

Deep Learning Approaches for Dependency Analysis in Large Ensemble Datasets

Fatemeh Farokhmanesh

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

Doktorin der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitz:

Prof. Dr. Nassir Navab

Prüfende der Dissertation:

1. Prof. Dr. Rüdiger Westermann
2. Prof. Dr. Lars Linsen

Die Dissertation wurde am 25.09.2024 bei der Technischen Universität München eingereicht und durch die TUM School of Computation, Information and Technology am 16.01.2025 angenommen.

ABSTRACT

The exponential growth of data volumes, particularly in the contexts of large-scale physical simulations, ensemble simulations, and extensive time series analysis, has fundamentally transformed the landscape of scientific research. These datasets, characterized by their vast size, high dimensionality, and complexity, hold the potential to extend new insights into the underlying structures. However, with this growth comes the tough challenge of effectively analyzing and interpreting the intricate dependencies within the data. While useful for small-sized datasets, traditional statistical methods struggle to efficiently capture the full scope of dependencies present in these large and complex datasets. This limitation necessitates the development of advanced deep learning methodologies that can handle the scale of these datasets and reveal the dependencies compactly and efficiently.

This dissertation addresses these challenges by introducing and evaluating deep learning-based approaches for dependency analysis in large ensemble datasets. Our research is structured in three central steps, each focusing on different datasets and aspects of dependency analysis.

Firstly, the concept of variable-to-variable (V2V) transfer in the context of large physical simulation datasets is investigated. Here, convolutional neural networks (CNNs) are employed to model and predict multiple target variables based on a set of input variables. CNNs, traditionally used in image processing due to their ability to capture spatial hierarchies, are adapted for this task by leveraging their capacity to identify patterns and dependencies between different variables in the data. It is found that the limited redundancies between pairs of variable fields restrict the application of the original V2V algorithm. Consequently, the predictive strength of variables is assessed by examining the learning behavior of V2V reconstruction networks through an ablation study. It is demonstrated that efficient V2V transfer can be achieved when groups of variables are considered for transfer, and an algorithm is proposed to implement this approach. Furthermore, the model behavior is explored using the feature analysis method, layer-wise relevance propagation (LRP). Regions where field values significantly impact local reconstruction accuracy are identified. The LRP results show that different model architectures learn distinct mapping functions influenced by their inductive biases. In this study, the U-Net model produced more physically interpretable relevance maps, while the ResNet architecture is more constrained in capturing spatial dependencies due to its network design.

Secondly, the focus of this dissertation is extended to the analysis of statistical dependencies, such as Pearson correlation and mutual information (MI), within and between

different variables in large ensemble simulations. Neural dependence fields (NDFs) are presented as the first neural network capable of efficiently learning and reconstructing statistical dependencies between physical variables across different spatial locations in large three-dimensional (3D) simulation ensembles. To capture non-linear dependencies, MI is used as a key measure, expanding the approach beyond linear dependence measures. The method is demonstrated on a weather forecast ensemble consisting of 1000 members, each containing multiple physical variables arranged on a 3D simulation grid. By eliminating the need for compute-intensive statistical estimators at runtime, the approach significantly reduces memory and computational demands for reconstructing dependence structures. This reduction enables seamless integration with a GPU-accelerated direct volume renderer, supporting real-time interaction and visualization of mutual dependencies for any chosen point in the domain.

Finally, this research analyzes MI between different time windows within large time series datasets by extending the NDF model to capture temporal dependencies. Time series data, characterized by its sequential nature and potential for long-range dependencies, presents unique challenges in terms of dependency analysis. To efficiently capture and represent the dependencies between synchronized and lagged time windows across different time series, time series neural mutual information fields (TNMIFs), a compact and efficient neural data structure is proposed. TNMIF is trained to efficiently reconstruct MI across various time window positions within large time series datasets. Additionally, the matrix representing mutual similarities between all time series, based on windowed MI estimates, is computed instantly. To capture dependencies in lagged time series, an extended version of the data structure is trained to reconstruct MI for positively (future) and negatively (past) lagged windows.

ZUSAMMENFASSUNG

Das exponentielle Wachstum der Datenmengen, insbesondere im Kontext großangelegter physikalischer Simulationen, Ensemble-Simulationen und umfangreicher Zeitreihenanalysen, hat die wissenschaftliche Forschung grundlegend verändert. Diese Datensätze, die sich durch ihre enorme Größe, hohe Dimensionalität und Komplexität auszeichnen, bieten das Potenzial, neue Einblicke in zugrunde liegende Strukturen zu gewinnen. Mit diesem Wachstum geht jedoch die große Herausforderung einher, die komplexen Abhängigkeiten in den Daten effektiv zu analysieren und zu interpretieren. Während traditionelle statistische Methoden bei kleineren Datensätzen nützlich sind, stoßen sie bei der effizienten Erfassung des vollen Umfangs der Abhängigkeiten in großen und komplexen Datensätzen an ihre Grenzen. Diese Einschränkung macht die Entwicklung fortschrittlicher Deep-Learning-Methoden notwendig, die in der Lage sind, mit der Größe dieser Datensätze umzugehen und die Abhängigkeiten kompakt und effizient zu erfassen.

Diese Dissertation greift diese Herausforderungen auf, indem sie Deep-Learning-basierte Ansätze zur Analyse von Abhängigkeiten in großen Ensemble-Datensätzen vorstellt und bewertet. Die Forschung ist in drei zentrale Schritte unterteilt, die sich jeweils auf unterschiedliche Datensätze und Aspekte der Abhängigkeitsanalyse konzentrieren.

Zunächst wird das Konzept des Variablen-zu-Variablen (V2V) Transfers im Kontext großer physikalischer Simulationsdatensätze untersucht. Hierbei werden Convolutional Neural Networks (CNNs) eingesetzt, um mehrere Zielvariablen basierend auf einer Reihe von Eingangsvariablen zu modellieren und vorherzusagen. CNNs, die traditionell in der Bildverarbeitung verwendet werden, um räumliche Hierarchien zu erfassen, werden für diese Aufgabe angepasst, indem ihre Fähigkeit genutzt wird, Muster und Abhängigkeiten zwischen verschiedenen Variablen in den Daten zu identifizieren. Es wird festgestellt, dass die begrenzten Redundanzen zwischen Variablenpaaren die Anwendung des ursprünglichen V2V-Algorithmus einschränken. Folglich wird die prädiktive Stärke der Variablen durch die Analyse des Lernverhaltens von V2V-Rekonstruktionsnetzwerken in einer Ablationsstudie bewertet. Es wird gezeigt, dass ein effizienter V2V-Transfer möglich wird, wenn Gruppen von Variablen für den Transfer betrachtet werden, und ein Algorithmus wird vorgeschlagen, um diesen Ansatz umzusetzen. Darüber hinaus wurde das Modellverhalten mithilfe der Feature-Analyse-Methode Layer-wise Relevance Propagation (LRP) untersucht. Regionen, in denen die Feldwerte die lokale Rekonstruktionsgenauigkeit signifikant beeinflussen, wurden identifiziert. Die LRP-Ergebnisse zeigen, dass verschiedene Modellarchitekturen unterschiedliche Abbildungsfunktionen lernen, die durch ihre induktiven Vorannahmen beeinflusst werden. In dieser Studie erzeugte das U-Net-Modell physikalisch interpretierbarere Relevanzkarten, während die ResNet-Architektur aufgrund

ihres Netzaufbaus stärker in der Erfassung räumlicher Abhängigkeiten eingeschränkt ist.

Zweitens wird der Fokus dieser Dissertation auf die Analyse statistischer Abhängigkeiten, wie zum Beispiel der Pearson-Korrelation und der gegenseitigen Information (MI), innerhalb und zwischen verschiedenen Variablen in großen Ensemble-Simulationen erweitert. Neural Dependence Fields (NDFs) werden als das erste neuronale Netzwerk vorgestellt, das in der Lage ist, statistische Abhängigkeiten zwischen physikalischen Variablen an verschiedenen räumlichen Positionen in großen 3D-Simulationsensembles effizient zu erlernen und zu rekonstruieren. Um nicht-lineare Abhängigkeiten zu erfassen, wird MI als zentrales Maß verwendet, wodurch der Ansatz über lineare Abhängigkeitsmaße hinausgeht. Die Methode wird an einem Wettervorhersage-Ensemble demonstriert, das aus 1000 Mitgliedern besteht, wobei jedes mehrere physikalische Variablen auf einem 3D-Simulationsgitter speichert. Durch die Eliminierung rechenintensiver statistischer Schätzer zur Laufzeit wird der Speicher- und Rechenaufwand für die Rekonstruktion von Abhängigkeitsstrukturen erheblich reduziert. Diese Reduzierung ermöglicht die nahtlose Integration mit einem GPU-beschleunigten Direct-Volume-Renderer, der die Echtzeit-Interaktion und Visualisierung gegenseitiger Abhängigkeiten für jeden ausgewählten Punkt im Datenbereich unterstützt.

Schließlich wird in dieser Forschung die MI zwischen verschiedenen Zeitfenstern in großen Zeitreihen-Datensätzen analysiert, indem das NDF-Modell erweitert wird, um zeitliche Abhängigkeiten zu erfassen. Zeitreihendaten, die durch ihre sequentielle Natur und das Potenzial für langfristige Abhängigkeiten gekennzeichnet sind, stellen einzigartige Herausforderungen für die Abhängigkeitsanalyse dar. Um die Abhängigkeiten zwischen synchronisierten und verzögerten Zeitfenstern über verschiedene Zeitreihen effizient zu erfassen und darzustellen, wird Time Series Neural Mutual Information Fields (TNMIFs), eine kompakte und effiziente neuronale Datenstruktur, vorgeschlagen. TNMIF wird trainiert, um MI über verschiedene Zeitfensterpositionen innerhalb großer Zeitreihen-Datensätze effizient zu rekonstruieren. Darüber hinaus wird die Matrix, die die gegenseitigen Ähnlichkeiten zwischen allen Zeitreihen basierend auf MI-Schätzungen von Zeitfenstern darstellt, sofort berechnet. Um Abhängigkeiten in verzögerten Zeitreihen zu erfassen, wurde eine erweiterte Version der Datenstruktur trainiert, um MI für positiv (zukünftige) und negativ (vergangene) verzögerte Fenster zu rekonstruieren.

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to all the people who supported me throughout my research journey.

First of all, I would like to thank my supervisor, Prof. Dr. Rüdiger Westermann, for his invaluable advice, feedback and support during my time in his research group. I also would like to thank the co-authors of the publications I worked on, Christoph Neuhauser, Kevin Höhle, Martin Weissmann, Takemasa Miyoshi, and Tobias Necker.

Special thanks to Susanne Weitz and Sebastian Wohner for their support in all organizational, administrative, and technical matters. I also thank my colleagues who have been great companions throughout my Ph.D. journey, in particular, Behdad, Benjamin, Björn, Christian, Christoph, Felix, Georg, Han, Hao, Josef, Junpeng, Junwei, Kevin, Luca, Ludwig, Mario, Nilam, Nils, Patrick, Philipp, Qiang, Rene, Sebastian, Shuvayan, and Simon.

Finally, I am deeply grateful to my family. To my partner in life and my best friend, Alireza. To My parents, Zhila and Hadi, and my brothers, Payam and Peyman. This journey wouldn't be possible without your support and encouragement.

CONTENTS

Abstract	iii
Zusammenfassung	v
Acknowledgments	vii
Contents	viii
Acronyms	xi
1 Introduction	1
1.1 Motivation	1
1.2 Contributions	3
1.3 List of Publications	5
1.4 Outline	6
2 Fundamentals and Related Work	7
2.1 Large Ensemble Datasets	7
2.1.1 Weather Benchmark Datasets	7
2.1.2 Convective-scale Ensemble Simulations	8
2.1.3 Large Time Series	9
2.2 Dependency Analysis	10
2.2.1 Statistical Dependencies	10
2.2.1.1 Pearson Correlation Coefficient	11
2.2.1.2 Mutual Information	12
2.2.2 Variable-to-Variable Transfer Dependence Measure	12

2.3	Deep Learning	15
2.3.1	Multilayer Perceptrons	16
2.3.2	Convolutional Neural Networks	19
2.3.2.1	U-Net	20
2.3.2.2	ResNet	22
2.3.3	Feature Analysis using Layer-wise Relevance Propagation	23
3	Dependency Analysis and Reconstruction using Deep Neural Networks	26
3.1	CNN-based Variable-to-Variable Transfer	26
3.1.1	Network Architectures	28
3.1.2	Evaluations and Results	28
3.2	Neural Field-based Statistical Dependency Reconstruction	31
3.2.1	Interactive Visualization of Statistical Dependencies in 3D Simulation Ensembles	31
3.2.1.1	Neural Fields	31
3.2.1.2	NDFs Training and Evaluations	34
3.2.2	Application of Neural Fields in Mutual Information Estimation in Large Time Series	35
3.2.2.1	Network Architecture	37
3.2.2.2	TNMIFs Evaluations	39
4	Paper A: Deep Learning-based Parameter Transfer in Meteorological Data	42
5	Paper B: Neural Fields for Interactive Visualization of Statistical Dependencies in 3D Simulation Ensembles	43
6	Paper C: A Compact and Efficient Neural Data Structure for Mutual Information Estimation in Large Timeseries	44
7	Conclusion and Outlook	45
7.1	Conclusion	45
7.2	Future Works	46

Bibliography	48
Paper A – Published Version	61
Paper B – Published Version	85
Paper C – Published Version	93
Reuse Permission Grant of Publication A	105
License of Publications B and C	106

ACRONYMS

CNN	convolutional neural network.
CSEns	convective-scale multi-variable ensemble.
I/O	input/output.
LRP	layer-wise relevance propagation.
MI	mutual information.
MLP	multilayer perceptron.
NDF	neural dependence field.
TNMIF	time series neural mutual information field.
V2V	variable-to-variable.

1 Introduction

1.1 Motivation

Effectively capturing dependencies in large multi-variable datasets in a compact and efficient way can address key challenges related to the storage and analysis of these datasets. In the realm of physical simulations, large-scale computing facilities often conduct simulations, with data being streamed to and stored on external file systems for archiving and subsequent examination. The volume of data produced has now reached levels where the speed of data transfer between computing devices and file systems—known as input/output (I/O) operations—constitutes a major bottleneck. Over the past decade, the computational capabilities of supercomputers have increased by about two orders of magnitude, while the storage and data retrieval capabilities have only improved by about one order of magnitude. This widening gap between computational power and I/O capabilities renders the traditional simulation workflow increasingly problematic, necessitating the elimination of streaming or the simulation of data that can be derived from the generated results.

In this research context, deep-learning-based V2V transfer, as proposed by [HZX⁺20], has emerged as a promising method for optimizing information transfer in scenarios where spatio-temporal multi-variable simulations can be executed much faster than the time it takes to store the resulting data on a file system. V2V transfer treats each simulated variable as an independent entity. It introduces an algorithm to identify groups of similar variables and a representative variable from each group, from which the other variables can be inferred. V2V transfer operates by representing all variables in a common feature space, the so-called latent-space, which a CNN learns. Within the latent-space, the method identifies subsets of similar variables. Then, for each subset, the most representative variable is selected. A network trained suitably is then used to reconstruct the remaining variables in each subset considering the representative one. By using this method, the bandwidth requirements for storing multi-variable simulation datasets are highly reduced. Rather than storing all the variables, the representative variables and the weights of the reconstruction networks must be stored, allowing a more efficient use of storage resources. However, in scenarios where the lack of similarities between pairs of variables exist, it prohibits the use of the original V2V transfer algorithm. In this dissertation, V2V transfer is considered an example of dependence measures, and an improved deep learning-based V2V transfer is proposed building upon the original V2V transfer, which enables the reconstruction of a group of variables from another group instead of transferring only between pair of variables. Then, a feature analysis

method is utilized to validate the dependencies CNN models learned between different variables.

To further uncover dependencies within and between different variables, it is crucial to analyze statistical dependencies. Estimating statistical dependencies between physical variables across different spatial locations is essential for understanding physical systems in various scientific and engineering fields. A critical application of this lies in meteorology, where accurate weather forecasting depends on large numerical simulations. It is crucial to analyze statistical relationships, such as spatio-temporal auto-correlations within forecast fields and correlations between different forecast variables, to convert volumetric ensemble data into reliable forecasts. The exponential increase in data production poses significant challenges, making these analyses increasingly costly when using traditional methods. As an example, consider a simulation ensemble comprising 1,000 members across a $250 \times 352 \times 20$ grid [NGW⁺20]. Storing all dependencies in such a scenario demands over three terabytes of memory, which is impractical. Moreover, calculating dependencies between any two points in real-time necessitates the entire ensemble to be available in working memory. This process, alongside the computation of more intricate measures like MI, takes several minutes to compute just for one single point the MI to all other points on a high-end multi-core CPU, thus impeding the interactive analysis of dependency structures.

Another important application is in the analysis of large time series datasets. Efficiently estimating the MI between two discrete random variables in this context is challenging. This process requires calculating the conditional entropy of the two variables by using their joint histograms to estimate the joint discrete probability density function. The Kraskov MI estimator [KSG04] provides entropy estimates based on the Chebyshev distances between the k -th nearest neighbors in the joint distribution space. For the large time series datasets, estimating Kraskov MI for all sliding windows in a selected time series and their corresponding windows in all other time series takes several seconds. This computation time increases further when considering dependencies between lagged time series, meaning that for each window in a selected time series, MI values are calculated with both future (positively lagged) and past (negatively lagged) windows in each of the other time series.

While the study of statistical dependence between random variables is well-established, with various measures available for assessing linear and non-linear relationships, determining these relationships in 3D ensemble fields or large time series remains challenging. The main challenges include the on-the-fly computation of dependencies, which can be computationally expensive, and the impracticality of storing all point-to-point dependencies due to the large amount of memory required. In this dissertation, these challenges are addressed using novel deep learning methods.

1.2 Contributions

Several contributions are presented in this thesis, developed in collaboration with other researchers, to address the challenges outlined above. The following list summarizes the concrete contributions of this thesis:

- The effectiveness of deep learning-based variable-to-variable (V2V) transfer [HZX⁺20] is evaluated in processing earth-system-related data and identifies limitations in the existing methodology. Based on these findings, an enhanced approach is proposed in which a single CNN is trained on meteorological archives to learn general relationships between subsets of variables. This method focuses on variable subsets that exhibit diverse characteristics and variations across two-dimensional space and time. The capabilities of the proposed method are illustrated using two exemplary datasets representing different types of meteorological data. The first dataset is a global reanalysis dataset from the WeatherBench benchmark dataset [RDS⁺20], which is used to evaluate the performance of this method on large spatial scales. The second dataset is an ensemble forecast dataset generated by Necker et al. [NGW⁺20], designed to analyze the sampling accuracy of spatio-temporal correlation patterns in convective-scale forecast ensembles. Due to the significant structural variability among the single-variable fields in each dataset and the lack of similarities between variable pairs, the original V2V transfer algorithm is not suitable for this application. In response to these observations, Farokhmanesh et al. [FHN⁺23a] propose a different V2V transfer strategy that considers the expressiveness of subsets of multiple variables rather than just pairs. To identify these subsets, a CNN-based model architecture is trained multiple times, each time removing different variable subsets from the input data. Through an ablation study, the predictive capabilities of the different models are assessed, depending on the variable subsets used as input sources. This process enables the networks to encode the inputs into a compact latent-space representation, capturing the relationships between the m input fields and the $n - m$ output fields; n is the total number of available variables. Given the high computational complexity of training, all $\binom{n}{m}$ possible networks to reconstruct $n - m$ fields from m given fields, a more computationally efficient strategy is proposed. This strategy involves iteratively removing the variables that are most difficult to predict from the remaining variables, thus avoiding redundant fields in the input. By this, the effectiveness of the proposed strategy in selecting the most representative variable subsets is demonstrated. In this method, the training behavior of the networks is monitored, using the convergence rates at early training stages (after a few epochs) as indicators of the difficulty of variable transfer (see Figure 3.1).

Beyond treating the networks as black-box models, the aim is to gain insight into the multi-variable relationships learned by the models. To achieve this, an adapted version of layer-wise relevance propagation (LRP) [BBM⁺15] is utilized. This method highlights the input areas and variables that are crucial to the reasoning

process of the models. This approach leads to identifying feature patterns in multi-variable space, such as recurring variable combinations that are important for achieving high accuracy, and to analyzing their correlation with specific weather conditions. (Paper A, [FHN⁺23a])

- Neural dependence fields (NDFs), a novel approach for compactly representing the dependency structures within large multi-variable ensembles, is introduced. The interpreting fields of two-point dependence measures are proposed in 3D ensembles as scalar fields, R , over the domain of position pairs in 3D space. Formally, this is expressed as $R : \Omega \times \Omega \rightarrow \mathbb{R}$, for $\Omega \subset \mathbb{R}^3$. Inspired by recent advances in neural scene representations and multi-dimensional tensor decomposition, a neural network architecture is designed that leverages self-similarity in dependency fields to learn a compact representation. This architecture also facilitates rapid sampling from the neural representation, allowing to bypass of the need to store the entire ensemble in the memory. As a result, significantly sped-up computation of dependency estimates is enabled, particularly for complex nonlinear dependence measures like MI. In the large ensemble simulation dataset considered in this study, the network can reconstruct dependencies between an arbitrary reference point and all other points in the domain in approximately 9 milliseconds. This efficiency makes it feasible to integrate the network into an interactive volume rendering pipeline, which supports instantaneous visualization and comparison of within- and between-variable dependency fields.

Contributions of this work can be summarized as the development of a compact neural network architecture for learning statistical point-to-point dependencies in large ensemble fields, integration of neural network-based dependency reconstruction into direct volume rendering, facilitating interactive visual exploration of dependencies within the 3D domain, and demonstration of interactive dependency analysis for a substantial meteorological multi-variable 3D ensemble field. The proposed method is versatile concerning the choice of dependency measure, supporting both linear measures, such as Pearson correlation, and computationally intensive nonlinear measures, such as MI. (Paper B, [FHN⁺23b])

- In large time series analysis, estimating Kraskov MI [KSG04] for all sliding windows within a selected time series and their corresponding windows across other time series is computationally expensive and time-consuming. This computation time further increases when incorporating dependencies between lagged time series. For each window in the selected time series, MI values must be computed for both positively lagged and negatively lagged windows in all other time series, significantly extending the overall processing time. To overcome this computational bottleneck, the use of neural networks for MI-based time series analysis, TNMIFs, is proposed. (Paper C, [FNW24]) Inspired by the research in Paper B [FHN⁺23b], in this study, it is demonstrated that a similar approach is applicable to time series data by focusing on estimating the MI between random variables represented by windows in different time series. This work proposes interpreting the field of two-window MI

measures in time series as scalar fields $\mathbb{R}$ over the domain of time series pairs. The network infers, in a single pass, the MI between the data in a selected time window and the corresponding windows in all other time series. This network inference is executed in less than 44 milliseconds, enabling users to interactively select a time series and instantly identify all time series with consistently high linear or nonlinear dependence over time. To enhance the analysis of dependencies between lagged time series, the network is trained to reconstruct MI for both positively and negatively lagged time windows. This capability allows for a comprehensive analysis of how the dependence between time series evolves over different time lags.

1.3 List of Publications

The methods described in this thesis have been originally proposed and published in the following peer-reviewed journal article and conference proceedings:

- **Paper A:** Fatemeh Farokhmanesh, Kevin Höhle, Tobias Necker, Martin Weissmann, Takemasa Miyoshi, and Rüdiger Westermann.
“Deep Learning-Based Parameter Transfer in Meteorological Data”.
In: *Artificial Intelligence for the Earth Systems 2023*. Volume 2, Issue 1, e220024.
DOI: 10.1175/AIES-D-22-0024.1.
- **Paper B:** Fatemeh Farokhmanesh, Kevin Höhle, Christoph Neuhauser, Tobias Necker, Martin Weissmann, Takemasa Miyoshi, and Rüdiger Westermann.
“Neural Fields for Interactive Visualization of Statistical Dependencies in 3D Simulation Ensembles”.
In: *Vision, Modeling, and Visualization, The Eurographics Association 2023*.
DOI: 10.2312/vmv.20231229.
- **Paper C:** Fatemeh Farokhmanesh, Christoph Neuhauser, and Rüdiger Westermann.
“A Compact and Efficient Neural Data Structure for Mutual Information Estimation in Large Timeseries”.
In: *Proceedings of the 36th International Conference on Scientific and Statistical Database Management 2024. Association for Computing Machinery, New York, NY, USA, 12 pages*.
DOI: 10.1145/3676288.3676295.

The list is in chronological order, according to the date of publication. The published version of each paper is incorporated at the end of this dissertation: Paper A can be found on Page 61, Paper B begins on Page 85, and Paper C is available at Page 93.

1.4 Outline

The main content of this dissertation is structured as follows. Chapter 2 discusses relevant fundamental concepts and provides an overview of related research. Chapter 3 describes the general proposed methodology for the dependency analysis of large datasets using deep learning. Considering the dependence measures as V2V transfer and statistical dependencies, it comprises two main parts. In Chapters 4 to 6, summaries of the publications included in this dissertation are given together with a list of individual contributions of the authors. Chapter 7 summarizes the results of this thesis and gives an outlook on future work. After Chapter 7, the individual publications and license or reuse permission forms are attached.

2 Fundamentals and Related Work

This chapter provides an extensive overview of the fundamental concepts and related work on large ensemble datasets, dependency analysis, deep learning, and their overlap.

2.1 Large Ensemble Datasets

Large datasets have become increasingly pivotal in advancing research across various domains, including meteorology, climate science, and time series analysis. This section introduces the fundamental concepts of large datasets, discusses the challenges associated with their analysis, and reviews related works that have leveraged such datasets in similar research contexts.

Large datasets are often characterized by their high volume, high dimensional variables, extensive temporal coverage, challenges, and opportunities for researchers [Mar15]. In scientific research, particularly in fields like meteorology and climatology, large datasets are instrumental in modeling and understanding complex phenomena [HTT⁺09]. These datasets typically encompass multi-dimensional arrays, such as girded weather data, ensemble predictions, and time series records, all of which require specialized handling and processing techniques.

The fundamental challenges associated with large datasets include the need for efficient data storage, retrieval, and processing. Traditional data processing tools are often inadequate for handling the scale of these datasets, leading to the adoption of distributed computing frameworks [DG08] and specialized data structures that facilitate scalable computation. Furthermore, the high dimensionality and temporal aspects of large datasets demand advanced analytical methods, including machine learning, deep learning, and statistical techniques, to uncover patterns and make predictions [Bis06]. In this thesis, three significant datasets are employed: a weather benchmark dataset, a convective-scale multi-variable ensemble (CSEns) dataset, and large time series datasets. The following describes each in detail.

2.1.1 Weather Benchmark Datasets

Weather benchmark datasets are crucial for the validation and comparison of weather forecasting models [BTB15]. These datasets typically include extensive historical records of atmospheric conditions collected from various sources such as satellite observations,

ground-based sensors, and climate models. A comprehensive weather benchmark dataset serves as a reference point for testing new forecasting techniques, ensuring that they can capture the inherent variability in weather patterns across different temporal and spatial scales.

A well-known example is the ERA5 dataset, which provides global, hourly estimates of a large number of atmospheric, oceanic, and land-surface variables [HBB⁺20]. Such datasets are indispensable for training and evaluating numerical weather prediction (NWP) models, which rely on vast amounts of data to predict weather conditions accurately [Ric00]. The use of large weather datasets is extensively documented in the literature, with studies highlighting their role in improving the accuracy and reliability of weather forecasts [RDS⁺20]. The WeatherBench dataset introduced by Rasp et al. [RDS⁺20] is particularly influential in the context of data-driven weather forecasting. WeatherBench provides a standardized benchmark for evaluating machine learning models against traditional NWP models, facilitating fair comparisons and driving innovation in forecasting techniques. By offering high-resolution, long-term historical weather data, WeatherBench allows researchers to assess the performance of various models across different forecast horizons and atmospheric variables. In paper A [FHN⁺23a], this dataset is utilized to validate the quality and applicability of the proposed deep learning-based V2V transfer method.

2.1.2 Convective-scale Ensemble Simulations

CSEns datasets are essential in capturing the uncertainty and variability intrinsic in weather systems, particularly at small scales where convection plays a significant role [CWK⁺12]. Convection involves vertical motion in the atmosphere, where warmer air rises and cooler air sinks. These datasets typically consist of multiple weather variables, such as temperature, humidity, wind speed, and pressure, simulated across different model runs (ensemble members) to represent a range of possible future states.

Such datasets are essential for probabilistic forecasting, where the goal is not just to predict a single outcome but to provide a range of possible scenarios, each with an associated probability [LP08]. The integration of multiple variables across ensemble members allows for a more comprehensive understanding of convective processes and their impacts. Researches leveraging CSEns datasets have demonstrated significant improvements in short-term weather forecasts, particularly for severe weather events such as thunderstorms, heavy rainfall, and tornadoes [GTPB11]. High spatial and temporal resolution is a key characteristic of CSEns datasets. They often resolve features down to a few kilometers or even less, capturing the fine-scale dynamics of severe weather. Notably, Necker et al. [NGW⁺20] introduced a large ensemble dataset comprising over 1,000 members, providing a rich source of data for exploring ensemble-based forecasting and model uncertainty. In papers A [FHN⁺23a] and Paper B [FHN⁺23b], this dataset is used to analyze within and between variable dependencies using novel deep learning methods.

2.1.3 Large Time Series

Time series datasets are present in various domains, such as finance, healthcare, and climate science [BJRL15]. A large time series dataset generally consists of a sequence of data points measured at successive time steps, where each sample can contain multiple variables observed simultaneously. The individual challenges presented by such datasets include the need to model temporal dependencies, handle missing data, and detect patterns or anomalies over extended periods [SSS00]. The literature on time series analysis is extensive, with numerous studies demonstrating the effectiveness of various techniques in forecasting, anomaly detection, and trend analysis [SSS00]. The increasing availability of large time series datasets is further directed research in this area, leading to new insights and methodologies that can handle the complexities of time-dependent data [HS97].

Several large time series datasets are proposed for advancing time series analysis. One of the most notable examples is the M3 competition dataset, introduced by Makridakis [MH00]. This dataset contains 3003 time series from diverse fields, including micro, macro, finance, industry, and demographic data, and it becomes a widely used benchmark for evaluating forecasting models. Following this, the M4 competition dataset, introduced, contains 100,000 time series from fields such as finance, macroeconomics, and industry [MSA18]. The M4 dataset, one of the largest publicly available, becomes a crucial benchmark for evaluating the performance of statistical and machine learning models in time series forecasting. Another important resource is the UCR time series classification archive, one of the largest repositories of univariate time series data for classification tasks. It includes over 128 datasets from a variety of sources, including sensor data, motion data, and image outline classification tasks. This archive is utilized for novel deep learning architectures experiments designed for time series tasks [DBK⁺19]. Additionally, the UCI electricity demand forecasting dataset, which contains electricity consumption data from a Portuguese power grid collected over several years, shows the use of large time series datasets in energy forecasting [CFD17]. Finally, the Yahoo Webscope anomaly detection dataset, which includes both synthetic and real-world time series data, is designed for evaluating anomaly detection algorithms [LAF15].

The primary objectives in time series analysis often include time series classification and measuring the similarity between time series or their subsequences over shorter time intervals. Middlehurst et al. [MSB23] provide a comprehensive review of various methods for time series analysis, including classification and feature extraction. Additionally, Fulcher et al. [FJ17], and Lubba et al. [LSK⁺19] introduce two influential toolboxes: the Highly comparative time series analysis (hctsa) and the canonical time series characteristics (catch22) toolboxes. These tools are designed to extract a broad spectrum of features, encompassing basic statistics, linear correlations, and entropy, thus covering a wide range of time series concepts. In the realm of deep learning, Fawaz et al. [IFLF⁺20] discuss the significant potential for improving time series classification by leveraging neural network architectures, citing the influence of AlexNet on this domain.

Their earlier work [IFFW⁺19] highlights ResNet (as developed by Wang et al. [WYO17]) as one of the most accurate deep learning models for time series classification up to that point.

Performing statistical dependence analysis and visualization of time series at the level of subsequences can become computationally demanding for large datasets [ANB⁺20]. Traditional approaches, such as those by Mueen et al. [MNL10] and Pietersz et al. [PG04], focus on analyzing entire time series, which limits their ability to uncover related subsequences within the data. Li et al. [LUG13] attempt to find correlations among subsequences but restrict their scope to synchronous subsequences, potentially skipping temporally misaligned patterns. To address these challenges, Ahsan et al. [ANB⁺20] propose an efficient solution for conducting correlation-based analytics on time series data, which includes tasks like retrieving the best match, grouping correlated subsequences, and self-correlation. Köthür et al. [KWS⁺15] expand upon windowed cross-correlation by introducing a visual analytic method for comparing two time series based on statistical dependence. However, this approach is limited in its scalability, mainly when dealing with more than two time series. Evers et al. [EBL23] present a novel approach for analyzing regional time series correlations within multi-field climate ensembles, focusing on selected segments extracted from these ensembles. This method offers new insights but is similarly focused on specific segments rather than the entire dataset. Paper C [FNW24] proposes a method to analyze the dependencies between a time window in a picked time series and synchronized or lagged time windows in all the other time series using large time series datasets from the UCR time series classification archive [DBK⁺19].

2.2 Dependency Analysis

Dependency analysis in this thesis refers to statistical dependency analysis within and between different variables and time series, as well as V2V transfer, an example of dependence measures in this dissertation inferring a certain subset of variables from another subset. In this section, fundamentals and the related works of the statistical dependence measures used in this thesis and the V2V transfer are introduced. Since the statistical dependence measures used in this thesis are the Pearson correlation coefficient and the MI, the explanation of methods such as Spearman’s rank correlation coefficient [Zar05] and Kendall rank correlation coefficient [Ken38] are excluded.

2.2.1 Statistical Dependencies

The term correlation traditionally refers to the linear relationship between two random variables, with Pearson correlation coefficient [Sti89] being the most commonly used metric for quantifying this type of relationship. However, the broader concept of dependence extends beyond just linear relationships and includes any form of association between variables, whether linear or nonlinear. Dependence can be characterized by more general,

information-theoretic measures such as MI [CT⁺91] (see Section 2.2.1.2). In this work, the focus is on both the Pearson correlation coefficient and MI, selecting these two from a wide spectrum of potential dependence measures available in the literature. Pearson correlation coefficient, being relatively simple and computationally inexpensive, captures linear dependencies effectively, while MI, though more computationally demanding, is a powerful tool for detecting complex, nonlinear relationships between variables. These two metrics thus represent opposite ends of the computational spectrum and allow us to capture a wide range of dependencies in our analysis [BMLC19].

2.2.1.1 Pearson Correlation Coefficient

The Pearson correlation coefficient, often referred to as Pearson’s r [Sti89, CHC⁺09], quantifies the linear correlation between pairs of random variables. It is widely used in data visualization to assess and explore relationships between variables.

Given a set of paired random samples, $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^N$, the Pearson correlation coefficient is defined as

$$r = \frac{\text{cov}(\mathbf{x}_1, \mathbf{x}_2)}{\sqrt{\text{var}(\mathbf{x}_1) \text{var}(\mathbf{x}_2)}}, \quad (2.1)$$

wherein $\text{var}(\cdot)$ and $\text{cov}(\cdot)$ denote sample variance and covariance of the respective random samples. Pearson’s r ranges from -1 to 1. The value of +1 indicates the highest positive correlation, -1 shows the highest negative correlation, and 0 indicates no linear correlation. The coefficient is straightforward to interpret and measures both the strength and direction of the relationship between variables. However, it is essential to use this method carefully when interpreting Pearson’s r in cases of nonlinear relationships or when outliers are present, as these factors can significantly distort the correlation coefficient.

One of the challenging topics is dependency visualization utilized for the interactive analysis of dependence measures. In this thesis, dependency visualization is applied within interactive workflows. Various techniques for visualizing correlations are proposed, including clustering methods [PW12, LWS18, EHL21], correlation matrices [CWMW11, EHL21], diagram views [STS06, BDSW13, ZMZM14, LS16], and feature-based approaches such as analyzing dependencies in flow fields using particle trajectories [BMLC19]. Correlation sub-sampling techniques [GW10, CWMW11] help identify prominent features within correlation fields for detailed analysis. These approaches complement the work in Paper B [FHN⁺23b], where volume rendering is utilized to demonstrate the network-based reconstruction for correlation visualization, focusing on identifying significant structures in correlation fields or developing effective visual encodings. The method proposed in Paper B [FHN⁺23b] integrates seamlessly with these techniques, reducing memory and computational demands when accessing dependencies in large 3D fields.

2.2.1.2 Mutual Information

MI is a widely used metric in machine learning, statistics, and information theory for measuring the similarity or correlation between pairs of random variables [CT⁺91]. Mathematically, MI is expressed as

$$I(X_1; X_2) = H(X_1) - H(X_1|X_2) = H(X_2) - H(X_2|X_1), \quad (2.2)$$

where X_1 and X_2 are random variables, $I(X_1; X_2)$ is the MI of X_1 and X_2 , $H(X_1)$ is the entropy of X_1 , and $H(X_1|X_2)$ is the conditional entropy of X_1 given X_2 . Notably, MI is symmetric under the exchange of variables, X_1 and X_2 .

Unlike linear correlation, MI can capture non-linear and non-monotonic dependencies that are not evident through covariance analysis. However, estimating MI from finite samples $\mathbf{x}_1$ and $\mathbf{x}_2$ of random variables X_1 and X_2 , i.e. computing $I(\mathbf{x}_1, \mathbf{x}_2)$ —is computationally expensive. Existing algorithms often struggle to scale effectively with large sample sizes [KSG04, MRL95]. While more recent copula-based and neural-network-based variational MI estimators offer improved performance in high-dimensional data spaces, they still present significant computational challenges [ZD11, BBR⁺18].

In this thesis, the ground truth MI fields are computed using the nearest-neighbor-based estimator developed by Kraskov et al. [KSG04], implemented in parallel. However, estimating MI fields for visualization purposes exceeds the time constraints typically required for interactive data analysis.

Mutual information neural estimation (MINE) is a recent method that leverages neural networks to estimate MI [BBR⁺18]. It does so by training a neural network to learn a lower bound on MI, which provides an estimate. MINE has demonstrated superior performance compared to traditional MI estimation methods on benchmark datasets. However, in this thesis scenario, inefficiencies arise because MINE operates on a member (or time windows)-wise basis and requires all ensemble members (or time windows) to be loaded into memory simultaneously, which can be impractical for large-scale data.

2.2.2 Variable-to-Variable Transfer Dependence Measure

In addition to statistical dependence measures within and between different variables and time series, this thesis analyzes the V2V as a method of identifying dependencies between different variables. The V2V approach is introduced by Han et al. [HZX⁺20] for multi-variable data, where networks are trained to infer specific variables from others, thus avoiding the need to store these variables. Conceptually, this work builds upon the notion of information transfer between scalar fields [WYG⁺11a] to derive transferable variable pairs. The overall goal of the V2V approach is to identify variables within a multi-variable dataset that can be reconstructed using other variables from the same dataset. [HZX⁺20] outline the V2V process in three conceptually distinct stages: feature learning, translation graph construction, and variable translation.

In the first stage, a CNN architecture, such as U-Net [RFB15], is trained in an autoencoder-like setting. The model is trained to encode and reconstruct snapshots of variable fields, and the same model is shared among all variables. This sharing ensures that the internal hidden parameters of the model—the latent-space representation—capture the similarities and dissimilarities between different variables. After training, all variable snapshots are mapped into this latent-space, where clusters of similar variables can be identified. Han et al. [HZX⁺20] suggest visualizing these features using a non-linear dimension reduction algorithm, specifically stochastic neighbor embedding (t-SNE) [vdMH08], to facilitate the detection of these clusters. variables that fall into the same cluster are considered a transferable variable group. In the second stage, the Kullback-Leibler (KL) divergence is used to estimate a measure of "transferable difficulty" between pairs of variables within the same transferable group. A pair of variables are considered for transfer if the Euclidean distance between their respective latent-space representations falls below a predefined threshold. variables from different transferable groups are not considered for transfer. A directed transfer graph is then constructed by chaining transferable pairs according to a minimum discrepancy criterion. In the final stage, CNNs are trained to learn the transfer mapping according to the translation graph. The directed graph, with nodes representing variable fields and edges indicating transferability from a source to a target variable, guides this process.

While V2V streamlines the identification of transferable variables by focusing on pairs of similar variables based on a distance threshold and t-distributed t-SNE visual analysis, it has some limitations. One significant shortcoming is that V2V does not evaluate the actual predictability of one variable based on another. Instead, it relies on an empirically determined approximate criterion, which might miss potentially valuable relationships if the threshold is not optimally set. Additionally, V2V may encounter ambiguity when determining source and target fields. The translation graph, which indicates transferability, can contain cycles. This occurs because the KL divergence, used to measure transferable difficulty, is not a metric and does not satisfy the triangle inequality. When only pairwise transferabilities are considered, the result might be that all variables within a set of similar variables are selected as both sources and targets for each other. This ambiguity can complicate the interpretation of the transferability and limit the expressiveness and reproducibility of the identified variable relationships.

This thesis evaluates the applicability of the V2V transfer method in the context of earth-system-related data. This assessment identifies several shortcomings of the original V2V methodology, particularly in its application to complex meteorological datasets. In response to these findings, an improved approach is proposed. This approach involves training a single CNN on extensive meteorological archives to learn generalized relationships between subsets of variables. This new method focuses on handling variable fields that exhibit vastly different characteristics and variations across two-dimensional (2D) space and time. In Paper A [FHN⁺23a], the effectiveness of the proposed approach is demonstrated using two exemplary datasets, each representing different modalities of meteorological data. The first global reanalysis dataset is sourced from the WeatherBench benchmark suite [RDS⁺20]. This dataset allowed for the study of the method's perfor-

mance on data covering large spatial scales, which is critical for understanding global weather patterns. The second dataset to evaluate the proposed method is an ensemble forecast dataset generated by [NGW⁺20]. This dataset is specifically designed to study the sampling accuracy of spatio-temporal correlation patterns within convective-scale forecast ensembles. The application of the improved CNN-based approach to this dataset demonstrated its potential in handling complex meteorological data, providing a more robust and scalable solution for analyzing relationships between different variables in earth-system-related data.

In the following, further available approaches to analyze relationships between different variables are reviewed. These methods include point-wise correlation coefficients [GHGM06, QCX⁺07, SWMW09, CWMW11] and gradient similarity measures [STS06]. Wang et al. [WYG⁺11b] investigated the information flow between variable pairs using transfer entropy to examine causal relationships. Biswas et al. [BDSW13] classified variables based on surprise and predictability derived from information theory, utilizing a graph-based representation for variable exploration. Liu et al. [LS16] developed a probabilistic association graph, incorporating the concepts of informativeness and uniqueness to reveal hidden associations among different variables. Tao et al. [TIW⁺18] focused on isosurface similarities across time and variable dimensions in time-varying multi-variable data, creating a matrix of isosurface similarity maps for visual exploration.

Furthermore, there are super-resolution and downscaling techniques that reconstruct high-resolution variable fields from corresponding low-resolution versions. Unlike analyzing the information transfer between different variables, these techniques transfer information between representations of the same variable at different spatial resolutions. Some of these methods can be applied for data compression by sub-sampling the variable fields and then reconstructing the initial fields from these sub-sampled versions. For example, [ROCN18] proposed a supervised convolutional neural network that interpolates low-resolution weather data into high-resolution outputs. Similarly, [PLPD18] introduced deep learning-based enhancement in Landsat super-resolution, while [CKS⁺20] developed a method that converts low-resolution climate data into high-resolution forecasts using Laplacian pyramid super-resolution networks. In contrast, downscaling approaches aim to predict additional high-resolution details from low-resolution variable fields without assuming prior knowledge of the original high-resolution data. For instance, [HKHW20] and [SGB21] trained models on a small set of paired low- and high-resolution simulations to avoid generating expensive high-resolution simulations at inference time. Although these techniques establish relationships between low- and high-resolution fields, their primary motivation is not data compression.

Super-resolution of scientific data is also explored from the perspective of scientific visualization, particularly since high-resolution simulations in meteorology complicate data analysis. Early works in this area demonstrate the capability of neural networks to upscale a low-resolution version of the data back to its original high-resolution form. This upscaling is performed in the spatial domain [ZHW⁺17, HW20, GYH⁺20], the temporal domain [HW19], and the spatio-temporal domain [HZCW21]. These works aim to avoid

storing high-resolution datasets, thereby reducing bandwidth and memory requirements. For example, by training networks to infer the full image from a low-resolution image of an iso-surface, [WCTW19] demonstrated improved rendering frame rates. In more recent work, [WITW20] trained a convolutional neural network to adaptively place image samples and reconstruct the full image from the generated unstructured set of samples.

In scenarios with severe I/O limitations, [STK19] introduced an in-situ approach for visualizing and post-processing high-resolution meteorological data. In-situ approaches process data immediately as it is produced by the simulation, without the need for storage resources. Röber and Engels [RE19] analyzed in-situ data processing approaches in climate science. Helbig et al. [HBB⁺15] proposed a visualization workflow where the first stage involves a data abstraction layer that downsamples the data spatially and temporally. [TVJ⁺17] introduced an image compression method using a recurrent neural network by saving the compressed latent-space produced by the network instead of the high-resolution data.

2.3 Deep Learning

This section provides a concise overview of the key deep learning techniques employed in this dissertation. Specifically, multilayer perceptrons (MLPs) and various types of CNNs are utilized as the core architectures in the experiments. For a more comprehensive review of the history of development and theoretical concepts of neural networks, the work by Bishop [Bis06] is referred to.

In its simplest form, a neural network is represented as a function $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$ that is trained to approximate an arbitrary target function $f_t : \Omega \subset \mathbb{R}^n \rightarrow \mathbb{R}^m$, where n represents the input dimensions and m the output dimensions. The parameters $\theta \in \mathbb{R}^k$ are optimized during this process. The input space Ω is a subset of $\mathbb{R}^n$, which contains the valid inputs for the problem being solved. The primary goal of neural network training is to find the set of k parameters that result in the best match between f_θ and f_t .

For this purpose, a set of N training pairs $\{(x_i, y_i)\}$, with $x_i \in \Omega$ and $y_i = f_t(x_i)$, is given, where $i = 1, \dots, N$. This set is referred to as the training set in supervised learning. The training process is then formulated as the minimization problem:

$$\theta = \operatorname{argmin}_{\theta} \frac{1}{N} \sum_{i=1}^N L(f_\theta(x_i), y_i), \quad (2.3)$$

where the function $L : \mathbb{R}^m \times \mathbb{R}^m \rightarrow \mathbb{R}$ is called the loss function. It measures the distance between the prediction of the network and the actual target values. After training, the network is expected to have learned a compact representation of the possibly complex target function f_t in the latent-space of k parameters. The goal is for the network to generalize well, meaning it should provide sensible outputs for new inputs $x \in \Omega$ that are

not part of the training set. This generalization ability describes how well the network can infer unseen data.

Common choices for the loss function include the mean squared error (MSE), defined as:

$$L_{\text{MSE}}(x, y) = \|x - y\|_2^2, \quad (2.4)$$

and the mean absolute error (MAE), known as L_1 -loss, defined as:

$$L_{\text{MAE}}(x, y) = \|x - y\|_1. \quad (2.5)$$

In addition to these basic loss functions, many others are developed for specific tasks. For instance, a smoothed version of the L_1 -loss is proposed to improve training stability [Gir15], and cross-entropy loss is frequently applied in classification tasks [RFB15]. If the output space represents images, specialized loss functions like the structural similarity index measure (SSIM) [WBSS04] or the learned perceptual image patch similarity (LPIPS) [ZIE⁺18] can be used.

The networks are usually trained using gradient-based optimization techniques. Gradients, denoted as $\hat{\theta} := \frac{\partial L}{\partial \theta}$, are computed via backpropagation. Gradient descent is then applied:

$$\theta_{i+1} = \theta_i - \alpha \hat{\theta}_i, \quad (2.6)$$

where α is the learning rate. In practical applications, it is often impossible to compute the gradients for the entire training set simultaneously due to memory constraints. Instead, batches of training data are processed sequentially, leading to the use of stochastic gradient descent (SGD). To further enhance optimization stability, advanced methods like SGD with momentum [SMDH13], Adam [KB14], Adagrad [DHS11], Adadelata [Zei12], or second-order approximations such as L-BFGS [NW99, BHNS16] are often employed.

2.3.1 Multilayer Perceptrons

A MLP is a class of feed-forward artificial neural networks that consist of multiple layers of neurons, where each neuron functions as a perceptron. It is a fundamental model in machine learning used to approximate functions that map input data to output predictions. The MLP operates by learning the weights of the connections between the neurons across different layers through a process known as training. At its core, a MLP consists of three types of layers: an input layer, one or more hidden layers, and an output layer. The input layer receives the input vector $x^1 = [x_1^1, x_2^1, \dots, x_n^1]$, where x_i^1 represents a feature of the input data. Each neuron in the subsequent layers performs a weighted sum of the inputs it receives, adds a bias term, and then applies an activation function

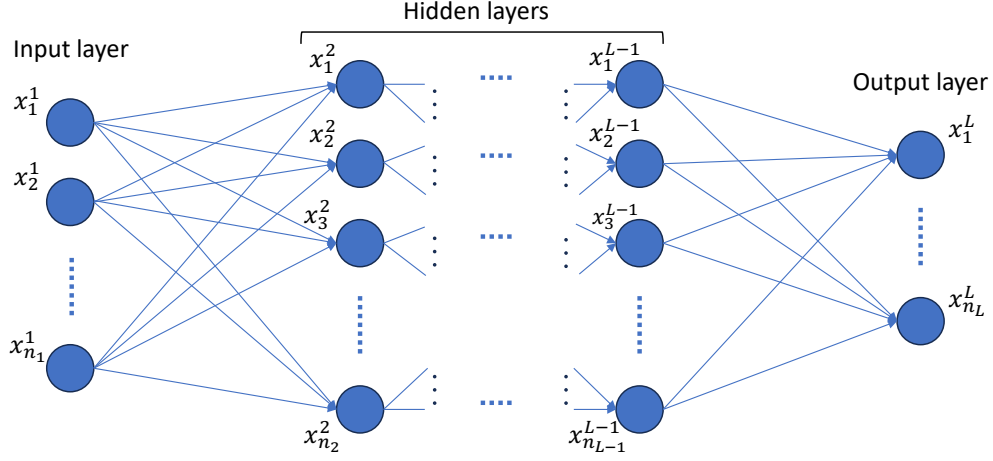


Figure 2.1: A multilayer perceptron (MLP) architecture, including the input layer, multiple hidden layers, and the output layer. Links indicate the weight multiplication.

to this sum. Mathematically, the output of a single neuron j in layer l is expressed as:

$$z_j^l = \sum_i w_{ji}^l x_i^{l-1} + b_j^l, \quad (2.7)$$

where:

- w_{ji}^l is the weight of the connection from neuron i in layer $l-1$ to neuron j in layer l .
- x_i^{l-1} is the output of neuron i from the previous layer.
- b_j^l is the bias term for neuron j in layer l .

The activation a_j^l of the neuron is then obtained by applying a to z_j^l :

$$x_j^l = a(z_j^l). \quad (2.8)$$

The structure of the corresponding MLP network (without considering bias) is illustrated in Figure 2.1, where the links represent the multiplication of the input values on the left by the corresponding weight values. When the bias and activation function are omitted, the inner sum can be expressed as a matrix-vector multiplication $\hat{x}^l = w^{l-1} x^{l-1}$. The bias term can be incorporated into the weight matrix by appending a value of 1 to the input vectors x^l . An important factor when developing MLPs is the choice of the activation function $a(\cdot)$. The three fundamental activation functions that are most commonly used in the early stages of deep learning are the Sigmoid, hyperbolic tangent (tanh), and rectified linear unit (ReLU) functions [JKRL09a, NH10]. Another important activation function used in this dissertation is SnakeAlt, proposed by Weiss et al. [WHW22]. It is

an improved version of the Snake activation function proposed by Ziyin et al. [ZHU20] that leads to higher reconstruction quality. These functions are defined as follows:

- Sigmoid: $a(x) := \frac{1}{1+\exp(-x)}$
- tanh: $a(x) := \tanh(x)$
- ReLU: $a(x) := \max(0, x)$
- Snake: $a(x) := x + \sin^2(x)$
- SnakeAlt: $a(x) := \frac{1}{2}x + \sin^2(x)$

In Figure 2.2, the two activation functions used in this dissertation, ReLU and SnakeAlt, are plotted.

In recent years, scene representation neural networks (SRNs) emerged as an application of MLP networks. SRNs are developed to derive compact representations of 3D models and scenes. They are initially proposed for mapping 2D or 3D position coordinates. Early examples include encoding surface models as implicit functions or occupancy maps using fully connected neural networks [MON⁺19, CZ19, PFS⁺19]. Over time, SRNs are evolved to encode a broader range of volumetric scene information, such as neural radiance fields [TSM⁺20, MST⁺21], and have come to be known as neural fields [XTS⁺22]. A neural field is a neural network that learns to parameterize spatio-temporal, multi-dimensional physical fields based on spatial coordinates. During inference, these coordinates are transformed into latent-space representations, which are then decoded to obtain the corresponding physical quantities.

Recent advancements in neural fields demonstrate that domain-specific input feature encodings can greatly enhance reconstruction quality. For example, Chabra et al. [CLI⁺20] introduced a method that organizes trainable parameters within a grid of latent features, allowing for more direct learning of spatial variations. This concept is further refined with adaptive data structures [MLL⁺21], fixed multi-resolution grids [TLY⁺21], and multi-resolution spatial hashing [MESK22, MRNK21a], which facilitate multi-scale learning of spatial feature maps. Comprehensive reviews of literature related to SRNs and neural scene representations can be found in [HSB⁺20, TFT⁺20].

In the field of scientific data visualization, Lu et al. [LJLB21] introduced SRNs for volumetric data compression, a process that was later accelerated and refined by Weiss et al. [WHW22] through the integration of trainable feature representations and efficient GPU implementation. Additionally, Höhle et al. [HWW22] employed neural fields to compress ensemble data by sharing model parameters across different ensemble members. Both studies illustrate the effective combination of volume rendering and network inference, which is also utilized in this research. In this dissertation, Papers B and C make use of neural fields. In Paper B, NDFs is introduced, which is a compact neural network architecture to learn statistical point-to-point dependencies in a large 3D ensemble dataset and enables interactive dependencies reconstruction. In Paper C, TNMIFs is proposed, where instant reconstruction of MI is enabled between time

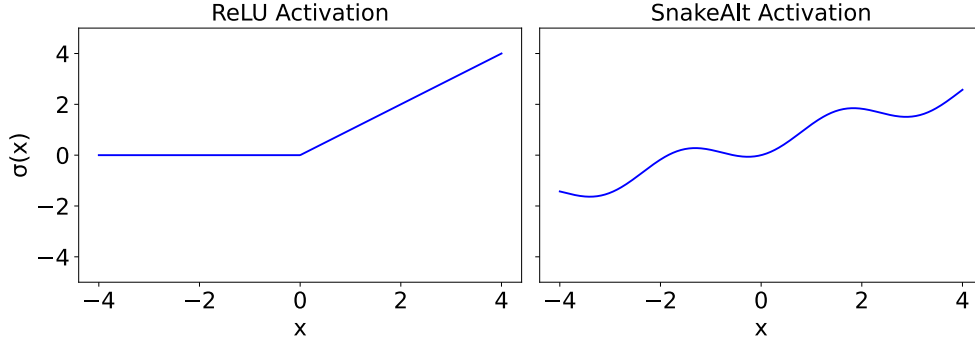


Figure 2.2: Plot of activation functions ReLU [JKRL09a, NH10] and SnakeAlt [WHW22].

windows of one time series and corresponding time windows or lagged time windows of all the other time series in a large time series dataset. This paper proposes to interpret the field of two-window MI measures in time series as scalar fields $\mathbb{R}$ over the domain of time series pairs. The network infers in one pass, for every window in a selected time series, the MI between the data in this window and the corresponding windows in all other time series, as well as shifted time windows with a specific shift value, for instance, ± 64 .

2.3.2 Convolutional Neural Networks

In 1980, Kunihiko Fukushima introduced the "Neocognitron" [Fuk80], a neural network model that aimed to emulate the visual cortex by enabling the recognition of patterns that remain stable despite shifts in position. This model organized neurons, referred to as "cells" by Fukushima, into a 2D grid, with each neuron following the same spatial input pattern. Building on this, LeCun et al. [LBD⁺89, LCJB⁺90] extended the Neocognitron to improve its performance for handwritten digit recognition tasks. Weng et al. [WAH93] introduced the concept of max pooling, further advancing the capabilities of neural networks for pattern recognition. A CNN consists of convolutional layers, which apply a series of filters or kernels across the input data to produce feature maps. These filters slide over the input, performing a convolution operation that emphasizes relevant patterns, such as edges, textures, or more complex structures in the data. Mathematically, the output of a convolutional layer can be expressed as:

$$Feature\ map = \sum_{m=1}^M \sum_{n=1}^N Input(i+m-1, j+n-1) \cdot Kernel(m, n). \quad (2.9)$$

The input represents the original input data, such as an image. If the input is an image, then $Input(i, j)$ represents the pixel value at position (i, j) in the image. In general, for a 2D input, $Input(i, j)$ is the value at row i and column j . the kernel is a small

matrix of weights, denoted as $Kernel(m, n)$, where m and n are the indices within the kernel matrix. For example, a 3×3 kernel has $M = 3$ and $N = 3$, so m and n range from 1 to 3. Common kernel sizes are 3×3 , 5×5 , or 7×7 , [LCJB⁺90, WAH93], but larger kernels are used by Krizhevsky et al. [KSH12], a kernel of size 11×11 in the first layer of their network architecture. The feature map is the output of the convolution operation, which represents the responses of the kernel as it slides over the input. Since cross-correlation offers a more intuitive interpretation, deep learning frameworks such as PyTorch [PGM⁺19] and TensorFlow [ABC⁺16] typically implement the cross-correlation operation instead of convolution. Cross-correlation is mathematically equivalent to convolution when the kernel is flipped. Following the convolutional layers, CNNs typically use activation functions like ReLU to introduce non-linearity and pooling layers, such as max pooling [WAH93], to downsample the spatial dimensions, reducing computational complexity and helping the network become invariant to small translations of the input.

One application area of CNNs is image classification tasks [JKRL09b, KSH12, RFB15, LGRN09]. Using large standardized datasets, such as the ImageNet database [RDS⁺15], facilitates the training of extensive network architectures and enables reproducible results and comparisons among different models. To categorize images into 1000 classes in the ImageNet database, reducing their resolution, for instance, from 224×224 pixels to a smaller size is necessary. One approach to achieve this resolution reduction is through pooling operations. Specifically, max pooling [WAH93] partitions the image into blocks of size $h \times w$ and substitutes each block with its maximum value. This operation reduces the spatial dimensions of the image from $hH \times wW$ to $H \times W$. Another technique for resolution reduction is strided convolutions [Goo16], where the stride determines the number of pixels skipped between input image locations when producing the output image. For example, Krizhevsky et al. [KSH12] employ both max pooling and strided convolutions within their architecture. For further reading about the CNNs, the article by Dumoulin and Visin [DV16] provides an extended description of the arithmetic behind the various convolution operations, how to handle the border, and visual explanations. In Paper A, The U-Net [RFB15] and ResNet [HZRS16a] architecture are utilized. The following explains the fundamentals of these two CNN architectures.

2.3.2.1 U-Net

U-Net is a CNN architecture developed for biomedical image segmentation, introduced by Ronneberger et al. [RFB15]. It is designed to precisely predict pixel-level labels, making it ideal for tasks such as tumor segmentation in medical images, where accurate localization is crucial. U-Net architecture consists of an encoder-decoder structure with skip connections that allow for better reconstruction of spatial information. Encoder, contracting path, is similar to a traditional CNN, this part captures high-level features while reducing the spatial dimensions of the input image through convolutional and max-pooling operations. Decoder, expanding path, upsamples the feature maps to reconstruct the original spatial resolution, allowing pixel-wise classification. The uniqueness of U-Net

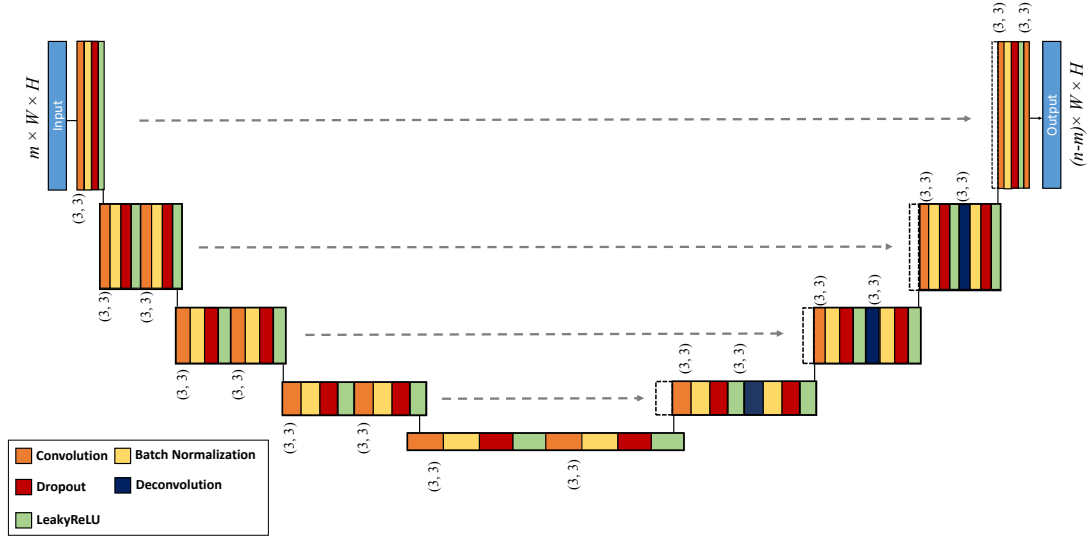


Figure 2.3: A schematic of U-Net Architecture is presented. The contracting branch consists of three convolutional blocks, each composed of two convolution layers followed by batch normalization, dropout, and a LeakyReLU activation function. The expansive branch includes three deconvolution blocks, each made up of a convolution and a deconvolution layer, with batch normalization, dropout, and LeakyReLU activation functions applied after each layer. The input block consists of a single convolution layer with 64 channels and a kernel size of (3, 3), followed by batch normalization, dropout, and LeakyReLU activation. Figure from Paper A, Farokhmanesh et al. [FHN⁺23a], © American Meteorological Society. Used with permission.

comes from its skip connections, which copy feature maps from the encoder to the corresponding layers in the decoder. These skip connections help preserve fine-grained spatial information, enabling more precise localization of features. These connections allow the network to transfer high-resolution features directly from the contracting path to the expanding path through the concatenation of the encoder feature map with the upsampled decoder map, helping the model retain spatial information that might otherwise be lost. This is particularly important for precise segmentation tasks, where boundary details are crucial. In Paper A [FHN⁺23a], the U-Net is one of the architectures used in this paper to reconstruct a subset of variables from the given variables (see Figure 2.3).

Further similar architectures to U-Net, including SegNet proposed by Badrinarayanan et al. [BKC17] which is an encoder-decoder architecture for segmentation tasks, and V-Net is another important architecture that is developed specifically for volumetric medical image segmentation, 3D images. V-Net extends the ideas of U-Net to 3D data [MNA16].

2.3.2.2 ResNet

ResNet, or residual network, is introduced by He et al. [HZRS16a, HZRS16b]. It addresses the problem of training very deep networks by introducing the concept of residual learning, which allows networks to learn residuals or differences between input and output rather than directly trying to learn the output. As neural networks grow deeper, they tend to face problems such as vanishing/exploding gradients, where the gradients become very small or grow too large during backpropagation, making it difficult to update weights effectively, and degradation problems; as the network depth increases, accuracy can start to degrade due to difficulties in optimization. Deeper networks should ideally perform better than shallower ones, but they often do not due to this degradation. The key innovation of ResNet is to solve these problems through residual learning. The core idea of ResNet is that instead of learning the desired mapping $H(x)$, the network learns the residual function $F(x) = H(x) - x$, which is easier to optimize. The hypothesis is that it is often easier for the network to learn the residual, or change, from the input to the output than to learn the output directly. The original input x is added back to the output of the residual function, allowing the network to learn this residual more efficiently.

A ResNet block can be described mathematically as:

$$y = \mathcal{F}(x, Wi) + x, \quad (2.10)$$

where x is the input, $\mathcal{F}(x, Wi)$ represents the function learned by the convolutional layers within the block, and x is added to the output of this function, identity mapping. This simple yet powerful idea allows ResNet to maintain high accuracy even as the number of layers increases significantly. A typical ResNet block consists of:

- Two or more convolutional layers,
- Batch normalization after each convolutional layer to stabilize training,
- A ReLU activation function,
- Skip connections or identity connections that bypass one or more layers.

Residual learning enables the training of much deeper networks, such as ResNet-50, with 50 layers, ResNet-101, with 101 layers [HZRS16a], and even deeper variants, without suffering from the vanishing gradient problem. ResNet-50 and ResNet-101 are widely used in various applications, from image classification to object detection, and they have set new standards for performance in many benchmarks. The utilized ResNet for V2V transfer in Paper A is shown in Figure 2.4. The networks in Paper A [FHN⁺23a] are not considered as a black box, and the learned relationships between different variables are analyzed using a feature analysis method, which is explained in the next section, Section 2.3.3.

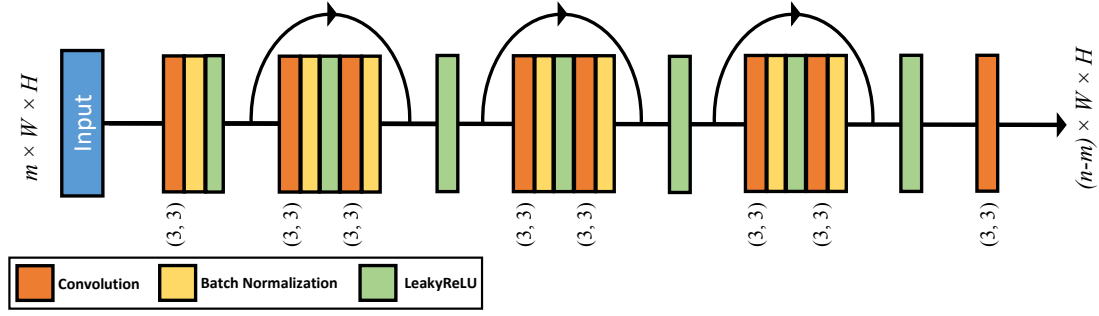


Figure 2.4: A schematic of ResNet architecture is provided. It consists of three residual blocks, each followed by a LeakyReLU activation function. The input layer processes the inputs through a convolution layer with 64 channels and a kernel size of (3, 3), followed by batch normalization and a LeakyReLU activation function. The network is composed of three residual blocks and an output convolution layer. Figure from Paper A, Farokhmanesh et al. [FHN⁺23a], © **American Meteorological Society. Used with permission.**

2.3.3 Feature Analysis using Layer-wise Relevance Propagation

The network architectures used in Paper A [FHN⁺23a] are introduced in Sections 2.3.2.1 and 2.3.2.2. To gain insight into the information learned by these models, a modified version of layer-wise relevance propagation (LRP) by Bach et al. [BBM⁺15] is employed in Paper A. The following explains the fundamentals of LRP and related works utilized LRP. LRP is an explainable AI method specifically designed for deep neural networks. LRP focuses on explaining the predictions of a neural network by propagating the prediction of the model back through the layers to attribute importance to the input features. This method provides relevance scores for each input feature, which shows how much each feature contributes to the final prediction. The core idea behind LRP is to redistribute the output prediction backward through the network, assigning relevance to each neuron in a way that the sum of the relevance scores in each layer matches the output prediction. The relevance of each neuron is attributed to the neurons in the previous layer, all the way back to the input layer, ultimately showing how much each input feature contributed to the prediction of the model. LRP decomposes the prediction $f(x)$ for a given input x by propagating relevance scores R_i backward through the network. For a neuron i in layer l with output z_i , LRP distributes the relevance score R_i^l of that neuron to the neurons j in the previous layer $l - 1$ based on the contributions z_{ij} from neuron j in the previous layer:

$$R_j^{l-1} = \sum_i \frac{z_{ij}}{\sum_k z_{ik}} R_i^l \quad (2.11)$$

Where:

- z_{ij} is the contribution of neuron j in layer $l - 1$ to neuron i in layer l ,

- R_i^l is the relevance score of neuron i in layer l ,
- R_j^{l-1} is the relevance score of neuron j in layer $l - 1$.

The propagation continues back through each layer until it reaches the input layer, resulting in relevance scores for each input feature.

One important property of LRP is relevance conservation. The sum of the relevance scores in any layer is equal to the output prediction of the model:

$$\sum_j R_j^{l-1} = \sum_i R_i^l = f(x) \quad (2.12)$$

This property ensures that the total relevance attributed to the input features exactly matches the output score of the model, making the method well-suited for understanding individual predictions.

LRP uses different propagation rules depending on the type of layer being explained:

LRP-0 Rule (Basic Rule) This rule works for linear layers and ensures that relevance is redistributed proportionally to the contribution of each neuron.

$$R_j^{l-1} = \sum_i \frac{z_{ij}}{\sum_k z_{ik}} R_i^l \quad (2.13)$$

LRP- ϵ Rule Adds a small term ϵ to the denominator to prevent division by zero and to stabilize the propagation, especially for neurons with small activations:

$$R_j^{l-1} = \sum_i \frac{z_{ij}}{\sum_k z_{ik} + \epsilon} R_i^l \quad (2.14)$$

LRP- γ Rule A modification for ReLU layers, which adjusts the amount of relevance propagated by scaling positive contributions. This is used to focus more on positive activations in the network.

LRP- $\alpha\beta$ Rule This rule separates the positive and negative contributions of activations and propagates them differently:

$$R_j^{l-1} = \alpha \sum_{i, z_{ij} > 0} \frac{z_{ij}}{\sum_k z_{ik}} R_i^l - \beta \sum_{i, z_{ij} < 0} \frac{z_{ij}}{\sum_k z_{ik}} R_i^l \quad (2.15)$$

Here, $\alpha + \beta = 1$, where α controls the amount of positive relevance and β the amount of negative relevance being propagated. This is useful when the goal is to distinguish between positive and negative contributions to a prediction.

LRP is widely used in various domains that require an explanation of neural network decisions, especially for image, text, and scientific data. Initially LRP applied in image classification tasks to interpret CNN decisions [MLB⁺17]. In the medical field, Binder et al. [BML⁺16] used LRP to explain brain tumor classification models. In natural language processing, Arras et al. [AMMS17] used LRP to analyze sentiment classification models by identifying important words, and Tjoa et al. [TG20] employed LRP in financial forecasting, enhancing transparency in stock market predictions. Extensions of LRP, such as improved rules for handling complex models [MBL⁺19], have further strengthened its use in intricate domains like climate science and physics, making LRP a key tool for enhancing the interpretability of AI systems across multiple fields.

3 Dependency Analysis and Reconstruction using Deep Neural Networks

This dissertation addresses the dependency analysis challenges inherent in large ensemble datasets such as large weather benchmarks, large ensemble simulations, and large time series datasets through the use of advanced deep learning techniques. The following provides an overview of the methodology employed across the three papers included in this dissertation. First, the use of deep neural networks for V2V transfer in meteorological datasets is discussed. The subsequent sections explore how neural fields can be applied to reconstruct statistical dependencies, such as Pearson correlation and MI, within large multivariate ensemble datasets as well as extensive time series data.

3.1 CNN-based Variable-to-Variable Transfer

To address the limitations of the original V2V transfer approach [HZX⁺20], an alternative variable selection procedure is proposed. This method replaces the single-variable auto-encoding CNN, along with the subsequent clustering and pairwise similarity search, with a more robust multi-variable CNN approach that incorporates loss-based tracking of the learning progress.

The process begins with a multi-variable time-varying dataset characterized by $n \in \mathbb{N}$ variables and $T \in \mathbb{N}$ time steps. This method considers a subset of m input fields, from which the remaining $n - m$ variable fields can be predicted. This setup allows for a more integrated approach to variable selection, leveraging the relationships among multiple variables simultaneously rather than relying on individual variable pairings. The use of loss-based tracking during the learning process enables the identification of the most informative and predictive input fields, enhancing the efficiency and accuracy of the variable selection procedure.

Given the computational intensity of training many networks for large n , a more efficient approach is required. A simpler method involves training only n networks, each predicting one variable from the remaining $n - 1$. The prediction loss of each network serves as an indicator of prediction difficulty: a high loss suggests low predictability, making that variable a strong candidate for the input subset. The variable with the highest loss is fixed as an input, and this process is repeated by selecting the next variable with the highest loss from the networks that already include the fixed input, continuing

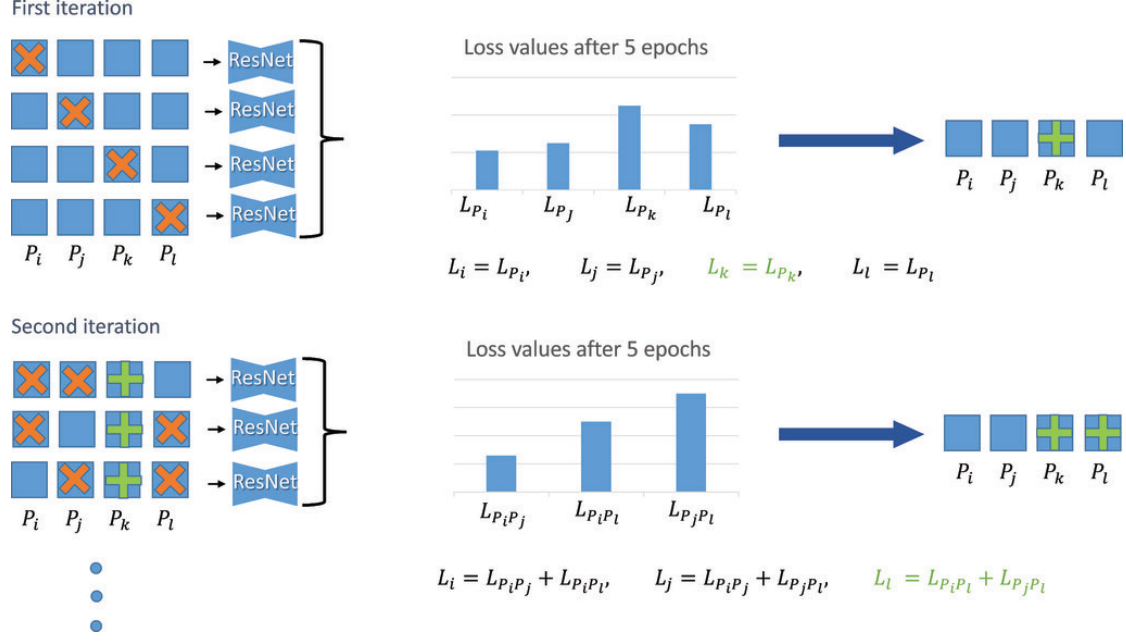


Figure 3.1: V2V Overview. In the i -th iteration, the same ResNet is repeatedly trained with varying combinations of $n - i$ input and i output variables. Output variables are marked with orange crosses, fixed input variables with green pluses, and free input variables with blue squares. At each iteration, a loss is calculated for each free variable by summing the losses of the networks where the variable is in the output set. The variable with the highest loss is then fixed in the input set. Figure from Farokhmanesh et al. [FHN⁺23a], © **American Meteorological Society. Used with permission.**

until the desired number of inputs is reached (see Figure 3.1). This approach replicates the input-output sets found by the exhaustive method but requires training only n networks. However, it only considers the prediction difficulty of individual variables, which may lead to overlooking stronger predictive combinations. For instance, if a subset of similar variables is easily predictable from each other, their associated losses will be low, making them less likely to be selected as inputs. As a result, the input set might lack these strongly correlated variables unless m is large enough to include at least one from each similar subset. However, setting m too large can render the method impractical. Compared to the V2V approach by Han et al. [HZX⁺20], the proposed strategy, while more computationally demanding, offers several key advantages. First, it provides a more accurate measure of transferability by directly training models to reconstruct variables. Second, it is not limited to pairwise variable selection but can reveal complex multi-variable relationships. Lastly, this method allows for the possibility of setting a loss threshold to determine when to stop iterations. Given the normalization of target variables, this threshold can be interpreted as a measure of acceptable relative error, offering a more intuitive and accessible stopping criterion than the distance threshold used in the V2V algorithm.

3.1.1 Network Architectures

In this study, a CNN is proposed to predict multiple output variables from a given set of input variables. One of the employed models is a ResNet architecture, which uses residual connections to bypass intermediate layers, preventing gradient decay and improving convergence [HZRS16a]. The proposed ResNet model in this dissertation includes three residual blocks, each comprising two convolutional layers with 64 channels and a kernel size of 3×3 , balancing computational cost and network depth. The network begins with an input block featuring a single convolutional layer, batch normalization, and a LeakyReLU [MHN⁺13] activation function. Input padding is applied to maintain spatial dimensions, using periodic padding for longitude in datasets with periodic boundaries and replication padding elsewhere. The final output layer is a single convolutional layer with $n - m$ output channels, completing the network architecture designed for robust variable prediction (see Figure 2.4).

As an alternative to the ResNet architecture, the potential of a U-Net architecture [RFB15] is also explored for loss-based variable selection. Unlike ResNet, which operates on a single spatial scale, the U-Net architecture extracts features across multiple spatial scales, enabling it to learn a broader range of variable relationships. The structure of the U-Net features two symmetric branches forming a characteristic U-shape (see Figure 2.3). The contracting branch consists of three convolutional blocks, each containing two convolution layers, followed by batch normalization, dropout, and a LeakyReLU activation function. The expansive branch features three deconvolution blocks, each comprising one convolution and one deconvolution layer. Similar to the contracting branch, each of these layers is followed by batch normalization, dropout, and a LeakyReLU activation function. The input block includes a single convolution layer with 64 channels and a kernel size of (3, 3), followed by batch normalization, dropout, and a LeakyReLU activation function.

In the experiments of Paper A, the U-Net architecture did not significantly improve reconstruction quality but did increase training time due to its higher computational complexity. A sample of the reconstruction quality of the U-Net is shown in Figure 3.2. Despite these findings, it is observed that ResNet and U-Net seem to internally learn different mappings, suggesting that they capture distinct aspects of the data, which is explained in detail in Paper A [FHN⁺23a].

3.1.2 Evaluations and Results

An alternative approach to performing deep learning-based V2V transfer is introduced. Instead of relying on the similarity of latent-space representations of variable fields to identify transferable variable pairs, a network is trained with various transfer scenarios to select the optimal variable settings. This method provides more flexibility for the network to leverage inter-variable relationships, allowing for improved transfer by learning variable combinations. It offers the potential for reducing bandwidth requirements in

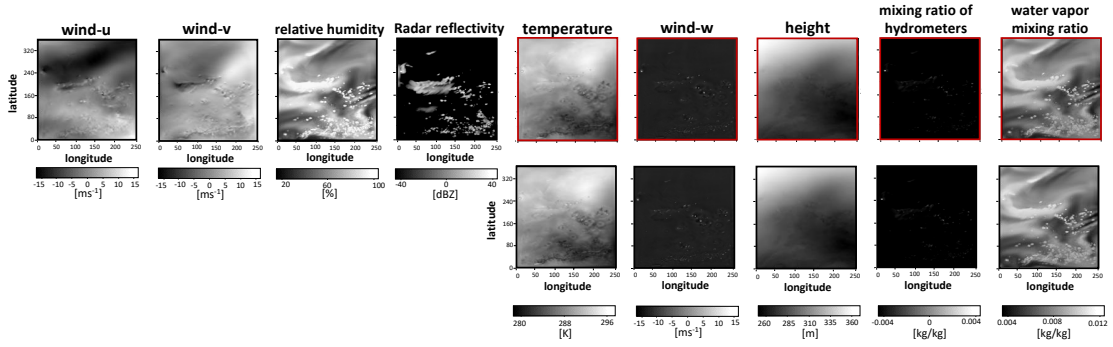


Figure 3.2: Reconstruction results for the WeatherBench dataset [RDS⁺20] are presented when the network is trained to predict three variable fields from three input fields and orography. The top row shows the initial variable fields, with red outlines indicating the fields the network is trained to predict from the others. The bottom row displays the predicted variable fields. Figure from Paper A, Farokhmanesh et al. [FHN⁺23a], © American Meteorological Society. Used with permission.

in-situ settings and facilitates more aggressive compression of multi-variable simulation data.

As demonstrated in Figure 3.3, while V2V transfer does not surpass classical lossy data compression schemes in terms of compression rate, it can effectively complement such schemes. Specifically, relevance maps generated through layer-wise relevance propagation LRP can guide spatially varying bitrates based on data importance. However, potential limitations of the method arise from challenges in comparing loss values across different variable fields. While L_1 -loss values work well for the data presented, variations in statistical distributions across fields can lead to misleadingly low or high loss values that may not accurately reflect reconstruction quality. In such cases, it is worth exploring alternative metrics that are less sensitive to distributional differences, such as relative improvement metrics or the structural similarity index (SSIM) [WBSS04], which relate more directly to visualization quality. Despite these considerations, the L_1 -loss is used as the primary metric for its general applicability, and its effectiveness is demonstrated in the current study.

Additionally, the regional variable structures that most significantly affect reconstruction quality are analyzed using an extended version of LRP. This analysis identified regions where field values have a strong impact on local reconstruction accuracy. LRP results also revealed that different neural network architectures learn distinct mapping functions based on their respective inductive biases. For instance, the U-Net model produced more physically interpretable relevance maps, whereas the ResNet architecture, due to its design, is more limited in capturing spatial dependencies. This insight can be valuable in operational model applications, offering a better understanding of model inference procedures and increasing trustworthiness in data-driven predictions.

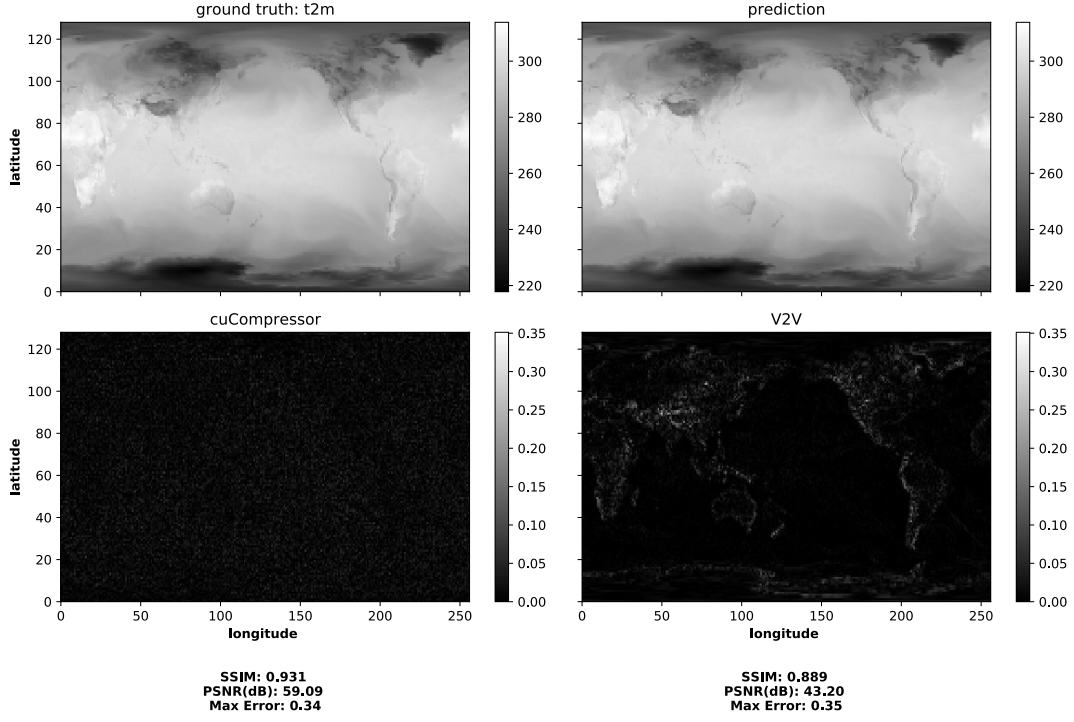


Figure 3.3: A comparison of the V2V transfer method and a standard compression algorithm for volumetric data is shown. The temperature field [RDS⁺20] is compressed at a ratio of 12:1 using the CUDA compression library [TRA⁺W12], which applies lossy compression techniques such as wavelet transforms, quantization, run-length encoding, and Huffman coding. The top left image shows the original field, while the top right shows the field predicted using the 3-to-3 V2V transfer method. The bottom images display the pixel-wise differences between the compressed and reconstructed fields for both methods. Figure from Paper A, Farokhmanesh et al. [FHN⁺23a], © American Meteorological Society. Used with permission.

Given that the method proposed in Paper A [FHN⁺23a] did not sufficiently uncover informative variable dependencies, additional efforts are made to explore alternative dependency analysis techniques, with a focus on spatial and temporal relationships between variables. The outcomes of this analysis led to the development of the method in Paper B [FHN⁺23b], which presents an interactive visualization tool for analyzing statistical dependencies in large 3D ensemble datasets, and Paper C [FNW24], where the temporal dependencies in a large time series dataset are analyzed. Section 3.2 gives a general overview of the methodology and evaluations of these methods.

3.2 Neural Field-based Statistical Dependency Reconstruction

This section provides an overview of the two remaining works, Paper B [FHN⁺23b] and Paper C [FNW24], incorporated in this dissertation. First, the statistical dependency analysis in CSEns dataset [NGW⁺20] is proposed utilizing neural fields, providing compact fields enabling interactive visualization of statistical dependencies such as Pearson correlation and MI, Section 3.2.1. Then, the application of neural fields in the reconstruction of MI in large time series datasets is explored, Section 3.2.2.

3.2.1 Interactive Visualization of Statistical Dependencies in 3D Simulation Ensembles

Estimating statistical dependencies between physical variables across spatial locations is critical for understanding complex systems in various scientific and engineering disciplines. This is particularly important in meteorology, where accurate weather forecasting depends on extensive numerical simulations. Forecast ensembles, which incorporate variations in initial conditions and model parameters, are used to account for uncertainty. Analyzing statistical relationships, such as spatio-temporal auto-correlations and cross-variable correlations, is essential for translating ensemble data into reliable weather forecasts.

While methods for assessing both linear and non-linear relationships between random variables are well-established, determining dependencies in 3D ensemble fields presents significant challenges. To address these issues, this dissertation introduces neural dependence fields (NDFs), a compact and efficient representation of the primary dependency structures in large, multi-variable ensembles. The proposed approach interprets fields of two-point dependency measures as scalar fields over the space of position pairs in 3D. The proposed neural network architecture is designed to capture self-similarities within the dependency fields, learning a compact representation.

For the ensemble dataset used in this study, the NDF model reconstructs dependencies between an arbitrary reference point and all other points in the domain in about 9 milliseconds. This rapid computation enables the integration of the neural model into an interactive volume rendering pipeline, allowing for immediate visualization and comparison of within- and between-variable dependencies. The proposed method supports both linear measures, like Pearson correlation, and more computationally expensive non-linear measures, such as MI. Despite a slight smoothing of fine details, the model accurately reconstructs the primary dependency structures within the dataset.

3.2.1.1 Neural Fields

The following explanation of statistical dependencies in ensemble fields is inspired by the explanations provided in Paper B [FHN⁺23b]. Statistical dependencies in ensemble datasets are quantified through bivariate dependency measures $\rho : \mathbb{R}^N \times \mathbb{R}^N \rightarrow \mathbb{R}$, which

are computed between paired random sample vectors. Consider a simulation domain $\Omega \subset \mathbb{R}^3$ in 3D space, and an ensemble $\mathcal{E} = \{E_i : 0 \leq i < N\}$ consisting of N multi-variable fields $E_i : \Omega \rightarrow \mathbb{R}^d$. The index i represents an arbitrary member of the ensemble. For each i and $0 \leq \nu < d$, the scalar field associated with variable ν in ensemble member E_i is denoted $E_i^\nu : \Omega \rightarrow \mathbb{R}$. For a given position $\mathbf{p} \in \Omega$, the local sample of variable values is defined as $\mathbf{e}^\nu(\mathbf{p}) := (E_i^\nu(\mathbf{p}) : 0 \leq i \leq N) \in \mathbb{R}^N$.

The dependency field for variables μ and ν , denoted $R_{\mu\nu} : \Omega \times \Omega \rightarrow \mathbb{R}$, is defined for all pairs of positions $(\mathbf{p}_\mu, \mathbf{p}_\nu) \in \Omega^2$. The value of the field at these positions is given by $R_{\mu\nu}(\mathbf{p}_\mu, \mathbf{p}_\nu) := \rho(\mathbf{e}^\mu(\mathbf{p}_\mu), \mathbf{e}^\nu(\mathbf{p}_\nu))$. Here, the positions $\mathbf{p}_\mu$ and $\mathbf{p}_\nu$ adjust the dependency value by varying the reference positions in fields μ and ν . In particular, the case where $\mu = \nu$ defines the self-dependency field $S_\mu := R_{\mu\mu}$, which is symmetric under the exchange of position coordinates, i.e., $S_\mu(\mathbf{p}_1, \mathbf{p}_2) = S_\mu(\mathbf{p}_2, \mathbf{p}_1)$ for all $\mathbf{p}_1, \mathbf{p}_2 \in \Omega$.

To efficiently handle large sets of these point-to-point dependence measures, the computationally expensive dependency calculations are preprocessed and encoded using memory- and compute-efficient neural scene representations. Specifically, the two-point dependency fields $R_{\mu\nu}$ are approximated by neural networks $\Phi_{\mu\nu} : \Omega \times \Omega \rightarrow \mathbb{R}$. The network is trained by solving the optimization problem:

$$\Phi_{\mu\nu} := \arg \min_{\Phi} \mathbb{E}_{(\mathbf{p}_1, \mathbf{p}_2) \sim \mathcal{U}(\Omega^2)} [d(\Phi(\mathbf{p}_1, \mathbf{p}_2), R_{\mu\nu}(\mathbf{p}_1, \mathbf{p}_2))], \quad (3.1)$$

where $\Phi(\cdot, \cdot)$ is the neural network, $d(\cdot, \cdot)$ is a similarity metric (such as L_1 or L_2 -loss), and the expectation is taken over position pairs sampled from the uniform distribution $\mathcal{U}(\Omega^2)$. Stochastic gradient descent is used to iteratively perform the optimization with batches of position pairs, avoiding excessive storage of dependency samples. After training, $\Phi_{\mu\nu}$ provides a compact encoding of the dependency fields, allowing rapid reconstruction of sample dependencies. This approach offers significant advantages over classical compression methods like TThresh and SZ, which are limited by fixed discretization and computational challenges. Furthermore, once the neural dependence network is trained, the entire ensemble dataset no longer needs to be retained in memory, enabling the method to scale efficiently to large ensemble sizes. NDFs are distinct from typical neural scene representations because they work in a 6D bi-spatial domain. This complexity makes training more difficult due to the curse of dimensionality. To tackle this, NDFs are designed to use sparse sampling more effectively, improving memory efficiency and avoiding the need to compute too many dependency samples.

The network architecture proposed for NDFs, as shown in Figure 3.4, is bipartite, meaning it has two variable-specific encoder networks (Enc_μ and Enc_ν) and a shared decoder. Each encoder processes one of the two positions between which the dependency is being calculated. The positions are converted into latent feature vectors, which are then combined through element-wise multiplication before being passed to the shared decoder. This design allows each encoder to only handle the 3D space $\Omega \subset \mathbb{R}^3$, improving the training process by using more efficient sampling. The model can then infer dependencies

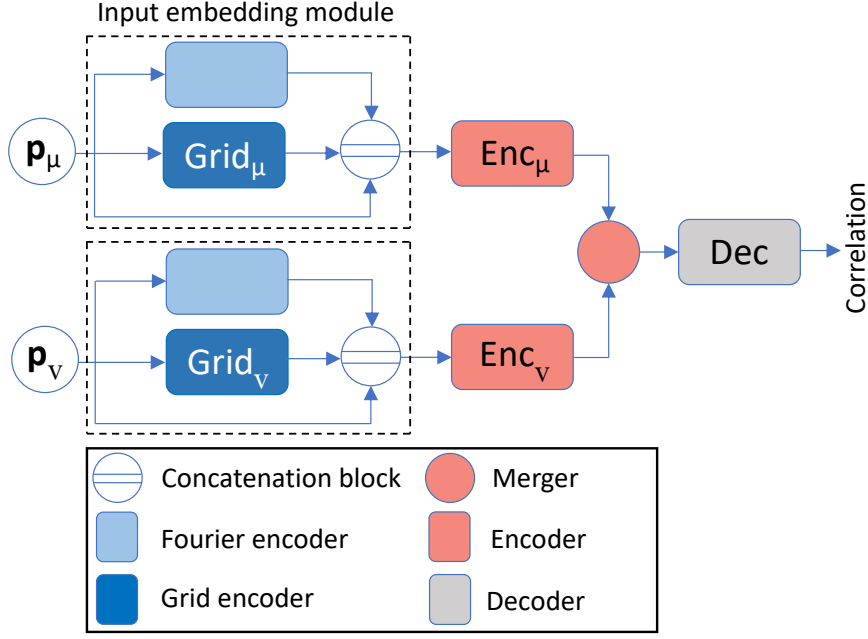


Figure 3.4: NDF architecture. The NDF architecture receives two positions as input: the reference position $\mathbf{p}_\mu$ and the query position $\mathbf{p}_\nu$. Corresponding to these positions are the hash grids Grid_μ and Grid_ν . Variable-specific encoders, Enc_μ and Enc_ν , are MLPs. The merger refers to the multiplication of the outputs from the two encoders. Figure from Paper B, Farokhmanesh et al. [FHN⁺23b].

between point pairs that are not explicitly trained, making the process faster and using less memory. For self-correlation fields, the NDFs are further simplified by using the same encoder for both positions ($\text{Enc}_\mu = \text{Enc}_\nu$). This keeps the model smaller and ensures that the dependency values are symmetric, meaning $\Phi_{\mu\mu}(\mathbf{p}_1, \mathbf{p}_2) = \Phi_{\mu\mu}(\mathbf{p}_2, \mathbf{p}_1)$, without needing extra training to learn this symmetry.

To improve the ability of the model to capture fine details and multi-scale spatial patterns, Fourier features [MST⁺21, TSM⁺20] and multi-resolution hash grids [MESK22] are used. These features are applied to the input positions before they are processed by the encoders. Fourier features map the coordinates into a higher-dimensional space, allowing the model to detect high-frequency patterns without increasing the number of trainable parameters. Multi-resolution hash grids, on the other hand, store feature vectors in hash tables, creating virtual grids at different scales. These grids allow feature retrieval using tri-linear interpolation, helping the model handle different spatial resolutions efficiently. The hash grids allow the model to represent various levels of detail while keeping memory usage low. During training, the grids are updated together using stochastic gradient descent, and hash collisions are naturally resolved through the randomness of the hash functions and multiple resolution levels. The key parameters for the hash grid include the size of the hash table (2^T for $T \in \mathbb{N}$), the number of resolution levels, and the grid

size for each level. In this work, six levels of resolution are used, starting with a grid size of 16^3 at the coarsest level, doubling at each finer level. This setup ensures the model captures fine details in the data without consuming excessive memory while still maintaining good reconstruction accuracy.

3.2.1.2 NDFs Training and Evaluations

During training, a rectangular simulation domain Ω is rescaled to fit within the symmetric unit cube, $\Omega = [-1, 1]^3$, with data samples placed on a regular grid. A total of 10^6 pairs of uniformly distributed random positions $(\mathbf{p}_\mu, \mathbf{p}_\nu)$ are generated from this domain. Using trilinear interpolation on the original dataset, the values $\mathbf{e}^\mu(\mathbf{p}_\mu)$ and $\mathbf{e}^\nu(\mathbf{p}_\nu)$ are retrieved, and the corresponding dependencies $R_{\mu\nu}(\mathbf{p}_\mu, \mathbf{p}_\nu)$ are computed. The model is trained by minimizing the L_1 -loss, although L_2 -loss has shown similar effectiveness. The Adam optimizer [KB14] is used, starting with a learning rate of 3×10^{-4} , and batches of 1000 samples are processed at a time. To improve the learning process, an adaptive learning rate scheduler is applied, reducing the rate by a factor of 10 if there is no progress after five consecutive steps. The sample pairs are renewed after every epoch, and training is performed for 200 epochs in total. Once training is completed, the NDF model can efficiently estimate dependencies between any two positions within the grid. To speed up calculations, multiple queries are processed in parallel using the tiny-cuda-nn framework [Mü21], which features an optimized fused MLP implementation [MRNK21b]. This implementation allows for rapid 16-bit inference on NVIDIA GPUs and supports multi-resolution hashed feature grids [MESK22]. Additionally, custom activation functions, such as *Snake* and *SnakeAlt* [WHW22], are incorporated into the framework, as simpler functions like ReLU are found to give less accurate results.

Performance is evaluated by comparing the runtime of NDF with standard implementations of Pearson correlation and MI estimators using the CSEns dataset [NGW⁺20]. The MI estimator used follows the method by Kraskov et al. [KSG04]. Testing is conducted on an NVIDIA RTX 3090 GPU and a 6-core Intel Xeon W-2235 CPU, with the NDF running on the GPU and the MI estimator executed on the CPU. Findings show that the NDF model runs about 26 times faster than the GPU-based Pearson correlation calculation. In comparison with the CPU-based MI estimator, the NDF model is over 114,000 times faster. After training, the NDF requires just 9 milliseconds to compute dependencies between points across the grid, consuming only 1 GB of memory—significantly less than the 7 GB required to store a full dataset for one variable. Training the NDF takes around one hour on the hardware used.

Although a GPU-based NDF estimator is not tested, it is expected that due to the computational demands of building nearest-neighbor search structures for MI, a GPU-optimized version can still be slower than NDFs, which take advantage of highly efficient fused MLP operations for fast calculations. Results displayed in Figure 3.5 demonstrate the ability of NDF to reconstruct both Pearson correlation coefficients and MI values. The experiments first focus on self-correlation fields for different variables and evaluate how

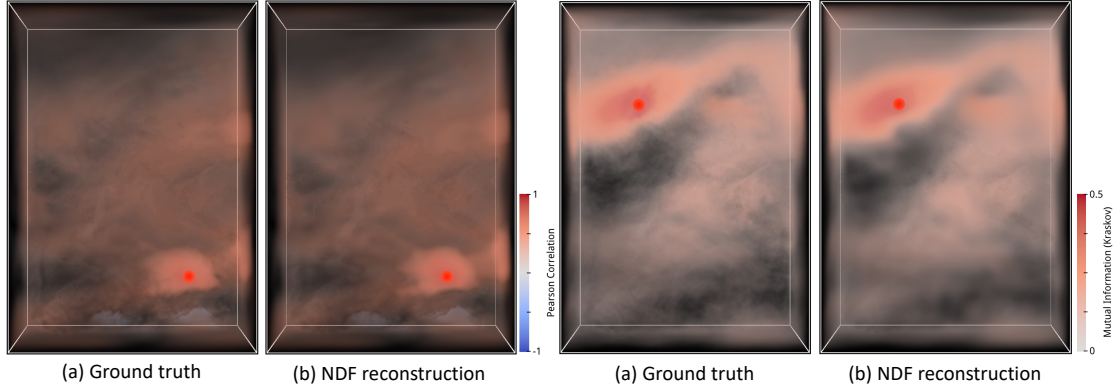


Figure 3.5: NDF model inference of the dependencies between data values at an arbitrary grid vertex (red dot) to all other vertices in a $250 \times 352 \times 20$ grid. Left: Pearson correlation. Right: mutual information. Figure inspired from Paper B, Farokhmanesh et al. [FHN⁺23b].

well NDFs capture these dependencies. NDFs are also used to reconstruct dependencies between different variables (see Figure 3.6).

3.2.2 Application of Neural Fields in Mutual Information Estimation in Large Time Series

Mutual information (MI) helps in measuring the linear and non-linear dependence between two random variables. Specifically, when a random variable X_2 is known, the MI $I(X_1, X_2)$ tells us how much information is provided about the variable X_1 . This makes MI particularly useful for analyzing time series data, where it can be applied to evaluate the dependence between different time windows, supporting tasks such as feature extraction and classification. Suppose there are T time series, each with a length of N . Each time series A^j ($0 \leq j < T$) is described by a sequence of N data points, $(a_1^j, a_2^j, \dots, a_N^j)$. The i -th data point in series A^j is denoted by $a_i^j \in \mathbb{R}$. When using a sliding window approach on a time series A^j of length N , $(N - l + 1)$ windows of size l can be created, each sliding by one position:

$$\text{Windows}(A^j) = \{A_{1,l}^j, A_{2,l}^j, \dots, A_{(N-l+1),l}^j\}. \quad (3.2)$$

Calculating MI between two discrete random variables can be complex and time-consuming. Estimating the joint entropy between two variables requires building their joint histograms to approximate their joint probability distributions. The Kraskov MI estimator [KSG04] simplifies this by using the k -nearest neighbors in the joint distribution space to estimate entropy. However, for time series, estimating MI for all sliding windows within a chosen series and their respective windows in other series can take several seconds. This time increases even more when considering lagged time series, where MI must be calculated for both past and future windows. To make this process faster, neural networks are introduced to estimate MI in time series data. Paper B [FHN⁺23b] shows

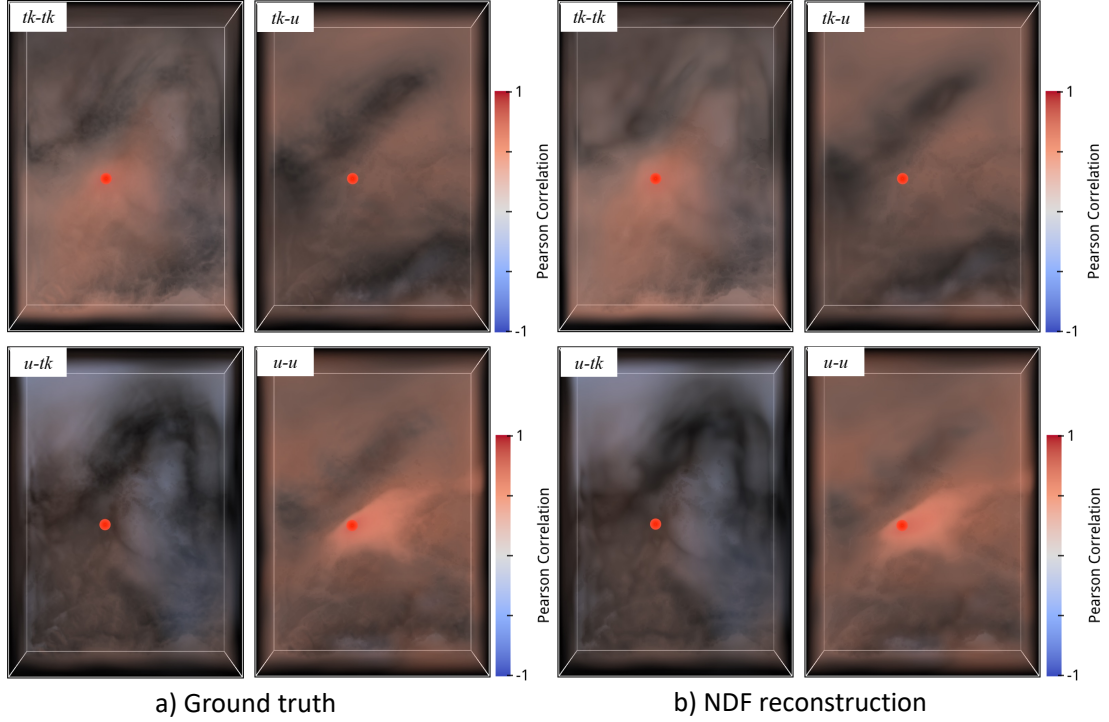


Figure 3.6: NDF Reconstruction between two different variables. Visualization of the dependencies between the temperature (tk) and the longitudinal component (u) of wind speed using Pearson correlation is shown. In a), the ground truth correlations are shown, while in b), the correlations reconstructed by the NDF model are displayed. The fields depict the correlations between either tk or u at specific points and the same or other variable at different points. Figure inspired from Paper B, Farokhmanesh et al. [FHN⁺23b].

that key dependency patterns in 3D physical fields can be learned using small neural networks. These networks are trained to compute the dependency between a point and all other points in the dataset. Once trained, this can be done in milliseconds, even for large datasets.

A similar idea is applied here for estimating MI between time windows in different time series. The MI values between window pairs are treated as scalar fields, with the neural network trained to estimate the MI between a chosen window in one time series and all corresponding windows in the other time series in a single pass. This estimation is done in approximately 44 milliseconds, allowing quick, interactive exploration where the user can select a time series and see which other series show high linear or non-linear dependence over time. To further extend this method to handle lagged time series, the network is trained to estimate MI for both past and future lagged windows. In the experiments, lags from -64 to 64 are used, but the network can be adapted to work with larger or smaller ranges as needed. After training, the MI between the windows in the chosen time series and all lagged windows in the other series can be estimated in about 1.1 seconds. This

approach is applied to collections where a time series and its shifted versions are present, with the MI used to determine the lag and correctly classify the shifted versions.

3.2.2.1 Network Architecture

The network architecture presented in paper B [FHN⁺23b] is adapted for estimating MI between windows in time series data. The 2D space for MI inference includes a continuous coordinate representing the sequence of data points and a discrete coordinate representing the different time series available in the datasets. TNMIFs operates within a four-dimensional (4D) bi-spatial domain. The proposed method, as shown in Figure 3.7, utilizes a network architecture consisting of two encoder networks and one decoder network. These networks are implemented as MLPs with L fully connected layers and C hidden channels, using the *SnakeAlt* activation function [WHW22]. Each encoder processes one of the two positions for which MI is to be reconstructed, converting these positions into latent feature vectors. The vectors are then combined through element-wise multiplication and passed to the decoder for the final prediction. This design allows each encoder to train on a marginal space $\Phi \subset \mathbb{P}$, maximizing the effective number of MI samples per volume and improving training efficiency. The model is also capable of extrapolating MI values for unseen point pairs during training, thus reducing both computation time and memory usage.

To ensure the model remains compact and maintains symmetry, identical encoders are used for both positions and consistent Hash encoding and Embedding are applied, as illustrated in Figure 3.7. This approach guarantees that it is symmetric under the exchange of window positions by design, thus avoiding the need for extensive training to learn this symmetry. In TNMIFs, the 4D feature domain is decomposed into an outer product of fields over a 2D domain. Experiments with alternative methods, such as concatenation or absolute difference, demonstrated reduced prediction accuracy, highlighting the importance of multiplication for achieving precise predictions, as supported by previous research [FKMW⁺23, FHN⁺23b].

To capture high-frequency patterns and multi-scale features, Fourier features [MST⁺21, TSM⁺20] and multi-resolution hash encoding [MESK22] are incorporated for the continuous coordinate, along with an embedding layer [PSM14, MSC⁺13] for the discrete coordinate. These encoded inputs, combined with the raw positions, are fed into the encoders, as indicated by the dashed rectangles in Figure 3.7. For the continuous dimension p_{ws} , Fourier features enhance the differentiation between spatially close positions by projecting them into a higher-dimensional space. This approach enables the model to resolve high-frequency patterns without increasing the number of trainable parameters or memory usage. The implementation found that setting $L = 12$ for Fourier frequencies yields effective results.

The multi-resolution hash encoding is applied to the continuous coordinate p_{ws} , using hash tables filled with trainable feature vectors to cover the domain at multiple resolu-

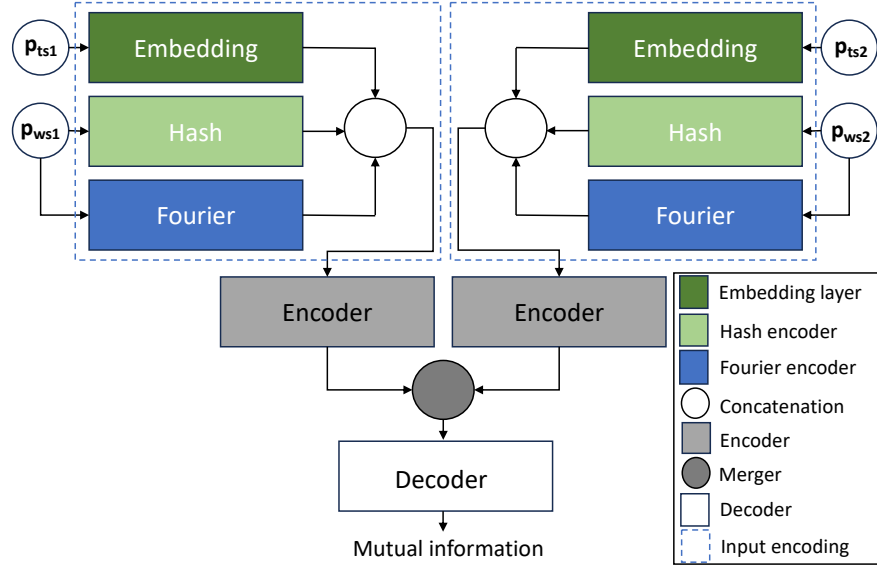


Figure 3.7: TNMIF architecture. The positions $(\mathbf{p}_{ts1}, \mathbf{p}_{ws1})$ and $(\mathbf{p}_{ts2}, \mathbf{p}_{ws2})$ are the reference and query positions, respectively. The Embedding layer is responsible for embedding the input features, while the Hash refers to the hash-grid encoding applied to the input data. The Fourier module applies Fourier feature encoding to the input. Both the Encoder modules and the Decoder module are implemented as MLPs. The Merger operation involves multiplying the outputs of the encoders. Figure from Paper C, Farokhmanesh et al. [FNW24].

tions [MESK22]. These vectors are combined across different resolution levels and trained together with subsequent model components. The expressiveness of the hash grid is controlled by the hash table size, 2^T for $T \in \mathbb{N}$, which also determines memory complexity. Other hyper-parameters include the feature vector dimension per resolution level, the number of resolution levels, and the grid resolution at each level. Eight resolution levels are used with virtual grids of size 16^3 at the coarsest level, doubling in size at each finer level. These parameters align the feature granularity with the resolution of the dataset, avoiding excessive memory usage while maintaining or improving reconstruction accuracy. Empirical testing indicated that a hash table size of $T = 30$ produces optimal results.

The embedding layer translates discrete variables into a continuous vector space, with each unique value represented by a learnable vector. These vectors are initialized randomly and refined during training, allowing the model to capture complex relationships and structures within the data. For time series data, this process involves converting discrete time series identifiers into continuous embeddings. This transformation enhances the network’s ability to discern and encode underlying patterns within the temporal data. Key parameters include the number of embeddings, which corresponds to the unique identifiers within the dataset, and the embedding dimension, which defines the complexity of the vector representation. The embedding dimension is optimized through experimentation to balance detail capture with computational efficiency.

3.2.2.2 TNMIFs Evaluations

Datasets The quality and performance of the proposed data structure are evaluated using two large time series datasets from the UCR time series classification archive [DBK⁺19]. The main quantitative analysis and performance evaluation is conducted on the CinCECG-Torso dataset. Additionally, the StarLightCurves dataset is used to further demonstrate the method’s performance and generalization.

The CinCECGTorso dataset contains 1,420 time series instances, each consisting of 1,639 time steps. The data originates from the computers in cardiology (CinC) challenges, which are hosted annually on Physionet, a repository of physiological signals and related data [GAG⁺00]. These ECG recordings are captured from multiple torso-surface sites and are collected from individuals participating in the challenges. Each individual corresponds to a distinct class in the dataset, with the four classes representing unique participants.

The StarLightCurves dataset [RPBA09], also from the UCR classification archive, consists of 1,000 training samples and 8,236 test samples, each with 1,024 time steps. This dataset illustrates the variability in brightness of celestial objects over time, with domain experts classifying these starlight curves into three distinct categories. For the experiments and evaluations, the 1,000-sample training set is utilized.

Training Two different training approaches are examined in this study: synchronous models and lagged models. In the synchronous models, sample pairs are created using synchronized time windows from different time series, which are then used for training. On the other hand, lagged models use sample pairs where a time window in one time series is paired with several lagged windows from another time series. The outputs of these models are MI values for each sample pair. These models are referred to as sync-TNMIF and lag-TNMIF, respectively. During the training process, data samples from the domain Φ are rescaled to fit within a symmetrical matrix space. For sync-TNMIF, 500,000 pairs of uniformly distributed positions $(\mathbf{p}_j, \mathbf{p}_k)$ are generated, while lag-TNMIF uses 1,000,000 pairs. These positions are used to retrieve the corresponding samples $\mathbf{a}(\mathbf{p}_j)$ and $\mathbf{a}(\mathbf{p}_k)$ from the original time series data. The mutual information $I(\mathbf{p}_j, \mathbf{p}_k)$ between each pair of samples is calculated. Both models are trained by minimizing the L_1 -loss function, though similar results are observed when using the L_2 -loss function in the experiments. The Adam optimizer [KB14] is used for the training process, starting with a learning rate of 7×10^{-4} , and training is conducted with batches of 1,024 samples. After every epoch, the training samples are refreshed to ensure consistent model performance. The sync-TNMIF model is trained for a total of 100 epochs, while lag-TNMIF is trained for 150 epochs.

The time series data is split into windows of a specified length l , with a fixed lag of $s = 1$. Choosing the correct window size is essential for time series analysis because it determines how the data is divided into subsequences that ideally align with individual temporal patterns, like heartbeats in ECG recordings or respiratory cycles. While manually setting

the window size is possible, automating the process is more efficient. In this study, the ClaSP algorithm is used to automatically find the best window size for the CinCECGTorso dataset. The ClaSP method, developed by Ermshaus et al. [ESL23b, ESL23a], serves as a benchmark tool to assess traditional and modern time series segmentation methods.

Performance All timings are evaluated on a system running Ubuntu 22.04, equipped with an Intel Xeon W-2235 6-core (12-thread) CPU and an NVIDIA RTX 3090 GPU. Ground truth MI fields are obtained by parallelizing Kraskov MI estimation on the multi-core CPU. Given two realization vectors of random variables X and Y , with joint samples $Z = \{(X_i, Y_i), i \in \{0, \dots, N - 1\}\}$, the Kraskov estimator relies on identifying the k -nearest neighbor for each joint sample. A k -d tree is used as the search structure in this implementation. Since one k -d tree is built for each grid point, parallelizing on the multi-core CPU is straightforward. Additionally, system calls are reduced by moving heap memory allocation outside the parallelization loop. Network training is performed in PyTorch. To achieve fast inference times, the tiny-cuda-nn GPU framework [Mü21], which offers a fast implementation of fully-fused MLP layers, is extended to support the embedding layer described in Section 3.2.2.1.

For the CinCECGTorso dataset, computation times are assessed for three scenarios: a) sync-TNMIF, estimating 1544 (the number of windows of length 96 in one selected time series) times 1419 (the number of remaining time series) MI measures, b) lag-TNMIF, estimating 1544 times 1419 times $2 \times 64 + 1 = 129$ MI measures (including negative and positive lags), and c) the same as a), but using the CPU-based Kraskov implementation. It is observed that the network implementation is approximately 632 times faster than the Kraskov MI implementation and does not require the time series or precomputed MI fields to be stored in memory during runtime. The slightly slower performance of the proposed model, compared to [FHN⁺23b], can be explained by differences in model architecture. The inclusion of an additional component in the encoding module, the embedding layer, and a larger number of time-window positions (at least 2,200,000) contributes to the slower performance. Furthermore, the encoder is applied to all window positions in the selected time series rather than just a single reference position, which results in less efficient memory access.

To further evaluate the performance of the lag-TNMIF model, an experiment is conducted to reconstruct MI between time windows within a selected time series and all lagged windows, with a maximum lag of ± 64 , for all other time series in class one of the CinCECGTorso dataset. Notably, considering a maximum lag of 64 reduced query times by about a factor of 10. This experiment highlights the efficiency of the lag-TNMIF model in reconstructing large volumes of lagged MI values, with the model achieving this task within 1.1 seconds. To demonstrate the usefulness of this approach, two specific time series similarity measures based on window-to-window MI are examined for the CinCECGTorso dataset: the time intervals between consecutive extreme events below the mean [LSK⁺19], and the interquartile range (IQR). The first measure focuses on the time gaps between extreme events that occur below the mean of all MI estimates between

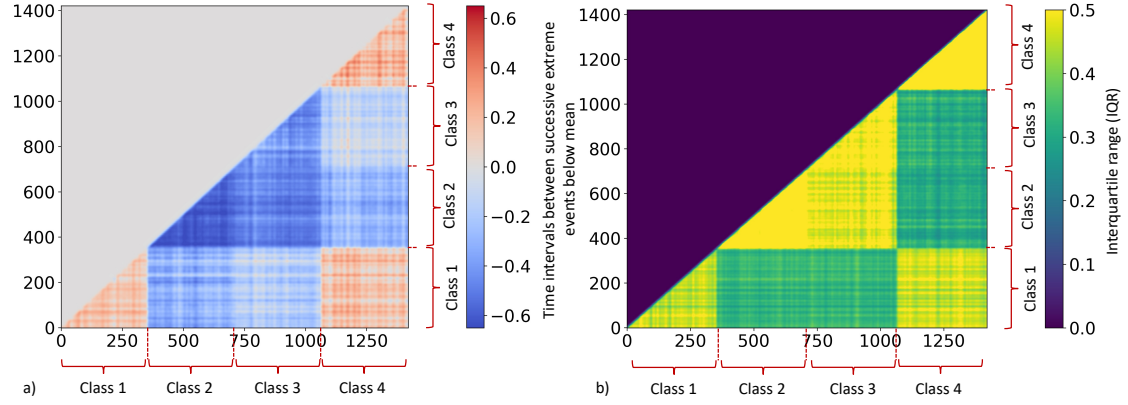


Figure 3.8: The vertical and horizontal axes represent the time series indices. Sync-TNMIF is used to compute all possible pairwise MI between the time series in just a few seconds. Statistical metrics, such as the IQR and time intervals between consecutive extreme events, are then calculated based on the pairwise MI results. Each point in the matrix represents a similarity value for a specific pair of time series. Figure from Paper C, Farokhmanesh et al. [FNW24].

different time series. The IQR, a statistical measure of data spread, is also calculated. It is determined by subtracting the first quartile (which represents the median of the lower half) from the third quartile (the median of the upper half). Both of these measures are computed and visualized for each pair of time series, as shown in 2D matrices (see Figure 3.8). These plots provide insight into the variations in temporal dynamics across different time series. Given that the dataset consists of four distinct classes, intra- and inter-class relationships are explored. The similarity matrix highlights strong similarities among time series within the same class. Additionally, both metrics showed comparable behavior between the first and fourth classes, as well as between the second and third classes. This analysis sheds light on the underlying structure and patterns in the dataset, which can be useful for interpretation and classification.

4 Paper A: Deep Learning-based Parameter Transfer in Meteorological Data

Abstract Numerical simulations in earth-system sciences involve multiple physical variables across space and time, resulting in significant I/O bandwidth demands and challenges in data analysis. To address this, deep-learning-based methods, such as variable-to-variable (V2V) transfer, are proposed to reduce bandwidth by identifying redundant variables and predicting them from others. In this study, the feasibility of V2V transfer for meteorological datasets is evaluated. It is found that the original V2V algorithm faces limitations due to limited redundancies between variable pairs. To overcome this, an ablation study is conducted to analyze the predictive capacity of reanalysis variables by assessing the learning behavior of V2V reconstruction networks. The study shows that V2V transfer becomes efficient when groups of variables are considered, leading to the proposal of an algorithm that implements this approach, offering new opportunities for meteorological data analysis. Additionally, the interpretability of the learned representations is explored using layer-wise relevance propagation (LRP), which identifies critical regions and variables for the reconstruction model. Applied to the meteorological data, this method reveals key relationships between landscape orography and regional weather patterns. This approach effectively reduces bandwidth requirements in numerical weather simulations and can complement existing data compression techniques. The identification of multi-variable features leads to further research on regional weather prediction and applications in other simulation data.

Author Contributions The method and idea were proposed by Rüdiger Westermann and Fatemeh Farokhmanesh. The implementation of network architecture, experiments, evaluations and ablation study was done by Fatemeh Farokhmanesh. Kevin Höhle proposed and implemented the feature analysis. Tobias Necker, Martin Weissmann and Takemasa Miyoshi provided one of the datasets used. Discussions with the co-authors led to the final paper, and Fatemeh Farokhmanesh, Kevin Höhle and Rüdiger Westermann wrote the manuscript.

Copyright © American Meteorological Society. Used with permission. “Deep Learning-Based Parameter Transfer in Meteorological Data”. In: *Artificial Intelligence for the Earth Systems 2023*. Volume 2, Issue 1. Permission of reuse in this dissertation was granted, as specified in the permission grant document printed on Page 105.

5 Paper B: Neural Fields for Interactive Visualization of Statistical Dependencies in 3D Simulation Ensembles

Abstract Neural dependence fields (NDFs) are introduced as a novel neural network designed to compactly represent and efficiently reconstruct statistical dependencies between physical variables at different spatial locations within large 3D simulation ensembles. Unlike traditional methods focused on linear dependencies, mutual information is employed as a measure of non-linear dependencies. The approach is demonstrated on a weather forecast ensemble consisting of 1000 members, each containing multiple physical variables on a $250 \times 352 \times 20$ simulation grid. By avoiding the use of compute-intensive statistical estimators at runtime, significant reductions in memory and computational demands are achieved for reconstructing the key dependence structures. This reduction enables the integration of the estimator into a GPU-accelerated direct volume renderer, allowing for interactive visualization of all mutual dependencies at any selected domain point.

Author Contributions Fatemeh Farokhmanesh, Kevin Höhle and Rüdiger Westermann proposed the method and idea. Fatemeh Farokhmanesh was responsible for the development of the model and analysis methodologies, such as experiment design and model performance evaluations. The dataset was provided by Tobias Necker, Martin Weissmann and Takemasa Miyoshi. Christoph Neuhauser implemented the GPU-based volume rendering pipeline enabling interactive visualization. Fatemeh Farokhmanesh, Kevin Höhle, Christoph Neuhauser and Rüdiger Westermann wrote the manuscript.

Copyright © 2023 The Author(s), Proceedings published by Eurographics © 2023 The Eurographics Association. This is an open access article under the terms of the Creative Commons Attribution 4.0 License, which permits use, distribution, and reproduction in any medium, provided the original work is properly cited. The terms of the license are printed on Page 106 and available online at <https://creativecommons.org/licenses/by/4.0/>

6 Paper C: A Compact and Efficient Neural Data Structure for Mutual Information Estimation in Large Timeseries

Abstract Database systems encounter significant challenges when using mutual information (MI) to analyze non-linear relationships in large time series, primarily due to the high computational and memory demands. These challenges are particularly evident in interactive workflows, where long response times hinder usability. To address this, time series neural MI fields (TNMIFs) are introduced as a compact data structure trained to efficiently reconstruct MI across varying time windows and positions in large time series datasets. The method is demonstrated on a dataset containing 1420 time series, each with 1639 time steps. The learned data structure, consuming only 45 megabytes, can provide MI estimates between windows in a selected time series and corresponding windows in all other series within 44 milliseconds. Furthermore, the matrix representing all mutual similarities between time series, based on windowed MI estimates, is computed in under 32 seconds. To accommodate dependence between lagged time series, an extended data structure is used to reconstruct MI for both positively (future) and negatively (past) lagged windows.

Author Contributions Fatemeh Farokhmanesh and Rüdiger Westermann proposed the method and idea. Fatemeh Farokhmanesh was responsible for the development of the model and analysis methodologies, such as experiment design and model performance evaluations. Christoph Neuhauser implemented and integrated the dictionary encoding to the tiny-cuda-nn framework enabling rapid dependency reconstruction of the models. Fatemeh Farokhmanesh and Rüdiger Westermann wrote the manuscript.

Copyright © 2024 The Owner/Author(s), Proceedings of the 36th International Conference on Scientific and Statistical Database Management © 2024 Association for Computing Machinery. This is an open access article under the terms of the Creative Commons Attribution 4.0 License, which permits use, distribution, and reproduction in any medium, provided the original work is properly cited. The terms of the license are printed on Page 106 and available online at <https://creativecommons.org/licenses/by/4.0/>

7 Conclusion and Outlook

This thesis presents a comprehensive exploration of deep learning methods designed to analyze and quantify dependencies in large ensemble datasets. Across the studies documented in Paper A [FHN⁺23a], Paper B [FHN⁺23b], and Paper C [FNW24], various approaches are proposed to address the challenges of understanding the relationships between variables, both in spatial and temporal contexts, with a strong emphasis on enhancing interpretability and computational efficiency.

7.1 Conclusion

In Paper A [FHN⁺23a], an alternative deep learning-based variable-to-variable (V2V) transfer method is developed, which trains networks using various transfer scenarios rather than relying solely on latent-space similarities between variable fields. This approach provides greater flexibility in capturing inter-variable relationships, enabling more efficient transfer of variables and potentially reducing bandwidth usage in in-situ simulations. While traditional compression schemes may outperform V2V in terms of compression rates, this method can complement them by using relevance maps to adapt bitrates according to data importance. The evaluation of different model architectures using LRP shows that the inductive biases of networks, such as U-Net and ResNet, lead to distinct learned mappings. U-Net generates more interpretable relevance maps, highlighting its strength in capturing spatial dependencies, whereas ResNet faces constraints in this area. These findings suggest that understanding the internal mechanisms of different models can improve the reliability of data-driven predictions and support more efficient simulation workflows.

However, as explored in this work, feature analysis using LRP revealed certain limitations. While LRP helped highlight the regions in variable fields that contribute significantly to reconstruction accuracy, it becomes clear that the discovered dependencies are not particularly informative for discerning within- and between-variable relationships. These observations signal the need for more robust and interpretable approaches to measure dependencies between variables, leading to the expanded focus on statistical dependencies in subsequent studies.

Building on the findings from Paper A [FHN⁺23a], Paper B [FHN⁺23b] introduces neural dependence fields (NDFs) as an advanced approach to analyze and visualize statistical dependencies, not only within individual variables but also between pairs of variables

across large 3D ensemble fields. The novelty of the NDF approach lies in its ability to infer both linear and non-linear dependencies, offering a compact and computationally efficient way to represent and reconstruct dependency structures. By utilizing GPU-accelerated volume rendering, NDFs allow for interactive visual analysis of these dependencies, making the real-time exploration of complex 3D dependence structures possible. Evaluations in Paper B [FHN⁺23b] demonstrate that NDFs effectively capture prominent dependency patterns in large datasets, though some finer details are smoothed out due to the limited capacity of the network. Importantly, Paper B [FHN⁺23b] highlighted the potential for expanding the NDF framework to time-varying fields, enabling deeper investigation into temporal dependencies.

Paper C [FNW24] followed as an extension of Paper B, focusing on applying the concepts of NDFs to time series data. The introduction of time series neural mutual information fields (TNMIFs) in Paper C [FNW24] extended the statistical dependency framework to large collections of time series, where the goal is to efficiently reconstruct MI between time windows across different time series. TNMIFs explored two training methodologies: sync-TNMIF for synchronized time windows across different time series, and lag-TNMIF, which account for lagged dependencies by analyzing a single window in relation to multiple lagged windows from other time series. The results in Paper C demonstrated that TNMIFs are able to effectively encode and reconstruct the principal MI structures, providing a compact representation that enabled rapid computation of time series similarity matrices. However, as with NDFs, limitations in network capacity led to some smoothing of fine details.

In summary, the works presented in Papers A, B, and C illustrate the progression from V2V transfer methods to the development of advanced statistical dependency models. The initial challenges encountered with LRP in Paper A—particularly its inability to capture informative non-linear dependencies between variables—underscored the need for new approaches that are able to better quantify and visualize these relationships. This leads to the introduction of NDFs in Paper B and their subsequent extension to time series analysis in Paper C.

7.2 Future Works

The findings from Paper A, Paper B, and Paper C suggest several promising directions for future research, building on the foundations established in this thesis.

In Paper A [FHN⁺23a], the focus on V2V transfer presents opportunities for further refinement, particularly in terms of extending the method to handle three-dimensional datasets and large ensemble forecasts. One of the key limitations identified in the current work is the relatively low mutual similarity between variable fields. Future research could explore how the V2V approach performs when applied to variable fields with stronger inter-variable relationships, such as those found in ensemble simulations. By using V2V to reconstruct a minimal subset of ensemble members that captures the full

ensemble spread, it may be possible to identify weather situations that are either under- or over-represented in the ensemble, which is highly valuable for operational meteorological applications. Additionally, integrating the temporal dimension into the analysis could further enhance the ability of the method to reconstruct data at specific time steps by incorporating information from previous time steps. This temporal aspect is particularly relevant for weather forecasting and other time-sensitive fields, where the evolution of variables over time is crucial. Another important direction for future work, as discussed in Paper A [FHN⁺23a], involves improving the LRP-based sensitivity analysis. While LRP helped identify regions of high relevance for reconstruction accuracy, selecting appropriate reference locations for point-to-field relevance mapping can be challenging. To address this, automated methods for detecting regions of high impact could be developed, allowing for more global field-to-field relevance analysis. Incorporating temporal coherence and recurrence into the analysis helps to reduce noise in the LRP maps and provides a more robust understanding of the dependencies between variables.

In Paper B, the future development of NDFs includes enhancing the capacity of the network to preserve finer details in the reconstructed dependencies. This could involve adding more stages to the encoder and decoder or experimenting with alternative architectures, such as diffusion networks, to improve the ability of the network to capture complex relationships. Additionally, the use of different loss functions, such as gradient regularization, may help to preserve finer spatial details that are currently smoothed out due to the limitations of the network. Extending NDFs to time-varying ensemble fields is another key direction, as this allows for the modeling of temporal dependencies, enabling new applications such as ensemble sensitivity analysis. The ability to decompose ensemble data into independent fields based on spatial and temporal variations could provide valuable insights for scientific workflows and real-time decision-making.

For Paper C, one key area for future research is enhancing the network to handle larger datasets and more complex time series structures by exploring deeper network architectures and more advanced training techniques. Integrating TNMIFs into interactive visualization workflows is also a valuable contribution, enabling users to dynamically select specific time windows or temporal features for analysis. Furthermore, the potential to simulate dynamic time warping through TNMIFs represents a promising direction for future work. By training the network on window pairs of varying lengths, it may be possible to estimate the similarity between time series that vary in speed. This could be achieved by embedding time-warping-based feature extraction within the TNMIF framework, allowing for more nuanced comparisons between time series data. Additional research into the relationships between time warping and MI variation will be necessary to fully realize this potential.

BIBLIOGRAPHY

- [ABC⁺16] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. {TensorFlow}: a system for {Large-Scale} machine learning. In *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, pages 265–283, 2016.
- [AMMS17] Leila Arras, Grégoire Montavon, Klaus-Robert Müller, and Wojciech Samek. Explaining recurrent neural network predictions in sentiment analysis. *arXiv preprint arXiv:1706.07206*, 2017.
- [ANB⁺20] Ramoza Ahsan, Rodica Neamtu, Muzammil Bashir, Elke A Rundensteiner, and Gabor Sarkozy. Correlation-based analytics of time series data. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 4482–4491. IEEE, 2020.
- [BBM⁺15] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140, 2015.
- [BBR⁺18] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. Mutual information neural estimation. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 531–540. PMLR, 10–15 Jul 2018.
- [BDSW13] Ayan Biswas, Soumya Dutta, Han-Wei Shen, and Jonathan Woodring. An information-aware framework for exploring multivariate data sets. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2683–2692, 2013.
- [BHNS16] Richard H Byrd, Samantha L Hansen, Jorge Nocedal, and Yoram Singer. A stochastic quasi-newton method for large-scale optimization. *SIAM Journal on Optimization*, 26(2):1008–1031, 2016.
- [Bis06] Christopher M Bishop. Pattern recognition and machine learning. *Springer google schola*, 2:1122–1128, 2006.

- [BJRL15] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.
- [BKC17] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495, 2017.
- [BML⁺16] Alexander Binder, Grégoire Montavon, Sebastian Lapuschkin, Klaus-Robert Müller, and Wojciech Samek. Layer-wise relevance propagation for neural networks with local renormalization layers. In *Artificial Neural Networks and Machine Learning–ICANN 2016: 25th International Conference on Artificial Neural Networks, Barcelona, Spain, September 6–9, 2016, Proceedings, Part II 25*, pages 63–71. Springer, 2016.
- [BMLC19] Marzieh Berenjkoub, Rodolfo Ostilla Monico, Robert S. Laramée, and Guoning Chen. Visual analysis of spatio-temporal relations of pairwise attributes in unsteady flow. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):1246–1256, 2019.
- [BTB15] Peter Bauer, Alan Thorpe, and Gilbert Brunet. The quiet revolution of numerical weather prediction. *Nature*, 525(7567):47–55, 2015.
- [CFD17] Luis M Candanedo, Véronique Feldheim, and Dominique Deramaix. Data driven prediction models of energy use of appliances in a low-energy house. *Energy and buildings*, 140:81–97, 2017.
- [CHC⁺09] Israel Cohen, Yiteng Huang, Jingdong Chen, Jacob Benesty, Jacob Benesty, Jingdong Chen, Yiteng Huang, and Israel Cohen. Pearson correlation coefficient. *Noise reduction in speech processing*, pages 1–4, 2009.
- [CKS⁺20] Jianxin Cheng, Qiuming Kuang, Chenkai Shen, Jin Liu, Xicheng Tan, and Wang Liu. Reslap: Generating high-resolution climate prediction through image super-resolution. *IEEE Access*, 8:39623–39634, 2020.
- [CLI⁺20] Rohan Chhabra, Jan E Lenssen, Eddy Ilg, Tanner Schmidt, Julian Straub, Steven Lovegrove, and Richard Newcombe. Deep local shapes: Learning local sdf priors for detailed 3d reconstruction. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIX 16*, pages 608–625. Springer, 2020.
- [CT⁺91] Thomas M Cover, Joy A Thomas, et al. Entropy, relative entropy and mutual information. *Elements of information theory*, 2(1):12–13, 1991.
- [CWK⁺12] Adam J Clark, Steven J Weiss, John S Kain, Israel L Jirak, Michael Coniglio, Christopher J Melick, Christopher Siewert, Ryan A Sobash, Patrick T Marsh, Andrew R Dean, et al. An overview of the 2010 hazardous weather

- testbed experimental forecast program spring experiment. *Bulletin of the American Meteorological Society*, 93(1):55–74, 2012.
- [CWMW11] Cheng-Kai Chen, Chaoli Wang, Kwan-Liu Ma, and Andrew T. Wittenberg. Static correlation visualization for large time-varying volume data. In *2011 IEEE Pacific Visualization Symposium*, pages 27–34, 2011.
- [CZ19] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5939–5948, 2019.
- [DBK⁺19] Hoang Anh Dau, Anthony Bagnall, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, and Eamonn Keogh. The ucr time series archive. *IEEE/CAA Journal of Automatica Sinica*, 6(6):1293–1305, 2019.
- [DG08] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, 2008.
- [DHS11] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.
- [DV16] Vincent Dumoulin and Francesco Visin. A guide to convolution arithmetic for deep learning. *arXiv preprint arXiv:1603.07285*, 2016.
- [EBL23] Marina Evers, Michael Böttinger, and Lars Linsen. Interactive Visual Analysis of Regional Time Series Correlation in Multi-field Climate Ensembles. In Soumya Dutta, Kathrin Feige, Karsten Rink, and Dirk Zeckzer, editors, *Workshop on Visualisation in Environmental Sciences (EnvirVis)*. The Eurographics Association, 2023.
- [EHL21] Marina Evers, Karim Huesmann, and Lars Linsen. Uncertainty-aware Visualization of Regional Time Series Correlation in Spatio-temporal Ensembles. *Computer Graphics Forum*, 2021.
- [ESL23a] Arik Ermshaus, Patrick Schäfer, and Ulf Leser. Clasp: parameter-free time series segmentation. *Data Mining and Knowledge Discovery*, 2023.
- [ESL23b] Arik Ermshaus, Patrick Schäfer, and Ulf Leser. Window size selection in unsupervised time series analytics: A review and benchmark. In *International Workshop on Advanced Analytics and Learning on Temporal Data*, pages 83–101. Springer, 2023.
- [FHN⁺23a] Fatemeh Farokhmanesh, Kevin Höhle, Tobias Necker, Martin Weissmann, Takemasa Miyoshi, and Rüdiger Westermann. Deep learning-based parameter transfer in meteorological data. *Artificial Intelligence for the Earth Systems*, 2(1):e220024, 2023.

- [FHN⁺23b] Fatemeh Farokhmanesh, Kevin Höhle, Christoph Neuhauser, Tobias Necker, Martin Weissmann, Takemasa Miyoshi, and Rüdiger Westermann. Neural Fields for Interactive Visualization of Statistical Dependencies in 3D Simulation Ensembles. In Michael Guthe and Thorsten Grosch, editors, *Vision, Modeling, and Visualization*. The Eurographics Association, 2023.
- [FJ17] Ben D Fulcher and Nick S Jones. hctsa: A computational framework for automated time-series phenotyping using massive feature extraction. *Cell systems*, 5(5):527–531, 2017.
- [FKMW⁺23] Sara Fridovich-Keil, Giacomo Meanti, Frederik Warburg, Benjamin Recht, and Angjoo Kanazawa. K-planes: Explicit radiance fields in space, time, and appearance. *arXiv preprint arXiv:2301.10241*, 2023.
- [FNW24] Fatemeh Farokhmanesh, Christoph Neuhauser, and Rüdiger Westermann. A compact and efficient neural data structure for mutual information estimation in large timeseries. In *Proceedings of the 36th International Conference on Scientific and Statistical Database Management, SSDBM '24*, New York, NY, USA, 2024. Association for Computing Machinery.
- [Fuk80] Kunihiro Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 36(4):193–202, 1980.
- [GAG⁺00] Ary L Goldberger, Luis AN Amaral, Leon Glass, Jeffrey M Hausdorff, Plamen Ch Ivanov, Roger G Mark, Joseph E Mietus, George B Moody, Chung-Kang Peng, and H Eugene Stanley. Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals. *circulation*, 101(23):e215–e220, 2000.
- [GHGM06] Markus Glatte, Jian Huang, Jinzhu Gao, and Colin Mollenhour. Scalable data servers for large multivariate volume visualization. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1291–1298, 2006.
- [Gir15] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015.
- [Goo16] Ian Goodfellow. Deep learning, 2016.
- [GTPB11] C Gebhardt, SE Theis, M Paulat, and Z Ben Bouallègue. Uncertainties in cosmo-de precipitation forecasts introduced by model perturbations and variation of lateral boundaries. *Atmospheric Research*, 100(2-3):168–177, 2011.
- [GW10] Yi Gu and Chaoli Wang. A study of hierarchical correlation clustering for scientific volume data. In *Advances in Visual Computing: 6th International Symposium, ISVC 2010, 1, 2010, Proceedings, Part III 6*, pages 437–446. Springer, 2010.

- [GYH⁺20] Li Guo, Shaojie Ye, Jun Han, Hao Zheng, Han Gao, Danny Z Chen, Jian-Xun Wang, and Chaoli Wang. SSR-VFD: Spatial super-resolution for vector field data analysis and visualization. In *2020 IEEE Pacific Visualization Symposium (PacificVis)*, pages 71–80. IEEE Computer Society, 2020.
- [HBB⁺15] Carolin Helbig, Lars Bilke, Hans-Stefan Bauer, Michael Böttinger, and Olaf Kolditz. Meva-an interactive visualization application for validation of multifaceted meteorological data with multiple 3d devices. *PloS one*, 10(4):e0123811, 2015.
- [HBB⁺20] Hans Hersbach, Bill Bell, Paul Berrisford, Shoji Hirahara, András Horányi, Joaquín Muñoz-Sabater, Julien Nicolas, Carole Peubey, Raluca Radu, Dinand Schepers, et al. The era5 global reanalysis. *Quarterly Journal of the Royal Meteorological Society*, 146(730):1999–2049, 2020.
- [HKHW20] Kevin Höhle, Michael Kern, Timothy Hewson, and Rüdiger Westermann. A comparative study of convolutional neural network models for wind field downscaling. *Meteorological Applications*, 27(6), 2020.
- [HS97] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- [HSB⁺20] Duong Hoang, Brian Summa, Harsh Bhatia, Peter Lindstrom, Pavol Klacanský, Will Usher, Peer-Timo Bremer, and Valerio Pascucci. Efficient and flexible hierarchical data layouts for a unified encoding of scalar field precision and resolution. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):603–613, 2020.
- [HTT⁺09] Tony Hey, Stewart Tansley, Kristin Michele Tolle, et al. *The fourth paradigm: data-intensive scientific discovery*, volume 1. Microsoft research Redmond, WA, 2009.
- [HW19] Jun Han and Chaoli Wang. TSR-TVD: Temporal super-resolution for time-varying data analysis and visualization. *IEEE transactions on visualization and computer graphics*, 26(1):205–215, 2019.
- [HW20] Jun Han and Chaoli Wang. SSR-TVD: Spatial super-resolution for time-varying data analysis and visualization. *IEEE Transactions on Visualization and Computer Graphics*, 2020.
- [HWW22] Kevin Höhle, Sebastian Weiss, and Rüdiger Westermann. Evaluation of volume representation networks for meteorological ensemble compression. 2022.
- [HZCW21] Jun Han, Hao Zheng, Danny Z. Chen, and Chaoli Wang. STNet: An end-to-end generative framework for synthesizing spatiotemporal super-resolution volumes. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–1, 2021.

- [HZRS16a] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [HZRS16b] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *Computer Vision—ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*, pages 630–645. Springer, 2016.
- [HZX⁺20] Jun Han, Hao Zheng, Yunhao Xing, Danny Z Chen, and Chaoli Wang. V2v: A deep learning approach to variable-to-variable selection and translation for multivariate time-varying data. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):1290–1300, 2020.
- [IFFW⁺19] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep learning for time series classification: a review. *Data mining and knowledge discovery*, 33(4):917–963, 2019.
- [IFLF⁺20] Hassan Ismail Fawaz, Benjamin Lucas, Germain Forestier, Charlotte Pelletier, Daniel F Schmidt, Jonathan Weber, Geoffrey I Webb, Lhassane Idoumghar, Pierre-Alain Muller, and François Petitjean. Inceptiontime: Finding alexnet for time series classification. *Data Mining and Knowledge Discovery*, 34(6):1936–1962, 2020.
- [JKRL09a] Kevin Jarrett, Koray Kavukcuoglu, Marc’Aurelio Ranzato, and Yann LeCun. What is the best multi-stage architecture for object recognition? In *2009 IEEE 12th international conference on computer vision*, pages 2146–2153. IEEE, 2009.
- [JKRL09b] Kevin Jarrett, Koray Kavukcuoglu, Marc’Aurelio Ranzato, and Yann LeCun. What is the best multi-stage architecture for object recognition? In *2009 IEEE 12th international conference on computer vision*, pages 2146–2153. IEEE, 2009.
- [KB14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [Ken38] Maurice G Kendall. A new measure of rank correlation. *Biometrika*, 30(1-2):81–93, 1938.
- [KSG04] Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. Estimating mutual information. *Physical Review E—Statistical, Nonlinear, and Soft Matter Physics*, 69(6):066138, 2004.
- [KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

- [KWS⁺15] Patrick Köthur, Carl Witt, Mike Sips, Norbert Marwan, Stefan Schinkel, and Doris Dransch. Visual analytics for correlation-based comparison of time series ensembles. In *Computer Graphics Forum*, volume 34, pages 411–420. Wiley Online Library, 2015.
- [LAF15] Nikolay Laptev, Saeed Amizadeh, and Ian Flint. Generic and scalable framework for automated time-series anomaly detection. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1939–1947, 2015.
- [LBD⁺89] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.
- [LCJB⁺90] Yann Le Cun, Lawrence D Jackel, Brian Boser, John S Denker, Hans Peter Graf, Isabelle Guyon, Don Henderson, Richard E Howard, and William Hubbard. Handwritten digit recognition: Applications of neural net chips and automatic learning. In *Neurocomputing: Algorithms, Architectures and Applications*, pages 303–318. Springer, 1990.
- [LGRN09] Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th annual international conference on machine learning*, pages 609–616, 2009.
- [LJLB21] Yuzhe Lu, Kairong Jiang, Joshua A Levine, and Matthew Berger. Compressive neural representations of volumetric scalar fields. In *Computer Graphics Forum*, volume 40, pages 135–146. Wiley Online Library, 2021.
- [LP08] Martin Leutbecher and Tim N Palmer. Ensemble forecasting. *Journal of computational physics*, 227(7):3515–3539, 2008.
- [LS16] Xiaotong Liu and Han-Wei Shen. Association analysis for visual exploration of multivariate scientific data sets. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):955–964, 2016.
- [LSK⁺19] Carl H Lubba, Sarab S Sethi, Philip Knaute, Simon R Schultz, Ben D Fulcher, and Nick S Jones. catch22: Canonical time-series characteristics: Selected through highly comparative time-series analysis. *Data Mining and Knowledge Discovery*, 33(6):1821–1852, 2019.
- [LUYG13] Yuhong Li, Leong Hou U, Man Lung Yiu, and Zhiguo Gong. Discovering longest-lasting correlation in sequence databases. *Proceedings of the VLDB Endowment*, 6(14):1666–1677, 2013.
- [LWS18] Tom Liebmann, Gunther H. Weber, and Gerik Scheuermann. Hierarchical correlation clustering in multiple 2d scalar fields. *Computer Graphics Forum*, 37(3):1–12, 2018.

- [Mar15] Bernard Marr. *Big Data: Using SMART big data, analytics and metrics to make better decisions and improve performance*. John Wiley & Sons, 2015.
- [MBL⁺19] Grégoire Montavon, Alexander Binder, Sebastian Lapuschkin, Wojciech Samek, and Klaus-Robert Müller. Layer-wise relevance propagation: an overview. *Explainable AI: interpreting, explaining and visualizing deep learning*, pages 193–209, 2019.
- [MESK22] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4), jul 2022.
- [MH00] Spyros Makridakis and Michele Hibon. The m3-competition: results, conclusions and implications. *International journal of forecasting*, 16(4):451–476, 2000.
- [MHN⁺13] Andrew L Maas, Awni Y Hannun, Andrew Y Ng, et al. Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, page 3. Atlanta, GA, 2013.
- [MLB⁺17] Grégoire Montavon, Sebastian Lapuschkin, Alexander Binder, Wojciech Samek, and Klaus-Robert Müller. Explaining nonlinear classification decisions with deep taylor decomposition. *Pattern recognition*, 65:211–222, 2017.
- [MLL⁺21] Julien NP Martel, David B Lindell, Connor Z Lin, Eric R Chan, Marco Monteiro, and Gordon Wetzstein. Acorn: Adaptive coordinate networks for neural scene representation. *arXiv preprint arXiv:2105.02788*, 2021.
- [MNA16] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 fourth international conference on 3D vision (3DV)*, pages 565–571. Ieee, 2016.
- [MNL10] Abdullah Mueen, Suman Nath, and Jie Liu. Fast approximate correlation for massive time-series data. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, pages 171–182, 2010.
- [MON⁺19] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4460–4470, 2019.
- [MRL95] Young-Il Moon, Balaji Rajagopalan, and Upmanu Lall. Estimation of mutual information using kernel density estimators. *Physical Review E*, 52(3):2318, 1995.

- [MRNK21a] Thomas Müller, Fabrice Rousselle, Jan Novák, and Alexander Keller. Real-time neural radiance caching for path tracing. *arXiv preprint arXiv:2106.12372*, 2021.
- [MRNK21b] Thomas Müller, Fabrice Rousselle, Jan Novák, and Alexander Keller. Real-time neural radiance caching for path tracing. *ACM Trans. Graph.*, 40(4), jul 2021.
- [MSA18] Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. The m4 competition: Results, findings, conclusion and way forward. *International Journal of forecasting*, 34(4):802–808, 2018.
- [MSB23] Matthew Middlehurst, Patrick Schäfer, and Anthony Bagnall. Bake off redux: a review and experimental evaluation of recent time series classification algorithms. *arXiv preprint arXiv:2304.13029*, 2023.
- [MSC⁺13] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013.
- [MST⁺21] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.
- [Mü21] Thomas Müller. *tiny-cuda-nn*, 4 2021.
- [NGW⁺20] Tobias Necker, Stefan Geiss, Martin Weissmann, Juan Ruiz, Takemasa Miyoshi, and Guo-Yuan Lien. A convective-scale 1,000-member ensemble simulation and potential applications. *Quarterly Journal of the Royal Meteorological Society*, 146(728):1423–1442, 2020.
- [NH10] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.
- [NW99] Jorge Nocedal and Stephen J Wright. *Numerical optimization*. Springer, 1999.
- [PFS⁺19] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 165–174, 2019.
- [PG04] Raoul Pietersz 4 and Patrick JF Groenen. Rank reduction of correlation matrices by majorization. *Quantitative Finance*, 4(6):649–662, 2004.

- [PGM⁺19] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.
- [PLPD18] Darren Pouliot, Rasim Latifovic, Jon Pasher, and Jason Duffe. Landsat super-resolution enhancement using convolution neural networks and sentinel-2 for training. *Remote Sensing*, 10(3):394, 2018.
- [PSM14] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.
- [PW12] T. Pfaffelmoser and R. Westermann. Visualization of global correlation structures in uncertain 2d scalar fields. In *Computer Graphics Forum*, volume 31, pages 1025–1034. Wiley Online Library, 2012.
- [QCX⁺07] Huamin Qu, Wing-Yi Chan, Anbang Xu, Kai-Lun Chung, Kai-Hon Lau, and Ping Guo. Visual analysis of the air pollution problem in hong kong. *IEEE Transactions on visualization and Computer Graphics*, 13(6):1408–1415, 2007.
- [RDS⁺15] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252, 2015.
- [RDS⁺20] Stephan Rasp, Peter D Dueben, Sebastian Scher, Jonathan A Weyn, Soukayna Mouatadid, and Nils Thuerey. Weatherbench: a benchmark data set for data-driven weather forecasting. *Journal of Advances in Modeling Earth Systems*, 12(11):e2020MS002203, 2020.
- [RE19] Niklas Röber and Jan Frederik Engels. In-situ processing in climate science. In *International Conference on High Performance Computing*, pages 612–622. Springer, 2019.
- [RFB15] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.
- [Ric00] David S Richardson. Skill and relative economic value of the ecmwf ensemble prediction system. *Quarterly Journal of the Royal Meteorological Society*, 126(563):649–667, 2000.

- [ROCN18] Eduardo Rocha Rodrigues, Igor Oliveira, Renato Cunha, and Marco Netto. Deepdownscale: a deep learning strategy for high-resolution weather forecast. In *2018 IEEE 14th International Conference on e-Science (e-Science)*, pages 415–422. IEEE, 2018.
- [RPBA09] Umaa Rebbapragada, Pavlos Protopapas, Carla E Brodley, and Charles Alcock. Finding anomalous periodic time series: An application to catalogs of periodic variable stars. *Machine learning*, 74:281–313, 2009.
- [SGB21] Agon Serifi, Tobias Günther, and Nikolina Ban. Spatio-temporal downscaling of climate data using convolutional and error-predicting neural networks. *Frontiers in Climate*, 3, 04 2021.
- [SMDH13] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147. PMLR, 2013.
- [SSS00] Robert H Shumway, David S Stoffer, and David S Stoffer. *Time series analysis and its applications*, volume 3. Springer, 2000.
- [Sti89] Stephen M Stigler. Francis galton’s account of the invention of correlation. *Statistical Science*, pages 73–79, 1989.
- [STK19] Takuto Sato, Osamu Tatebe, and Hiroyuki Kusaka. In-situ data analysis system for high resolution meteorological large eddy simulation model. In *Proceedings of the 6th IEEE/ACM International Conference on Big Data Computing, Applications and Technologies*, pages 155–158, 2019.
- [STS06] Natascha Sauber, Holger Theisel, and Hans-Peter Seidel. Multifield-graphs: An approach to visualizing correlations in multifield scalar data. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):917–924, 2006.
- [SWMW09] Jeffrey Sukharev, Chaoli Wang, Kwan-Liu Ma, and Andrew T Wittenberg. Correlation study of time-varying multivariate climate data sets. In *2009 IEEE pacific visualization symposium*, pages 161–168. IEEE, 2009.
- [TFT⁺20] Ayush Tewari, Ohad Fried, Justus Thies, Vincent Sitzmann, Stephen Lombardi, Kalyan Sunkavalli, Ricardo Martin-Brualla, Tomas Simon, Jason Saragih, Matthias Nießner, et al. State of the art on neural rendering. In *Computer Graphics Forum*, volume 39, pages 701–727. Wiley Online Library, 2020.
- [TG20] Erico Tjoa and Cuntai Guan. A survey on explainable artificial intelligence (xai): Toward medical xai. *IEEE transactions on neural networks and learning systems*, 32(11):4793–4813, 2020.

- [TIW⁺18] Jun Tao, Martin Imre, Chaoli Wang, Nitesh V Chawla, Hanqi Guo, Gökhan Sever, and Seung Hyun Kim. Exploring time-varying multivariate volume data using matrix of isosurface similarity maps. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):1236–1245, 2018.
- [TLY⁺21] Towaki Takikawa, Joey Litalien, Kangxue Yin, Karsten Kreis, Charles Loop, Derek Nowrouzezahrai, Alec Jacobson, Morgan McGuire, and Sanja Fidler. Neural geometric level of detail: Real-time rendering with implicit 3d shapes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11358–11367, 2021.
- [TRAW12] Marc Treib, Florian Reichl, Stefan Auer, and Rüdiger Westermann. Interactive editing of gigasample terrain fields. *Computer Graphics Forum (Proc. Eurographics)*, 31(2):383–392, 2012.
- [TSM⁺20] Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in Neural Information Processing Systems*, 33:7537–7547, 2020.
- [TVJ⁺17] George Toderici, Damien Vincent, Nick Johnston, Sung Jin Hwang, David Minnen, Joel Shor, and Michele Covell. Full resolution image compression with recurrent neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 5306–5314, 2017.
- [vdMH08] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008.
- [WAH93] John J Weng, Narendra Ahuja, and Thomas S Huang. Learning recognition and segmentation of 3-d objects from 2-d images. In *1993 (4th) International Conference on Computer Vision*, pages 121–128. IEEE, 1993.
- [WBSS04] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.
- [WCTW19] S. Weiss, M. Chu, N. Thuerey, and R. Westermann. Volumetric isosurface rendering with deep learning-based super-resolution. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–1, 2019.
- [WHW22] Sebastian Weiss, Philipp Hermüller, and Rüdiger Westermann. Fast neural representations for direct volume rendering. In *Computer Graphics Forum*, volume 41, pages 196–211. Wiley Online Library, 2022.
- [WITW20] Sebastian Weiss, Mustafa Işık, Justus Thies, and Rüdiger Westermann. Learning adaptive sampling and reconstruction for volume visualization. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–1, 2020.

- [WYG⁺11a] Chaoli Wang, Hongfeng Yu, Ray W. Grout, Kwan-Liu Ma, and Jacqueline H. Chen. Analyzing information transfer in time-varying multivariate data. In *2011 IEEE Pacific Visualization Symposium*, pages 99–106, 2011.
- [WYG⁺11b] Chaoli Wang, Hongfeng Yu, Ray W Grout, Kwan-Liu Ma, and Jacqueline H Chen. Analyzing information transfer in time-varying multivariate data. In *2011 IEEE Pacific Visualization Symposium*, pages 99–106. IEEE, 2011.
- [WYO17] Zhiguang Wang, Weizhong Yan, and Tim Oates. Time series classification from scratch with deep neural networks: A strong baseline. In *2017 International joint conference on neural networks (IJCNN)*, pages 1578–1585. IEEE, 2017.
- [XTS⁺22] Yiheng Xie, Towaki Takikawa, Shunsuke Saito, Or Litany, Shiqin Yan, Numair Khan, Federico Tombari, James Tompkin, Vincent Sitzmann, and Srinath Sridhar. Neural fields in visual computing and beyond. In *Computer Graphics Forum*, volume 41, pages 641–676. Wiley Online Library, 2022.
- [Zar05] Jerrold H Zar. Spearman rank correlation. *Encyclopedia of biostatistics*, 7, 2005.
- [ZD11] X Zeng and TS Durrani. Estimation of mutual information using copula density function. *Electronics letters*, 47(8):493–494, 2011.
- [Zei12] Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.
- [ZHU20] Liu Ziyin, Tilman Hartwig, and Masahito Ueda. Neural networks fail to learn periodic functions and how to fix it. *Advances in Neural Information Processing Systems*, 33:1583–1594, 2020.
- [ZHW⁺17] Zhenglei Zhou, Yule Hou, Qirui Wang, Guangxiang Chen, Jiawei Lu, Yubo Tao, and Hai Lin. Volume upscaling with convolutional neural networks. In *Proceedings of the Computer Graphics International Conference*, pages 1–6, 2017.
- [ZIE⁺18] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.
- [ZMZM14] Zhiyuan Zhang, Kevin T McDonnell, Erez Zadok, and Klaus Mueller. Visual correlation analysis of numerical and categorical data on the correlation map. *IEEE transactions on visualization and computer graphics*, 21(2):289–303, 2014.

Deep Learning–Based Parameter Transfer in Meteorological Data

FATEMEH FAROKHMANESH,^a KEVIN HÖHLEIN,^a TOBIAS NECKER,^b MARTIN WEISSMANN,^b
TAKEMASA MIYOSHI,^c AND RÜDIGER WESTERMANN^a

^a *Department of Informatics, Technical University of Munich, Munich, Germany*

^b *Institut für Meteorologie und Geophysik, Universität Wien, Vienna, Austria*

^c *RIKEN Center for Computational Science, Kobe, Japan*

(Manuscript received 24 March 2022, in final form 13 October 2022)

ABSTRACT: Numerical simulations in Earth-system sciences consider a multitude of physical parameters in space and time, leading to severe input/output (I/O) bandwidth requirements and challenges in subsequent data analysis tasks. Deep learning–based identification of redundant parameters and prediction of those from other parameters, that is, variable-to-variable (V2V) transfer, has been proposed as an approach to lessening the bandwidth requirements and streamlining subsequent data analysis. In this paper, we examine the applicability of V2V to meteorological reanalysis data. We find that redundancies within pairs of parameter fields are limited, which hinders application of the original V2V algorithm. Therefore, we assess the predictive strength of reanalysis parameters by analyzing the learning behavior of V2V reconstruction networks in an ablation study. We demonstrate that efficient V2V transfer becomes possible when considering groups of parameter fields for transfer and propose an algorithm to implement this. We investigate further whether the neural networks trained in the V2V process can yield insightful representations of recurring patterns in the data. The interpretability of these representations is assessed via layerwise relevance propagation that highlights field areas and parameters of high importance for the reconstruction model. Applied to reanalysis data, this allows for uncovering mutual relationships between landscape orography and different regional weather situations. We see our approach as an effective means to reduce bandwidth requirements in numerical weather simulations, which can be used on top of conventional data compression schemes. The proposed identification of multiparameter features can spawn further research on the importance of regional weather situations for parameter prediction and also in other kinds of simulation data.

KEYWORDS: Pattern detection; Data science; Deep learning; Model interpretation and visualization; Neural networks

1. Introduction

The rapid increase in available computing power has enabled a broad adoption of simulation-based research methodologies in Earth-system sciences. Numerical simulations of spatiotemporal dynamical systems consider a multitude of physical parameters and are carried out at high resolution in space and time. To account for the uncertainty in the representation of certain physical processes, in meteorology and climate modeling numerical ensemble simulations are carried out with varying magnitudes of initial condition uncertainty. Simulations are performed routinely by weather centers worldwide, and in research we see increasing use of unique superensembles consisting of hundreds and even thousands of members (Necker et al. 2020).

In classical workflow scenarios, simulations are run on large-scale computing facilities and data are streamed to and stored on external file systems for archiving and subsequent analysis. However, the volume of generated data has reached an order of magnitude where the speed of data transfer between computing device and file system—so-called input/output (I/O) operations—imposes a major bottleneck. For instance, over the last decade the ability to compute increased about two orders of magnitude on supercomputers, while the ability to store and load data only increased about one order of magnitude.

The divergence between computing and I/O renders the classical simulation workflow increasingly problematic and

requires us to avoid streaming or even simulating data that can be recovered from the generated results. When using data compression, this can significantly reduce the time it requires to bring the data to the compression stage (e.g., when using distributed memory architectures) and perform the compression.

Within this line of research, deep learning–based variable-to-variable (V2V) transfer has been proposed recently by Han et al. (2021) for optimizing information transfer in situations where spatiotemporal multiparameter simulations can be carried out in far less time than it requires to store the data on a file system. V2V considers each simulated parameter as a separate entity and proposes an algorithm to identify groups of similar parameters and one representative member from which the other parameters in this group can be inferred.

V2V represents all simulated parameter fields in a common feature space (the so-called latent space) that is learnt by a convolutional neural network (CNN) and identifies subsets of similar parameters in this space. For each subset, the most representative member is determined, and another suitably trained network then learns to reconstruct all other member in one subset from the representative one. The bandwidth requirements for storing the multiparameter data are reduced to the bandwidth required for storing the representative parameter fields and the weight parameterization of the reconstruction networks.

Corresponding author: Fatemeh Farokhmanesh, fatemeh.farokhmanesh@tum.de

Publisher's Note: The article was revised on 10 July 2024 to include additional authors and updated Acknowledgments and Data Availability Statement sections; see the associated corrigendum.

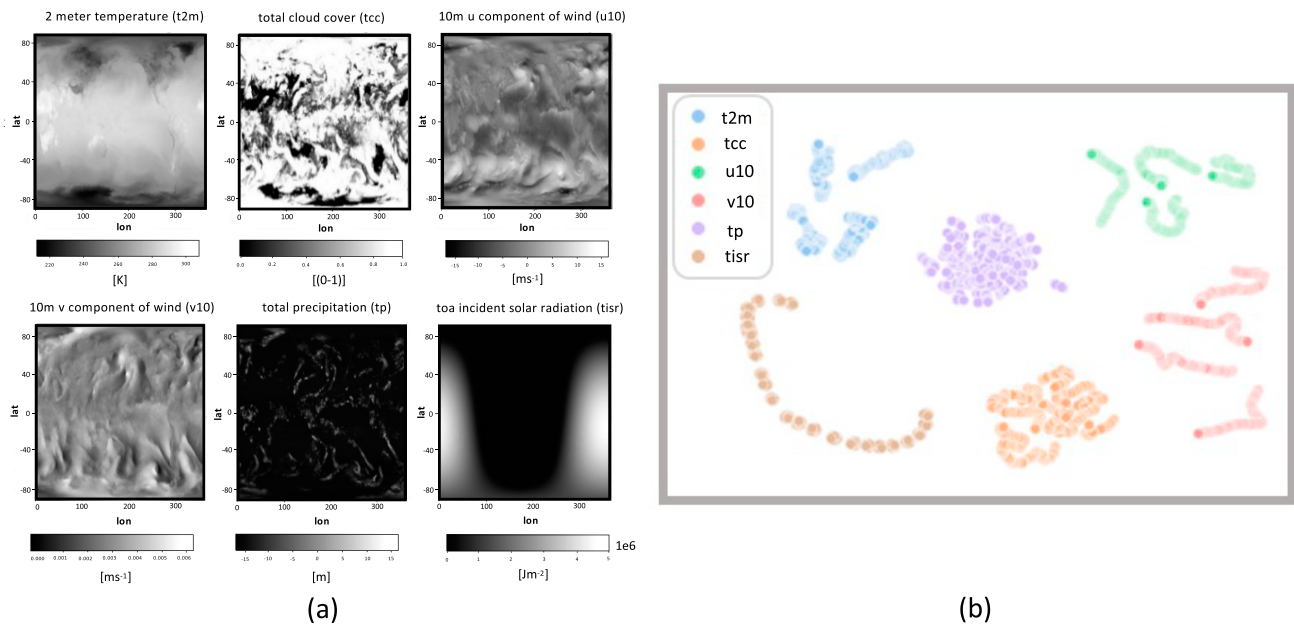


FIG. 1. Different parameter fields in the ERA5 reanalysis dataset. (a) Grayscale visualizations of the parameter fields at a particular time. (b) t-SNE projections of latent-space features of the parameter fields (different parameters indicated by colors) at different times (note that projections for different initializations of t-SNE yield similar groupings).

Contribution

In this work, we assess the applicability of V2V transfer to Earth-system related data and identify shortcomings of the proposed methodology. In consideration of these findings, we propose an improved approach, in which a single CNN is trained on meteorological archives to learn general relationships between subsets of parameters, thereby focusing on subsets of parameter fields with vastly different characteristics and variation in 2D space and time. We demonstrate the capabilities of the proposed approach using two exemplary datasets, representing different modalities of meteorological data. We consider a global reanalysis dataset, which is taken from the WeatherBench (WB) benchmark suite (Rasp et al. 2020), and study the performance of the proposed approach on data on large spatial scales. Furthermore, we apply the proposed method to an ensemble forecast dataset, which was generated by Necker et al. (2020) to study the sampling accuracy of spatiotemporal correlation patterns in convective-scale forecast ensembles. Figure 1, as well as Fig. B1, demonstrate significant structural variability between the single-parameter fields in each dataset. Figure 1a shows snapshots of the parameter fields at a particular point in time. Figure 1b shows a two-dimensional embedding [computed with t -distributed stochastic neighbor embedding (t-SNE); van der Maaten and Hinton 2008] of the V2V latent-space representations of these fields at different time points, which are used to search for parameter similarities. Fig. B1 displays the same overview for the convective-scale ensemble (CSEns) dataset. It can be seen that visible clusters involve only a single parameter, and clusters of distinct parameters are situated at roughly the same distance from one another. The lack of similarities between pairs of parameters prohibits the use of the original V2V algorithm.

Based on these observations, we propose a different strategy for V2V transfer, which considers the expressiveness of

subsets of multiple parameters instead of transferring only between pairs of them. To identify these subsets, a CNN-based model architecture is trained multiple times on multiparameter fields with varying parameter subsets removed from the input data. In an ablation study, we then shed light on the prediction skills of the different models, depending on the parameter subsets that are used as source. For $n \in \mathbb{N}$ originally available parameter fields, the networks are designed to learn mappings from m ($< n$) parameter fields (the input) to predict the remaining $n - m$ ones (the output). Doing so, the networks encode the inputs into a compact latent-space representation, in which relationships between the m input fields and the $n - m$ output fields are encoded. The networks are trained via standard backpropagation with a loss function, which measures the reconstruction accuracy for all $n - m$ parameters in common. We demonstrate, using numerical and visualization-based quality metrics, that the networks efficiently learn to reconstruct the unseen parameters, thereby not overfitting to the provided training samples but generalizing to simulation snapshots that have not been seen before.

Due to the high computational complexity of training all $\binom{n}{m}$ different networks for reconstructing $n - m$ fields from m given fields, we propose a computationally less involved strategy and demonstrate its effectiveness for selecting the most representative members. Conceptually, this strategy builds upon removing iteratively those parameters that are most difficult to predict from the remaining parameters, simultaneously avoiding keeping redundant fields in the input. Furthermore, the networks' training behaviors are monitored and the convergence rates at early training stages after few epochs of learning are used as indicators of the difficulty of parameter transfer. For instance, for one of our use cases comprising nine different parameters and selecting four of them to predict the

remaining five parameters, training requires only roughly six hours on a low-size deep learning cluster with six midsize GPUs, equipped with 11 GB of graphics memory, each.

Beyond considering the networks purely as black-box models, we further try to gain insight into the multiparameter relationships that are learned by the models. For this purpose, we employ an adapted version of layerwise relevance propagation (LRP; [Bach et al. 2015](#)), a method for highlighting input areas and parameters, which are important for the reasoning process of the models. This offers the opportunity to uncover feature patterns in multiparameter space, that is, reoccurring parameter combinations, which are recognized as important for the network to achieve high accuracy and analyze their correspondence to certain weather situations.

The remainder of this paper is structured as follows: In [section 2](#), we review related work. In [section 2b](#), we summarize the original V2V algorithm and highlight algorithmic shortcomings. Building up on these findings, we present our extended V2V approach in [section 4](#). We introduce the example datasets used for subsequent experiments in [section 3](#) and describe the network architectures used for the experiments in [section 4b](#). The ablation study, as well as the LRP analysis of the models, are carried out in [section 5](#). We conclude the paper in [section 6](#).

2. Related work

In recent years, machine learning with powerful deep learning architectures has found applications in various fields of climate science and meteorology ([Reichstein et al. 2019](#)). Many of the possible applications exploit the efficiency and flexibility of CNN architectures when applied to inference tasks involving grid-structured data.

a. Superresolution and downscaling techniques

Related to our approach are so-called superresolution and downscaling techniques, which reconstruct high-resolution parameter fields from corresponding low-resolution versions. In contrast to V2V approaches, the information transfer occurs between representations of the same parameter with different spatial resolutions. Some of these approaches can be used for data compression, in principle. Such methods operate by first subsampling the parameter fields and subsequently reconstructing the initial fields from the subsampled versions. For example, [Rodrigues et al. \(2018\)](#) proposed a supervised convolutional neural network that interpolates low-resolution weather data into a high-resolution output. [Pouliot et al. \(2018\)](#) introduced deep learning-based enhancement in Landsat superresolution. [Cheng et al. \(2020\)](#) proposed a method that converts low-resolution climate data to high-resolution climate forecasts using Laplacian pyramid superresolution networks. Downscaling approaches, in contrast, aim at predicting additional high-resolution details from low-resolution parameter fields, without assuming prior knowledge of the original high-resolution data. For instance, [Höhlein et al. \(2020\)](#) and [Serifi et al. \(2021\)](#) train on a small set of paired low- and high-resolution simulation pairs to circumvent generating expensive high-resolution simulations at inference time at all. These techniques, even though they establish

relationships between low- and high-resolution fields, are not motivated by the idea to compress the data.

Super-resolution of scientific data has also been investigated from the perspective of scientific visualization, since high-resolution simulations in meteorology make analysis of these datasets challenging. Early works on data superresolution demonstrate the capabilities of neural networks to learn upscaling a low-resolution version of the data to the initial high-resolution dataset. Upscaling is performed in the spatial domain ([Zhou et al. 2017](#); [Han and Wang 2022](#); [Guo et al. 2020](#)), in the temporal domain ([Han and Wang 2020](#)), and in the spatiotemporal domain ([Han et al. 2022](#)). Underlying these works is the goal to avoid storing the high-resolution datasets and, thus, reduce bandwidth and memory requirements. By training networks to infer the full image from a low-resolution image of an iso-surface, [Weiss et al. \(2019\)](#) demonstrate improved rendering frame rates. In recent work by [Weiss et al. \(2020\)](#) a convolutional neural network learns to adaptively place image samples and reconstruct the full image from the generated unstructured set of samples.

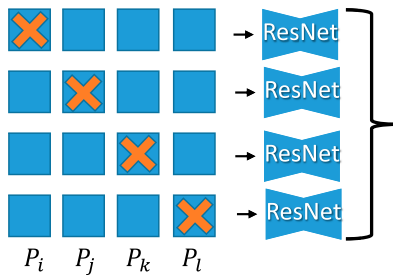
To work in situations with severe I/O limitations, [Sato et al. \(2019\)](#) introduced a so-called in situ approach for visualizing and postprocessing high-resolution meteorological data. In situ approaches process the data instantly when it is produced by the simulation, without involving storage resources. [Röber and Engels \(2019\)](#) analyzed in situ data processing approaches in climate science. [Helbig et al. \(2015\)](#) proposed a visualization workflow where the first stage is a data abstraction layer that downsamples the data spatially and temporally. [Toderici et al. \(2017\)](#) proposed an image compression method using a recurrent neural network by saving the compressed latent space produced by the network instead of the high-resolution data.

A different approach for data compression has been introduced by [Han et al. \(2021\)](#) for multiparameter data, by training networks to infer certain parameters from others, and, thus, to avoid storing these parameters. Conceptually, this work builds upon the notion of information transfer between scalar fields ([Wang et al. 2011](#)) to derive transferable parameter pairs.

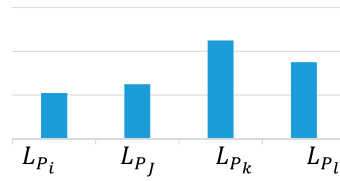
b. V2V transfer

The overall goal of V2V lies in identifying those variables in a multiparameter dataset that can be redundantly reconstructed given other parameters from the same dataset. [Han et al. \(2021\)](#) subdivide the V2V process into 3 conceptually distinct stages: Feature learning, translation graph construction and variable translation. First, a CNN model architecture such as UNet ([Ronneberger et al. 2015](#)) is trained in an autoencoder-like setting to encode and reconstruct snapshots of parameter fields. The same model is shared between all parameters, such that the internal hidden variables of the model, referred to as the latent-space representation, are informed about similarities and dissimilarities between different parameters. After training, all parameter snapshots are mapped into the latent space where clusters of similar parameters are detected. [Han et al. \(2021\)](#) propose to find clusters through visual examination of the latent-space features. To visualize the features, they apply a nonlinear dimension reduction algorithm,

First iteration

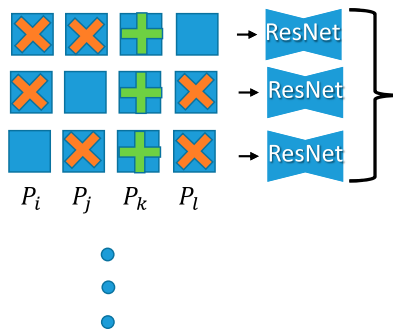


Loss values after 5 epochs

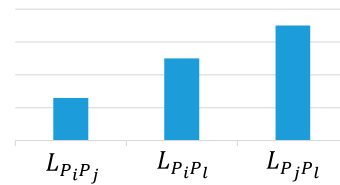


$$L_i = L_{P_i}, \quad L_j = L_{P_j}, \quad L_k = L_{P_k}, \quad L_l = L_{P_l}$$

Second iteration



Loss values after 5 epochs



$$L_i = L_{P_i P_j} + L_{P_i P_l}, \quad L_j = L_{P_i P_j} + L_{P_j P_l}, \quad L_l = L_{P_i P_l} + L_{P_j P_l}$$

FIG. 2. Method overview. In the i th iteration, the same ResNet is trained multiple times using different combinations of $n - i$ input and i output parameters. Orange crosses indicate the output parameters, green pluses indicate the parameters that are fixed in the input set, and blue squares indicate the remaining parameters in the input set. In each iteration, for each free parameter a loss is computed by adding the losses of those networks in which the parameter is in the output set. The parameter with the maximum loss is fixed in the input set. For the WB dataset, orography is used in every input, but is not predicted.

called t-SNE (van der Maaten and Hinton 2008). Parameters in the same cluster become a transferable variable group.

In the translation-graph construction stage, the Kullback–Leibler divergence is used to estimate a measure of so-called *transferable difficulty* for pairs of parameters inside a transferable parameter group. Parameter pairs are considered for transfer, if the Euclidean distance between their respective latent-feature representations is smaller than a predefined distance threshold. Parameters in different transferable parameter groups are not considered for transfer. A directed transfer graph is then constructed by chaining transferable pairs according to a minimum discrepancy criterion. Finally, in the variable transfer stage, CNNs are trained to learn the transfer mapping according to the translation graph.

V2V reduces the search for transferable variables to pairs of similar variables based on a distance threshold criterion and the visual analysis of a t-SNE projection. This leads to a number of shortcomings regarding the expressiveness of the identified parameter relations and reproducibility of the results. V2V does not assess true predictability of one variable based on another, but uses an empirically determined approximate criterion, which might overlook potentially valuable relationships when the threshold value is not set optimally.

Furthermore, V2V cannot always decide unambiguously the source and target fields. This decision is based on the translation graph where the nodes correspond to the parameter

fields and directed edges indicate transferability from a source to a target variable. In this graph, however, cycles can occur. This happens because the *transferable difficulty* is based on the Kullback–Leibler divergence, which is not a metric and does not satisfy the triangle inequality in general. When only pairwise transferabilities are considered, this can result in the selection of all parameters in a set of similar parameters as sources and targets of one another, respectively.

3. Datasets

We validate our approach with the WeatherBench dataset, which is a benchmark dataset for data-driven, medium-range climate prediction problems (Rasp et al. 2020), and the convective-scale forecast ensemble generated by Necker et al. (2020) (see appendix B).

In both cases, the proposed neural network models receive as input an array of shape $m \times H \times W$, with m the number of physical input parameters, and $H \in \mathbb{N}$ and $W \in \mathbb{N}$ denoting the spatial dimensions of the field. Assuming an initial number of $n \in \mathbb{N}$ physical parameters, the model output is a field of shape $(n - m) \times H \times W$, which contains reconstructions of the parameters that have not been considered in the input (see Fig. 2).

WB is based on ERA5 atmospheric reanalysis data (Hersbach et al. 2020) generated regularly at the European Centre for

Medium-Range Weather Forecasting (ECMWF) through data assimilation procedures, combining spatiotemporal numerical simulations and observation data. To facilitate the accessibility to machine learning workflows and accelerate studies in weather prediction, WB provides regridded ERA5 reanalysis data on regular latitude–longitude grids with three different resolutions and 13 different pressure levels. The data are available hourly for 40 years from 1979 to 2018.

We consider a selection of 2D single-level fields with a resolution of $1.405^\circ 25'$ in latitude and longitude, resulting in a domain size of 128×256 vertices for global data. The selected physical parameters are 2-m temperature (t2m), total cloud cover (tcc), u and v components of 10-m wind (u10, v10), total precipitation (tp), and top-of-atmosphere incident solar radiation (tcsr). As in previous studies (e.g., Höhle et al. 2020), where the prediction accuracy of convolutional neural networks could be improved by using orography information, orography height is added as an additional constant predictor. To facilitate model training, all field values are standardized before training the models. This normalization scheme enforces equal variation in all parameter fields under consideration, which is helpful for ensuring comparability of reconstruction accuracy metrics. We utilize two different global and local standardizations. In local standardization, rescaling is computed for each grid location from the statistics of all time steps, while in global standardization, mean and standard deviation values are computed over the whole domain for all time steps. Global standardization performs better in our case and helps to reduce the reconstruction error. We refer this to the fact that the local standardization can enhance uniformity of the data but destroys spatial coherence patterns. We use the 23 first years of WB during training. Of these, 20 years serve as training data for fitting the models, and three years are reserved for validation. The remaining years are left out for testing and visualization.

4. Method

To overcome the shortcomings of V2V, we propose an alternative parameter selection procedure. It replaces the single-parameter autoencoding CNN and subsequent clustering and pairwise similarity search with multiparameter CNNs and loss-based tracking of the learning progress. Initially, given a multivariate time-varying dataset with $n \in \mathbb{N}$ parameters and $T \in \mathbb{N}$ time steps, the user selects the number m of input fields from which the remaining $n - m$ parameter fields are predicted.

a. Parameter selection

The most straightforward—yet computationally demanding—approach is to launch $\binom{n}{m}$ training runs for the different parameter configurations and select the network with the lowest loss. By starting from $n - 1$ inputs and one output and proceeding iteratively with decreasing number of inputs, the procedure can also be made dependent on a predefined loss threshold, that is, by launching for all k , $1 \leq k \leq m$, a batch of $\binom{n}{k}$ training runs and stopping once the minimum loss exceeds a given threshold. The parameter configuration for

which the minimum loss is achieved is then selected for variable transfer. In our current implementation we consider only the user-specified number of input parameters.

Since for large values of n , the described procedure requires training too many networks, a computationally less expensive alternative needs to be developed. A straightforward approach is to train only n networks, where each network predicts one single parameter from the remaining $n - 1$ parameters, and use the networks' losses as indicators of how difficult the prediction of a parameter is. If a loss value is high, predictability is low, and, thus, the parameter predicted with the highest loss is fixed as one of the input parameters. Then, of all trained networks where this parameter is already contained in the input set, the one with the largest loss is selected, and the variable that is predicted is added to the input. This procedure is repeated until the specified number of inputs is reached.

In our experiments, this approach finds exactly the same input and output sets as the exhaustive training procedure, yet it requires training only n networks. In general, however, since only the difficulties of predicting each parameter individually from all other parameters are considered, parameter combinations with higher prediction strength can be overlooked. For instance, consider a subset of similar parameters, each of which can be predicted at high accuracy from the other parameters in this subset. In this situation, the loss values of the reconstruction networks will be low, and it becomes unlikely that one of these parameters will be selected as input. Consequently, either the input solely comprises parameters from which the ones in the subset cannot be well predicted, or m needs to be so large that of each subset of similar parameters at least one is selected. However, since parameters from the most similar subset will be considered last, m can become too large to be of any practical relevance.

To address these shortcomings, we propose a strategy with lower computational complexity than the first strategy, and which differs from the second strategy in that it considers subsets of parameters in both the inputs and predicted outputs. As in the previous strategy, n networks are trained initially, with each network predicting a single parameter from the remaining $n - 1$ parameters, and the parameter predicted by the network with the highest loss is fixed in the input. In the next iteration, all networks with $n - 2$ inputs (including the fixed parameter) and two outputs are trained. For all but the fixed parameter, the overall loss is computed by adding the losses of all networks where this parameter is in the predicted set. The parameter with the highest loss is fixed, and the procedure moves on with $n - 3$ inputs, and three outputs now containing the two fixed parameters (see Fig. 2 for a graphical overview of the proposed approach). This strategy, in case of a similar subset of parameters, recognizes when a certain output cannot be well predicted and then fixes an input that is necessary to achieve higher accuracy.

In our experiments, all parameter fields are normalized before training to equilibrate differences in parameter magnitudes and variation. If the dataset contains static fields like orography, these fields are concatenated to the inputs to serve as

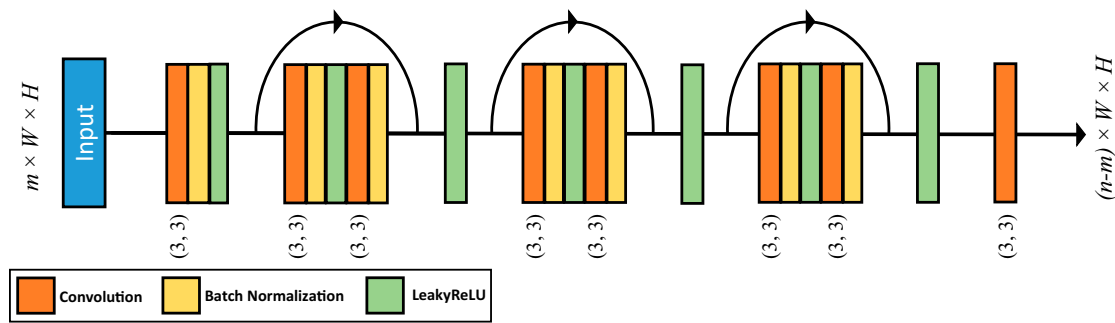


FIG. 3. Schematic of the used ResNet architecture. It consists of three residual blocks, each followed by a LeakyReLU activation function. The m ($< n$) input fields are fed into the input convolution layer with 64 channels and kernel size of $(3, 3)$, a batch normalization layer, and LeakyReLU activation function. The network comprises three residual blocks and a final single convolution layer. The network predicts $n - m$ output fields.

additional information for the models. In particular, orography is used with the WB dataset to enable the networks learning dependencies between the parameters and land/seascape, and, thus, enhance their inferencing skills. The quality of the reconstruction of all parameters is measured by a suitable loss function, for example, L_1 loss, and the model weights are optimized using standard backpropagation. We monitor both the training and validation loss to avoid overfitting.

A network's loss curve indicates how difficult it is for the network to achieve an accurate reconstruction depending on the current input and output parameters. That is, depending on which parameters are used, the reconstruction error decreases more or less quickly. While saturation of both losses typically happens after 70 epochs, our experiments show that already after few epochs of training the reconstruction error clearly reveals the differences between different parameter combinations. In particular, when comparing the loss curves in these early stages with the loss curves after convergence, the relative behavior of the networks does not change. This indicates that network training does not need to be performed until convergence but can be stopped after few epochs to obtain an indication of the reconstruction quality. In particular, we consider loss values after five epochs of training, resulting in roughly one hour (for training 20 networks) on a low-size deep learning cluster with six midsize GPUs to determine three input and three output parameters for the WB dataset, comprising six parameters. For the CSEns dataset, comprising nine different parameters, the proposed procedure requires roughly six hours for training 87 networks to determine the four input parameters that best predict the remaining five output parameters.

Compared to the V2V approach by Han et al. (2021), the proposed strategy is computationally more expensive, yet it exhibits a number of advantages: First, we obtain a more accurate measure of transferability, since our models are directly trained to reconstruct parameters. Second, the proposed approach is not constrained to selecting pairs of parameters but can uncover multiparameter relationships. Last, the method, in principle, enables to set a loss threshold for triggering the stopping of iterations. Due to normalization of the target

parameters, this threshold can be interpreted as a measure of acceptable relative error and is thus more accessible than the distance threshold in the latent-space features, which is considered in the original V2V algorithm.

b. Network architectures

In this study, we propose to train a deep CNN architecture to predict a certain number of output parameters from a given set of input parameters. In general, deeper networks can have higher prediction quality, yet they can easily lead to convergence problems in the optimization process due to vanishing gradients (Glorot and Bengio 2010). In early layers of the network, gradient estimation causes an exponential decay of the gradient magnitudes, so that the parameters cannot change significantly in the training process. An efficient way to overcome this problem is to utilize shortcut or residual connections as in ResNet architectures (He et al. 2016). In such architectures, outputs of earlier layers are added to the output of later layers, thus circumventing the accumulation of intermediate gradients.

In this study, we select a ResNet architecture with three residual blocks. A schematic representation is shown in Fig. 3. The input block of the model consists of a single convolution layer with kernel size of $(3, 3)$, 64 channels, batch normalization, and leaky rectified linear unit (LeakyReLU). We use input padding before each convolution layer. To account for periodic boundary conditions in the longitude direction of the WB dataset, we employ a periodic padding scheme in this dimension, and replication padding elsewhere. After that, there are three residual blocks, and each block has two convolution layers with 64 channels and kernel size $(3, 3)$. Since the number of parameters grows with the kernel size, it is cost efficient to select a kernel of size 3. After the first convolution layer in the residual block, the network utilizes a batch normalization layer, a LeakyReLU layer, the second convolution layer, and another batch normalization layer. Batch normalization is used to achieve improved stability and convergence (Ioffe and Szegedy 2015). After each residual block, a LeakyReLU activation function guarantees nonlinearity of the mapping. The final layer is a single convolution layer with kernel size of $(3, 3)$ and $n - m$ output channels.

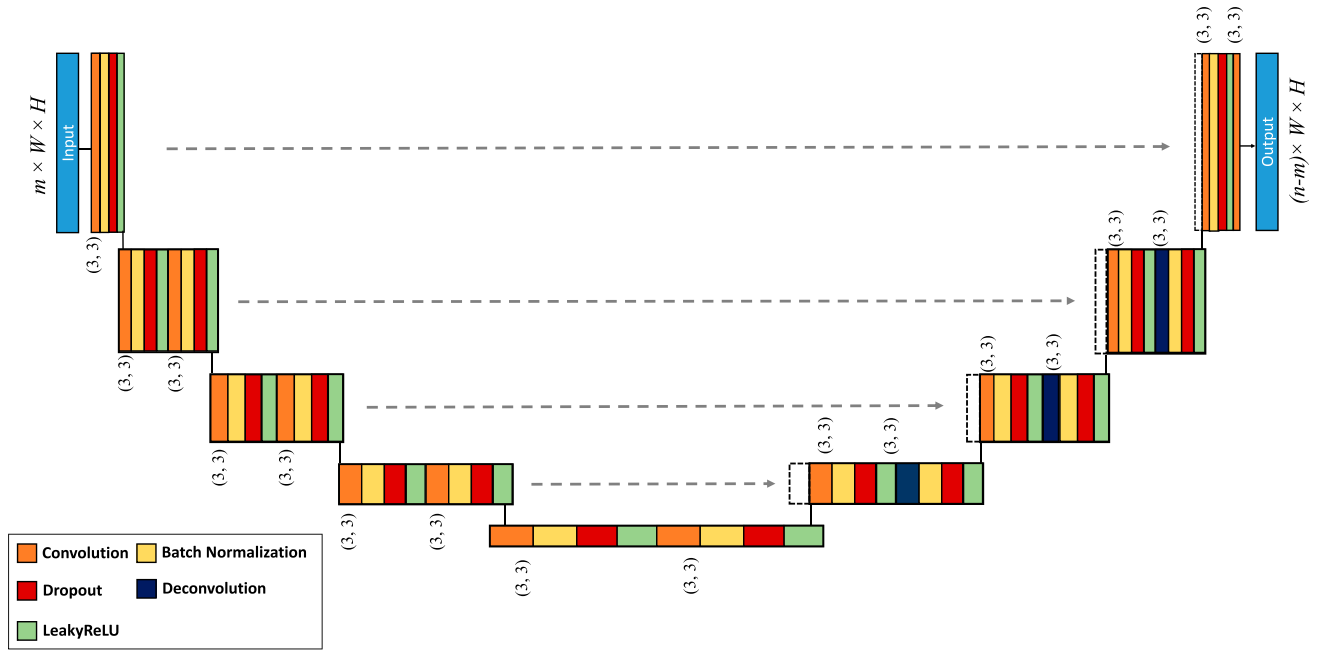


FIG. 4. Schematic of the used UNet architecture. The contracting branch is comprised of three convolutional blocks, each consisting of two convolution layers with subsequent batch normalization, dropout, and a LeakyReLU activation function. The expansive branch includes three deconvolution blocks, each consisting of one convolution, and one deconvolution layer. Each of these layers is followed by a batch normalization layer, dropout, and LeakyReLU activation function. The m ($< n$) input fields are fed into the input block, which contains a single convolution layer with 64 channels and kernel size of $(3, 3)$, followed by batch normalization layer, dropout, and LeakyReLU activation function. The number of reconstructed fields is $n - m$.

As an alternative to the ResNet architecture, we also analyzed the potential of a UNet architecture (Ronneberger et al. 2015) for loss-based parameter selection.

In contrast to the ResNet architecture, which operates on a single spatial scale throughout the whole architecture, the UNet architecture allows for the extraction of features on multiple spatial scales, which offers the possibility of learning a wider range of parameter relationships. The UNet consists of two symmetric branches, which give it the characteristic u-shape, as seen in Fig. 4. In the encoding branch, the data are encoded into an abstract reduced feature representation, and in the decoding path the feature representations are then decoded to reconstruct the predicted fields at the target resolution. During the encoding step, the resolution is iteratively reduced, and the number of feature channels is increased at the same time. In the decoding step, while reducing the number of feature channels, the features are supersampled to a higher resolution. The paths are connected by skip connections, which concatenate feature channels from the encoder with corresponding features from the decoder, in order to precisely preserve and localize the information in the data that could be lost in the encoding stage. The bottommost layer of the UNet, that is, the bottleneck layer, enforces the model to learn a compact representation of the input containing the globally most relevant information to recover it.

In our experiments, the UNet architecture did not improve the reconstruction quality significantly, yet it increased the training time due to its higher computational complexity. A sample of the reconstruction quality of the UNet architecture

is shown in Fig. A2. Nevertheless, we found that ResNet and UNet seem to learn different mappings internally, which we discuss in more detail in section 5c.

The presented architectures have been designed through empirical experimentation, trading of model flexibility and reconstruction quality against applicability to diverse datasets and computational efficiency. Especially for data fields on spherical geometries, more sophisticated network designs exist (see, e.g., the survey by Cao et al. 2020). Such architectures, however, come at higher computational complexity or require careful data-specific selection of hyperparameters to achieve better performance than standard CNNs, and are thus not considered in the present study.

The error between target fields and predictions is measured in terms of L_1 distance, which we prefer over L_2 due to empirically less pronounced suppression of outlying predictions.

5. Experiments

In an exhaustive ablation, we demonstrate the feasibility and reliability of our approach using the WB reanalysis dataset as a use case. The results of applying the proposed strategy to the CSEns dataset are shown in appendix B.

Via this ablation study, we aim to answer the following questions:

- 1) Which is the minimal set of input parameters from which the remaining parameters can be reconstructed accurately? This number indicates how aggressively the initial parameter set can be reduced.

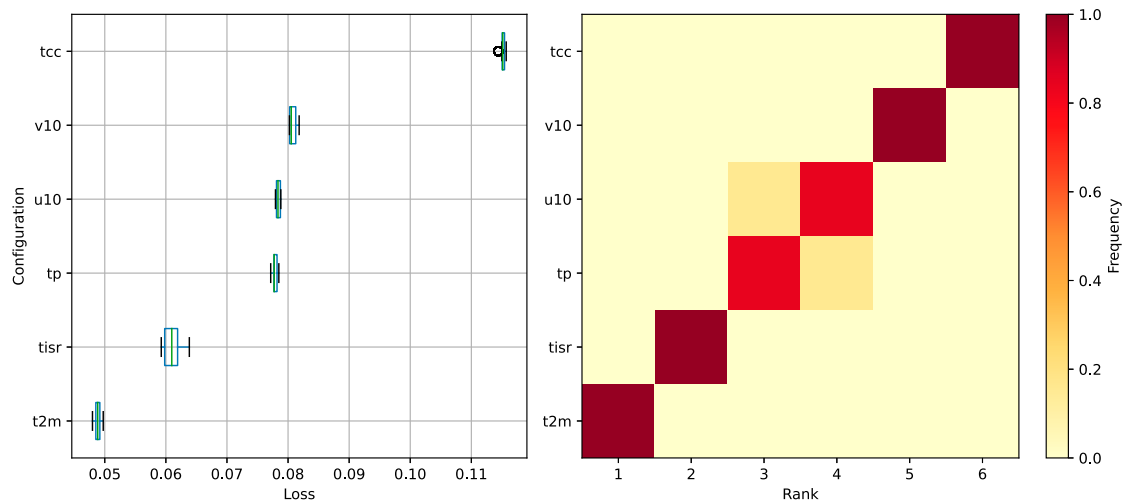


FIG. 5. Distribution and ranking of losses for different network configurations. Networks are trained for different parameter configuration ($m = 1$, i.e., one parameter is left out and is to be predicted) with different weight initializations. (left) Boxplot of the loss statistics. (right) Heat map of the observed ranking order.

- 2) Over which geographic regions do parameters strongly affect the network's prediction quality?

To answer the first question, we use loss-based parameter selection via the ResNet architecture. Both architectures are used subsequently to answer the second question by visualizing the sensitivity of the local prediction accuracy to regional changes of the input parameters.

a. Validation of the extended V2V approach

To validate the reliability and reproducibility of the proposed loss-based parameter selection, we train networks for different parameter configurations multiple times with different random weight initializations and compare the order of the observed losses after five training epochs. For brevity, we first show results only for the case $m = 1$, which results in six different parameter configurations. For every configuration, we train 10 models and sample model ensembles by randomly picking one of the 10 models for each configuration. For each sample, we then rank the model configurations according to the observed loss value after five training epochs and assess the consistency of the ranking order among different samples. Figure 5 illustrates the observed loss statistics. We find that the separation in loss magnitude between different parameter configurations is typically larger than the variance of losses for each configuration (see Fig. 5, left). As a result, the ranking of losses is consistent between different runs. This is seen in the heat chart in Fig. 5 (right), which visualizes the frequency of how often a particular loss rank is observed for each of the parameters and suggests an almost perfect one-to-one mapping between parameters and ranks. Both charts together confirm that our loss-tracking approach constitutes a reproducible criterion for selecting parameter configurations. Nevertheless, we observe that clustering of losses may occur, that is, different configurations may result in very similar loss statistics (e.g., parameters tp, u10, and v10 in Fig. 5, left). Due to the overall small variation

in losses per configuration, we conjecture that all of the possible outcomes are equally well suited for further evaluations.

b. Ablation study

For the WB data comprising six parameters, we start with applying the loss-based selection procedure to predict one masked out parameter using the remaining five input parameters. This number is then increased to two and finally three predicted parameters, with four and three input parameters, respectively. This means that during the first iteration six networks, then $\binom{5}{2}$ networks and finally $\binom{4}{3}$ networks are trained. We do not go beyond three masked out parameters, since significantly reduced reconstruction quality is observed in this case.

To justify our decision to use the network losses after five epochs as indicators of the difficulty to predict a certain parameter or parameter combination, we analyze the loss values for five and 70 epochs of training of all networks that were trained. Figure 6 shows the losses of all networks trained for five epochs by the proposed loss-based selection approach (top and bottom left charts), and the losses of all $\binom{6}{3}$ possible networks trained for five epochs (dark blue bars in bottom right). The loss values indicate that the loss-based selection approach finds the parameter combination yielding the lowest loss. Note that this is also confirmed for the CSEns dataset, as shown in Figs. B2 and B3 in appendix B. As shown by the overlayed loss values of the networks trained for 70 epochs (green bars in bottom right), training for five epochs shows very similar relative differences between different parameter combinations. This result is also confirmed by the comparison of the loss values for 5 and 70 of the CSEns dataset. When only one parameter is predicted from the remaining five parameters (plus orography), it can be seen that 2-m temperature and total cloud cover, respectively, are the parameters that are easiest and most difficult to

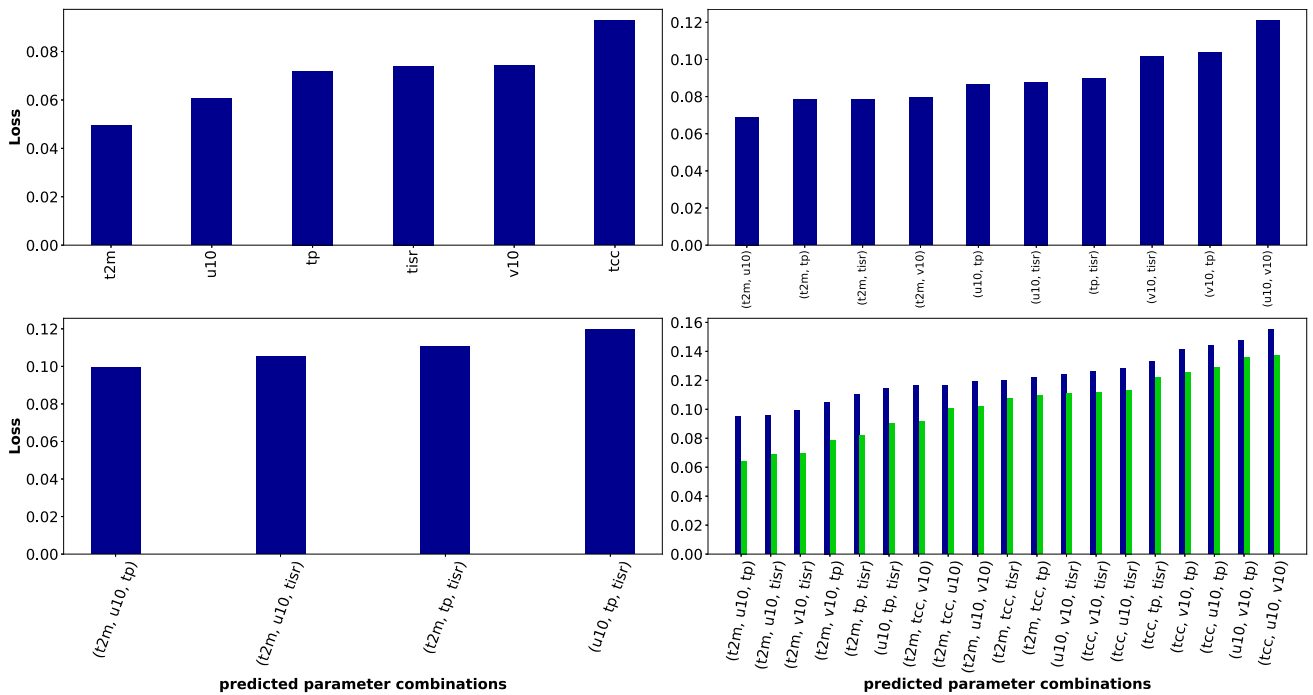


FIG. 6. Bar charts showing the losses of all networks trained for 3-to-3 parameter transfer with the WB dataset using the proposed iterative loss-based approach [(top left) first iteration, (top right) second iteration, (bottom left) third iteration], and (bottom right) of all possible $\binom{6}{3}$ networks. Blue bars represent losses after five epochs of training, green bars indicate losses after 70 epochs.

reconstruct. Thus, total cloud cover is the first parameter that is fixed in the input.

Figure 7 shows the initial parameter fields (including orography) and the reconstruction results of the three parameters that have been masked out by the loss-based procedure. For comparison, the reconstruction results of the three worst parameter combinations are shown in Fig. A1 in appendix A. Notably, when all $\binom{6}{3}$ parameter

combinations are evaluated, the very same combination is determined.

In Fig. 8, for the selected parameter combination the resulting pixelwise differences between the reconstructions and the initial parameter fields are shown. The quality of the reconstructed fields is measured using the image statistics structural similarity index measure (SSIM; Wang et al. 2004) and the peak signal-to-noise ratio (PSNR), with the initial parameter

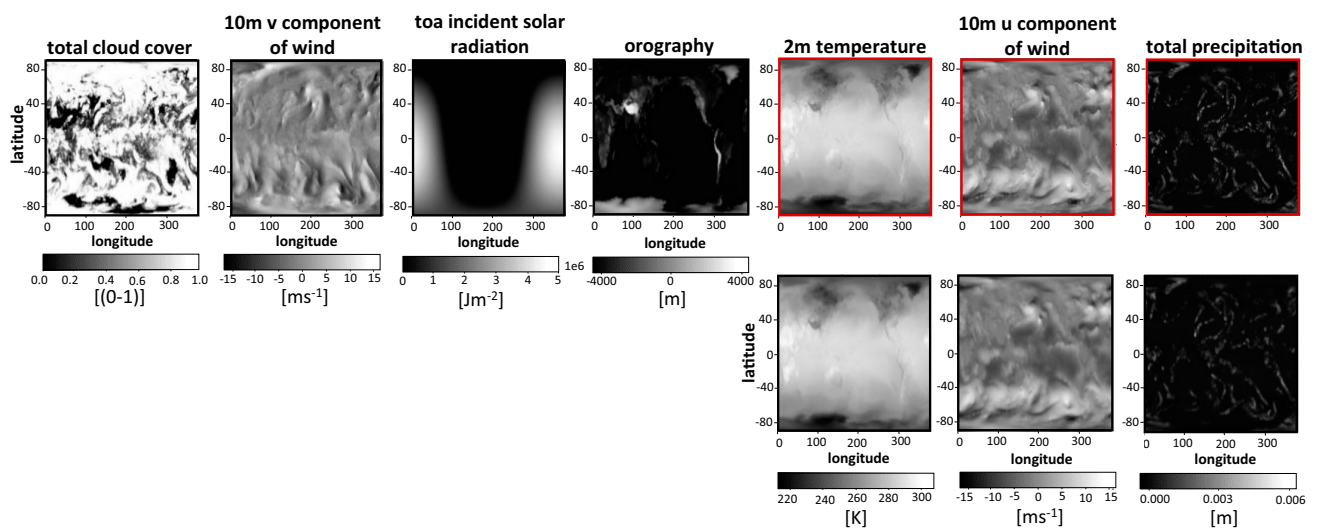


FIG. 7. Reconstruction results for the WB dataset when the network is trained to predict three parameter fields from three input fields and orography. (top) The initial parameter fields. A red outline indicates those fields the network has learned to predict from the others. (bottom) Predicted parameter fields.

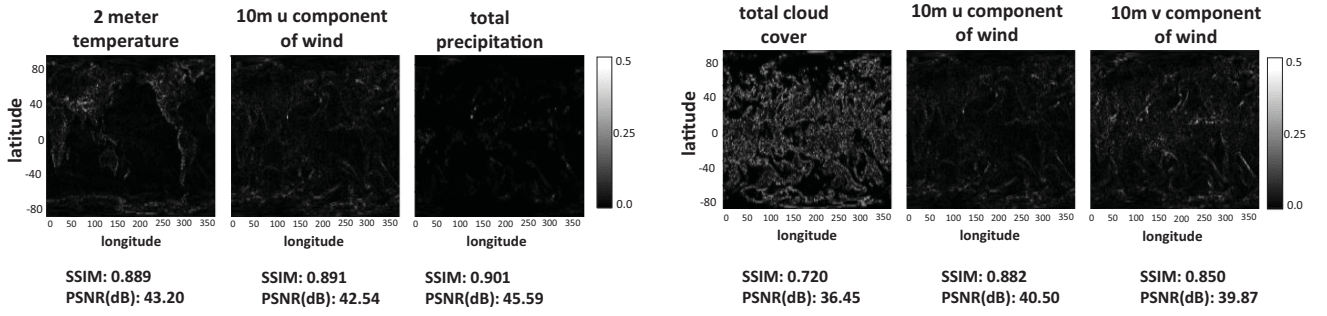


FIG. 8. Pixelwise differences between the initial and predicted fields when using the (left) best and (right) worst parameter combination. Per-pixel values are scaled by a factor of 10 for better visibility. Corresponding SSIM and PSNR (dB) values are given below each image.

fields as references. It can be seen that even when one-half of the parameters are masked out, they can still be reconstructed at high accuracy by the network. In addition, the reconstruction quality that is achieved by the worst parameter combination is shown, that is, the parameter combination yielding the highest loss of all possible $\binom{6}{3}$ parameter combinations. The results indicate the importance of a suitable procedure for finding the best parameter combination. The pixelwise error plots indicate significantly different reconstruction quality between the best and worst parameter set.

c. Feature analysis

While the potential of the selected network architecture for V2V can be concluded from the results of the ablation study, no information can be drawn about what kind of dependencies are exploited by the networks. To shed light on this aspect, we use LRP to localize the sensitivity of the reconstruction results to changes in the input parameter fields.

In its original form, LRP has been introduced as an explainability algorithm for image classification models (Bach et al. 2015), which is achieved by measuring neuron activations and back-propagated gradient information to identify image regions that exert a strong effect on the classifier output. LRP, thereby, builds on the concept of pixelwise decomposition of the classifier score. That is, given a classification score mapping of the form $f: \Omega \rightarrow \mathbb{R}$, where $\Omega \subseteq \mathbb{R}^{m \times H \times W}$ is the input domain (e.g., the space of images with $H \times W$ pixels and m channels per pixel), and $f(x) > 0$ (< 0) indicates evidence for presence (absence) of a particular feature, LRP attempts to find a set of relevance values $R_{kij} \in \mathbb{R}$ associated with the pixel values, such that the classification score can be approximated as

$$f(x) \approx \sum_{k=0}^{m-1} \sum_{i=0}^{H-1} \sum_{j=0}^{W-1} R_{kij}, \quad (1)$$

and $R_{kij} > 0$ (< 0) indicates that pixel channel k at position (i, j) contributes evidence in favor of (against) the presence of the feature in question. In the case of deep neural networks, which are composed of linear transformations with elementwise activation functions, suitable relevance values can be computed via iterative relevance back-propagation, subject to propagation rules (Bach et al. 2015).

Deviating from the setting of standard LRP, the input of V2V models is not an image, but a multidimensional array, representing a multiparameter field, and the output is not a univariate classification score, but a multidimensional multiparameter field. The difference in input modalities is only of limited importance, since the multiparameter fields can be interpreted directly as multichannel images. However, the complexity of the model output prevents straightforward application of standard LRP. We therefore propose to use an adapted variant of LRP to gain insight into spatio-temporal relevance and correlation patterns between model predictions and inputs.

Given a model mapping of the form $f: \Omega \rightarrow \mathbb{R}^{(n-m) \times H \times W}$, we propose adding an additional selector layer $s: \mathbb{R}^{(n-m) \times H \times W} \rightarrow \mathbb{R}$ at the end of the model, such that the output of the combined model, $s[f(x)] \in \mathbb{R}$, admits an additive decomposition according to Eq. (1). Possible choices for s include summation operators, such as global (or local) averaging of field values or deviation measures, or selection operations, which select single pixels and output channels for computing LRP relevances. Depending on the choice of selector layer, different aspects of the input-output relationship can be investigated. For instance, a selector function returning the mean value of the output channel $0 \leq c < m$ inside a region defined by the pixel set $I \subseteq \{(i, j): 0 \leq i < H, 0 \leq j < W\}$, that is,

$$s_I^{(c)}(x) := \langle [x]_{kij} \rangle_{(i,j) \in I, k=c}, \quad (2)$$

where $[x]_{kij}$ denotes selection of the element at position (k, i, j) in the array x , yields positive relevance for input regions that favor an increase in the averaged quantity according to Eq. (1). In contrast, functions of the form

$$\delta_I^{(c)}(x; x_0, p) := \langle \|x - x_0\|_{kij}^p \rangle_{(i,j) \in I, k=c}, \quad (3)$$

$p > 0$, yield positive relevance for regions that increase the deviation between the model prediction x and a certain reference prediction x_0 within the region I .

Figures 9 and 10 show relevance maps for the global atmospheric situation at 0800 UTC 15 May 2004 as seen in the WB dataset, using the ResNet (Fig. 9) and the UNet model (Fig. 10).

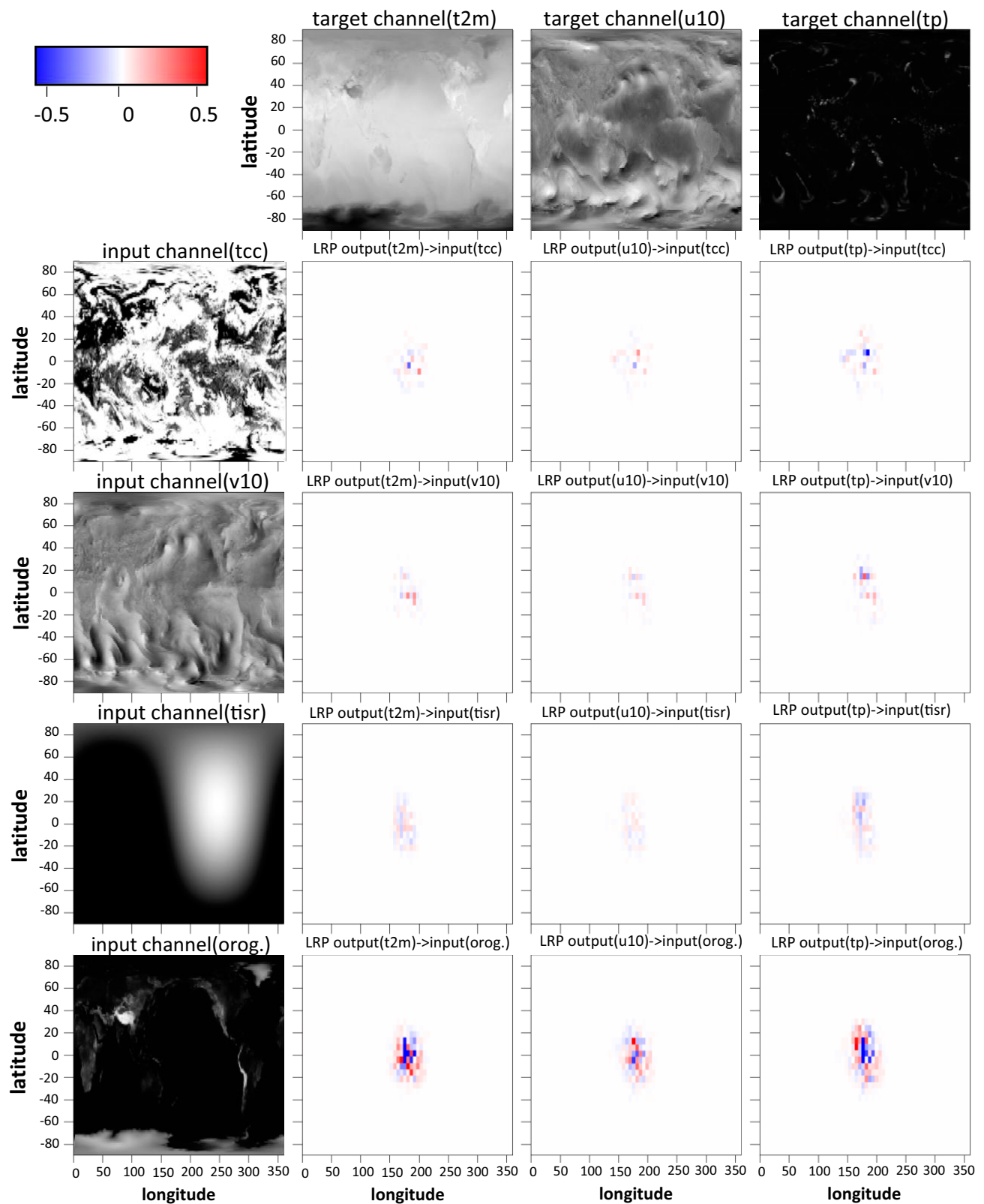


FIG. 9. LRP relevance maps with deviation-based selector function for the ResNet model in the best I/O configuration w.r.t. the proposed selection procedure. Time stamp of data sample: 0800 UTC 15 May 2004.

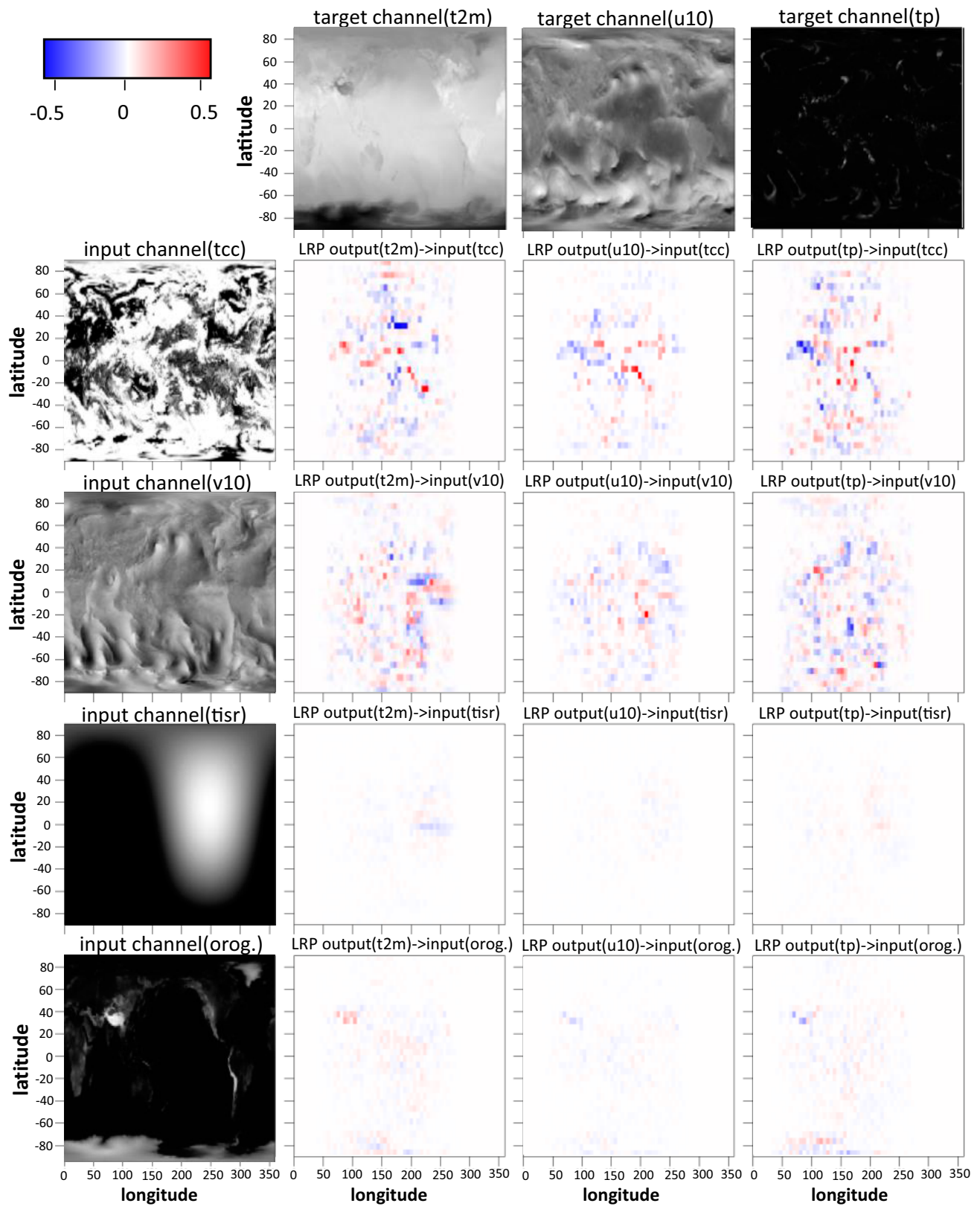


FIG. 10. LRP relevance maps with deviation-based selector function for the UNet model in the best I/O configuration w.r.t. the proposed selection procedure. Time stamp of data sample: 0800 UTC 15 May 2004.

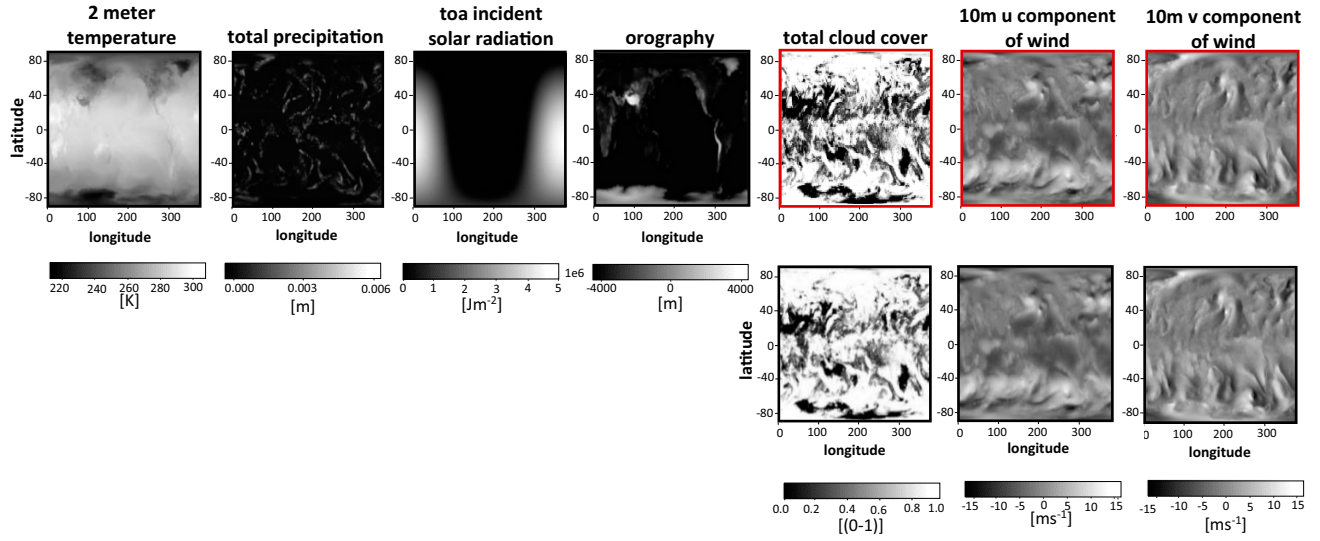


FIG. A1. Reconstruction results for the WeatherBench dataset when the network is trained to predict three parameter fields (worst combination) from four input fields. (top) The initial parameter fields. A red outline indicates those fields the network has learned to predict from the others. (bottom) Predicted parameter fields.

We employ an absolute-difference-based selector function with a focus on single pixel deviations of the predicted quantities from the respective target value, that is, $\delta_I^{(c)}(x; x_0, 1)$ with $I = \{(i^*, j^*)\}$, $0 \leq i^* < H$, $0 \leq j^* < W$, and x_0 denoting the target field. This reveals information about what parts in the data push the separate prediction channels away from the actual target value. Figures are shown for $(i^*, j^*) = (63127)$, which corresponds to the center pixel in the image, located at $0^\circ, 180^\circ$. Relevance maps for other dates and pixel indices look similar. All output channels are treated separately, yielding a matrix of relevance maps, which visualize relationships between channel-wise prediction errors and model inputs. The back-propagation of relevance values according to our proposed feature selection is carried out using the standard LRP algorithm, which is

available in the Captum model interpretability library for Pytorch (Kokhlikyan et al. 2020).

While the relevance patterns appear noisy in both cases, the structure of the relevance maps differs significantly between the model architectures, despite being trained on the same task and with the same set of training data. The ResNet architecture favors relevance distributions that are concentrated around the reference location. This is consistent with the inductive bias of the architecture, which arises from the use of convolution layers with small kernel sizes (see section 4b). Positive and negative relevances appear to be distributed randomly throughout the map, but significant differences are observed in the magnitude of the relevances. Relevance values w.r.t. orography possess larger magnitude in both positive and negative orientation than the remaining parameters. For the

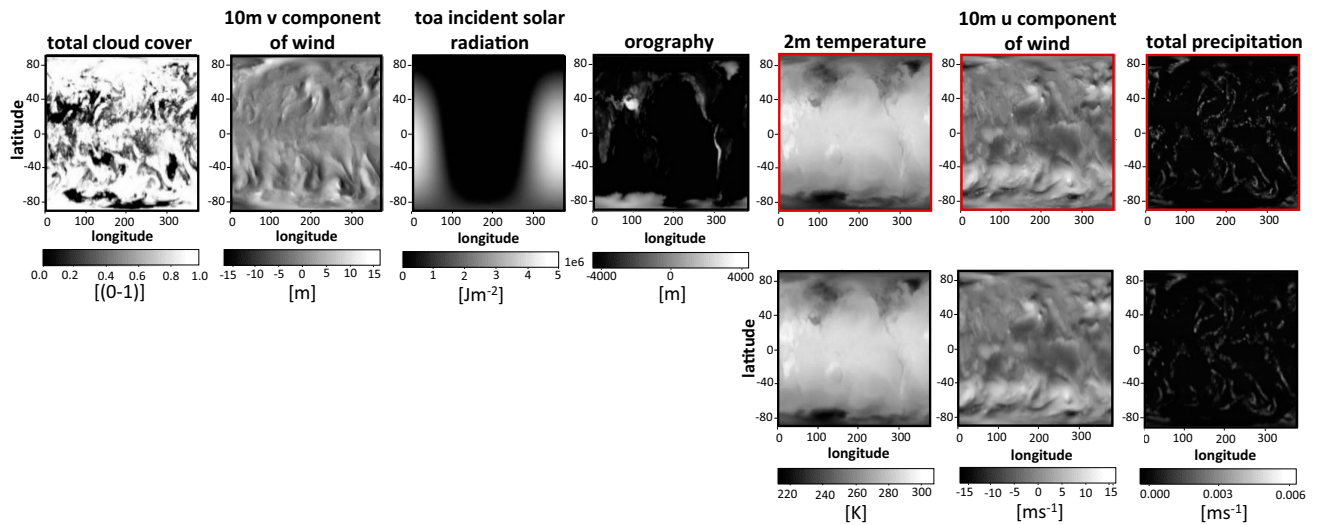


FIG. A2. As in Fig. 7, but using UNet for training.

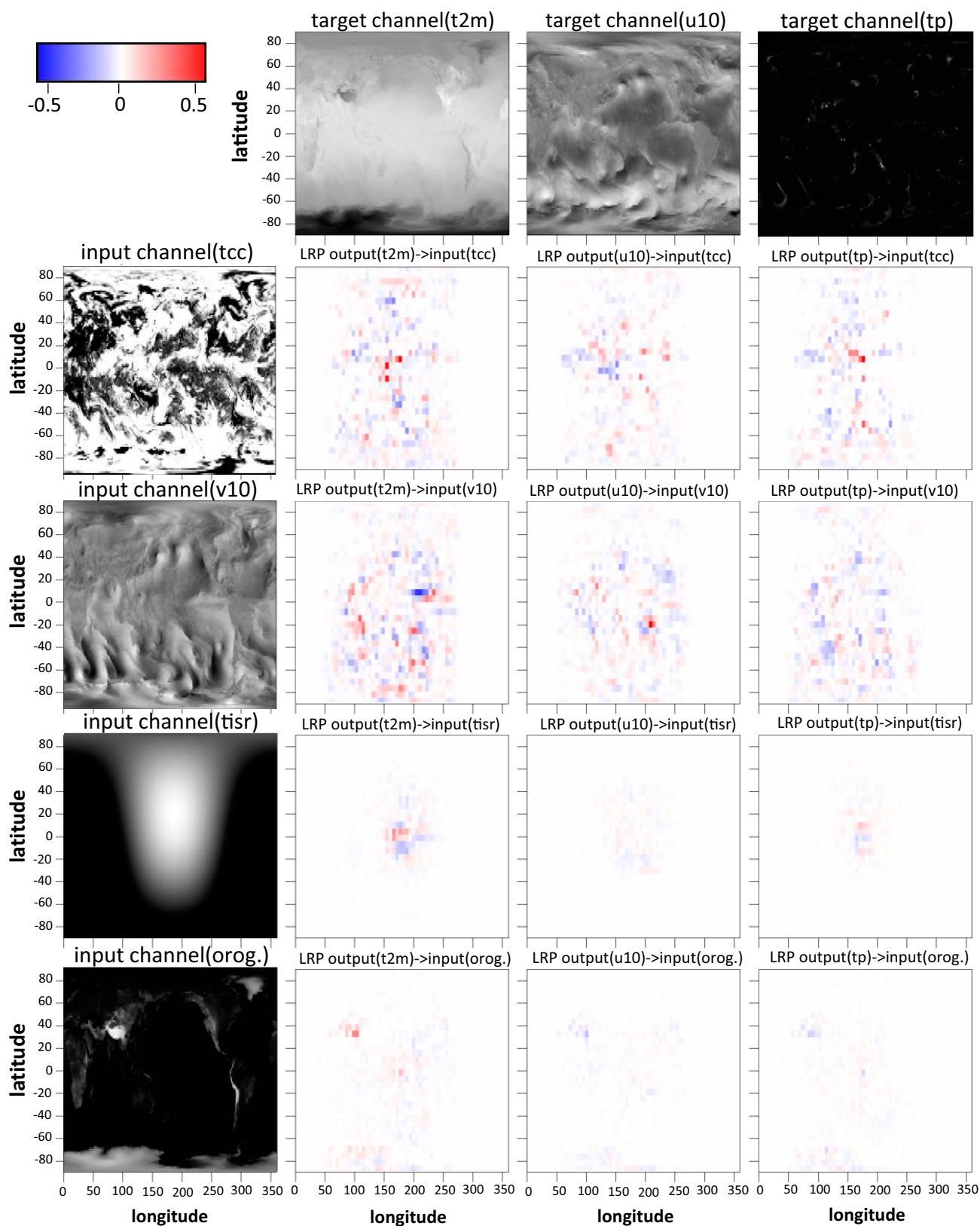


FIG. A3. LRP relevance maps with deviation-based selector function for the UNet model in the best I/O configuration for WB data. Time stamp of data sample: 1200 UTC 15 May 2004.

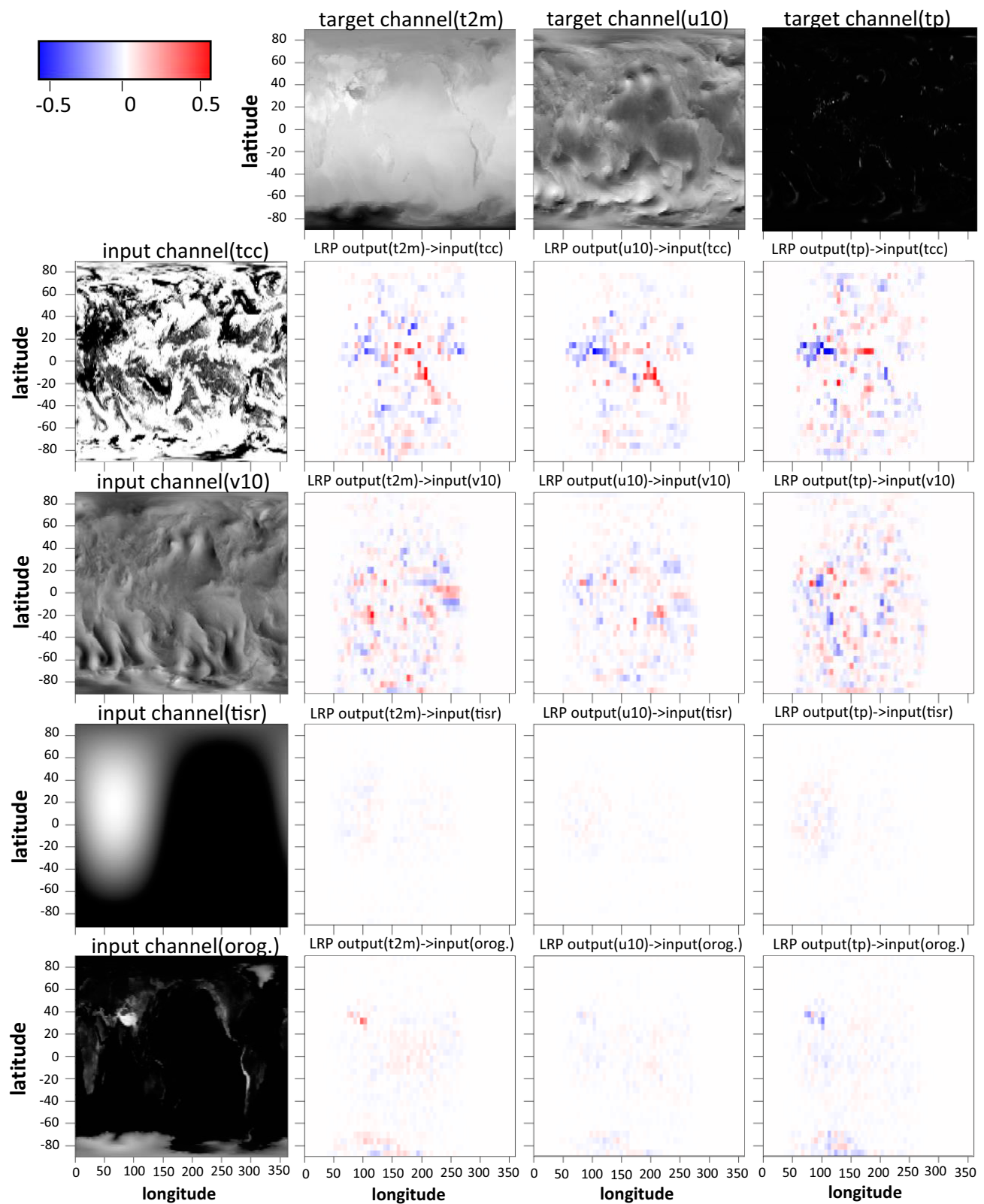


FIG. A4. LRP relevance maps with deviation-based selector function for the UNet model in the best I/O configuration for WB data. Time stamp of data sample: 2000 UTC 15 May 2004.

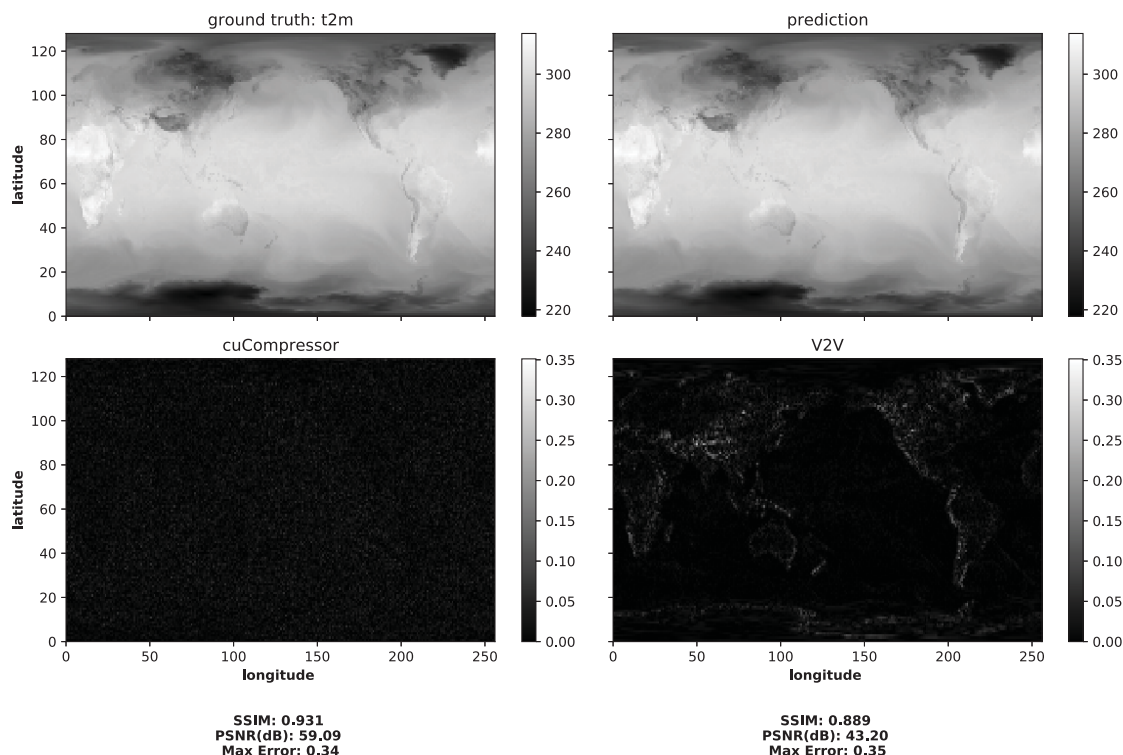


FIG. A5. Quality comparison of V2V against a dedicated compression algorithm for volumetric data. Parameter field $t2m$ compressed at a rate of 12:1 with the publicly available compute unified device architecture (CUDA) compression library by Treib et al. (2012), which provides lossy compression using a combination of the discrete wavelet transform, coefficient quantization, run-length encoding, and Huffman coding. (top left) Original field. (top right) Parameter field predicted using 3-to-3 V2V transfer. (bottom) Pixelwise differences for reconstructed compressed field and V2V reconstruction.

cloud-cover input, relevance values are concentrated on a small number of pixels, which obtain a relevance with notably higher amplitude than that of surrounding pixels. Similarly, the large degree of variability in the relevance maps makes it difficult to identify. For the UNet, relevance values are distributed over a larger spatial domain and relevance magnitude is largest for the cloud-cover field and the v component of the wind field. Likely, this is caused by the multi-scale properties of the UNet architecture and confirms that the UNet manages to learn features on larger spatial scales. Notably, the distribution of relevance values also displays stronger spatial correlations, which might suggest that the model learns to pay attention to spatially coherent features in the data. Also, in contrast to ResNet, the relevance of the orography field is smaller. Intuitively, this relevance attribution appears more understandable, since the selected reference pixel corresponds to a location in the middle of the Pacific Ocean, where the impact of orography on physical processes should be weak.

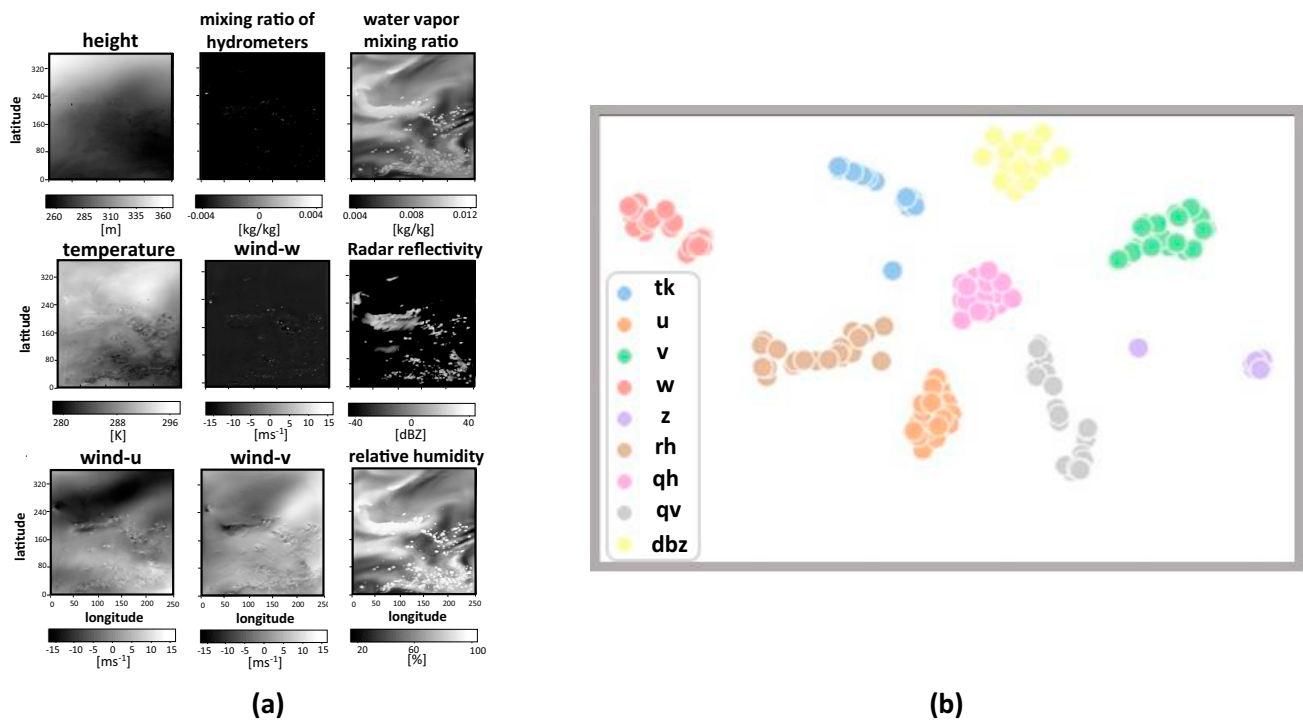
A prominent feature of the WB dataset is the temporal coherence of subsequent samples, which is determined by the day-night cycle, as well as the seasonal cycle. To assess the stability of the relevance maps, as well as the impact of the day-night cycle on the model mapping, we show two additional relevance maps for the UNet model applied to data samples from 1200 to 2000 LT 15 May 2004 in Figs. A3 and A4 in appendix A. The

figures show that the relevance maps for 0800 and 2000 LT look very similar. In particular, clusters of spatially coherent regions of positive or negative relevance are preserved, which suggests a stability and coherence in the visual structure of the relevance maps. The maps for 1200 LT deviate slightly and show larger relevance as for the 2-m temperature with respect to the top-of-atmosphere incoming solar radiation, which is consistent with physical intuition.

Overall, we conclude that the different architectures learn distinct mappings, despite being trained on the same task and achieving similar prediction accuracy. Yet, we find that some aspects of the dependency structure can be partly reverse engineered via thorough investigation of the relevance maps. Similar statements apply to the relevance maps for models trained on the CSEns dataset. Exemplary relevance maps for this dataset are shown in Fig. B8. A more detailed analysis of the derived relevance maps, as well as the study of more specific meteorological events and weather situations at selected times and locations, is, however, beyond the scope of this paper and will be addressed in future work.

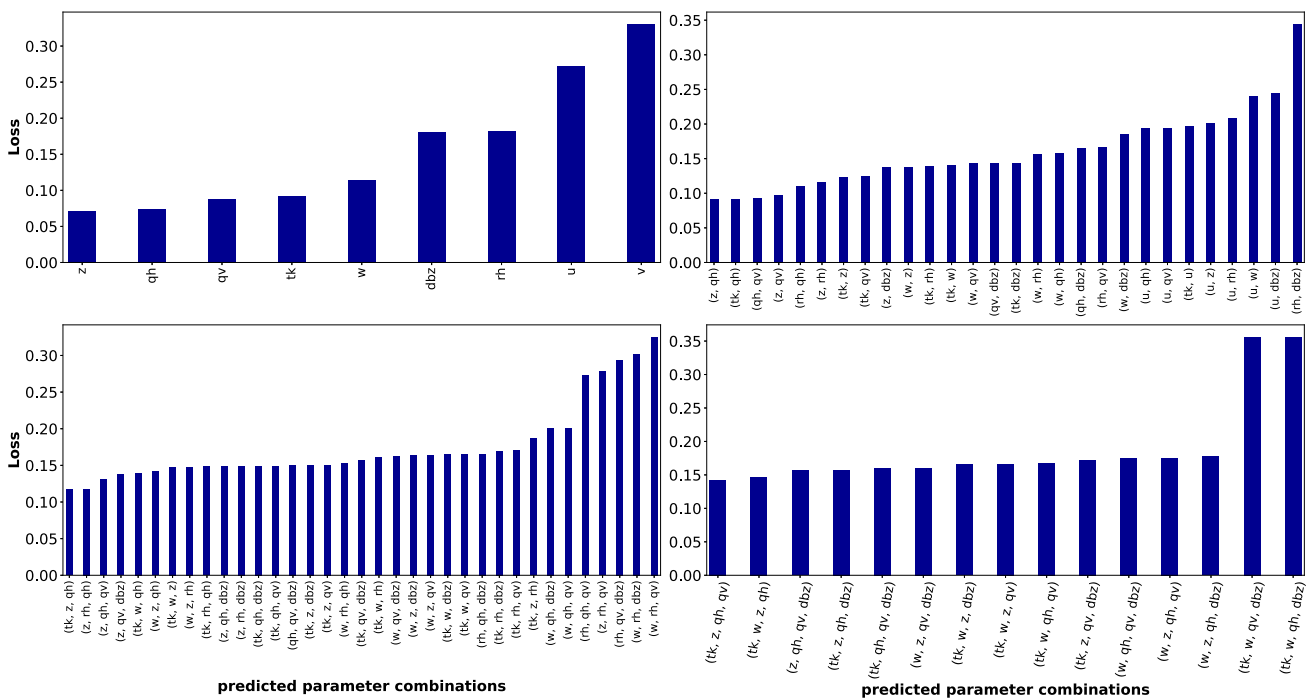
6. Conclusions

We have introduced an alternative way to perform deep learning-based variable-to-variable transfer. Instead of building



upon the similarity of latent-space representations of parameter fields to determine transferable parameter pairs, we train a network using different transfer scenarios and select the best parameter setting. In this way, we give more flexibility

to the network to exploit interparameter relationships, that is, to learn parameter combinations for improved transfer. This allows for saving bandwidth in in situ settings and can help to more aggressively compress multiparameter simulation data.



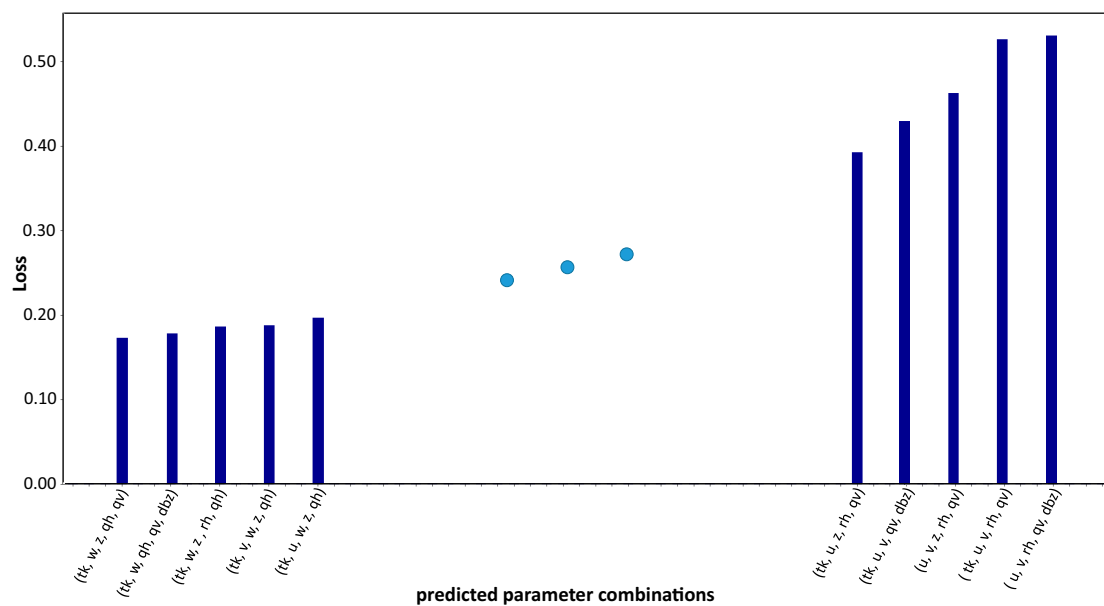


FIG. B3. Bar chart showing the losses of the best (left) and worst (right) possible networks for 4-to-5 parameter transfer with the CSEns dataset. All $\binom{9}{5}$ possible models have been trained for five epochs. Configurations with intermediate losses have been omitted from the chart for clarity of the visualization.

As shown in Figs. A5 and B9 in appendixes A and B, V2V transfer cannot compete with classical lossy data compression schemes in terms of compression rate, yet it may effectively support such schemes when the structure of the relevance maps generated via LRP is exploited to select spatially varying bitrates according to the importance of the data values. Other potential limitations arise due to intricacies in comparing the loss values of different parameter fields. Comparing L_1 loss values performs well on our data, which we verify by showing visualizations of the reconstructed fields. Yet it may happen that differences in the statistical distribution of field values result in deceptively low or high loss values for certain fields, which do not accurately reflect the reconstruction quality. In such cases, it may be useful to explore alternative metrics that

are more robust to differences in data distributions, such as relative improvement metrics that compare against simpler baseline models such as climatologies, or metrics like SSIM (Wang et al. 2004) that relate to visualization quality. The choice of such metrics, however, may depend strongly on the dataset at hand, as well as on the intended application. For reasons of general applicability, we rely on the L_1 loss in this study and demonstrate that it can serve as a reasonable default choice.

We have further analyzed the regional parameter structures that have the most significant effect on the reconstruction quality. By using an extension to layerwise relevance propagation (LRP), we were able to determine regions over which the field values have a large effect on the local reconstruction

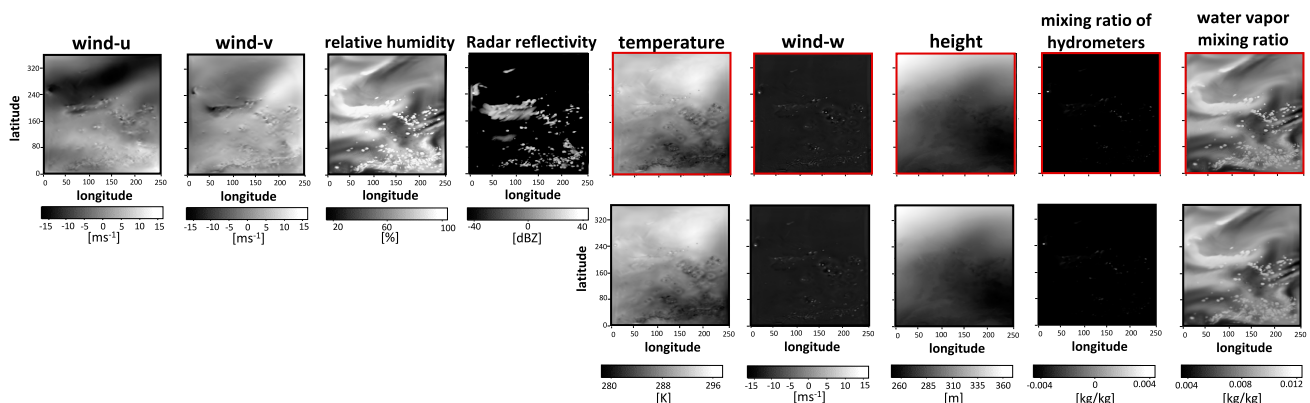


FIG. B4. Reconstruction results for the CSEns datasets when the network is trained to predict five parameter fields from four input fields. (top) The initial parameter fields. A red outline indicates those fields the network has learned to predict from the others. (bottom) Predicted parameter fields.

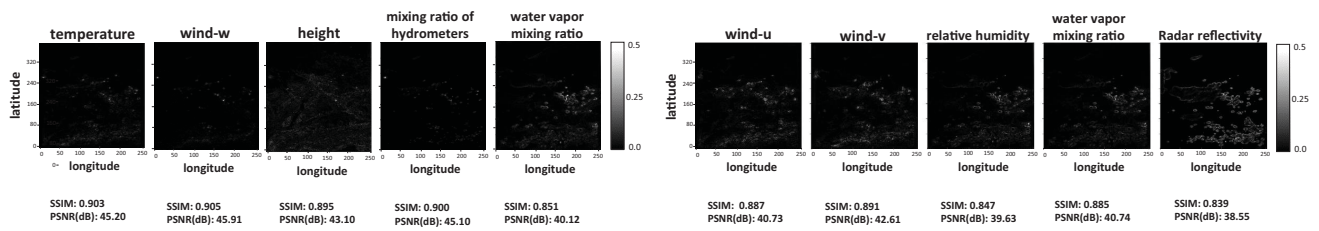


FIG. B5. As in Fig. 8, but using the CSEns dataset.

accuracy. LRP results demonstrate that different model architectures learn different mapping functions, depending on the inductive bias of the used architecture. In our study, the use of the UNet model led to more physically interpretable relevance maps, while the mappings learned by the ResNet architecture are constrained in learning spatial dependencies due to the construction of the network architecture. Information like this may help in operational model applications to gain a better understanding of model-driven inference procedures and increase trustworthiness of data-driven model predictions.

In the future, we will shed light on the use of the proposed V2V approach with 3D and especially large forecast ensembles. In our current use cases, all parameter fields show rather low mutual similarities, and, thus, one can expect our approach to perform even more effective once parameter fields with certain similarities and more pronounced spatial relationships are given, like ensemble simulations. One specific task we envision is to analyze the representativeness of the single members captured by a grand ensemble, by using V2V to reconstruct an as small as possible subset of all members capturing the full ensemble spread. This can facilitate guidance toward weather situations that are under- or overrepresented in the ensemble and reveal situations that are intrinsically difficult to resolve. Furthermore, we intend to consider the temporal evolution of the fields to improve the reconstruction at a certain time, that is, by letting the network train on multiple time steps from the past.

Finally, together with meteorologists and climatologists we intend to further analyze the sensitivity maps that have been derived via LRP. Such an analysis includes the extraction of

specific local weather events such as jet cores or fronts, and to set them into relation to the regions that have been deemed important for achieving high reconstruction accuracy. A limitation of the current LRP approach lies in the necessity of selecting reference locations, for which point-to-field relevance maps shall be computed. In exploratory data analysis tasks, it might be nontrivial to make sensible decisions about which locations to look at in the first place. We therefore plan on refining the LRP-based analysis procedures to detect regions of high impact in an automated fashion and with a more global view to enable the interactive exploration field-to-field relevance relations. In a similar line of reasoning, we intend to include the time dimension in the analysis, for example, by using temporal coherence and recurrence in the data to reduce the noise level of the derived LRP maps via temporal filtering or climatological summarization of relevances. Further efforts will be put on the investigation of alternative mechanisms for pursuing a sensitivity analysis, focusing more on spatial as well as temporal relationships between different parameters.

Acknowledgments. This study has been done within the subproject A7 “Visualization of coherence and variation in meteorological dynamics” of the Transregional Collaborative Research Center SFB/TRR 165 “Waves to Weather” (www.wavestoweather.de) funded by the German Research Foundation (DFG). The authors acknowledge crucial contributions by Juan Ruiz and the RIKEN Data Assimilation Research Team for conducting the 1000-member ensemble simulation.

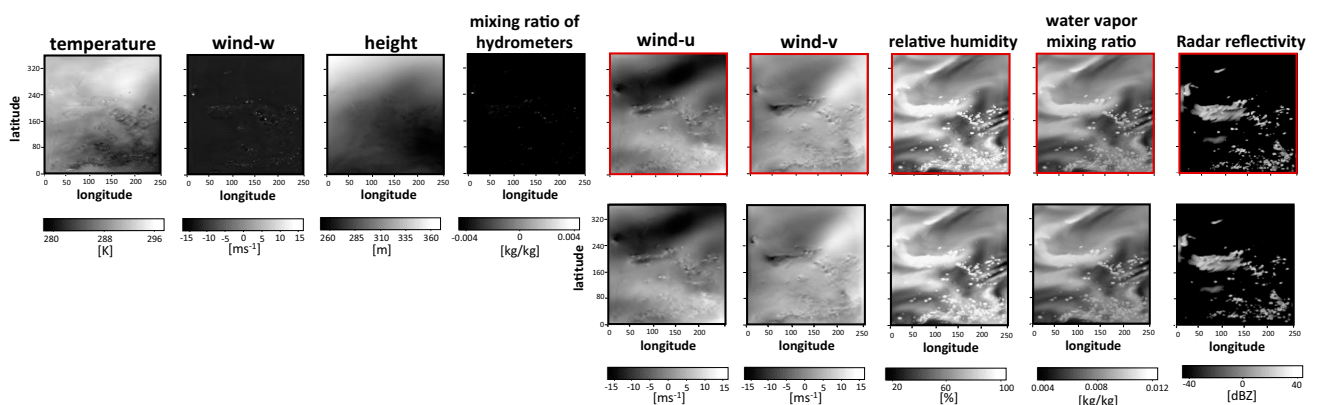


FIG. B6. As in Fig. A1, but using the CSEns dataset.

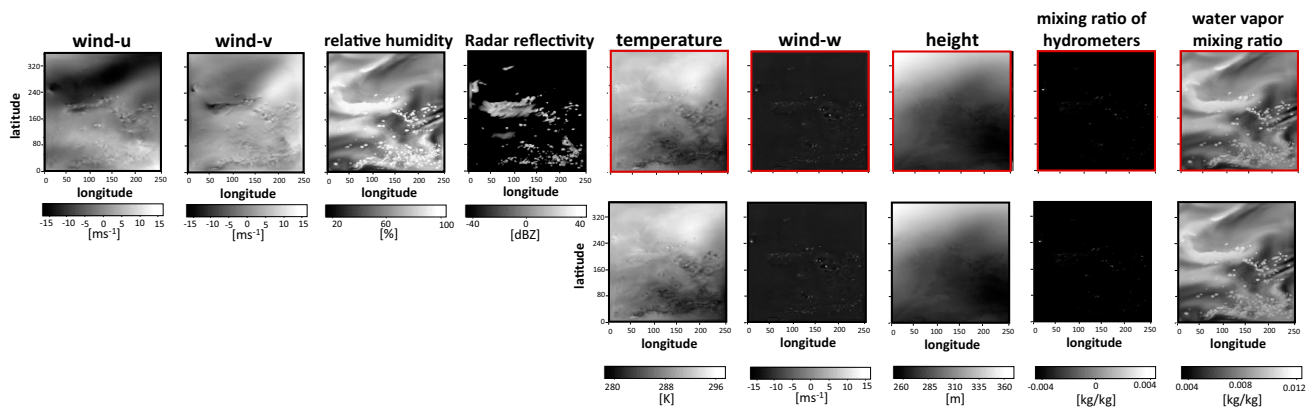


FIG. B7. As in Fig. B4, but using the UNet architecture instead of the ResNet.

Data availability statement. The code for the experiments is made publicly available at <https://github.com/FatemehFarokhmahesh/DNN-based-Parameter-Transfer-in-Meteorological-Data.git>. WeatherBench dataset (Rasp et al. 2020 [RDS+20]) is publicly

available at <https://github.com/pangeo-data/WeatherBench>. Access to the convective-scale ensemble simulation dataset can be granted upon-requested from the owners of the original dataset (Necker et al. 2020 [NGW+20]).

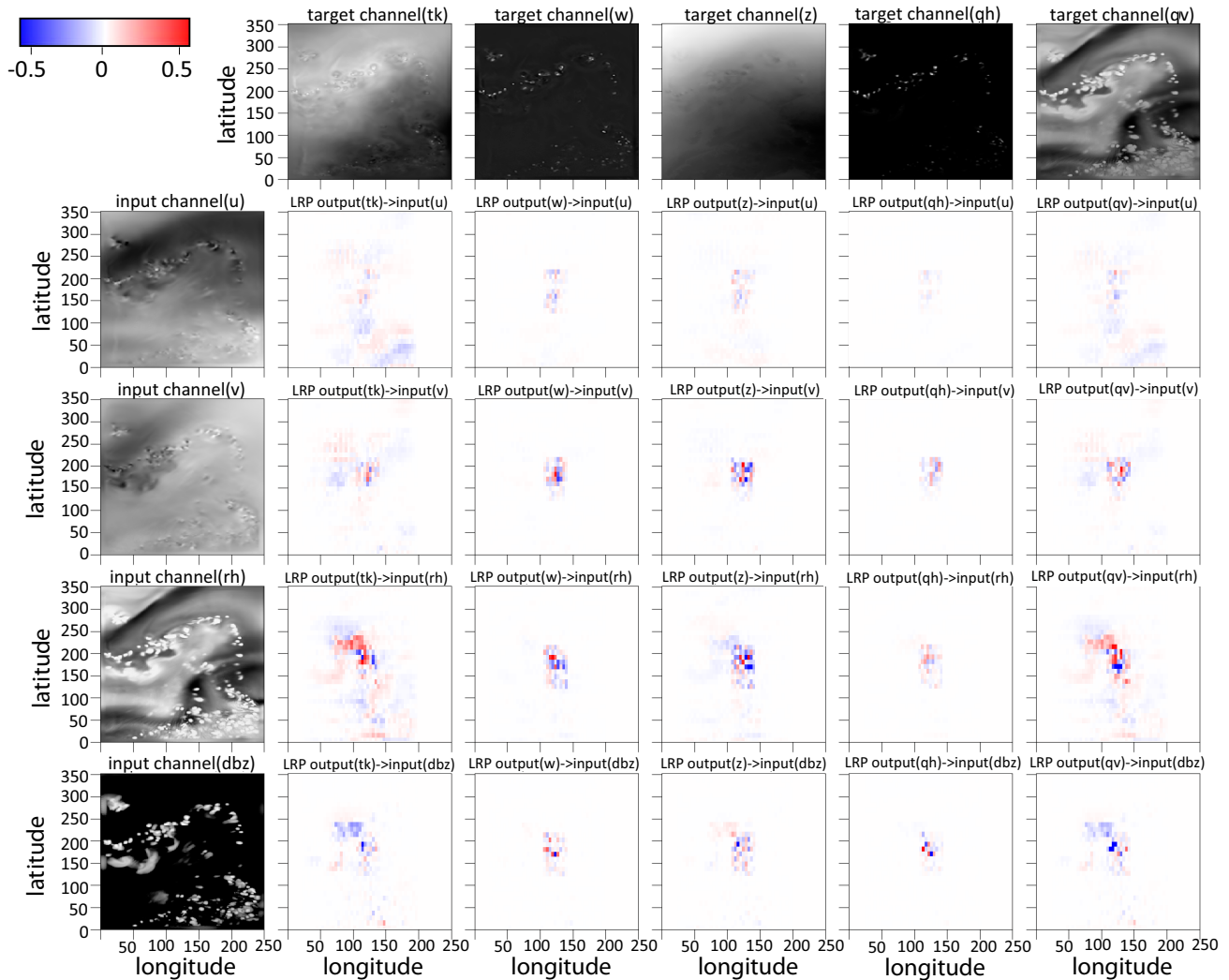


FIG. B8. LRP relevance maps with deviation-based selector function for the UNet model in the best I/O configuration for CSEns data. Time stamp of data sample: 1700 UTC 1 Jun 2016.

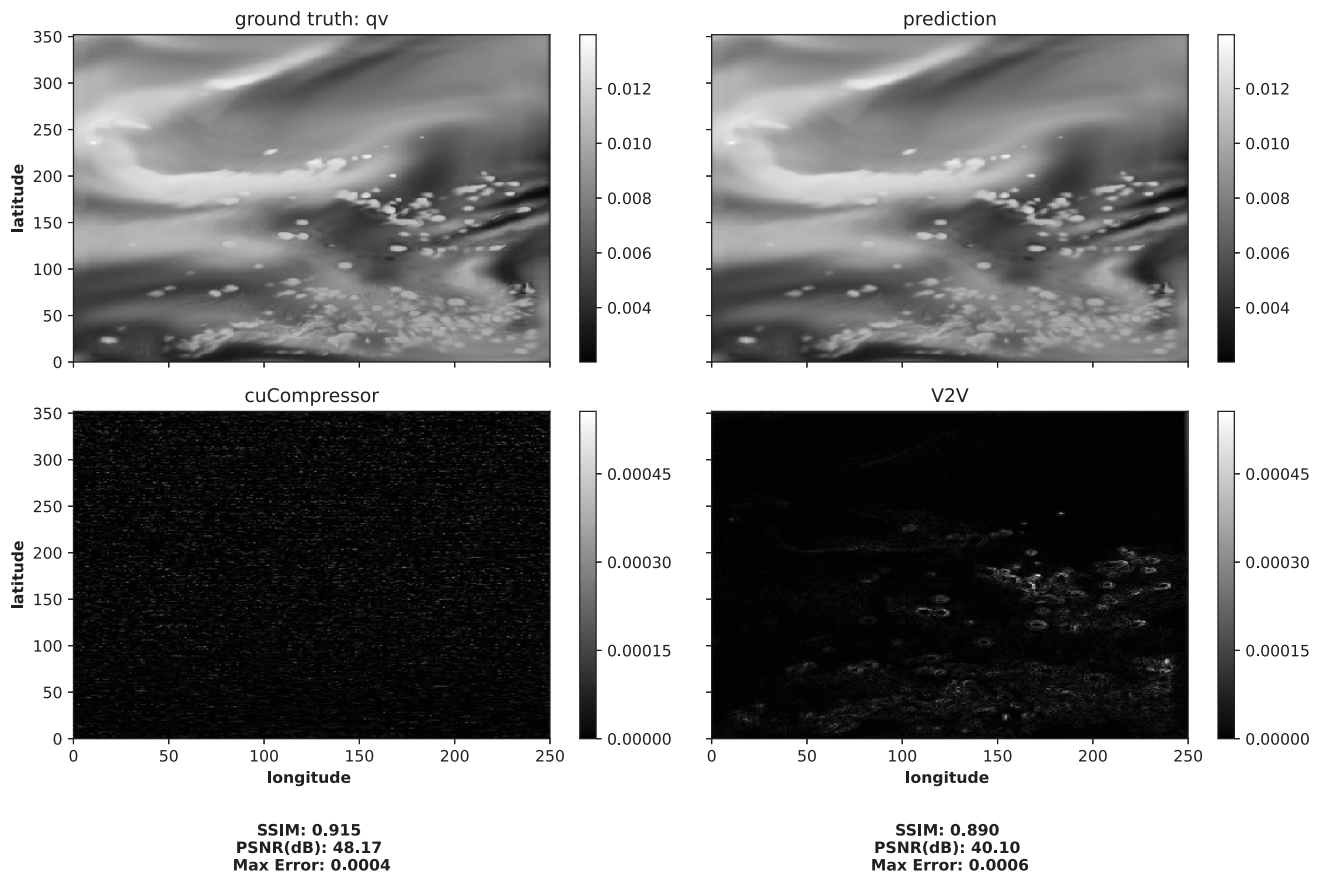


FIG. B9. Quality comparison of V2V against a dedicated compression algorithm for volumetric data. Parameter field qv compressed at a rate of 12:1 with the publicly available CUDA compression library by Treib et al. (2012), which provides lossy compression using a combination of the discrete wavelet transform, coefficient quantization, run-length encoding, and Huffman coding. (top left) Original field, (top right) parameter field predicted using 4-to-5 V2V transfer, and (bottom) pixelwise differences for reconstructed compressed field and V2V reconstruction.

APPENDIX A

Supplementary Visualizations for the WeatherBench Dataset

Figures A1–A5 illustrate specific aspects of V2V transfer in the first dataset, the WB reanalysis dataset.

APPENDIX B

Variable-to-Variable Transfer for the Convective-Scale Ensemble

The CSEns, generated by Necker et al. (2020), contains 1000 runs of a 3D atmospheric dynamics model over a rectangular domain in central Europe. Data are stored on a regular grid with 352×250 nodes, which corresponds to a horizontal grid spacing of 3 km and allows the resolution of convective effects in the model dynamics. The simulation covers a time interval of six hours, with a period of one hour between successive time steps, and comprises 30 levels in height. In the lower levels, some of the data are invalid due to grid cells falling below the level of the surface topography. Levels with missing values are omitted. 3D data

are available for a total of 9 different parameters, which are temperature (tk), u , v , and w component of winds (u , v , w), geopotential height (z), relative humidity (rh), mixing ratio of all hydrometeors (qh), water vapor mixing ratio (qv), and radar reflectivity (dBZ). Structural differences are observed not only between different parameters, but also between different time steps and height levels of the same parameter. Specifically, the variation in the fields decreases with increasing distance from the Earth surface, due to decreasing influence of boundary layer effects, and complexity increases with increasing simulation time due to a strengthening of convective activity. To enable a fair comparison between the CSEns and WB, we consider data only for the three lowermost levels without missing values, as well as the three latest time steps, which show the highest field complexity. We further split time-variate 3D fields both in time and height to obtain a sequence of plain 2D fields. We then consider data for 200 members for training, five members for validation, and the remaining members for testing and visualization.

Figures B1–B9 illustrate V2V transfer in the second dataset, the CSEns by Necker et al. (2020), which were excluded from the main paper to improve readability.

REFERENCES

- Bach, S., A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek, 2015: On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLOS ONE*, **10**, e0130140, <https://doi.org/10.1371/journal.pone.0130140>.
- Cao, W., Z. Yan, Z. He, and Z. He, 2020: A comprehensive survey on geometric deep learning. *IEEE Access*, **8**, 35 929–35 949, <https://doi.org/10.1109/ACCESS.2020.2975067>.
- Cheng, J., Q. Kuang, C. Shen, J. Liu, X. Tan, and W. Liu, 2020: ResLap: Generating high-resolution climate prediction through image super-resolution. *IEEE Access*, **8**, 39 623–39 634, <https://doi.org/10.1109/ACCESS.2020.2974785>.
- Glorot, X., and Y. Bengio, 2010: Understanding the difficulty of training deep feedforward neural networks. *Proc. 13th Int. Conf. on Artificial Intelligence and Statistics*, Sardinia, Italy, PMLR, 249–256, <http://proceedings.mlr.press/v9/glorot10a/glorot10a.pdf>.
- Guo, L., S. Ye, J. Han, H. Zheng, H. Gao, D. Z. Chen, J.-X. Wang, and C. Wang, 2020: SSR-VFD: Spatial super-resolution for vector field data analysis and visualization. *2020 IEEE Pacific Visualization Symp.*, Tianjin, China, Institute of Electrical and Electronics Engineers, 71–80, <https://doi.org/10.1109/PacificVis48177.2020.8737>.
- Han, J., and C. Wang, 2020: TSR-TVD: Temporal super-resolution for time-varying data analysis and visualization. *IEEE Trans. Visualization Comput. Graphics*, **26**, 205–215, <https://doi.org/10.1109/TVCG.2019.2934255>.
- , and —, 2022: SSR-TVD: Spatial super-resolution for time-varying data analysis and visualization. *IEEE Trans. Visualization Comput. Graphics*, **28**, 2445–2456, <https://doi.org/10.1109/TVCG.2020.3032123>.
- , H. Zheng, Y. Xing, D. Z. Chen, and C. Wang, 2021: V2V: A deep learning approach to variable-to-variable selection and translation for multivariate time-varying data. *IEEE Trans. Visualization Comput. Graphics*, **27**, 1290–1300, <https://doi.org/10.1109/TVCG.2020.3030346>.
- , —, D. Z. Chen, and C. Wang, 2022: STNet: An end-to-end generative framework for synthesizing spatiotemporal super-resolution volumes. *IEEE Trans. Visualization Comput. Graphics*, **28**, 270–280, <https://doi.org/10.1109/TVCG.2021.3114815>.
- He, K., X. Zhang, S. Ren, and J. Sun, 2016: Deep residual learning for image recognition. *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, Las Vegas, NV, Institute of Electrical and Electronics Engineers, 770–778, <https://doi.org/10.1109/CVPR.2016.90>.
- Helbig, C., L. Bilke, H.-S. Bauer, M. Böttinger, and O. Kolditz, 2015: MEVA—An interactive visualization application for validation of multifaceted meteorological data with multiple 3D devices. *PLOS ONE*, **10**, e0123811, <https://doi.org/10.1371/journal.pone.0123811>.
- Hersbach, H., and Coauthors, 2020: The ERA5 global reanalysis. *Quart. J. Roy. Meteor. Soc.*, **146**, 1999–2049, <https://doi.org/10.1002/qj.3803>.
- Höhlen, K., M. Kern, T. Hewson, and R. Westermann, 2020: A comparative study of convolutional neural network models for wind field downscaling. *Meteor. Appl.*, **27**, e1961, <https://doi.org/10.1002/met.1961>.
- Ioffe, S., and C. Szegedy, 2015: Batch normalization: Accelerating deep network training by reducing internal covariate shift. *Int. Conf. on Machine Learning*, Lille, France, PMLR, 448–456, <http://proceedings.mlr.press/v37/ioffe15.pdf>.
- Kokhlikyan, N., and Coauthors, 2020: Captum: A unified and generic model interpretability library for PyTorch. arXiv, 2009.07896v1, <https://doi.org/10.48550/arXiv.2009.07896>.
- Necker, T., S. Geiss, M. Weissmann, J. Ruiz, T. Miyoshi, and G.-Y. Lien, 2020: A convective-scale 1,000-member ensemble simulation and potential applications. *Quart. J. Roy. Meteor. Soc.*, **146**, 1423–1442, <https://doi.org/10.1002/qj.3744>.
- Pouliot, D., R. Latifovic, J. Pasher, and J. Duffe, 2018: Landsat super-resolution enhancement using convolution neural networks and Sentinel-2 for training. *Remote Sens.*, **10**, 394, <https://doi.org/10.3390/rs10030394>.
- Rasp, S., P. D. Dueben, S. Scher, J. A. Weyn, S. Mouatadid, and N. Thuerey, 2020: WeatherBench: A benchmark data set for data-driven weather forecasting. *J. Adv. Model. Earth Syst.*, **12**, e2020MS002203, <https://doi.org/10.1029/2020MS002203>.
- Reichstein, M., G. Camps-Valls, B. Stevens, M. Jung, J. Denzler, N. Carvalhais, and Prabhat, 2019: Deep learning and process understanding for data-driven Earth system science. *Nature*, **566**, 195–204, <https://doi.org/10.1038/s41586-019-0912-1>.
- Röber, N., and J. F. Engels, 2019: In-situ processing in climate science. *High Performance Computing*, Springer, 612–622, https://doi.org/10.1007/978-3-030-34356-9_46.
- Rodrigues, E. R., I. Oliveira, R. Cunha, and M. Netto, 2018: DeepDownscale: A deep learning strategy for high-resolution weather forecast. *2018 IEEE 14th Int. Conf. on e-Science*, Amsterdam, Netherlands, Institute of Electrical and Electronics Engineers, 415–422, <https://doi.org/10.1109/eScience.2018.00130>.
- Ronneberger, O., P. Fischer, and T. Brox, 2015: U-net: Convolutional networks for biomedical image segmentation. *Medical Image Computing and Computer Assisted Intervention*, N. Navab et al., Eds., Lecture Notes in Computer Science, Vol. 9351, Springer, 234–241.
- Sato, T., O. Tatebe, and H. Kusaka, 2019: In-situ data analysis system for high resolution meteorological large eddy simulation model. *Proc. Sixth IEEE/ACM Int. Conf. on Big Data Computing, Applications and Technologies*, New York, NY, Association for Computing Machinery, 155–158, <https://doi.org/10.1145/3365109.3368769>.
- Serifi, A., T. Günther, and N. Ban, 2021: Spatio-temporal downscaling of climate data using convolutional and error-predicting neural networks. *Front. Climate*, **3**, 656479, <https://doi.org/10.3389/fclim.2021.656479>.
- Toderici, G., D. Vincent, N. Johnston, S. Jin Hwang, D. Minnen, J. Shor, and M. Covell, 2017: Full resolution image compression with recurrent neural networks. *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, Honolulu, HI, Institute of Electrical and Electronics Engineers, 5306–5314, <https://doi.org/10.1109/CVPR.2017.577>.
- Treib, M., F. Reichl, S. Auer, and R. Westermann, 2012: Interactive editing of GigaSample terrain fields. *Comput. Graphics Forum*, **31**, 383–392, <https://doi.org/10.1111/j.1467-8659.2012.03017.x>.
- van der Maaten, L., and G. Hinton, 2008: Visualizing data using t-SNE. *J. Mach. Learn. Res.*, **9**, 2579–2605.
- Wang, C., H. Yu, R. W. Grout, K.-L. Ma, and J. H. Chen, 2011: Analyzing information transfer in time-varying multivariate data. *2011 IEEE Pacific Visualization Symp.*, Hong Kong, China, Institute of Electrical and Electronics Engineers, 99–106, <https://doi.org/10.1109/PACIFICVIS.2011.5742378>.

- Wang, Z., A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, 2004: Image quality assessment: From error visibility to structural similarity. *IEEE Trans. Image Process.*, **13**, 600–612, <https://doi.org/10.1109/TIP.2003.819861>.
- Weiss, S., M. Chu, N. Thuerey, and R. Westermann, 2019: Volumetric isosurface rendering with deep learning-based super-resolution. arXiv, 1906.06520v1, <https://doi.org/10.48550/arXiv.1906.06520>.
- , M. Işık, J. Thies, and R. Westermann, 2020: Learning adaptive sampling and reconstruction for volume visualization. arXiv, 2007.10093v1, <https://doi.org/10.48550/arXiv.2007.10093>.
- Zhou, Z., Y. Hou, Q. Wang, G. Chen, J. Lu, Y. Tao, and H. Lin, 2017: Volume upscaling with convolutional neural networks. *Proc. Computer Graphics Int. Conf.*, New York, NY, Association for Computing Machinery, 1–6, <https://doi.org/10.1145/3095140.3095178>.

 CORRIGENDUMFATEMEH FAROKHMANESH,^a KEVIN HÖHLEIN,^a AND RÜDIGER WESTERMANN^a^a *Department of Computer Science, School of Computation, Information and Technology,
Technical University of Munich, Munich, Germany*

(Manuscript received and in final form 4 April 2024)

We became aware that one of the datasets used in our published paper, Deep Learning–Based Parameter Transfer in Meteorological Data (Farokhmanesh et al. 2023), was proprietary and that the provision of the dataset was not acknowledged properly. This oversight was unintentional and originated from a misunderstanding regarding the use of the dataset. Upon reflection and further communication with the owners of the convective-scale ensemble dataset (Necker et al. 2020) that has been used in experiments in our original publication, we realized that the data owners should be included as coauthors in the publication. The rationale behind this correction is rooted in the realization that our assumptions about the data use policy were not aligned with the expectations of the data owners. In the original publication, we cited the publication in which the dataset is presented (Necker et al. 2020) and acted in good faith that ownership rights were not violated. However, it has come to our attention that the owners of the dataset should have been acknowledged as coauthors to explicitly recognize their contribution. We acknowledge our misunderstanding of the data use policy and the inadvertent omission of the data owners as coauthors.

REFERENCES

- Farokhmanesh, F., K. Höhle, and R. Westermann, 2023: Deep learning–based parameter transfer in meteorological data. *Artif. Intell. Earth Syst.*, **2**, e220024, <https://doi.org/10.1175/AIES-D-22-0024.1>.
- Necker, T., S. Geiss, M. Weissmann, J. Ruiz, T. Miyoshi, and G.-Y. Lien, 2020: A convective-scale 1,000-member ensemble simulation and potential applications. *Quart. J. Roy. Meteor. Soc.*, **146**, 1423–1442, <https://doi.org/10.1002/qj.3744>.

 Denotes content that is immediately available upon publication as open access.

Corresponding author: Fatemeh Farokhmanesh, fatemeh.farokhmanesh@tum.de

DOI: 10.1175/AIES-D-24-0016.1 e240016

© 2024 American Meteorological Society. This published article is licensed under the terms of the default AMS reuse license. For information regarding reuse of this content and general copyright information, consult the AMS Copyright Policy (www.ametsoc.org/PUBSReuseLicenses).

Neural Fields for Interactive Visualization of Statistical Dependencies in 3D Simulation Ensembles

F. Farokhmanesh¹ , K. Höhle¹ , C. Neuhauser¹ , T. Necker² , M. Weissmann² , T. Miyoshi³  and R. Westermann¹ 

¹Technical University of Munich, Department of Computer Science, School of Computation, Information and Technology, Germany

²University of Vienna, Department of Meteorology and Geophysics, Austria

³RIKEN Center for Computational Science, Kobe, Japan

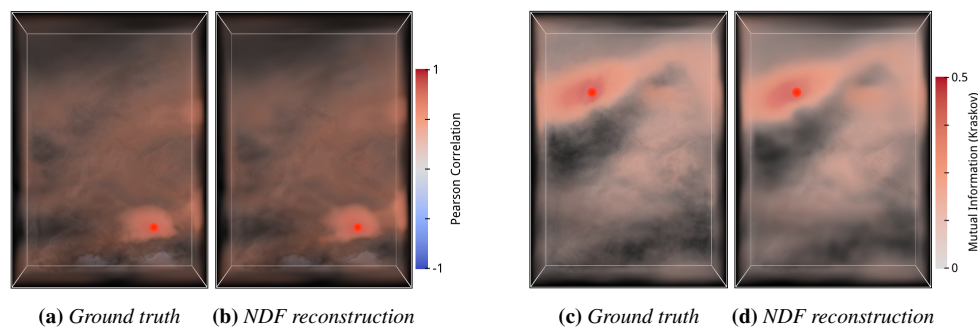


Figure 1: Neural dependence fields have learned to infer 1500 billion point-to-point Pearson correlation (left) or mutual information estimates (right) in a 1000-member simulation ensemble. Inference of the dependencies between data values at an arbitrary grid vertex (red dot) to all other vertices in a $250 \times 352 \times 20$ grid takes 9 ms on a high-end GPU. Ground truth volume renderings and network results, respectively, are shown in Figures 1a, 1c and 1b, 1d. The network requires only 1 GB at runtime.

Abstract

We present neural dependence fields (NDFs) – the first neural network that learns to compactly represent and efficiently reconstruct the statistical dependencies between the values of physical variables at different spatial locations in large 3D simulation ensembles. Going beyond linear dependencies, we consider mutual information as an exemplary measure of non-linear dependence. We demonstrate learning and reconstruction with a large weather forecast ensemble comprising 1000 members, each storing multiple physical variables at a $250 \times 352 \times 20$ simulation grid. By circumventing compute-intensive statistical estimators at runtime, we demonstrate significantly reduced memory and computation requirements for reconstructing the major dependence structures. This enables embedding the estimator into a GPU-accelerated direct volume renderer and interactively visualizing all mutual dependencies for a selected domain point.

CCS Concepts

• *Computing methodologies* → *Neural networks*; *Computer graphics*; • *Applied computing* → *Earth and atmospheric sciences*;

1. Introduction

Estimating statistical dependencies between physical variables at different spatial locations is crucial for understanding physical systems in various scientific and engineering fields. An important application lies in meteorology, where accurate weather forecasting relies on extensive numerical simulations. Weather forecasts need to account for randomness, and ensembles of simulations with varying initial conditions and model specifications are used to

quantify uncertainty. It is essential to analyze statistical relations, such as spatio-temporal auto-correlations within forecast fields or correlations between different forecast variables, to translate volumetric ensemble fields into reliable forecasts. Studying statistical dependence between random variables is well-researched, and measures exist for assessing linear and non-linear relationships. However, determining relations in 3D ensemble fields presents challenges. Computing dependencies on the fly may be compu-

tationally costly, and storing all point-to-point correlations leads to an explosion in required memory. For example, in a simulation ensemble with 1000 members on a $250 \times 352 \times 20$ grid, storing all correlations would need over 3 terabytes of memory, making it infeasible. Additionally, computing correlations between arbitrary point pairs on the fly requires the entire ensemble to fit into the working memory, and more complex measures like mutual information (MI) take roughly 16 minutes on a recent multi-core CPU, hindering interactive analysis of correlation structures.

In this work, we address these challenges by introducing neural dependence fields (NDFs), a novel compact representation of the major correlation structures in large multi-variable ensembles. For this, we propose to interpret fields of two-point correlation measures in 3D ensembles as scalar fields R over the domain of position pairs in 3D space, i.e., $R : \Omega \times \Omega \rightarrow \mathbb{R}$, for $\Omega \subset \mathbb{R}^3$. Taking inspiration from recent progress in neural scene representations and multi-dimensional tensor decomposition, we design a neural network architecture that exploits self-similarity in the correlation fields to learn a compact representation thereof. At the same time, the network enables fast sampling out of the neural representation. Thus, we can avoid holding the ensemble in memory and are able to speed up the computation of correlation estimates significantly, especially for complex non-linear dependence measures. For the ensemble considered in this work, it takes roughly 9 ms to reconstruct dependencies between an arbitrary reference point and all other points in the domain. This allows embedding the network into an interactive volume rendering pipeline, which enables instant visualization and comparison of single-variable auto-correlation and inter-variable correlation fields. In summary, our contributions are:

- A compact neural network architecture to learn statistical point-to-point correlations in large ensemble fields.
- The embedding of neural network-based correlation reconstruction into direct volume rendering to enable interactive visual exploration of the dependencies in the 3D domain.
- A demonstration of interactive correlation analysis for a large meteorological 3D ensemble field.

The proposed method is agnostic towards the choice of correlation measure, such that both linear (e.g., Pearson correlation) and compute-intensive non-linear measures (e.g., MI) are supported. The network manages to reconstruct the major correlation structures faithfully, despite showing a tendency to smooth out fine details (cf. Fig. 1). In view of the complexity of the information to be learned, our results demonstrate the potential of network-based correlation learning and open the door for future research in this field, e.g., by looking into more powerful architectures or specialized loss functions. The code for the project is publicly available at [FH23, NS23].

2. Related work

Scene representation networks and neural fields Scene representation networks (SRNs) are neural networks trained to derive compact representations of 3D models and scenes. Originally, they were proposed for 2D or 3D position coordinate mapping. Early examples include encoding surface models as implicit functions or occupation maps using fully-connected neural networks [MON*19, CZ19, PFS*19]. Later, they evolved to encode

diverse volumetric scenery information, such as neural radiance fields [TSM*20, MST*21] and were named neural fields [XTS*22]. A neural field is a neural network that learns a parametrization of spatio-temporal multi-dimensional physical fields over spatial coordinates. In inference, coordinates are transformed into latent-space representations and then decoded to obtain the physical quantity.

Recent work on neural fields has shown that domain-oriented input feature encodings can significantly boost reconstruction quality. Chabra et al. [CLI*20] proposed laying out trainable parameters in a grid of latent features to learn spatial variations more directly. Refinements include adaptive data structures [MLL*21], fixed multi-resolution grids [TLY*21], and multi-resolution spatial hashing [MESK22, MRNK21a], enabling multi-scale learning of spatial feature maps. Comprehensive reviews of SRN-related literature focusing on neural scene representations are available [HSB*20, TFT*20].

In scientific data visualization, Lu et al. [LJLB21] introduced SRNs for volumetric data compression, which was sped up and refined by Weiss et al. [WHW22] through the use of trainable feature representations in combination with an efficient GPU implementation. Höhle et al. [HWW22] employ neural fields for compressing ensemble data by sharing model parameters between different ensemble members. Both works demonstrate the combination of volume rendering and network inference as used in this study.

Correlation visualization Volume rendering was chosen to demonstrate network-based reconstruction for correlation visualization in interactive workflows. However, alternative techniques for correlation visualization have been proposed, including clustering [PW12, LWS18, EHL21], correlation matrices [CWMW11, EHL21], diagram views [STS06, BDSW13, ZMZM14, LS16], and specific feature-based approaches like analyzing dependencies in flow fields with particle trajectories [BMLC19]. Correlation subsampling [GW10, CWMW11] identifies prominent features in correlation fields for analysis. These approaches complement our contribution, as they address significant structures in correlation fields or develop effective visual encodings. Our approach seamlessly integrates with these techniques, reducing memory and computation requirements for accessing correlations in large 3D fields.

3. Statistical dependence in ensemble fields

We quantify statistical dependencies in ensemble datasets through bivariate correlation measures $\rho : \mathbb{R}^N \times \mathbb{R}^N \rightarrow \mathbb{R}$ between vectors of paired random samples. For this, let $\Omega \subset \mathbb{R}^3$ be a simulation domain in 3D space, and let $\mathcal{E} = \{E_i : 0 \leq i < N\}$ be an ensemble of N multi-variable fields $E_i : \Omega \rightarrow \mathbb{R}^d$. The index i suggests a fixed but arbitrary enumeration of the members. For all i and $0 \leq v < d$, let $E_i^v : \Omega \rightarrow \mathbb{R}$ denote the scalar field associated with variable v in member E_i . For a given position $\mathbf{p} \in \Omega$, we refer to the local sample of variable values as $\mathbf{e}^v(\mathbf{p}) := (E_i^v(\mathbf{p}) : 0 \leq i < N) \in \mathbb{R}^N$. Then, for variables μ and v , we consider the field of μ - v -correlations, $R_{\mu v} : \Omega \times \Omega \rightarrow \mathbb{R}$, where for all pairs of positions $(\mathbf{p}_\mu, \mathbf{p}_v) \in \Omega^2$ the field value is defined as $R_{\mu v}(\mathbf{p}_\mu, \mathbf{p}_v) := \rho(\mathbf{e}^\mu(\mathbf{p}_\mu), \mathbf{e}^v(\mathbf{p}_v))$. Note that the indexing of the position variables is used to imply that position $\mathbf{p}_\mu$ (position $\mathbf{p}_v$) alters the value of $R_{\mu v}(\cdot, \cdot)$ by changing the reference

position in field μ (field v), respectively. Special attention is paid to the case $\mu = v$, which we refer to as the μ -self-correlation field $S_\mu := R_{\mu\mu}$. Notably, S_μ is symmetric under exchange of position coordinates, i.e., $S_\mu(\mathbf{p}_1, \mathbf{p}_2) = S_\mu(\mathbf{p}_2, \mathbf{p}_1)$ for all $\mathbf{p}_1, \mathbf{p}_2 \in \Omega$.

This work uses Pearson correlation and MI from the wide range of possible dependence measures. Both represent opposite sides of the spectrum of computational cost and indicate different kinds of dependence [BMLC19].

3.1. Pearson product-moment correlation coefficient

The Pearson correlation coefficient, or Pearson's r , measures the linear correlation between random variable pairs. It is commonly used in data visualization to explore relationships between variables. Given a set of paired random samples, $\mathbf{e}_1, \mathbf{e}_2 \in \mathbb{R}^N$, the Pearson correlation coefficient is defined as

$$r = \frac{\text{cov}(\mathbf{e}_1, \mathbf{e}_2)}{\sqrt{\text{var}(\mathbf{e}_1) \text{var}(\mathbf{e}_2)}}, \quad (1)$$

wherein $\text{var}(\cdot)$ and $\text{cov}(\cdot)$ denote sample variance and covariance of the respective random samples. With a range of -1 to 1, Pearson correlation indicates correlation (+1), anti-correlation (-1), or the absence of correlation (0). It is easy to interpret, quantifying the strength and direction of the relationship between variables. However, caution is needed when the relationship is nonlinear or when outliers are present, as they can significantly affect the correlation coefficient.

3.2. Mutual information

MI is widely used in machine learning, statistics, and information theory [CT*91] to measure similarity or correlation between random variable pairs. Unlike linear correlation, MI can detect nonlinear and non-monotonic dependencies that are not evident in covariance. Mathematically, this is expressed as

$$I(X_1; X_2) = H(X_1) - H(X_1|X_2) = H(X_2) - H(X_2|X_1), \quad (2)$$

wherein X_1 and X_2 are random variables, $I(X_1; X_2)$ is the MI of X_1 and X_2 , $H(X_1)$ is the entropy of X_1 , and $H(X_1|X_2)$ is the conditional entropy of X_1 given X_2 . Note that MI is symmetric under the exchange of X_1 and X_2 .

Estimating MI from finite samples $\mathbf{e}_1$ and $\mathbf{e}_2$ of random variables X_1 and X_2 , i.e. computing $I(\mathbf{e}_1, \mathbf{e}_2)$, is computationally expensive. Existing algorithms struggle to scale with large sample sizes [KSG04, MRL95]. More recent copula-based and neural-network-based variational MI estimators offer better performance in high-dimensional data spaces but still pose computational challenges [ZD11, BBR*18]. In our study, we compute ground truth MI fields using the nearest-neighbor-based estimator of Kraskov et al. [KSG04], implemented in parallel. However, detailed performance analysis in section 6 shows that estimating MI fields for visualization exceeds the time constraints of interactive data analysis. Mutual information neural estimation (MINE) is a recent method that uses neural networks to estimate MI [BBR*18]. It trains a neural network to learn a lower bound on MI, which provides an estimate. MINE has shown better performance than traditional MI estimation methods on benchmark datasets. Nevertheless, in our sce-

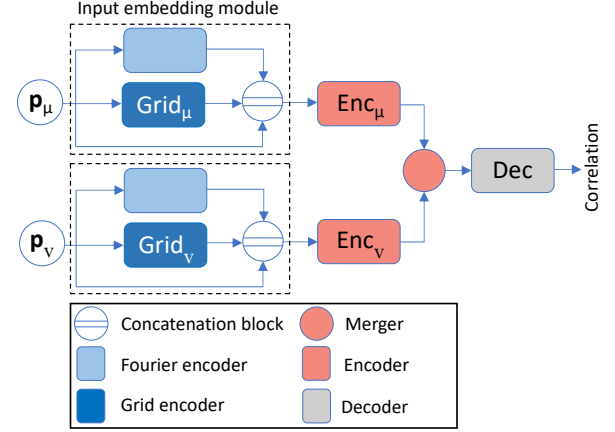


Figure 2: NDF architecture. $\mathbf{p}_\mu$ and $\mathbf{p}_v$ are respectively the reference and query positions. Grid_μ and Grid_v , respectively, are the hash grids. Variable-specific encoders Enc_μ and Enc_v are MLPs. Since the feature grids involve trainable parameters, each encoder is equipped with a separate grid. Merger indicates the multiplication of the encoder outputs.

nario, the inefficiency arises because MINE operates on a member-wise basis and requires all members to be present in memory.

4. Neural dependence fields

To enable the use of large sets of point-to-point dependence measures, we perform the computationally expensive calculations of these measures in a preprocess and encode the two-point μ - v -correlation fields $R_{\mu v}$ (as defined in section 3) with memory- and compute-efficient neural scene representations, $\Phi_{\mu v} : \Omega \times \Omega \rightarrow \mathbb{R}$. The network is obtained by solving the optimization problem

$$\Phi_{\mu v} := \arg \min_{\Phi} \mathbb{E}_{(\mathbf{p}_1, \mathbf{p}_2) \sim \mathcal{U}(\Omega^2)} [d(\Phi(\mathbf{p}_1, \mathbf{p}_2), R_{\mu v}(\mathbf{p}_1, \mathbf{p}_2))], \quad (3)$$

wherein $\Phi(\cdot, \cdot)$ is the neural network, $d(\cdot, \cdot)$ is a similarity metric, such as L_1 or L_2 loss, and the expectation $\mathbb{E}[\cdot]$ is taken over samples of position pairs from a uniform distribution, $\mathcal{U}(\Omega^2)$, with support Ω^2 . The optimization is carried out iteratively using stochastic gradient descent. Using tractably sized batches of position pairs simultaneously avoids storing excessive amounts of correlation samples. After training, $\Phi_{\mu v}$ is a compact correlation field encoding, enabling rapid sample reconstruction. Classical compression methods like TThresh and SZ cannot achieve similar compaction due to fixed discretization and computational infeasibility. Additionally, once the correlation network is trained, there is no need to keep the entire ensemble dataset in memory, allowing the approach to scale efficiently to large ensemble sizes.

4.1. Network architecture

NDFs differ from classical neural scene representations as they operate on a bi-spatial domain with six dimensions (6D). Due to the curse of dimensionality, training efficiency is lowered by the additional dimensions since covering the domain adequately with samples becomes exponentially more complex. To overcome this, we

construct NDFs to utilize sparse sampling information efficiently, which improves memory efficiency and avoids the computation of excessive amounts of correlation samples.

As shown in Figure 2, we propose a bipartite network architecture, which consists of two variable-specific encoder networks, Enc_μ , and Enc_ν , along with a shared decoder network. All networks are implemented as multi-layer perceptrons (MLPs) with l fully-connected layers and c hidden channels using *SnakeAlt* activation [WHW22]. Each encoder model receives information about one of the two positions between which the correlation should be reconstructed. Positions are translated into a latent feature vector, which is merged via element-wise multiplication and forwarded to the decoder for the final prediction. This architecture allows each encoder to be trained on only the marginal space $\Omega \subset \mathbb{R}^3$, which improves the training efficiency by increasing the effective amount of correlation samples per volume. In combination with the spatial coherence of the ensemble fields, this enables the model to infer correlations even for point pairs that were not seen during training, thus saving computation time and memory requirements.

The decomposition is similar in spirit to the approach used in TensorRF [CXG*22] or K-planes [FKMW*23], where 3D feature tensors are decomposed into linear combinations of tensor products between lower-dimensional feature vectors and matrices for higher parameter efficiency. In the proposed NDF, features over a 6D domain are decomposed into an outer product of fields over a 3D domain. The accuracy of predictions relies heavily on using multiplication for feature merging. Alternative combination methods, such as concatenation, addition or absolute difference, result in a substantial drop in prediction fidelity, which is in line with findings in [FKMW*23]. Deviating from TensorRF and K-planes, we found applying MLPs before and after feature merging beneficial, which we validate in more detail in section 6.

For self-correlation fields S_μ (as defined in section 3) and the corresponding NDFs $\Phi_{\mu\mu}$, we further constrain the architecture to use identical encoders for both positions, i.e., $\text{Enc}_\mu = \text{Enc}_\nu$ (and $\text{Grid}_\mu = \text{Grid}_\nu$, see below for details). This helps to keep the models small and ensures symmetry of the learned fields under exchange of the query positions on an architectural level, i.e., $\Phi_{\mu\mu}(\mathbf{p}_1, \mathbf{p}_2) = \Phi_{\mu\mu}(\mathbf{p}_2, \mathbf{p}_1)$ is fulfilled trivially by design and does not need to be learned in expensive training iterations. To improve the capability of the encoders to learn high-frequency patterns as well as spatially distributed and multi-scale features, we employ Fourier features [MST*21, TSM*20] as well as multi-resolution hash-grids [MESK22] on the position coordinates, which are concatenated to the raw positions before being processed by the encoders. The input embedding modules are marked with dashed rectangles in Figure 2.

4.2. Input embedding

For a given vector of input coordinates, $\mathbf{p} = (p_x, p_y, p_z) \in \mathbb{R}^3$, Fourier features increase the spread between spatially close positions by embedding the position information into a higher-dimensional space using the fixed feature mapping

$$\mathbf{f}_{ij} = (\sin(\omega_i \mathbf{n}_j \cdot \mathbf{p}), \cos(\omega_i \mathbf{n}_j \cdot \mathbf{p})), \quad (4)$$

wherein $\omega_i = 2^i \pi$ for $0 \leq i < L \in \mathbb{N}$, and $\mathbf{n}_j \in \mathbb{R}^3$ are the axis-aligned unit vectors for $j \in \{x, y, z\}$. With Fourier features, the model can better resolve high-frequency patterns while not affecting the number of trainable parameters (and thus memory consumption) due to the fixed functional form of the mapping. In our implementation, we empirically determined $L = 12$ as a good choice for the number of Fourier frequencies.

Multi-resolution hash grids use hash tables filled with trainable feature vectors to populate the domain at various resolutions [MESK22]. By hashing 3D grid indices, vectors are assigned to regular grid positions at multiple scales, enabling retrieval of feature vectors at arbitrary positions through tri-linear interpolation. This allows the creation of virtual feature grids at any resolution with a fixed memory budget.

The feature vectors for different resolution levels are concatenated and trained jointly with subsequent model parts using stochastic gradient descent. Hash collisions equilibrate during training due to the pseudo-random hash mapping and multiple resolutions, ensuring adaptive and local feature capacity distribution. The critical parameter for the expressiveness of the hash grid is the hash table size, 2^T for $T \in \mathbb{N}$, which also determines memory complexity. Other hyper-parameters include the dimension of the feature vectors per resolution level, the number of resolution levels, and the grid resolution on each level. We use 6 resolution levels with virtual grids of size 16^3 on the coarsest level, doubling with each finer level. These parameters ensure that the feature granularity matches the spatial resolution of the original dataset, avoiding higher memory consumption with finer levels while maintaining or improving the reconstruction accuracy.

4.3. Training

During training, we use a rectangular simulation domain Ω with data samples on a regular grid, rescaled to fill the symmetric unit cube, i.e., $\Omega = [-1, 1]^3$. We generate 10^6 pairs of uniformly distributed random positions $(\mathbf{p}_\mu, \mathbf{p}_\nu)$ in $[-1, 1]^3$, retrieve samples $\mathbf{e}^\mu(\mathbf{p}_\mu)$ and $\mathbf{e}^\nu(\mathbf{p}_\nu)$ using trilinear interpolation in the original ensemble dataset, and compute correlations $R_{\mu\nu}(\mathbf{p}_\mu, \mathbf{p}_\nu)$. The models are trained to optimize the L_1 loss as a similarity measure, with L_2 loss yielding similar results in our experiments. We use the Adam optimizer [KB14] with an initial learning rate of 3×10^{-4} and 1000 samples per batch. An adaptive learning rate scheduler reduces the learning rate by a factor of 0.1 after 5 passes without improvement in reconstruction accuracy. After every epoch, the training samples are renewed. The total training duration is 200 epochs.

5. Correlation visualization

Once trained, the network can estimate dependencies for any position pair. Multiple queries can be batched efficiently for parallel processing on a GPU using the tiny-cuda-nn framework [Mü21], which features a fully-fused MLP implementation [MRNK21b] with fast 16-bit inference using tensor cores on NVIDIA GPUs and provides functionality for the multi-resolution hashed feature grids [MESK22]. The custom activation functions *Snake* and *SnakeAlt* [WHW22] were added to a fork of the library to support

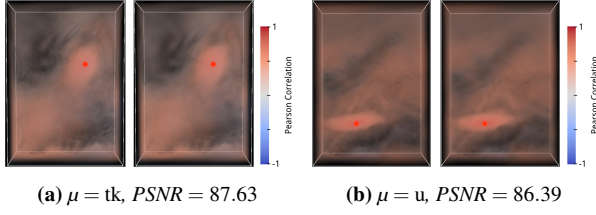


Figure 3: Point-to-point Pearson self-correlations S_μ for different variables μ (temperature tk and longitudinal component u of wind speed) and reference positions $\mathbf{p}_{\text{ref}}$. Figures show ground truth (left) and NDF reconstruction (right). The reference is shown in red.

the model architecture, as simpler activation functions like ReLU led to inferior reconstruction accuracy.

Network training is performed in PyTorch using the Python bindings of the library. To enable interactive visualizations, the binary weights of the MLP encoder and decoder can then be loaded by a tiny-cuda-nn module into whatever GPU correlation visualization is used. In our primary use case, we access the network from a GPU-based volume renderer implemented in Vulkan [The23]. The renderer is tied to a graphical interface, in which the user is able to select reference points $\mathbf{p}_{\text{ref}} \in [-1, 1]^3$, for which correlation samples are reconstructed and displayed as a density field. Specifically, we consider the case of displaying volumetric correlation fields, $\phi: [-1, 1]^3 \rightarrow \mathbb{R}$, where $\phi(\mathbf{p}) := \Phi_{\mu\nu}(\mathbf{p}, \mathbf{p}_{\text{ref}})$ or $\Phi_{\mu\nu}(\mathbf{p}_{\text{ref}}, \mathbf{p})$.

For visualization, the correlation fields are sampled on a grid with resolution $X \times Y \times Z$. A CUDA input buffer is prepared for the reference point $\mathbf{p}_{\text{ref}}$ and passed to the tiny-cuda-nn encoder module to get the encoded reference vector. The grid is divided into $\lceil (X \times Y \times Z)/M \rceil$ query batches of size M . Each batch is encoded, and the reference and query features are multiplied before being passed to the tiny-cuda-nn decoder module to obtain correlation estimates in an output buffer shared between Vulkan and CUDA, preventing race conditions with shared semaphores. Finally, the shared output buffer is copied to a 3D Vulkan image for visualization in the volume renderer.

6. Performance and Quality Analysis

Here, we showcase the NDF model for interactive visual analysis of spatial statistical dependencies in a large weather forecast ensemble. We examine the network’s reconstruction speed and memory requirements and compare the results to ground truth dependence fields obtained using Pearson correlation coefficients and MI on GPU and CPU.

6.1. Dataset

We validate our approach using a convective-scale multi-variable ensemble dataset (CSEns) by Necker et al. [NGW*20]. It consists of 1000 numerical simulations of a 3D atmospheric dynamics model over a rectangular region in central Europe, with a grid size of 250×352 nodes and 20 discrete height levels. The simulations span six hours, and we select the last time step with the most interesting features for visualization [HWW22]. The dataset

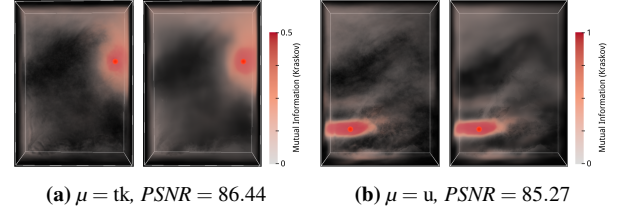


Figure 4: Point-to-point MI self-correlations S_μ for different variables μ (temperature tk and longitudinal component u of wind speed) and reference positions $\mathbf{p}_{\text{ref}}$. Figures show ground truth (left) and NDF reconstruction (right). The reference is shown in red.

includes 3D data for nine meteorological variables. For validation, we compute ground truth correlation fields for temperature (tk) and longitudinal wind (u) using all 1000 ensemble members. The generalization of our approach to multiple time steps and inter-temporal dependencies will be explored in future work.

6.2. NDF model performance

To shed light on the performance of the NDFs, we conducted a comparative analysis of the runtime of NDFs against reference implementations of Pearson correlation and MI. The implementation of the MI estimator follows Kraskov et al. [KSG04]. All performance measurements are based on the CSEns dataset and were performed on an NVIDIA RTX 3090 GPU and a 6-core Intel Xeon W-2235 CPU, respectively. Notably, a parallel MI estimator is only available on the CPU, whereas we restricted ourselves to a GPU implementation of NDF-based reconstruction. Table 1 shows that our model is about 26x faster than the GPU implementation of the Pearson correlation coefficient. Compared to the CPU implementation of the MI estimator, the factor is $114,106\times$. Once the NDF model is trained, it takes 9 ms to reconstruct the dependencies between the data values at a selected grid point and all other grid points. The network model requires roughly 1 GB of memory while keeping the entire dataset in memory for one variable would amount to 7 GB. Training of the NDF takes approximately one hour on the aforementioned machine. Even though it is difficult to estimate the performance of a GPU implementation of the MI estimator, it can be assumed that even an optimized GPU-accelerated estimator will be significantly slower than the NDF model. The MI estimator must construct search structures for nearest-neighbour queries for all requested samples, a much more elaborate process than passing data through fully-fused MLP kernels.

Table 1: Performance comparison. For a selected grid point, the dependence measures are computed for all other grid points in a $250 \times 352 \times 20$ simulation grid using a single variable. A GPU implementation of the MI estimator is not available.

	NDF (ours)	Pearson	MI [KSG04]
CPU	—	4772 ms	1026957 ms
GPU	9 ms	234 ms	—

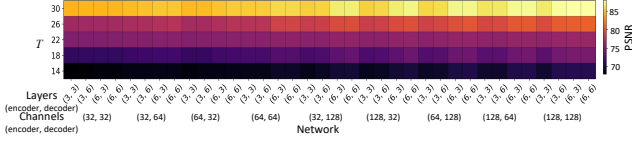


Figure 5: Impact of hash table size and MLP hyperparameters on NDFs’ reconstruction quality. The horizontal axis displays various encoder and decoder configurations, including the number of layers and hidden channels. The vertical axis shows the log-2 hash table size T from section 4.1. The plot considers variable temperature (tk) and Pearson correlation, with similar behavior observed for other variables and similarity metrics.

6.3. NDF model accuracy

Figures 1, 3, and 4 visually compare the network’s ability to reconstruct major dependence structures in the variable fields. While fine details may not be equally well reproduced due to the limited number of correlation samples during model training, the proposed bipartite NDF architecture uses sample information efficiently by increasing the effective sampling density for the encoder parts of the model (see section 4.1). Furthermore, experiments with increased sample counts did not significantly improve reconstruction, indicating that sampling density is not the limiting factor for reconstruction accuracy. A trade-off between model capacity (memory size) and reconstruction quality is observed, suggesting that the quality is primarily bounded by the model’s ability to store training information rather than sample availability. The figures displayed dependence fields for selected reference points, representing 3D slices in a much larger 6D correlation space where the model was trained. Storing all possible two-point correlations in 32-bit floating point format would require 6 TB of memory space for the CSEns grid of size $250 \times 352 \times 20$. The resulting network size of 1 GB corresponds to an effective compression factor of over $6,000\times$, showing promising results.

Figure 5 shows the trade-off between model size and reconstruction quality. NDFs are trained with various settings for the hash-table size 2^T and different complexities of encoder and decoder MLPs, using Pearson self-correlation fields of variable tk . PSNR values are computed on a set of 10^6 position pairs, sampled uniformly from the grid domain. Models are trained for a shorter duration of 50 epochs for efficiency. The figure indicates a minor advantage for models with more complex encoder and decoder MLPs. However, the most significant factor affecting achievable PSNR is the hash table size, which also strongly affects model memory consumption. Increasing the table size leads to higher PSNR values. A doubling of the table size 2^T roughly doubles the model memory requirements, while the volume of the MLP parameterization is limited to only a few kB, making it negligible. In our implementation, we set the maximum table size to less than 2^{32} . For further experiments, we choose $T = 30$ with 6-layer encoders and decoders, each having 128 channels per layer.

To validate our design choices against TensorRF [CXG*22] and K-planes [FKMW*23], we compare the reconstruction quality of the proposed architecture against a TensorRF-like model, where the MLP in the encoder part is omitted, predicting based solely on the

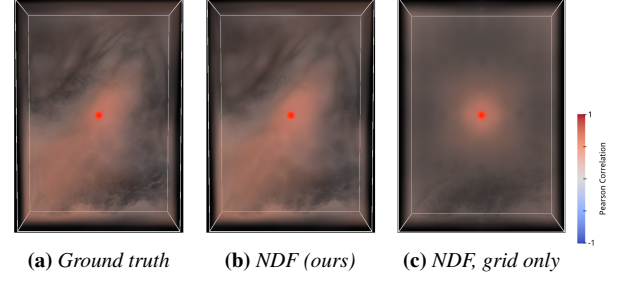


Figure 6: Accuracy comparison between NDFs with MLP encoder (complete) and pure grid model (without encoder) using Pearson self-correlation fields for variable temperature (tk).

outputs of the input embedding. Figure 6 displays reconstructed Pearson correlation fields for both approaches, highlighting the significant added value of using the MLP before feature merging.

6.4. Additional experiments

The following are quantitative results using NDFs for reconstructing Pearson correlation coefficients and MI values. We begin with experiments on variable self-correlation fields S_μ and their corresponding networks $\Phi_{\mu\mu}$.

Single-point experiment Firstly, the user selects a point in the 3D domain, and the dependence field $\phi(\mathbf{r})$ is instantly displayed via volume rendering (see supplementary video). Volumetric visualizations, including reconstruction and rendering, can be generated below 10 ms. Moving the reference point interactively highlights different regions with high correlation. This allows the user to identify areas of high internal correlation or observe how correlations change with distance from the reference. The transfer function can be adjusted interactively for better visibility of specific features during the analysis.

Multi-point experiment Secondly, the user selects two points in the domain, and the difference between the reconstructed dependence fields for each point is visualized (Figure 7a for Pearson correlation and Figure 7b for MI). The images show the correlation decay around the points and reveal additional structures in other areas within the fields. This allows users to efficiently analyze the differences in the dependence structures related to different points in the 3D domain.

Multi-variable experiment Our last experiment demonstrates the use of NDFs for analyzing spatial dependencies between different variables. For a set of $d \in \mathbb{N}$ variable fields, i.e., $V = \{v_1, v_2, \dots, v_d\}$, this requires training of $d(d+1)/2$ NDFs $\Phi_{v_i v_j}$, for each of the combinations $v_i, v_j \in V$ with $1 \leq i < j \leq d$. For a pair of two different physical variables, such as $v_1 = tk$ and $v_2 = u$, this amounts to training three NDFs, $\Phi_{tk,tk}$, $\Phi_{u,u}$ and $\Phi_{tk,u}$, which emulate the corresponding correlation fields. Note that no separate model is required for $\Phi_{u,tk}$ if the underlying correlation measure ρ (see section 3) is symmetric under exchange of arguments, since due to the symmetric architecture of NDFs $\Phi_{u,tk}(\mathbf{p}_1, \mathbf{p}_2) = \Phi_{tk,u}(\mathbf{p}_2, \mathbf{p}_1)$ for all $\mathbf{p}_1, \mathbf{p}_2 \in \Omega$.

Elaborating on the example of tk and u , we propose a matrix-like

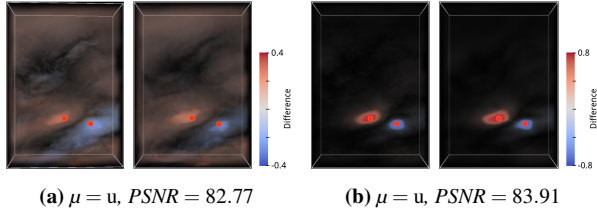


Figure 7: Difference field visualization for multiple points in longitudinal component u of wind speed. a) Pearson correlation field for two different points, ground truth (left), model reconstruction (right). b) MI for two different points, ground truth (left), and model reconstruction (right).

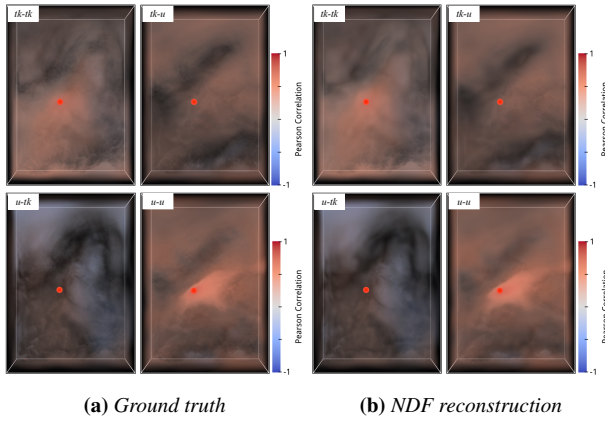


Figure 8: Visualizing dependencies between variables temperature (tk) and longitudinal component u of wind speed using Pearson correlation coefficients. a) Ground truth, b) NDF model reconstruction. Fields show correlations between variables tk or u at selected points to the same or different variable at other points.

arrangement of linked volumetric correlation visualizations in the spirit of standard correlation matrix visualizations, i.e., correlation volume matrices. For this, a single reference point is selected, and correlation fields with respect to this point are rendered for all NDF configurations, i.e., all combinations of variables. Figure 8 shows an example of this.

NDFs enable easier visualization with multiple variables as they reduce computation time and memory usage compared to on-the-fly computations using raw data. The standard method would require all variables' data in GPU memory, and even high-end GPUs with 24 GB of memory can only load two variable fields simultaneously in practice. NDFs of 1 GB each allow networks for up to four variables to fit into the same memory. Exploring the reuse of variable-specific encoder grids in different networks could prevent the memory requirements of NDFs from growing quadratically with the number of variables.

7. Conclusion and Future Work

We have introduced and evaluated neural dependence fields (NDFs), a novel approach for encoding and visualizing statistical

dependencies in large 3D ensemble fields. NDFs infer spatial dependencies within single variables and in pairs of different variables. They offer compact representations of linear and non-linear dependence patterns in large ensembles, facilitating the rapid reconstruction of correlation samples from the compact representation. We demonstrated interactive visual analysis of 3D dependence structures through GPU-accelerated direct volume rendering.

Our evaluations show that NDFs faithfully encode and reconstruct the prominent dependence structures in 3D fields while smoothing out some details due to limited network capacities. In the future, we aim to enhance these capacities by adding network stages in the encoder and decoder, exploring alternative architectures like diffusion networks, and trying different loss functions for preserving fine details better (e.g., gradient regularization [LJLB21]). Additionally, we plan to extend NDFs to infer temporal dependence structures in time-varying ensemble fields by decomposing the data into independent fields with spatial variation. This extension would enable new application scenarios in scientific workflows, such as ensemble sensitivity analysis [KRRW19].

Acknowledgments

The research leading to these results has been done within the sub-project “A7” of the Transregional Collaborative Research Center SFB / TRR 165 “Waves to Weather” (www.wavestoweather.de) funded by the German Research Foundation (DFG). The authors acknowledge crucial contributions by Juan Ruiz and the RIKEN Data Assimilation Research Team for conducting the 1000-member ensemble simulation.

References

- [BBR*18] BELGHAZI M. I., BARATIN A., RAJESHWAR S., OZAI S., BENGIO Y., COURVILLE A., HJELM D.: Mutual information neural estimation. In *Proceedings of the 35th International Conference on Machine Learning* (10–15 Jul 2018), Dy J., Krause A., (Eds.), vol. 80 of *Proceedings of Machine Learning Research*, PMLR, pp. 531–540. 3
- [BDSW13] BISWAS A., DUTTA S., SHEN H.-W., WOODRING J.: An information-aware framework for exploring multivariate data sets. *IEEE Transactions on Visualization and Computer Graphics* 19, 12 (2013), 2683–2692. doi:10.1109/TVCG.2013.133. 2
- [BMLC19] BERENJKOUB M., MONICO R. O., LARAMEE R. S., CHEN G.: Visual analysis of spatio-temporal relations of pairwise attributes in unsteady flow. *IEEE Transactions on Visualization and Computer Graphics* 25, 1 (2019), 1246–1256. doi:10.1109/TVCG.2018.2864817. 2, 3
- [CLI*20] CHABRA R., LENNSEN J. E., ILG E., SCHMIDT T., STRAUB J., LOVEGROVE S., NEWCOMBE R.: Deep local shapes: Learning local sdf priors for detailed 3d reconstruction. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIX 16* (2020), Springer, pp. 608–625. 2
- [CT*91] COVER T. M., THOMAS J. A., ET AL.: Entropy, relative entropy and mutual information. *Elements of information theory* 2, 1 (1991), 12–13. 3
- [CWMW11] CHEN C.-K., WANG C., MA K.-L., WITTENBERG A. T.: Static correlation visualization for large time-varying volume data. In *2011 IEEE Pacific Visualization Symposium* (2011), pp. 27–34. doi:10.1109/PACIFICVIS.2011.5742369. 2
- [CXG*22] CHEN A., XU Z., GEIGER A., YU J., SU H.: Tensorf: Tensorial radiance fields. In *Computer Vision–ECCV 2022: 17th European Conference, 2022, Proceedings, Part XXXII* (2022), Springer, pp. 333–350. 4, 6

- [CZ19] CHEN Z., ZHANG H.: Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2019), pp. 5939–5948. 2
- [EHL21] EVERS M., HUESMANN K., LINSSEN L.: Uncertainty-aware Visualization of Regional Time Series Correlation in Spatio-temporal Ensembles. *Computer Graphics Forum* (2021). doi:10.1111/cgfm.14326. 2
- [FH23] FAROKHMANESH F., HÖHLEIN K.: Neural Fields for Interactive Visualization of Statistical Dependencies in 3D Simulation Ensembles: Code for Experiments, July 2023. doi:10.5281/zenodo.8186686. 2
- [FKMW*23] FRIDOVICH-KEIL S., MEANTI G., WARBURG F., RECHT B., KANAZAWA A.: K-planes: Explicit radiance fields in space, time, and appearance. *arXiv preprint arXiv:2301.10241* (2023). 4, 6
- [GW10] GU Y., WANG C.: A study of hierarchical correlation clustering for scientific volume data. In *Advances in Visual Computing: 6th International Symposium, ISVC 2010, 1, 2010, Proceedings, Part III* 6 (2010), Springer, pp. 437–446. 2
- [HSB*20] HOANG D., SUMMA B., BHATIA H., LINDSTROM P., KLANCANSKY P., USHER W., BREMER P.-T., PASCUCCI V.: Efficient and flexible hierarchical data layouts for a unified encoding of scalar field precision and resolution. *IEEE Transactions on Visualization and Computer Graphics* 27, 2 (2020), 603–613. 2
- [HWW22] HÖHLEIN K., WEISS S., WESTERMANN R.: Evaluation of volume representation networks for meteorological ensemble compression. 2, 5
- [KB14] KINGMA D. P., BA J.: Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014). 4
- [KRRW19] KUMPF A., RAUTENHAUS M., RIEMER M., WESTERMANN R.: Visual analysis of the temporal evolution of ensemble forecast sensitivities. *IEEE Transactions on Visualization and Computer Graphics* 25, 1 (2019). doi:10.1109/TVCG.2018.2864901. 7
- [KSG04] KRASKOV A., STÖGBAUER H., GRASSBERGER P.: Estimating mutual information. *Phys. Rev. E* 69 (Jun 2004), 066138. doi:10.1103/PhysRevE.69.066138. 3, 5
- [LJLB21] LU Y., JIANG K., LEVINE J. A., BERGER M.: Compressive neural representations of volumetric scalar fields. In *Computer Graphics Forum* (2021), vol. 40, Wiley Online Library, pp. 135–146. 2, 7
- [LS16] LIU X., SHEN H.-W.: Association analysis for visual exploration of multivariate scientific data sets. *IEEE Transactions on Visualization and Computer Graphics* 22, 1 (2016), 955–964. doi:10.1109/TVCG.2015.2467431. 2
- [LWS18] LIEBMAN T., WEBER G. H., SCHEUERMANN G.: Hierarchical correlation clustering in multiple 2d scalar fields. *Computer Graphics Forum* 37, 3 (2018), 1–12. doi:https://doi.org/10.1111/cgfm.13396. 2
- [MESK22] MÜLLER T., EVANS A., SCHIED C., KELLER A.: Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.* 41, 4 (jul 2022). doi:10.1145/3528223.3530127. 2, 4
- [MLL*21] MARTEL J. N., LINDELL D. B., LIN C. Z., CHAN E. R., MONTEIRO M., WETZSTEIN G.: Acorn: Adaptive coordinate networks for neural scene representation. *arXiv preprint arXiv:2105.02788* (2021). 2
- [MON*19] MESCHEDER L., OECHSLE M., NIEMEYER M., NOWOZIN S., GEIGER A.: Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (2019), pp. 4460–4470. 2
- [MRL95] MOON Y.-I., RAJAGOPALAN B., LALL U.: Estimation of mutual information using kernel density estimators. *Physical Review E* 52, 3 (1995), 2318. 3
- [MRNK21a] MÜLLER T., ROUSSELLE F., NOVÁK J., KELLER A.: Real-time neural radiance caching for path tracing. *arXiv preprint arXiv:2106.12372* (2021). 2
- [MRNK21b] MÜLLER T., ROUSSELLE F., NOVÁK J., KELLER A.: Real-time neural radiance caching for path tracing. *ACM Trans. Graph.* 40, 4 (jul 2021). doi:10.1145/3450626.3459812. 4
- [MST*21] MILDENHALL B., SRINIVASAN P. P., TANCİK M., BARRON J. T., RAMAMOORTHY R., NG R.: Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM* 65, 1 (2021), 99–106. 2, 4
- [Mü21] MÜLLER T.: tiny-cuda-nn, 4 2021. URL: <https://github.com/NVlabs/tiny-cuda-nn>. 4
- [NGW*20] NECKER T., GEISS S., WEISSMANN M., RUIZ J., MIYOSHI T., LIEN G.-Y.: A convective-scale 1,000-member ensemble simulation and potential applications. *Quarterly Journal of the Royal Meteorological Society* 146, 728 (2020), 1423–1442. doi:https://doi.org/10.1002/qj.3744. 5
- [NS23] NEUHAUSER C., STUMPFEGGER J.: chrismile/Corrender: A correlation field renderer using the Vulkan graphics API, v2023-07-29, July 2023. doi:10.5281/zenodo.8195623. 2
- [PFS*19] PARK J. J., FLORENCE P., STRAUB J., NEWCOMBE R., LOVEGROVE S.: DeepSDF: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (2019), pp. 165–174. 2
- [PW12] PFAFFELMOSE T., WESTERMANN R.: Visualization of global correlation structures in uncertain 2d scalar fields. In *Computer Graphics Forum* (2012), vol. 31, Wiley Online Library, pp. 1025–1034. 2
- [STS06] SAUBER N., THEISEL H., SEIDEL H.-P.: Multifield-graphs: An approach to visualizing correlations in multifield scalar data. *IEEE Transactions on Visualization and Computer Graphics* 12, 5 (2006), 917–924. doi:10.1109/TVCG.2006.165. 2
- [TFT*20] TEWARI A., FRIED O., THIES J., SITZMANN V., LOMBARDI S., SUNKAVALLI K., MARTIN-BRUALLA R., SIMON T., SARAGIH J., NIESSNER M., ET AL.: State of the art on neural rendering. In *Computer Graphics Forum* (2020), vol. 39, Wiley Online Library, pp. 701–727. 2
- [The23] THE KHROSOS VULKAN WORKING GROUP: Vulkan 1.3.245 - A Specification. <https://registry.khronos.org/vulkan/specs/1.3-extensions/html/vkspec.html>, 2023. Accessed: 2023-03-30. 5
- [TLY*21] TAKIKAWA T., LITALIEN J., YIN K., KREIS K., LOOP C., NOWROUZEZAHRAI D., JACOBSON A., MCGUIRE M., FIDLER S.: Neural geometric level of detail: Real-time rendering with implicit 3d shapes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2021), pp. 11358–11367. 2
- [TSM*20] TANCİK M., SRINIVASAN P., MILDENHALL B., FRIDOVICH-KEIL S., RAGHAVAN N., SINGHAL U., RAMAMOORTHY R., BARRON J., NG R.: Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in Neural Information Processing Systems* 33 (2020), 7537–7547. 2, 4
- [WHW22] WEISS S., HERMÜLLER P., WESTERMANN R.: Fast neural representations for direct volume rendering. In *Computer Graphics Forum* (2022), vol. 41, Wiley Online Library, pp. 196–211. 2, 4
- [XTS*22] XIE Y., TAKIKAWA T., SAITO S., LITANY O., YAN S., KHAN N., TOMBARI F., TOMPKIN J., SITZMANN V., SRIDHAR S.: Neural fields in visual computing and beyond. In *Computer Graphics Forum* (2022), vol. 41, Wiley Online Library, pp. 641–676. 2
- [ZD11] ZENG X., DURRANI T.: Estimation of mutual information using copula density function. *Electronics letters* 47, 8 (2011), 493–494. 3
- [ZMZM14] ZHANG Z., McDONNELL K. T., ZADOK E., MUELLER K.: Visual correlation analysis of numerical and categorical data on the correlation map. *IEEE transactions on visualization and computer graphics* 21, 2 (2014), 289–303. 2

A Compact and Efficient Neural Data Structure for Mutual Information Estimation in Large Timeseries

Fatemeh Farokhmanesh
Technical University of Munich
Munich, Germany
fatemeh.farokhmanesh@tum.de

Christoph Neuhauser
Technical University of Munich
Munich, Germany
christoph.neuhauser@tum.de

Rüdiger Westermann
Technical University of Munich
Munich, Germany
westermann@tum.de

ABSTRACT

Database systems face challenges when using mutual information (MI) for analyzing non-linear relationships between large timeseries, due to computational and memory requirements. Interactive workflows are especially hindered by long response times. To address these challenges, we present timeseries neural MI fields (TNMIFs), a compact data structure that has been trained to reconstruct MI efficiently across various time-windows and window positions in large timeseries. We demonstrate learning and reconstruction with a large timeseries dataset comprising 1420 timeseries, each storing data at 1639 timesteps. While the learned data structure consumes only 45 megabytes, it answers queries for the MI estimates between the windows in a selected timeseries and the corresponding windows in all other timeseries within 44 milliseconds. Given a measure of similarity between timeseries based on windowed MI estimates, even the matrix showing all mutual timeseries similarities can be computed in less than 32 seconds. To support measuring dependence between lagged timeseries, an extended data structure learns to reconstruct MI to positively (future) and negatively (past) lagged windows. Using a maximum lag of 64 in both directions decreases query times by about a factor of 10.

CCS CONCEPTS

• Information systems → Data management systems.

KEYWORDS

Timeseries Analysis, Neural Data Structures, Mutual Information

ACM Reference Format:

Fatemeh Farokhmanesh, Christoph Neuhauser, and Rüdiger Westermann. 2024. A Compact and Efficient Neural Data Structure for Mutual Information Estimation in Large Timeseries. In *36th International Conference on Scientific and Statistical Database Management (SSDBM 2024)*, July 10–12, 2024, Rennes, France. ACM, New York, NY, USA, 12 pages. <https://doi.org/10.1145/3676288.3676295>

1 INTRODUCTION

Mutual information (MI) is used to express types of dependence between two random variables. I.e., when a random variable Y is

known, the MI $I(X, Y)$ measures how much this tells about the variable X . In contrast to the classical Pearson or Spearman correlation coefficients, MI captures linear as well as non-linear dependencies. This property makes MI appealing for the analysis of timeseries, e.g., by using the dependence between the data in different time-windows for classification and feature extraction.

We denote the number of timeseries in a collection as T and the length of each timeseries as N . Each timeseries A^j ($0 \leq j < T$) is represented as $(a_1^j, a_2^j, \dots, a_N^j)$, forming an ordered sequence of N data points. The i -th value of series A^j is denoted as $a_i^j \in \mathbb{R}$. Regarding the sliding window approach, considering a timeseries A^j with a length of N , there exist $(N - l + 1)$ sliding time-windows (or just windows) of length l (with an increment of 1):

$$\text{Windows}(A^j) = \{A_{1,l}^j, A_{2,l}^j, \dots, A_{(N-l+1),l}^j\} \quad (1)$$

An efficient estimation of the MI between two discrete random variables is challenging. It requires computing the conditional entropy of the two variables by using their joint histograms to estimate the joint discrete probability density function. The Kraskov MI estimator [21] derives entropy estimates from the Chebyshev distances between the k -th nearest neighbors in the joint distribution space. For the timeseries we consider, estimating Kraskov MI for all sliding windows in a selected timeseries and their corresponding windows in all other timeseries takes several seconds. The computation time further increases when the dependence between lagged timeseries is considered, i.e., for each window in a selected timeseries, the MI values to a set of positively (future) and negatively (past) lagged windows in each of the other timeseries are computed.

To address the computational bottleneck, we introduce the use of neural networks for MI-based timeseries analysis. It has been shown recently [11] that the major correlation structures in an ensemble of 3D physical fields can be learned by a compact neural network architecture. The network exploits the smoothness of the spatial correlation structures in physical fields. It learns a function that outputs the correlation values for a selected point to all other points in the domain. Evaluating the function, i.e., passing the coordinates of the input point and the output points through the network, requires only a few milliseconds, even for large numbers of ensemble members.

We demonstrate that a similar approach is possible for timeseries by considering the problem of estimating the MI between random variables that are represented by windows in different timeseries. We propose to interpret the field of two-window MI measures in timeseries as scalar fields $\mathbb{R}$ over the domain of timeseries pairs. The network infers in one pass, for every window in a selected timeseries, the MI between the data in this window and the corresponding windows in all other timeseries. Network inference is



This work is licensed under a Creative Commons Attribution International 4.0 License.

SSDBM 2024, July 10–12, 2024, Rennes, France
© 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-1020-9/24/07
<https://doi.org/10.1145/3676288.3676295>

performed in less than 44 milliseconds. This enables, for instance, to interactively select a timeseries and instantly convey all timeseries with constantly high linear or non-linear dependence over time.

To facilitate the analysis of dependence between lagged timeseries, the network is trained to reconstruct MI for both positively and negatively lagged time-windows. In our experiments, we consider all lags from -64 to 64. However, as required, the models can be trained on larger or smaller maximum lag values. It takes approximately 1.1 seconds to infer the MI values between the windows within the selected timeseries and all lagged windows in every other timeseries. We demonstrate this type of query with collections containing a timeseries as well as shifted versions of it. By using the MI measures between lagged timeseries, the lag can be determined, and shifted versions can be correctly classified.

2 RELATED WORK

Statistical dependence in timeseries datasets. Studying timeseries datasets is a common task across many scientific fields. The main goals are to measure the similarity between timeseries or subsequences over shorter time intervals for categorization and feature detection, as well as anomaly detection and temporal extrapolation. In a recent study conducted by Middlehurst et al. [27], various methods for timeseries analysis, including classification and feature extraction, are comprehensively reviewed. Fulcher et al. [13] and Lubba et al. [24] introduce two prominent toolboxes: the highly comparative analysis toolbox and the canonical timeseries characteristics toolbox. These toolboxes are designed to extract a diverse array of features, spanning basic statistics of timeseries values, linear correlations, and entropy, thereby covering a wide range of timeseries concepts. Fawaz et al. [18] discuss the impact of AlexNet and observe that there is significant potential for improvement in deep learning for time series classification, given the similarities in the data. Fawaz et al. [17] found that up to that point, ResNet (Wang et al. [44]) was the most accurate deep learner for timeseries classification.

Statistical dependence analysis is one of the most important tasks in the field of timeseries analysis and visualization. However, conducting these analyses at the level of subsequences of timeseries becomes excessively expensive for large datasets [1]. Mueen et al. [31], and Pietersz et al. [39] consider timeseries as a whole, which restricts their capacity to uncover related subsequences in the timeseries dataset. Li et al. [22] seek correlations among subsequences but confine their attention solely to synchronous subsequences between two timeseries, and as a result, the temporally misaligned subsequences are possibly overlooked. Ashan et al. [1] devise an efficient solution for conducting correlation-based analytics operations on timeseries data, encompassing tasks such as retrieving the best match, grouping correlated subsequences of two timeseries, and self-correlation. Köthür et al. [20] introduce a visual analytic method that expands upon windowed cross-correlation for comparing visually two timeseries based on statistical dependence. Notably, this approach does not scale to more than two timeseries. Evers et al. [10] introduce a novel approach for analyzing regional timeseries correlations within multi-field climate ensembles. Their method focuses solely on sets of selected segments extracted from these multi-field ensembles.

Neural fields and scene representation networks. Scene representation networks (SRNs) are neural networks trained to derive compact representations of 3D models and scenes. Initially, they were proposed for mapping 2D or 3D position coordinates. Early implementations involved encoding surface models as implicit functions or occupancy maps using fully connected neural networks [5, 26, 37]. Subsequently, they evolved to encode diverse volumetric scenery information, such as neural radiance fields [29, 42] and were termed neural fields [46]. A neural field is a neural network that learns a parametrization of spatio-temporal multi-dimensional physical fields over spatial coordinates. During inference, coordinates are transformed into latent-space representations and then decoded to obtain the physical quantity.

Recent advancements in neural fields have demonstrated that domain-oriented input feature encodings can significantly enhance reconstruction quality. Chabra et al. [4] proposed laying out trainable parameters in a grid of latent features to learn spatial variations more directly. Refinements include adaptive data structures [25], fixed multi-resolution grids [41], and multi-resolution spatial hashing [32, 33], enabling multi-scale learning of spatial feature maps. Comprehensive reviews of SRN-related literature focusing on neural scene representations are available [15, 43].

Lu et al. [23] introduced SRNs for volumetric data compression, which was sped up and refined by Weiss et al. [45] through the use of trainable feature representations in combination with an efficient GPU implementation. Höhle et al. [16] employ neural fields for compressing ensemble data by sharing model parameters between different ensemble members. Farokhmanesh et al. [11] utilize neural fields and present a neural network to efficiently learn and reconstruct statistical dependence structures via a compact neural network architecture. Our approach integrates with these methods, effectively reducing memory and computational demands for estimating MI in large timeseries datasets.

3 TIMESERIES DEPENDENCE ESTIMATION

As mentioned in Section 1, each timeseries in a collection is split into time-windows of size l . We assess non-linear dependence in timeseries datasets by employing MI measures between two windows, synchronous or lagged, with length l , $i : \mathbb{R}^l \times \mathbb{R}^l \rightarrow \mathbb{R}$. Consider a 2D space with one continuous dimension (sequence of data points) and one discrete dimension (collection of timeseries). We denote this space as $\mathbb{P}$, and let $\Phi \subset \mathbb{P}$. Further consider $\mathcal{W} = \{A_i : 0 < i < l\}$ as time-window that contains l timesteps $A_i : \Phi \rightarrow \mathbb{R}$. For a specific position $\mathbf{p} \in \Phi$, we refer to the sample of time-window as $\mathbf{a}(\mathbf{p}) := (A_i(\mathbf{p}) : 0 \leq i < l) \in \mathbb{R}^l$. The MI field is then a mapping $I : \Phi \times \Phi \rightarrow \mathbb{R}$, where for all pairs of positions $(\mathbf{p}_j, \mathbf{p}_k) \in \Phi^2$ the MI value is defined as $I(\mathbf{p}_j, \mathbf{p}_k) := i(\mathbf{a}(\mathbf{p}_j), \mathbf{a}(\mathbf{p}_k))$. Note that $I(\cdot, \cdot)$ is symmetric under exchange of position coordinates, i.e., $I(\mathbf{p}_1, \mathbf{p}_2) = I(\mathbf{p}_2, \mathbf{p}_1)$.

3.1 Mutual Information

MI is widely applied in machine learning, statistics, and information theory [6] to assess how similar or correlated pairs of random variables are. MI has the capability to uncover non-linear and non-monotonic relationships, which may go unnoticed when only considering covariance. In mathematical terms, this is expressed as

follows:

$$I(X; Y) = H(X) - H(X|Y) = H(Y) - H(Y|X), \quad (2)$$

where X and Y represent two random variables. The term $I(X; Y)$ denotes the MI between X and Y , $H(X)$ stands for the entropy of Y , and $H(X|Y)$ represents the conditional entropy of X given Y . Notably, MI is symmetric under the exchange of X and Y .

Measuring MI comes with significant computational demands. Traditional methods struggle to handle large datasets [21, 30]. Recent approaches like copula-based and neural network-based variational MI estimators show promise for high-dimensional data but still present computational challenges [2, 47]. In our research, we compute ground truth MI fields using a CPU implementation of the nearest neighbor-based estimator of Kraskov et al. [36]. However, our performance analysis in Section 5.2 shows that the time required for computing timeseries MI fields significantly exceeds what is acceptable in interactive data analysis workflows.

4 NEURAL MI FIELDS

In the following, we introduce a neural network for MI estimation, which we call timeseries neural MI fields (TNMIFs). A TNMIF can be seen as a compact data structure that encodes query parameters into a compact latent code and estimates the final MI information from this code. Initially, we conduct computationally expensive MI calculations as a preprocessing step to facilitate the utilization of extensive collections of window-to-window MI measures. Then, we encode the MI fields between time-windows, denoted as I , as described in Section 3, using efficient neural fields $\Theta : \Phi \times \Phi \rightarrow \mathbb{R}$ capable of addressing both computational and memory constraints. The network is trained by solving the following optimization problem:

$$\Theta = \arg \min_{\Theta} \frac{1}{N} \sum_{(p_i, p_j)} \mathcal{L}(\theta(p_i, p_j), I(p_i, p_j)). \quad (3)$$

Here, $\theta(\cdot, \cdot)$ represents the neural network, $\mathcal{L}(\cdot, \cdot)$ denotes a similarity metric such as L_1 -loss. The mean is computed over samples of window position pairs. The optimization process is executed iteratively using stochastic gradient descent. Upon completion of training, Θ serves as a concise representation of MI fields, facilitating efficient sample reconstruction, which helps alleviate the need to store vast quantities of MI samples. Furthermore, once the MI network is trained, there is no necessity to retain the entire collection of timeseries in memory, enabling the approach to efficiently scale to larger collections with larger numbers of samples.

4.1 Network Architecture

We draw inspiration from the network architecture proposed in [11] and perform a number of changes to adapt the network to the estimation of window-to-window MI in timeseries data. As discussed in Section 3, the 2D space used for timeseries MI inference consists of two coordinates: one continuous coordinate representing the sequence of data points and one discrete coordinate representing the various timeseries. TNMIFs operate within a four-dimensional (4D) bi-spatial domain.

As depicted in Figure 1, we use a network architecture comprising two encoder networks and a decoder network. These networks are implemented as multi-layer perceptrons (MLPs) with L fully connected layers and C hidden channels, utilizing the *SnakeAlt*

activation [45]. Each encoder model receives information regarding one of the two positions between which the MI is to be reconstructed. The positions are translated into latent feature vectors, which are then merged via element-wise multiplication and passed to the decoder for the final prediction. This design enables each encoder to be trained solely on the marginal space $\Phi \subset \mathbb{P}$, thereby enhancing training efficiency by maximizing the effective number of MI samples per volume. Consequently, the model can extrapolate MI measures even for point pairs not encountered during training, reducing computational time and memory requirements.

We further constrain the architecture by employing identical encoders for both positions and, as shown in Figure 1, with consistent Hash encoding and Embedding. This approach facilitates model compactness and ensures symmetry of the learned fields when query positions are exchanged. Specifically, the architectural design ensures that $\Theta(p_i, p_j) = \Theta(p_j, p_i)$ is trivially fulfilled, eliminating the need for costly training iterations to learn this symmetry.

In TNMIFs, features spanning a 4D domain are decomposed into an outer product of fields across a 2D domain. Our experimentation with alternative combination methods, such as concatenation and absolute difference, decreased prediction accuracy. This underscores the critical role of multiplication in feature merging for achieving precision in predictions, as highlighted by prior research studies [11, 12].

To enhance the capacity of the encoders to capture high-frequency patterns and multi-scale features, we utilize Fourier features [29, 42] and multi-resolution hash encoding [32] for the continuous coordinate, along with an embedding layer [28, 38] for the discrete coordinate. Subsequently, these three input encodings are concatenated with the raw positions before being fed into the encoders. The input encoding modules, highlighted by dashed rectangles in Figure 1, illustrate this process.

For a given input coordinate vector, $\mathbf{p} = (p_{ts}, p_{ws}) \in \mathbb{P}$, we apply Fourier features to the continuous dimension p_{ws} . Fourier features expand the representation of spatially close positions by projecting position information into a higher-dimensional space using fixed feature mapping. This enhancement enables the model to effectively capture high-frequency patterns without increasing the number of trainable parameters and, consequently, memory consumption, thanks to the fixed functional form of the mapping. In our implementation, we empirically found that setting $L = 12$ yields favorable results for the number of Fourier frequencies.

Multi-resolution hash encoding uses hash tables filled with trainable feature vectors to populate the domain at various resolutions [32]. This allows the creation of virtual feature grids at any resolution with a fixed memory budget. Here, the hash-encoding is applied to the continuous coordinate p_{ws} . The feature vectors corresponding to different resolution levels are combined and trained collectively with subsequent model components. A crucial parameter dictating the expressiveness of the hash grid is the hash table size, denoted as 2^T for $T \in \mathbb{N}$, which also determines the memory complexity. Other hyper-parameters include the dimension of the feature vectors per resolution level, the number of resolution levels, and the grid resolution at each level. Specifically, we employ 8 resolution levels with virtual grids of size 16^3 at the coarsest level, doubling in size with each subsequent finer level. These parameters

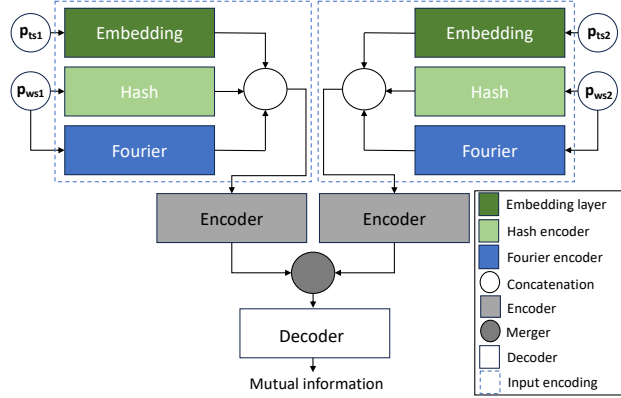


Figure 1: TNMIF network architecture. (p_{ts1} , p_{ws1}) and (p_{ts2} , p_{ws2}) denote the reference and query positions, respectively. Embedding is the embedding layer. Hash is the hash-grid encoding technique. The Fourier module applies Fourier feature encoding to the input. Both the Encoder modules and the Decoder module are MLPs. The Merger operation denotes the multiplication of the encoder outputs, facilitating the integration of information across different layers.

ensure that the level of detail in the features matches the level of detail in the original dataset. This prevents using too much memory when looking at very detailed parts of the dataset while still keeping the accuracy of the reconstruction the same or making it better. Based on our empirical findings, we determined that setting $T = 30$ for the hash table size produces favorable outcomes.

The embedding layer transforms each unique value in the categorical variable into a learnable vector of fixed size called an embedding vector. These embedding vectors are initialized randomly and updated during training to minimize the loss function. In the context of encoding a timeseries dataset, we use the embedding layer to convert discrete dimension p_{ts} into continuous embeddings. These embeddings capture the underlying relationships and patterns in the data, allowing the neural network to learn from them effectively. Our experiments consider two crucial parameters for the embedding layer: the number of embeddings and the embedding dimension. The number of embeddings is set equal to the number of timeseries in our dataset, ensuring each timeseries has a corresponding embedding vector. The embedding dimension is another important parameter, representing the size of each embedding vector. We set the embedding dimension to 40. This dimensionality has been chosen based on experimentation and empirical observation, aiming to strike a balance between the expressiveness of the embedding representations and computational efficiency.

4.2 Datasets

We demonstrate the quality and performance of the proposed data structure with two large timeseries datasets from the UCR timeseries classification archive [7]. The results are primarily presented using the CinCECGTorso dataset for our main quantitative analysis and performance evaluation. Additionally, we include findings from

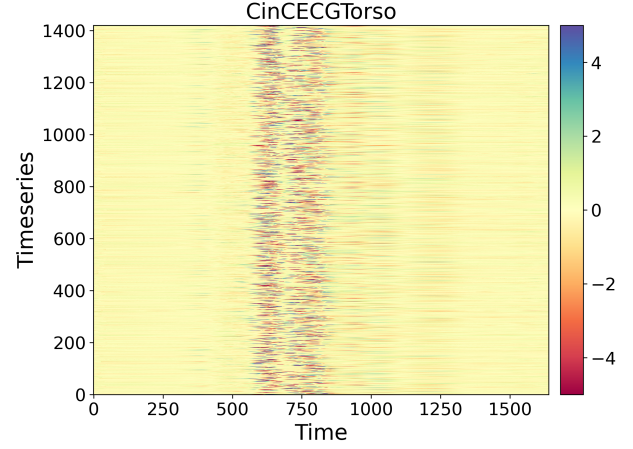


Figure 2: CinCECGTorso dataset: 1420 timeseries with 1639 timesteps. Note the visual similarity of all timeseries.

the StarLightCurves dataset to further indicate the performance and generalization of the method.

CinCECGTorso. CinCECGTorso comprises 1420 timeseries instances, with each instance consisting of 1639 timesteps. The data originates from the computers in cardiology (CinC) challenges, an annual competition co-located with the Computers in Cardiology conference series. The CinC challenges are hosted on Physionet, a repository of physiological signals and related data [14].

The CinCECGTorso dataset is derived from electrocardiogram (ECG) recordings captured from multiple torso-surface sites. These recordings are obtained from individuals participating in the CinC challenges, with each participant representing a distinct class in the dataset. The dataset comprises samples from four different classes, each corresponding to a unique individual. These classes enable the exploration of variations in ECG signals across different individuals. Figure 2 shows a color plot of all timeseries.

StarLightCurves. StarLightCurves [40], also available from the UCR classification archive, comprises 1000 training samples and 8236 test samples, each characterized by 1024 timesteps. This dataset offers an insightful view of the variability of celestial objects. A starlight curve shows the temporal variations in the brightness of a celestial entity over time. These curves have been classified into three distinct classes by domain experts. In our study, we utilized the 1000-sample training set for conducting experiments and evaluations. Figure 3 shows a colorplot of all timeseries.

4.3 Training

We explore two distinct training methodologies: synchronous models and lagged models. In synchronous models, sample pairs represent corresponding (i.e., synchronized) time-windows in different timeseries, forming the basis for model training. Conversely, lagged models utilize sample pairs where for a time-window in one timeseries, multiple lagged windows in some other timeseries are considered. The output of these models is a set of MI values for each

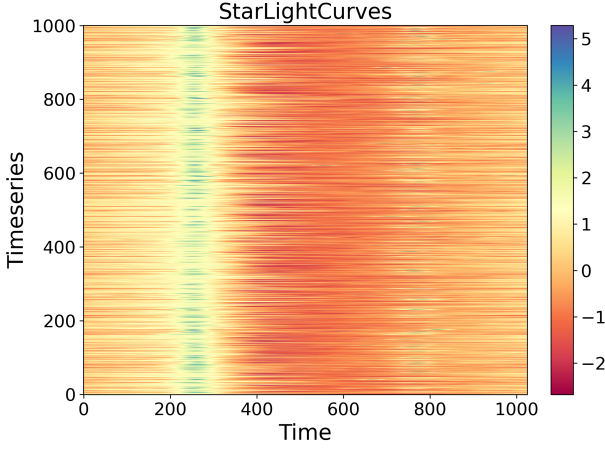


Figure 3: StarLightCurves dataset: 1000 timeseries with 1024 timesteps. Note the visual similarity of all timeseries.

sample pair. We will subsequently call the two models sync-TNMIF and lag-TNMIF, respectively.

During the training phase, our approach involves utilizing data samples from the domain Φ , which are appropriately rescaled to occupy a symmetric matrix space. We generate a substantial set of 5×10^5 (for sync-TNMIF) and 10^6 (for lag-TNMIF) pairs of positions $(\mathbf{p}_j, \mathbf{p}_k)$, uniformly distributed in the matrix. These positions serve as reference points to retrieve corresponding samples $\mathbf{a}(\mathbf{p}_j)$ and $\mathbf{a}(\mathbf{p}_k)$ from the original timeseries data.

Subsequently, we compute the MI $I(\mathbf{p}_j, \mathbf{p}_k)$ between these pairs of samples. Our models are trained to optimize the L_1 loss function, effectively quantifying similarity. Notably, we find that employing the L_2 -loss function yields comparable outcomes in our experimental setup.

For the optimization process, we leverage the Adam optimizer [19], initialized with a learning rate of 7×10^{-4} , and conduct training with batches containing 1024 samples each. Following each epoch, we refresh the training samples to ensure model robustness. The training regime spans a total of 100 epochs for sync-TNMIF and 150 epochs for lag-TNMIF.

As denoted in Sections 1 and 3, we split the timeseries into windows with a specified length l and a constant lag $s = 1$. Window size selection algorithms in timeseries analysis aim to determine an appropriate window size by considering various data characteristics and analytical requirements. These algorithms take into account factors such as pattern frequency, desired granularity, and computational constraints. The window size parameter is crucial for segmenting data into subsequences, ideally aligned with the length of a single temporal pattern, such as a heartbeat in an ECG recording or a breathing cycle in a respiratory signal. While manual specification of window size is feasible, automating this process is preferred for efficiency.

Our study utilizes the ClaSP algorithm to determine the optimal window size for the CinCECGTorso timeseries dataset. Developed by Ermhaus et al. [8, 9], ClaSP is a benchmark tool that evaluates classical and recent timeseries segmentation strategies.

Their research has identified dominant Fourier frequency, highest autocorrelation, and subsequence summary statistics as effective strategies across various scenarios, as observed in UCR timeseries classification datasets[7].

For CinCECGTorso, the optimal window size determined by the ClaSP algorithm is 48. Additionally, we explored multiples of this size, such as a window size of 96, to further assess model performance. The window size of 64 is optimal for the StarLightCurves dataset using ClaSP.

5 RESULTS

In the following, we analyze the quality and performance of the proposed network architecture for MI estimation with the CinCECGTorso dataset. While sync-TNMIF has been trained for 2 hours, the lagged version requires longer training times of roughly 3 hours. During network training, only 0.3% (sync-TNMIF) and 0.009% (lag-TNMIF) of all possible window-to-window MI estimates were seen by the network. The network requires roughly 45 MB of memory, which is only a small fraction of the memory that would be required to store all possible MI values.

5.1 Quantitative Analysis

The reconstruction quality of the TNMIF model is assessed using the synchronous and lagged versions. The MI fields estimated by the networks are compared to the ground truth fields to demonstrate both the capacities and limitations of TNMIF.

Figures 4 and 5 demonstrate sync-TNMIF’s capability to simultaneously estimate MI between each time-window in a selected timeseries (first row in each map) and the corresponding window in each of the other timeseries. As can be seen, the overall structure of the MI field is very well preserved. This is remarkable, because only a small number of all possibly queried window pairs have been seen by the network during training. The proposed bi-partite TNMIF architecture efficiently leverages sample information by effectively augmenting the sampling density for the encoder components.

On the other hand, some of the fine details are not reconstructed equally well. We first attributed this to the restricted number of MI samples used for model training, yet the property remained even when pursuing experiments with significantly increased sample counts. This suggests that sampling density does not represent the primary bottleneck for accuracy. Instead, a valid trade-off between model capacity and reconstruction fidelity is apparent, indicating that quality is predominantly constrained by the capability of the model to retain training information, rather than sample availability. Notably, the shown MI fields are with respect to a selected reference timeseries, acting as 2D sections within a much larger 4D MI space where the model was trained. Thus, the network needs to encode a gigantic amount of information, i.e., storing all possible window-to-window MI measures in 32-bit floating point format for the CinCECGTorso dataset would require 55 GB of memory. The network size of 45 MB translates to a noteworthy 1200× compression, underscoring promising outcomes.

To further evaluate the quality of the sync-TNMIF model, two similar timeseries from each of the four classes in the CinCECGTorso dataset are selected, and the ground truth and estimated MI values per window pair are computed. The corresponding MI

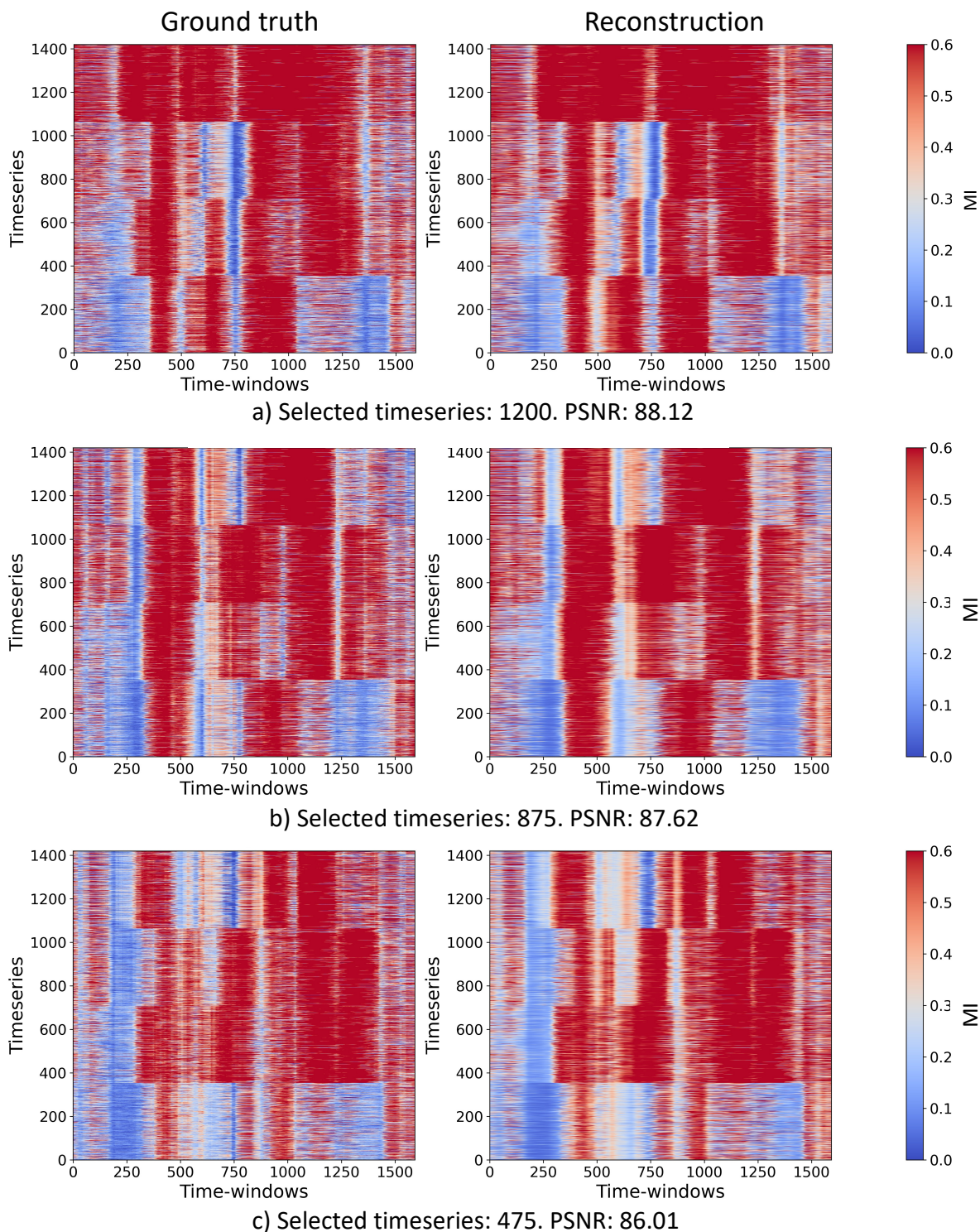


Figure 4: sync-TNMIF. 2D maps show window-to-window MI between selected timeseries (bottom row) in CinCECGTorso and each of the other timeseries using a window size of 48. Timeseries are grouped according to class index.

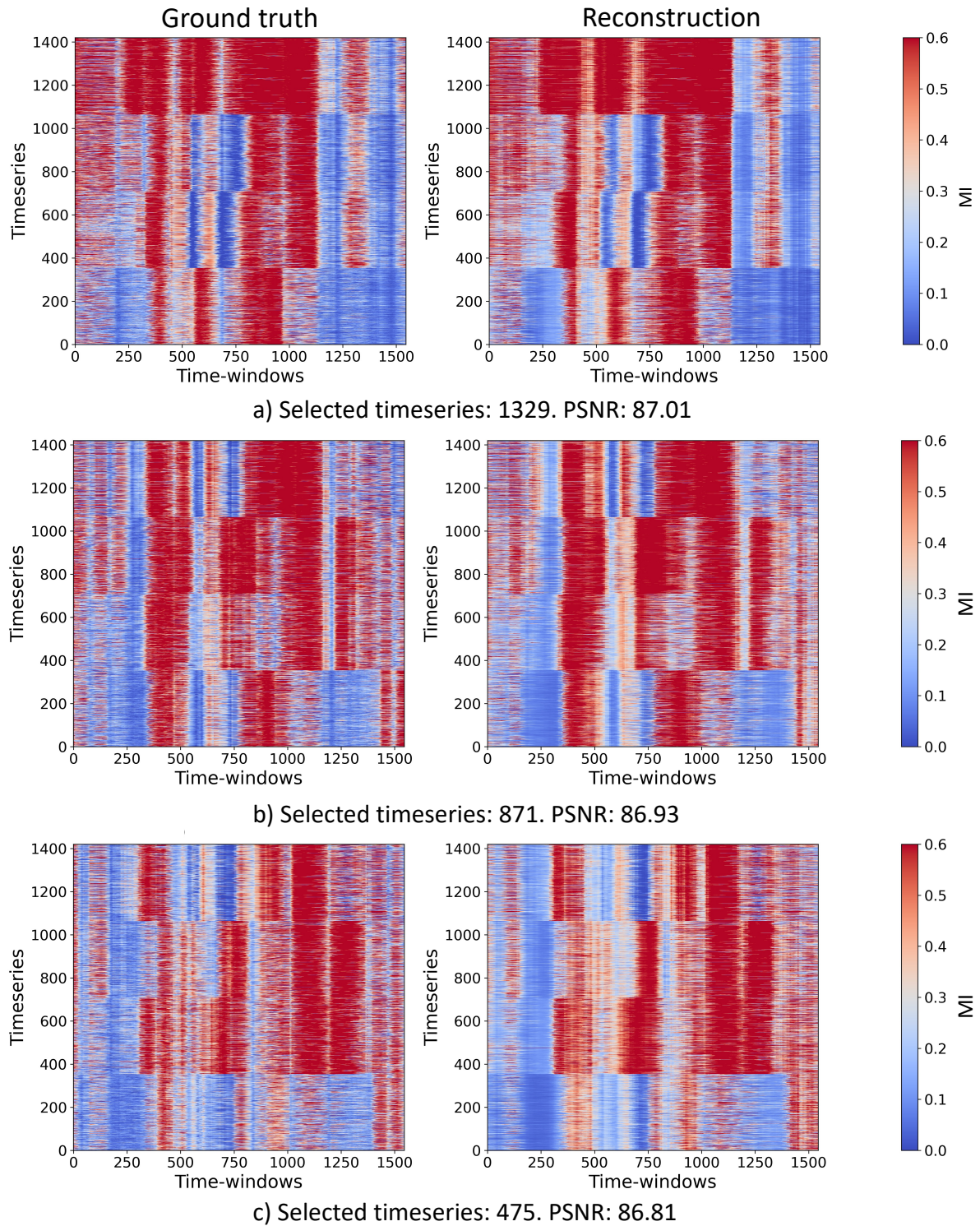


Figure 5: sync-TNMIF. 2D maps show window-to-window MI between selected timeseries (bottom row) in CinCECGTorso and each of the other timeseries using a window size of 96. Timeseries are grouped according to class index.

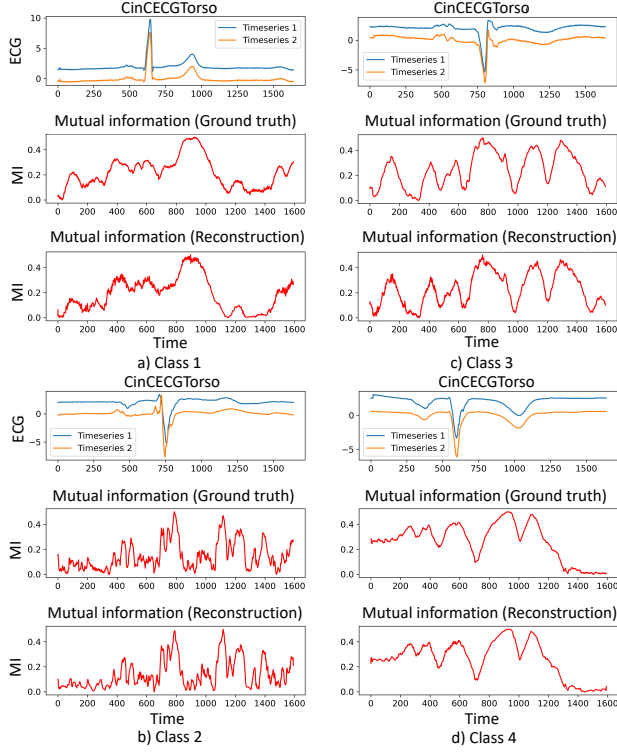


Figure 6: In a) to d), from each class in the CinCECGTorso dataset, two timeseries are selected and composed into time-windows of size 96. Window-to-window MI curves reconstructed by the sync-TNMIF model accurately resemble the ground truth curves of MI between the two timeseries.

curves are shown in Figure 6, demonstrating the capability of sync-TNMIF to accurately estimate window-to-window MI. Note that, since MI captures both linear and non-linear relationships, even if two timeseries are similar their relationship might be non-linear. This reflects in lower MI values and corresponding variation of the curves depicting MI for consecutive window pairs.

To verify the generalizability of our approach, we apply our model to an additional large timeseries. In Figure 11, sync-TNMIF performance is validated by visualizing the window-to-window MI between a picked timeseries (first row in each map) and all the others in the StarLightCurves dataset. The proficiency of the model in reconstructing and preserving the primary MI structures across various large timeseries datasets highlights the remarkable capabilities of TNMIFs.

To evaluate how well the lag-TNMIF model can estimate MI to lagged windows in other timeseries, we pursue the following experiment: From the CinCECGTorso dataset, we select a single timeseries and generate various lagged copies of it by using random shifts between -32 and 32. The window size is set to 48. The lag-TNMIF model is then trained with the same window size and lags between ± 32 , so that for each window in the initial timeseries the shifted windows in all copies are covered.

Simultaneously for all windows w_j in the initial timeseries, the lag-TNMIF estimates the MI between the window and all lagged windows in each shifted timeseries. For a window w_j in the initial timeseries and a shifted timeseries t , we denote with $W_{j,l}^t$, $l \in [-32, 32]$, the set of MI values between w_j and the lagged windows in timeseries t . The lag-TNMIF model estimates $W_{j,l}^t$, $l \in [-32, 32]$, for all timeseries t in a single inference pass.

For each t and j , the lag at which the maximum MI occurs in $W_{j,l}^t$ is then computed. Ideally, the maximum should be exactly the random value by which timeseries t is generated. However, determining the lag value for a single time-window may not always be accurate, even when exact MI values are used. This is due to the possible presence of additional similar time-windows within the query window of the lagged timeseries. To address this issue, for each shifted copy the average of the per-window maxima is computed. In addition, for each window in the initial timeseries the maximum MI to the lagged windows in each shifted version is color coded. Figure 7 shows the results for a selected timeseries and 20 randomly shifted copies of it. It demonstrates high agreement between the applied lags and those which are derived from estimated MI via the lag-TNMIF model.

The quality of the lag-TNMIF model is further evaluated using timeseries pairs from each of the four classes in CinCECGTorso (see Figure 8). Specifically, we select a pair of highly similar timeseries from each class, and shift one timeseries of each pair by 32 into the future. The lag-TNMIF model estimates the MI between each time-window in the non-shifted timeseries and all lagged time-windows in the shifted one, considering a lag of ± 48 . Subsequently, the location of the time-window with the highest MI value is identified. We repeat this computation using the Kraskov MI estimator. Figure 8 shows the results for all selected timeseries pairs, including the per-window MI maxima estimated by the network and the average over all maxima. As can be seen, from the MI values estimated by the lag-TNMIF in one single inference step, the applied lags can be determined accurately. I.e., the network can serve as a data structure that can accurately and efficiently detect similar timeseries based on MI, even if they are shifted in time.

5.2 Performance

All timings have been evaluated on a system running Ubuntu 22.04 with an Intel Xeon W-2235 CPU 6-core (12-thread) CPU and an NVIDIA RTX 3090 GPU. To obtain ground truth MI fields, Kraskov MI estimation has been parallelized on the multi-core CPU. Given two realization vectors of random variables X and Y , with the joint samples $Z = \{(X_i, Y_i), i \in \{0, \dots, N-1\}\}$, the Kraskov estimator builds upon finding the k -nearest neighbor of each joint sample. In our implementation, a k -d tree is used as a search structure. Since there is one k -d tree per grid point, the multi-core parallelization on the CPU is straightforward. Furthermore, we reduce system calls by moving the allocation of heap memory outside of the parallelization loop.

Network training is performed in PyTorch. To enable fast inference times of the network, we have extended the tiny-cuda-nn GPU framework [34], which provides a fast implementation of fully-fused MLP layers, to support the embedding layer described

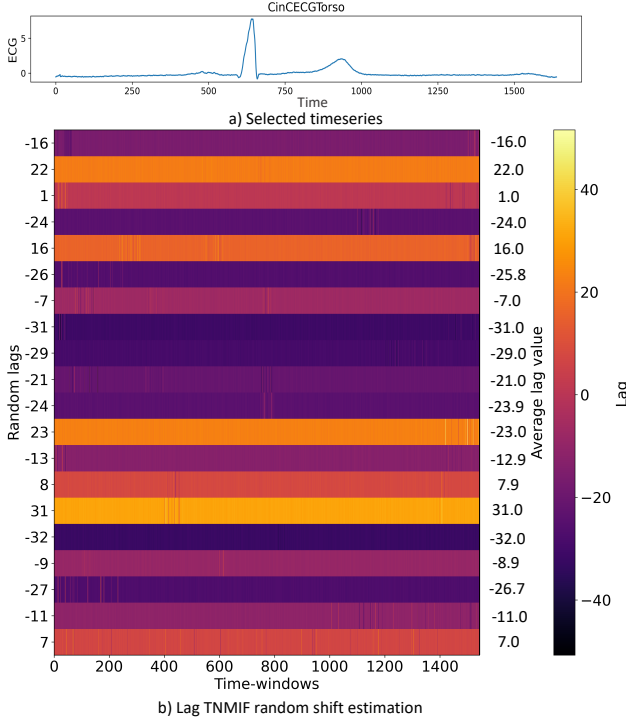


Figure 7: Shifted copies of a selected timeseries from CinCECGTorso dataset, depicted in a), generated by random shifts between ± 32 . The lag TNMIF model (with lags considered between ± 48) estimates the MI between the windows in the original timeseries and the lagged windows in all copies. The lag at which the maximum MI occurs is encoded by color in b) for 20 randomly shifted copies. The average lag value over all windows is shown on the right axis.

in Section 4.1. The code for the experiments is publicly available at https://github.com/FatemehFarokhmanesh/TNMIF_models.

For sync- and lag-TNMIF, Table 1 shows the performance that is achieved with our CPU Kraskov implementation in comparison to network estimation. For the CinCECGTorso dataset, computation times are given for a) sync-TNMIF estimating 1544 (number of windows of length 96 in one selected timeseries) times 1419 (number of remaining timeseries) MI measures, b) lag-TNMIF estimating 1544 (number of windows of size 96 in one selected timeseries) times 1419 (number of remaining timeseries) times $2 \times 64 + 1 = 129$ (number of negative and positive lags) MI measures, and c) same as a) but using the CPU Kraskov implementation. It can be seen that

Table 1: Performance comparison. For the CinCECGTorso dataset, computation times for sync and lag-TNMIF and CPU Kraskov.

	sync-TNMIF	lag-TNMIF	MI [21]
Time	44 ms	1100 ms	27788 ms

SSDBM 2024, July 10–12, 2024, Rennes, France

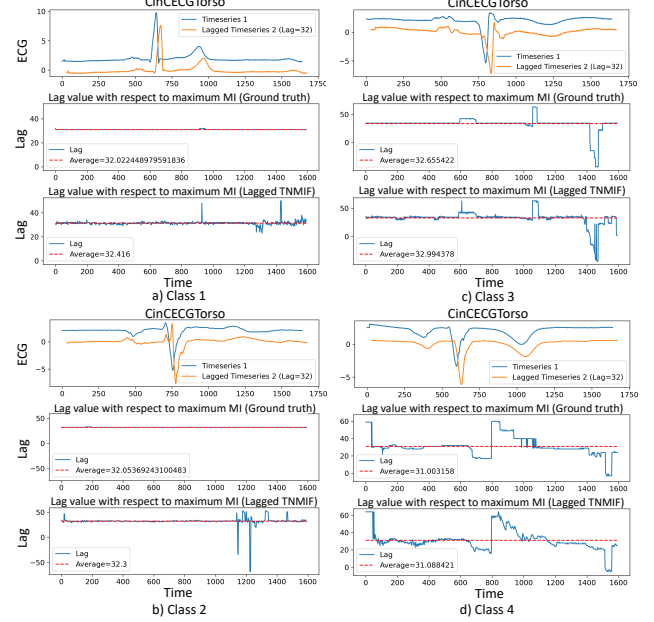


Figure 8: In a) to d), from each class in the CinCECGTorso dataset two timeseries are selected. One is shifted by a lag of 32. lag-TNMIF estimates for each window in the non-shifted timeseries the MI values of all windows shifted by ± 48 in the other timeseries. Red and blue curves, respectively, indicate the mean lag (over all windows) at which the maximum MI occurs and the per-window lag at which the maximum MI occurs.

the network is approximately about a factor of $632\times$ faster than the Kraskov MI implementation, yet it neither needs the timeseries nor any precomputed MI fields in memory at runtime. The slower performance of the model compared to [11] can be explained by differences in the model architecture. Our model incorporates an additional component in encoding module, embedding layer, and a larger number of time-window positions, at least 2,200,000. Additionally, the encoder is not only applied to a single reference position, but to all window positions in the selected timeseries, which leads to less memory access coherency.

To further shed light on the performance and efficiency of the lag-TNMIF model, we conducted an experiment wherein the model reconstructed MI between time-windows within a selected timeseries and all lagged windows, with a maximum lag of ± 64 , across all other timeseries belonging to class one in the CinCECGTorso dataset. Remarkably, considering a maximum lag of 64 decreases query times by about a factor of 10. The results of this experiment, depicted in the 3D volume shown in Figure 9, showcase the efficiency of the lag-TNMIF model in reconstructing large amounts of lagged MI values. Notably, the model achieved this within 1.1 seconds, underscoring its capability to efficiently and accurately capturing significant and complex temporal relationships instantaneously. We use the application Corrender [35] to visualize the 3D volume.

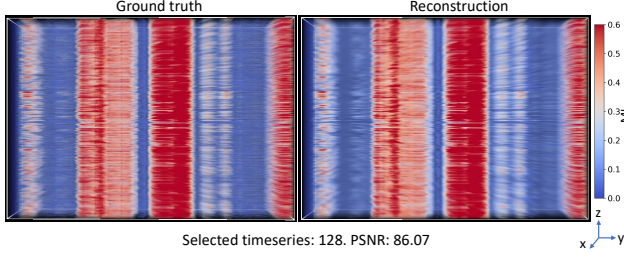


Figure 9: Result of one inference pass of lag-TNMIF, performed in 1.1 seconds. 3D maps show MI between windows in selected timeseries in CinCEGTorso (bottom row) and lagged windows in each of the other timeseries. Time-windows are arranged along the y -axis, timeseries are arranged along the z -axis. For each window, all considered lagged windows are arranged along the x -axis. Window size is 48, lags between -64 and 64 have been considered.

5.3 Timeseries Similarity Matrix

Assuming that each of T timeseries is split into the same number of time-windows, the sync-TNMIF model enables rapid estimation of the MI values between all pairs of corresponding windows. The model estimates these values in roughly 32 seconds, compared to 5 hours using CPU Kraskov MI. From these window-to-window estimates, for each timeseries pair a similarity measure can be derived and plotted in a 2D similarity matrix. Since MI is symmetric under the exchange of positions, only half of the MI estimates are required.

To illustrate the utility of this approach, we examine two specific timeseries similarity measures based on window-to-window MI for the CinCEGTorso dataset: the time intervals between successive extreme events below the mean [24] and the interquartile range (IQR). The first measure considers the durations between the occurrences of extreme events that fall below the mean of all timeseries-to-timeseries MI estimates. The IQR is a statistical measure that indicates the dispersion or spread of data points within a dataset. I.e., it is computed as the difference between the third quartile, which represents the median of the upper half of the timeseries-to-timeseries MI estimates, and the first quartile, which denotes the median of the lower half of these estimates.

The two measures are computed and visualized for each timeseries-pair in the 2D matrices in Figure 10. The plots provide insights into the differences in the temporal dynamics of different timeseries. Since the dataset comprises four distinct classes, the intra- and inter-class relationships can be examined. The similarity matrix illustrates high similarities among the timeseries within each class. Additionally, both metrics exhibit comparable behavior between the first and fourth classes, as well as between the second and third classes. This analysis provides insights into the underlying structure and patterns present in the dataset, which are valuable for interpretation and classification purposes.

6 CONCLUSION AND FUTURE WORK

We have introduced and assessed the effectiveness of timeseries neural MI fields (TNMIFs), a compact data structure that is trained

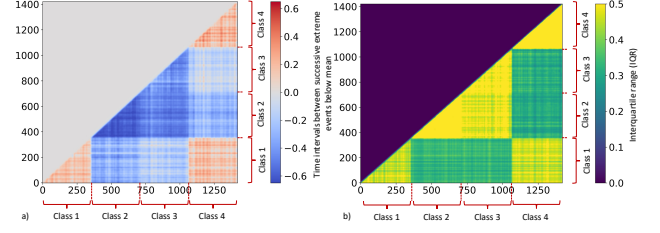


Figure 10: Vertical and horizontal coordinates show the timeseries indices. All possible pairwise MI between the timeseries is computed using sync-TNMIF within a few seconds. Statistical measures of the IQR and the time intervals between successive extreme events are computed over the measures derived from the pairwise MI computations. Each point shows a similarity value for a timeseries-pair.

to efficiently reconstruct MI between synchronous or lagged time-windows in large timeseries collections. With sync-TNMIF and lag-TNMIF, we investigated two distinct training methodologies, where pairs of samples represent synchronized time-windows from different timeseries or a single window and multiple lagged windows from another timeseries. Our results demonstrate that TNMIFs can accurately and efficiently encode and reconstruct the principal MI structures present in timeseries fields, while tending to smooth some fine details due to the limitations of network capacity. The TNMIFs capability to rapidly reconstruction MI in large timeseries enables the computation of timeseries similarity matrices from the reconstructed estimates.

By considering deeper encoders and decoders or utilizing alternative network architectures, we seek to improve the network capacity in the future. We further aim to integrate the models into an interactive visualization workflow, combined with mechanisms to select specific time-windows or specific temporal features. Lastly, we intend to shed light on the use of TNMIFs to estimate the similarity between two timeseries varying in speed, i.e., to simulate dynamic time warping. Firstly, this requires training with window pairs of different lengths. Secondly, we see especially the embedding of neural networks for time-warping-based feature extraction [3] as a promising strategy. This requires further analysis of the relationships between warping and MI variation.

ACKNOWLEDGMENTS

The authors would like to thank Prof. Eamonn Keogh and all the people who have contributed to the UCR timeseries classification archive for their selfless work. The research leading to these results has been done within the subproject “A7” of the Transregional Collaborative Research Center SFB / TRR 165 “Waves to Weather” (www.wavestoweather.de) funded by the German Research Foundation (DFG).

REFERENCES

- [1] Ramoza Ahsan, Rodica Neamtu, Muzammil Bashir, Elke A Rundensteiner, and Gabor Sarkozy. 2020. Correlation-Based Analytics of Time Series Data. In *2020 IEEE International Conference on Big Data (Big Data)*. IEEE, 4482–4491.
- [2] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. 2018. Mutual Information Neural

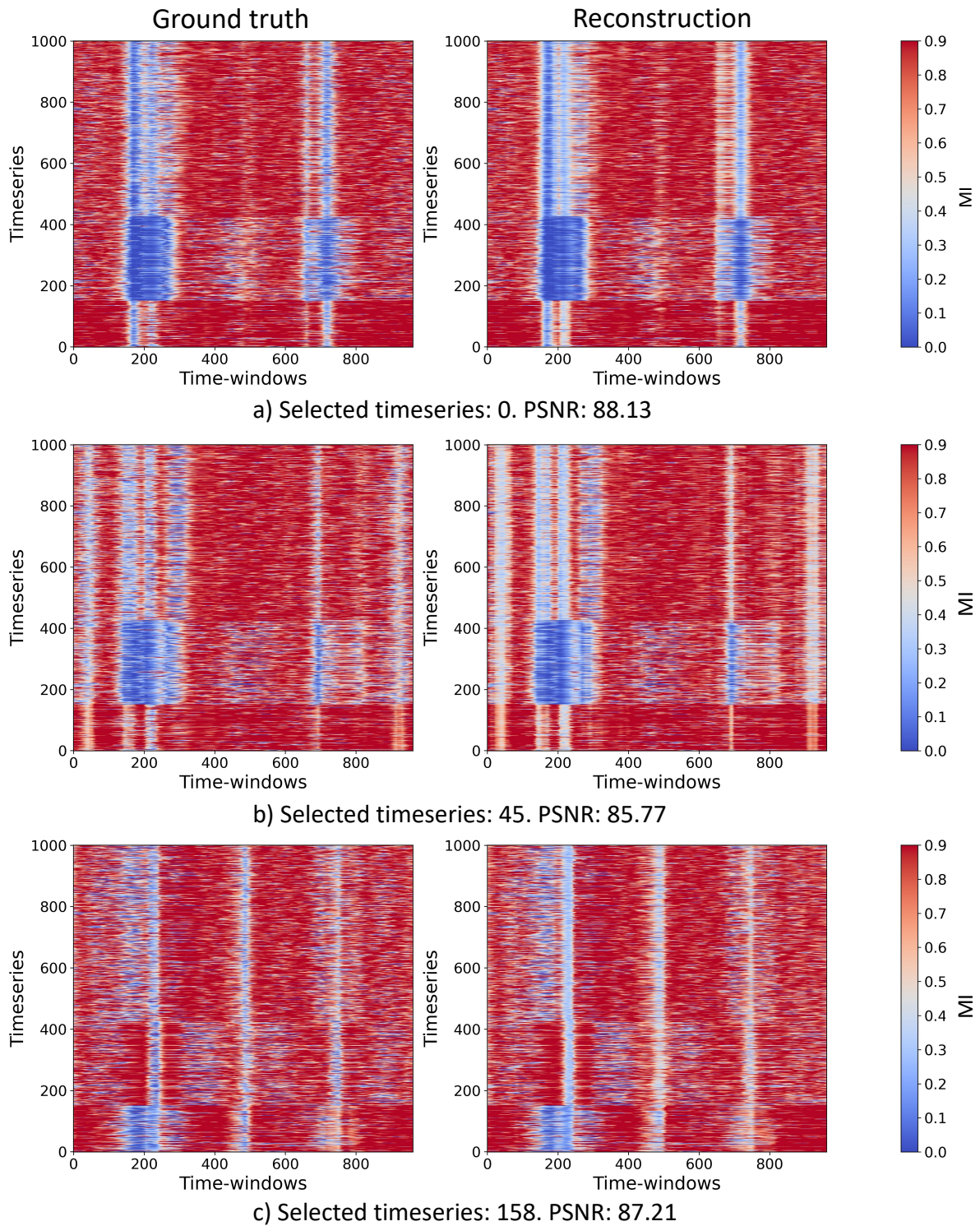


Figure 11: sync-TNMIF. 2D maps show window-to-window MI between selected timeseries in CinCECGTorso (bottom row) and each of the other timeseries using a window size of 64. Timeseries are grouped according to class index.

- Estimation. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*, Jennifer Dy and Andreas Krause (Eds.). PMLR, 531–540.
- [3] Xingyu Cai, Tingyang Xu, Jinfeng Yi, Junzhou Huang, and Sanguthevar Rajasekaran. 2019. Dtnet: a dynamic time warping network. *Advances in neural information processing systems* 32 (2019).
 - [4] Rohan Chhabra, Jan E Lenssen, Eddy Ilg, Tanner Schmidt, Julian Straub, Steven Lovegrove, and Richard Newcombe. 2020. Deep local shapes: Learning local sdf priors for detailed 3d reconstruction. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIX* 16. Springer, 608–625.
 - [5] Zhiqin Chen and Hao Zhang. 2019. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 5939–5948.
 - [6] Thomas M Cover, Joy A Thomas, et al. 1991. Entropy, relative entropy and mutual information. *Elements of information theory* 2, 1 (1991), 12–13.
 - [7] Hoang Anh Dau, Anthony Bagnall, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, and Eamonn Keogh. 2019. The UCR time series archive. *IEEE/CAA Journal of Automatica Sinica* 6, 6 (2019), 1293–1305.
 - [8] Arik Ermshaus, Patrick Schäfer, and Ulf Leser. 2023. ClaSP: parameter-free time series segmentation. *Data Mining and Knowledge Discovery* (2023).
 - [9] Arik Ermshaus, Patrick Schäfer, and Ulf Leser. 2023. Window size selection in unsupervised time series analytics: A review and benchmark. In *International Workshop on Advanced Analytics and Learning on Temporal Data*. Springer, 83–101.
 - [10] Marina Evers, Michael Böttinger, and Lars Linsen. 2023. Interactive Visual Analysis of Regional Time Series Correlation in Multi-field Climate Ensembles. In *Workshop on Visualisation in Environmental Sciences (EnvirVis)*, Soumya Dutta, Kathrin Feige, Karsten Rink, and Dirk Zeckzer (Eds.). The Eurographics Association. <https://doi.org/10.2312/envirvis.20231108>
 - [11] Fatemeh Farokhmanesh, Kevin Höhle, Christoph Neuhauser, Tobias Necker, Martin Weissmann, Takemasa Miyoshi, and Rüdiger Westermann. 2023. Neural Fields for Interactive Visualization of Statistical Dependencies in 3D Simulation Ensembles. In *Vision, Modeling, and Visualization*, Michael Guthe and Thorsten Grosch (Eds.). The Eurographics Association. <https://doi.org/10.2312/vmv.20231229>
 - [12] Sara Fridovich-Keil, Giacomo Meanti, Frederik Warburg, Benjamin Recht, and Angjoo Kanazawa. 2023. K-planes: Explicit radiance fields in space, time, and appearance. *arXiv preprint arXiv:2301.10241* (2023).
 - [13] Ben D Fulcher and Nick S Jones. 2017. hctsa: A computational framework for automated time-series phenotyping using massive feature extraction. *Cell systems* 5, 5 (2017), 527–531.
 - [14] Ary L Goldberger, Luis AN Amaral, Leon Glass, Jeffrey M Hausdorff, Plamen Ch Ivanov, Roger G Mark, Joseph E Mietus, George B Moody, Chung-Kang Peng, and H Eugene Stanley. 2000. PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals. *circulation* 101, 23 (2000), e215–e220.
 - [15] Duong Hoang, Brian Summa, Harsh Bhatia, Peter Lindstrom, Pavol Klacansky, Will Usher, Peer-Timo Bremer, and Valerio Pascucci. 2020. Efficient and flexible hierarchical data layouts for a unified encoding of scalar field precision and resolution. *IEEE Transactions on Visualization and Computer Graphics* 27, 2 (2020), 603–613.
 - [16] Kevin Höhle, Sebastian Weiss, and Rüdiger Westermann. 2022. Evaluation of Volume Representation Networks for Meteorological Ensemble Compression. (2022).
 - [17] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. 2019. Deep learning for time series classification: a review. *Data mining and knowledge discovery* 33, 4 (2019), 917–963.
 - [18] Hassan Ismail Fawaz, Benjamin Lucas, Germain Forestier, Charlotte Pelletier, Daniel F Schmidt, Jonathan Weber, Geoffrey I Webb, Lhassane Idoumghar, Pierre-Alain Muller, and François Petitjean. 2020. Inceptiontime: Finding alexnet for time series classification. *Data Mining and Knowledge Discovery* 34, 6 (2020), 1936–1962.
 - [19] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).
 - [20] Patrick Köthür, Carl Witt, Mike Sips, Norbert Marwan, Stefan Schinkel, and Doris Dransch. 2015. Visual analytics for correlation-based comparison of time series ensembles. In *Computer Graphics Forum*, Vol. 34. Wiley Online Library, 411–420.
 - [21] Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. 2004. Estimating mutual information. *Phys. Rev. E* 69 (Jun 2004), 066138. Issue 6. <https://doi.org/10.1103/PhysRevE.69.066138>
 - [22] Yuhong Li, Leong Hou U, Man Lung Yiu, and Zhiguo Gong. 2013. Discovering longest-lasting correlation in sequence databases. *Proceedings of the VLDB Endowment* 6, 14 (2013), 1666–1677.
 - [23] Yuzhe Lu, Kairong Jiang, Joshua A Levine, and Matthew Berger. 2021. Compressive neural representations of volumetric scalar fields. In *Computer Graphics Forum*, Vol. 40. Wiley Online Library, 135–146.
 - [24] Carl H Lubba, Sarab S Sethi, Philip Knaute, Simon R Schultz, Ben D Fulcher, and Nick S Jones. 2019. catch22: CAnonical Time-series CHaracteristics: Selected through highly comparative time-series analysis. *Data Mining and Knowledge Discovery* 33, 6 (2019), 1821–1852.
 - [25] Julien NP Martel, David B Lindell, Connor Z Lin, Eric R Chan, Marco Monteiro, and Gordon Wetzstein. 2021. Acorn: Adaptive coordinate networks for neural scene representation. *arXiv preprint arXiv:2105.02788* (2021).
 - [26] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. 2019. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 4460–4470.
 - [27] Matthew Middlehurst, Patrick Schäfer, and Anthony Bagnall. 2023. Bake off redux: a review and experimental evaluation of recent time series classification algorithms. *arXiv preprint arXiv:2304.13029* (2023).
 - [28] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems* 26 (2013).
 - [29] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. 2021. Nerf: Representing scenes as neural radiance fields for view synthesis. *Commun. ACM* 65, 1 (2021), 99–106.
 - [30] Young-II Moon, Balaji Rajagopalan, and Upmanu Lall. 1995. Estimation of mutual information using kernel density estimators. *Physical Review E* 52, 3 (1995), 2318.
 - [31] Abdullah Mueen, Suman Nath, and Jie Liu. 2010. Fast approximate correlation for massive time-series data. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*. 171–182.
 - [32] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. 2022. Instant Neural Graphics Primitives with a Multiresolution Hash Encoding. *ACM Trans. Graph.* 41, 4, Article 102 (jul 2022), 15 pages. <https://doi.org/10.1145/3528223.3530127>
 - [33] Thomas Müller, Fabrice Rousselle, Jan Novák, and Alexander Keller. 2021. Real-time neural radiance caching for path tracing. *arXiv preprint arXiv:2106.12372* (2021).
 - [34] Thomas Müller. 2021. *tiny-cuda-nn*. <https://github.com/NVlabs/tiny-cuda-nn>
 - [35] Christoph Neuhauser and Josef Stumpffegger. 2024. *chrismile/Correnderer: A correlation field renderer using the Vulkan graphics API, v2024-04-26*. <https://doi.org/10.5281/zenodo.11069528>
 - [36] Christoph Neuhauser, Josef Stumpffegger, and Rüdiger Westermann. 2023. Adaptive Sampling of 3D Spatial Correlations for Focus+ Context Visualization. *IEEE Transactions on Visualization and Computer Graphics* (2023).
 - [37] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. 2019. DeepSDF: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 165–174.
 - [38] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.
 - [39] Raoul Pietersz 4 and Patrick JF Groenen. 2004. Rank reduction of correlation matrices by majorization. *Quantitative Finance* 4, 6 (2004), 649–662.
 - [40] Umaa Rebbapragada, Pavlos Protopapas, Carla E Brodley, and Charles Alcock. 2009. Finding anomalous periodic time series: An application to catalogs of periodic variable stars. *Machine learning* 74 (2009), 281–313.
 - [41] Takaki Takikawa, Joey Litalien, Kangxue Yin, Karsten Kreis, Charles Loop, Derek Nowrouzezahrai, Alec Jacobson, Morgan McGuire, and Sanja Fidler. 2021. Neural geometric level of detail: Real-time rendering with implicit 3D shapes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 11358–11367.
 - [42] Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. 2020. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in Neural Information Processing Systems* 33 (2020), 7537–7547.
 - [43] Ayush Tewari, Ohad Fried, Justus Thies, Vincent Sitzmann, Stephen Lombardi, Kalyan Sunkavalli, Ricardo Martin-Brualla, Tomas Simon, Jason Saragih, Matthias Nießner, et al. 2020. State of the art on neural rendering. In *Computer Graphics Forum*, Vol. 39. Wiley Online Library, 701–727.
 - [44] Zhiguang Wang, Weizhong Yan, and Tim Oates. 2017. Time series classification from scratch with deep neural networks: A strong baseline. In *2017 International joint conference on neural networks (IJCNN)*. IEEE, 1578–1585.
 - [45] Sebastian Weiss, Philipp Hermüller, and Rüdiger Westermann. 2022. Fast neural representations for direct volume rendering. In *Computer Graphics Forum*, Vol. 41. Wiley Online Library, 196–211.
 - [46] Yiheng Xie, Takaki Takikawa, Shunsuke Saito, Or Litany, Shiqin Yan, Numair Khan, Federico Tombari, James Tompkin, Vincent Sitzmann, and Srinath Sridhar. 2022. Neural fields in visual computing and beyond. In *Computer Graphics Forum*, Vol. 41. Wiley Online Library, 641–676.
 - [47] X Zeng and TS Durrani. 2011. Estimation of mutual information using copula density function. *Electronics letters* 47, 8 (2011), 493–494.

From: **Gumbel, Erin** <egumbel@ametsoc.org>
Date: Thu, Sep 12, 2024 at 12:11 PM
Subject: Re: Permission Request Form [#2501]
To: <fatemeh.farokhmanesh@tum.de>
Cc: <permissions@ametsoc.org>

Dear Mx. Farokhmanesh,

Thank you for your email. As noted in the follow-up screen upon submission of your request, it can take up to two weeks for us to prepare a response.

This signed message constitutes permission to use the material requested below.

You may include your paper, "Farokhmanesh, F., K. Höhle, T. Necker, M. Weissmann, T. Miyoshi, and R. Westermann, 2023: Deep Learning–Based Parameter Transfer in Meteorological Data. *Artif. Intell. Earth Syst.*, 2, e220024", in your Technische Universität München dissertation, *Deep Learning Approaches for Dependency Analysis in Large Ensemble Datasets*, with the following conditions:

1. Include the complete bibliographic citation of the original source.
2. Include the following statement with that citation: © **American Meteorological Society. Used with permission.**

Congratulations on completing your dissertation! If you have any questions or need additional information, please feel free to contact me.

Best,



Ms. Erin Gumbel, she/her/hers
Senior Peer Review Support Associate
Senior Permissions Specialist
egumbel@ametsoc.org
617-226-3926

ATTRIBUTION 4.0 INTERNATIONAL

Deed

Canonical URL : <https://creativecommons.org/licenses/by/4.0/>

[See the legal code](#)

You are free to:

Share — copy and redistribute the material in any medium or format for any purpose, even commercially.

Adapt — remix, transform, and build upon the material for any purpose, even commercially.

The licensor cannot revoke these freedoms as long as you follow the license terms.

Under the following terms:



Attribution — You must give [appropriate credit](#), provide a link to the license, and [indicate if changes were made](#). You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.

No additional restrictions — You may not apply legal terms or [technological measures](#) that legally restrict others from doing anything the license permits.

Notices:

You do not have to comply with the license for elements of the material in the public domain or where your use is permitted by an applicable [exception or limitation](#).

No warranties are given. The license may not give you all of the permissions necessary for your intended use. For example, other rights such as [publicity, privacy, or moral rights](#) may limit how you use the material.