomas Schlegl, Philipp Seeböck, Sebastian M Waldstein, Ursula Schmidt-Erfurth, and Georg Langs. "Unsupervised anomaly detection with generative adversarial networks to guide marker discovery". In: *International conference on information processing in medical imaging.* 2017.

[Sch87] J.C. Schlimmer. *Mushroom.* UCI Machine Learning Repository. 1987.

[SEG22] Mahalakshmi Sabanayagam, Pascal Esser, and Debarghya Ghoshdastidar. *Representation Power of Graph Convolutions : Neural Tangent Kernel Analysis*. 2022. URL: https://arxiv.org/abs/2210.09809.

[Ser+18] Pierre Sermanet, Corey Lynch, Yevgen Chebotar, Jasmine Hsu, Eric Jang, Stefan Schaal, Sergey Levine, and Brain. "Time-contrastive networks: Self-supervised learning from video". In: *IEEE international conference on robotics and automation*. 2018.

[Sha+22] Anshul Shah, Suvrit Sra, Rama Chellappa, and Anoop Cherian. "Max-Margin Contrastive Learning". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. 2022.

[She62] R. N. Shepard. "The analysis of proximities: Multidimensional scaling with an unknown distance function. I." In: *Psychometrika* 27.2 (1962), pp. 125–140.

[Shi+23] Cheng Shi, Liming Pan, Hong Hu, and Ivan Dokmanić. *Homophily modulates double descent generalization in graph convolution networks*. 2023. arXiv: 2212.13069 [cs.LG].

[SHS01] Bernhard Schölkopf, Ralf Herbrich, and Alexander J. Smola. "A Generalized Representer Theorem". In: *Annual Conference on Computational Learning Theory*. 2001.

[Sig+89] Vince Sigillito., Wing S., Hutton L., and Baker K. *Ionosphere*. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5W01B. 1989.

[Sim51] E. H. Simpson. "The Interpretation of Interaction in Contingency Tables". In: *Journal of the Royal Statistical Society. Series B (Methodological)* (1951).

[SM00] J. Shi and J. Malik. "Normalized cuts and image segmentation". In: *IEEE Transactions on pattern analysis and machine intelligence* 22.8 (2000), pp. 888–905.

[SMA21] Nikunj Saunshi, Sadhika Malladi, and Sanjeev Arora. "A Mathematical Exploration of Why Language Models Help Solve Downstream Tasks". In: *International Conference on Learning Representations*. 2021.

[SMG14] Andrew M. Saxe, James L. McClelland, and Surya Ganguli. "Exact solutions to the nonlinear dynamics of learning in deep linear neural networks". In: *International Conference on Learning Representations*. 2014.

[Son+17] Tzu-Hsi Song, Victor Sanchez, Hesham ElDaly, and Nasir M. Rajpoot. "Hybrid deep autoencoder with Curvature Gaussian for detection of various types of cells in bone marrow trephine biopsy images". In: *IEEE 14th International Symposium on Biomedical Imaging*. 2017.

[Sou+18] Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, and Nathan Srebro. "The Implicit Bias of Gradient Descent on Separable Data". In: *Journal of Machine Learning Research* (2018).

[SS02] Bernhard Schölkopf and Alexander Johannes Smola. *Learning with Kernels: support vector machines, regularization, optimization, and beyond.* Adaptive computation and machine learning series. MIT Press, 2002.

[SSBD14] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms.* Cambridge University Press, 2014.

[SSM97] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. “Kernel principal component analysis”. In: *International conference on artificial neural networks.* Springer. 1997.

[SSM98] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. “Nonlinear Component Analysis as a Kernel Eigenvalue Problem”. In: *Neural Computation* (1998).

[Ste+19] Steffen Steffen, Alexei Baevski, Ronan Collobert, and Michael Auli. *wav2vec: Unsupervised Pre-training for Speech Recognition.* 2019. DOI: 10.48550/ARXIV.1904.05862.

[STH18] Franco Scarselli, Ah Chung Tsoi, and Markus Hagenbuchner. In: *The Vapnik–Chervonenkis dimension of graph and recursive neural networks.* 2018.

[SVL14] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. “Sequence to Sequence Learning with Neural Networks”. In: *Advances in neural information processing systems.* 2014.

[SY14] Mayu Sakurada and Takehisa Yairi. “Anomaly Detection Using Autoencoders with Nonlinear Dimensionality Reduction”. In: *Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis.* 2014.

[TBH23] Lindsay C Todman, Alex Bush, and Amelia SC Hood. “‘Small Data’for big insights in ecology”. In: *Trends in Ecology & Evolution* (2023).

[TBK14] Ilya Tolstikhin, Gilles Blanchard, and Marius Kloft. “Localized Complexities for Transductive Learning”. In: *Conference on Learning Theory.* 2014.

[The+17] L. Theis, W. Shi, A. Cunningham, and F. Huszár. “Lossy Image Compression with Compressive Autoencoders”. In: *International Conference on Learning Representations.* 2017.

[Tia+14] Fei Tian, Bin Gao, Qing Cui, Enhong Chen, and Tie-Yan Liu. “Learning Deep Representations for Graph Clustering”. In: *Proceedings of the AAAI Conference on Artificial Intelligence* (2014).

[Tia22] Yuandong Tian. “Understanding Deep Contrastive Learning via Coordinate-wise Optimization”. In: *Advances in Neural Information Processing Systems.* 2022.

[TK22] Nikolaos Tsilivis and Julia Kempe. “What Can the Neural Tangent Kernel Tell Us About Adversarial Robustness?” In: *Advances in Neural Information Processing Systems* (2022).

[TKH21] Christopher Tosh, Akshay Krishnamurthy, and Daniel Hsu. "Contrastive learning, multi-view redundancy, and linear models". In: *Algorithmic Learning Theory.* Proceedings of Machine Learning Research. 2021.

[TKM20] Ryan Theisen, Jason M. Klusowski, and Michael W. Mahoney. "Good linear classifiers are abundant in the interpolating regime". In: *Computing Research Repository* (2020).

[TLP16] Ilya O. Tolstikhin and David Lopez-Paz. "Minimax Lower Bounds for Realizable Transductive Classification". In: *ArXiv.* Vol. 1602.03027. 2016.

[Ukk17] A. Ukkonen. "Crowdsourced correlation clustering with relative distance comparisons". In: *arXiv preprint arXiv:1709.08459* (2017).

[Van+23] Leena Chennuru Vankadara, Michael Lohaus, Siavash Haghiri, Faiz Ul Wahab, and Ulrike von Luxburg. "Insights into Ordinal Embedding Algorithms: A Systematic Evaluation". In: *Journal of Machine Learning Research* (2023).

[Vap82] Vladimir Vapnik. "Estimation of Dependences Based on Empirical Data". In: *Springer Series in Statistics.* 1982.

[Vap98] V.N. Vapnik. "Statistical Learning Theory". In: *A Wiley-Interscience publication.* 1998.

[VC71] V. N. Vapnik and A. Ya. Chervonenkis. "On the Uniform Convergence of Relative Frequencies of Events to Their Probabilities". In: *Theory of Probability & Its Applications.* 1971.

[Vel+18] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. "Graph Attention Networks". In: *International Conference on Learning Representations.* 2018.

[Vin+10] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, Pierre-Antoine Manzagol, and Léon Bottou. "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion." In: *Journal of machine learning research* (2010).

[von07] U. von Luxburg. "A tutorial on spectral clustering". In: *Statistics and computing* 17.4 (2007), pp. 395–416.

[VZ19] Saurabh Verma and Zhi-Li Zhang. "Stability and Generalization of Graph Convolutional Neural Networks". In: *Computing Research Repository.* 2019.

[Wag82] Clifford H. Wagner. "Simpson's Paradox in Real Life". In: *The American Statistician* (1982).

[Wah90] Grace Wahba. *Spline Models for Observational Data.* SIAM, 1990.

[Wan+15] Zhangyang Wang, Yingzhen Yang, Shiyu Chang, Jinyan Li, Simon Fong, and Thomas S. Huang. "A Joint Optimization Framework of Sparse Coding and Discriminative Clustering". In: *Proceedings of the 24th International Conference on Artificial Intelligence.* AAAI Press, 2015.

[Wan+16] Zhangyang Wang, Shiyu Chang, Jiayu Zhou, and Meng Wang. "Learning A Task-Specific Deep Architecture For Clustering". In: Proceedings of the 2016 SIAM International Conference on Data Mining, 2016.

[WKB14] M. Wilber, S. Kwak, and S. Belongie. "Cost-Effective HITs for Relative Similarity Comparisons". In: *Human Computation and Crowdsourcing (HCOMP)*. Pittsburgh, 2014.

[Wol+95] William Wolberg, Olvi Mangasarian, Nick Street, and W. Street. *Breast Cancer Wisconsin (Diagnostic)*. UCI Machine Learning Repository. 1995.

[Wu+18] Le Wu, Peijie Sun, Richang Hong, Yanjie Fu, Xiting Wang, and Meng Wang. "Socialgcn: An efficient graph convolutional network based model for social recommendation". In: *arXiv preprint arXiv:1811.02815* (2018).

[Wu+20] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S Yu. "A Comprehensive Survey on Graph Neural Networks". In: *IEEE transactions on neural networks and learning systems*. 2020.

[WXM21] Colin Wei, Sang Michael Xie, and Tengyu Ma. "Why Do Pretrained Language Models Help in Downstream Tasks? An Analysis of Head and Prompt Tuning". In: *Advances in Neural Information Processing Systems*. 2021.

[XGF16] Junyuan Xie, Ross Girshick, and Ali Farhadi. "Unsupervised Deep Embedding for Clustering Analysis". In: (2016).

[Xin+02] Eric Xing, Michael Jordan, Stuart J Russell, and Andrew Ng. "Distance Metric Learning with Application to Clustering with Side-Information". In: *Advances in Neural Information Processing Systems*. Ed. by S. Becker, S. Thrun, and K. Obermayer. 2002.

[XJL20] M. Xu, V. Jog, and P.-L. Loh. "Optimal rates for community estimation in the weighted stochastic block model". In: *The Annals of Statistics* 48.1 (2020), pp. 183–204.

[XNZ08] Shiming Xiang, Feiping Nie, and Changshui Zhang. "Learning a Mahalanobis distance metric for data clustering and classification". In: *Pattern Recognition* (2008).

[Xu+19] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. "How Powerful are Graph Neural Networks?" In: *International Conference on Learning Representations*. 2019.

[Xu+23] Pengcheng Xu, Xiaobo Ji, Minjie Li, and Wencong Lu. "Small data machine learning in materials science". In: *npj Computational Materials* (2023).

[XX15] Pengtao Xie and Eric P. Xing. "Integrating Image Clustering and Codebook Learning". In: 2015.

[Yan+17] Bo Yang, Xiao Fu, Nicholas D. Sidiropoulos, and Mingyi Hong. "Towards K-Means-Friendly Spaces: Simultaneous Deep Learning and Clustering". In: *International Conference on Machine Learning*. 2017.

[YHM94] Wei-Yong Yan, U. Helmke, and J.B. Moore. “Global analysis of Oja’s flow for neural networks”. In: *IEEE Transactions on Neural Networks* (1994).

[Yin+18] Rex Ying, Jiaxuan You, Christopher Morris, Xiang Ren, William L. Hamilton, and Jure Leskovec. “Hierarchical Graph Representation Learning with Differentiable Pooling”. In: *Advances in Neural Information Processing Systems*. 2018.

[YMA21] AFOUDI Yassine, LAZAAR Mohamed, and Mohammed Al Achhab. “Intelligent recommender system based on unsupervised machine learning and demographic attributes”. In: *Simulation Modelling Practice and Theory* (2021).

[You87] F. W. Young. *Multidimensional scaling: History, theory, and applications.* Lawrence Erlbaum Associates, 1987.

[YP16] S.-Y. Yun and A. Proutiere. “Optimal cluster recovery in the labeled stochastic block model”. In: *Advances in Neural Information Processing Systems*. 2016, pp. 965–973.

[YS16] B. Yan and P. Sarkar. “On robustness of kernel clustering”. In: *Advances in Neural Information Processing Systems*. 2016, pp. 3098–3106.

[YSC17] Bowei Yan, Purnamrita Sarkar, and Xiuyuan Cheng. “Provable Estimation of the Number of Blocks in Block Models”. In: *International Conference on Artificial Intelligence and Statistics*. 2017.

[YSC18] B. Yan, P. Sarkar, and X. Cheng. “Provable Estimation of the Number of Blocks in Block Models”. In: *International Conference on Artificial Intelligence and Statistics*. 2018, pp. 1185–1194.

[Zen+17] Kun Zeng, Jun Yu, Ruxin Wang, Cuihua Li, and Dacheng Tao. “Coupled Deep Autoencoder for Single Image Super-Resolution”. In: *IEEE Transactions on Cybernetics* (2017).

[Zha+17] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. “Understanding deep learning requires rethinking generalization”. In: *International Conference on Learning Representations*. 2017.

[Zha+18] Muhan Zhang, Zhicheng Cui, Marion Neumann, and Yixin Chen. “An End-to-End Deep Learning Architecture for Graph Classification”. In: *AAAI Conference on Artificial Intelligence*. 2018.

[Zha+22a] Chaoning Zhang, Chenshuang Zhang, Junha Song, John Seon Keun Yi, Kang Zhang, and In So Kweon. “A survey on masked autoencoder for self-supervised learning in vision and beyond”. In: *arXiv preprint arXiv:2208.00173* (2022).

[Zha+22b] Chaoning Zhang, Chenshuang Zhang, Junha Song, John Seon Keun Yi, Kang Zhang, and In So Kweon. “A Survey on Masked Autoencoder for Self-supervised Learning in Vision and Beyond”. In: (2022).

[Zha+23] Runtian Zhai, Bingbin Liu, Andrej Risteski, Zico Kolter, and Pradeep Ravikumar. “Understanding Augmentation-based Self-Supervised Representation Learning via RKHS Approximation”. In: *arXiv preprint arXiv:2306.00788* (2023).

[Zho+20] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. "Graph neural networks: A review of methods and applications". In: *AI open* (2020).

[Zhu+23] Zhijian Zhuo, Yifei Wang, Jinwen Ma, and Yisen Wang. "Towards a Unified Theoretical Understanding of Non-contrastive Learning via Rank Differential Mechanism". In: *The Eleventh International Conference on Learning Representations*. 2023.

[ZLZ20] Pengfei Zhou, Tianyi Li, and Pan Zhang. "Phase transitions and optimal algorithms for semisupervised classifications on graphs: From belief propagation to graph convolution network". In: *Physical Review Research*. 2020.

[ZP17] Chong Zhou and Randy C. Paffenroth. "Anomaly Detection with Robust Deep Autoencoders". In: *International Conference on Knowledge Discovery and Data Mining* (2017).