Quality Control and Fault Classification of Laser Welded Hairpins in Electrical Motors

Johannes Vater*†§, Matthias Pollach*‡§, Claus Lenz‡, Daniel Winkle† and Alois Knoll§
† BMW Group, Munich, Germany, ‡ Cognition Factory GmbH, Munich, Germany,
§Chair of Robotics, Artificial Intelligence and Real-time Systems, Technical University Munich, Munich, Germany
Email: Johannes.JV.Vater@bmw.de, matthias.pollach@tum.de, lenz@cognitionfacory.com, Daniel.Winkle@bmw.de, knoll@in.tum.de
*Authors contributed equally

***Abstract*— We present the development, evaluation, and comparison of different neural network architectures using different input data to detect and classify quality deviations in the welding of hairpins. Hairpins are copper rods that are located in the stator of electric motors in electric cars. We use both 3D data and grayscale images as input. The primary challenges are that only a small dataset is available and that high network accuracy is essential to prevent defects in the usage of an electrical engine and to enable a focused rework process. We were able to achieve a 99% accuracy using either 3D data or grayscale images.**

***Index Terms*—machine learning, convolutional neural networks, electric motors, hairpin, quality control, production**

I. Introduction

The automotive industry is currently facing a decade of fundamental changes in its drive concepts. In the future, electric mobility will play a significant role in individual transport. Moreover, the disruptive nature of electromobility and its technologies have created new challenges [1]. This has been accompanied by a profound change in the structure of the vehicles and the components used. This development has far-reaching effects on production technology because the components of an electric powertrain clearly differ from those of a conventional powertrain. In addition, the lack of expert knowledge regarding new production techniques is problematic and has an impact on the quality of the components produced.

The use case presented in this paper is the welding of hairpins. Hairpins are a novel technology designed to increase the efficiency of an electric motor by replacing the traditional copper windings in the stator of an electric motor with thick copper bars. The free ends of the hairpins are connected via laser welding. This process is error-prone and difficult to monitor continuously. Currently, occurring defects cannot be classified and reworked during production.

Therefore, this paper presents an approach to classify quality deviations for the welding of hairpins, which enables a rework process that is part of the production process. Different network architectures of convolutional neural networks (CNN) are analyzed and compared. In addition, different inputs of the CNN are used and the results are evaluated: 3D scans and grayscale images are used for this purpose. Grayscale images offer potential financial savings in comparison to expensive 3D scans.

II. State of the Art

In recent years, many different architectures for object detection and object recognition have evolved and significant improvements with respect to performance have been achieved. The most common architectures for object detection are single-stage and two-stage detection networks. Single-stage networks, such as SSD [2] or YOLO [3], combine object detection and object classification in a single stage, whereas two-stage detection networks are more complex and separate the object detection task from the object classification task. There are two-stage architectures based on region convolutional neural network (R-CNN) [4] and architectures evolving from it like Fast R-CNN [5], Faster [6] or Mask R-CNN [7]. The detection of hairpins is considered to be a solved problem. Consequently, the focus of this study is on object classification.

The longer it takes to detect missed production errors, the greater the expenses for correction. Therefore, quality control is crucial and the number of missed faults needs to be minimized. However, only a few machine learning (ML)-based applications to detect quality deviations are common in this area [8], [9]. Further approaches of ML algorithms in the production of electric engines are summarized by Mayr et al. [10]. Additionally, there is only one application of ML-based methods for detecting quality deviations in the welding process of hairpins [11]. Mayr et al. used a charge-coupled device (CCD) camera to detect quality deviations with the help of a CNN. However, the accuracy of the network was only in the range of 61% to 91%. This is too low for industrial applications. This paper uses two approaches to improve the accuracy of such detections.

- First, 3D scans are applied instead of images from a CCD camera. These scans serve as the input for a CNN that was self-developed to be able to detect quality deviations in the welding.
- Because such a 3D scanner is considerably more expensive than a common industrial camera, different network architectures are compared to achieve a higher accuracy with common grayscale images.

III. Methodology

A. Business Understanding

An important step in the production of the stator is the deformation of the copper rod into a shape similar to a hairpin. These hairpins are subsequently inserted into the stator lamination stack, followed by twisting the exposed ends of the hairpins and connecting them by laser welding [12]. However, the problem with this process is that copper has strong reflective properties and therefore cam hardly absorb any radiation. Thus, in contrast to the welding of steel or aluminum, a higher laser power has to be applied, which results in characteristic defects during welding [13]. Therefore, we analyzed the welding process in detail and identified types of errors that regularly occur during the welding process. The classification of welding is divided into four classes as shown in Fig. 1: Correct welding (CW), insufficient welding (IW), weld spatter (WS) and weld craters (WC).

Currently, there is no automated fault classification system. Therefore, the stator passes all further processing steps until its final inspection at the end of the production line. In case of a faulty weld seam, the stator is removed from the production line, disassembled and manually re-welded. This process is time-consuming and expensive. To establish a more efficient process, welding defects have to be detected earlier to enable targeted re-welding in large scale series production.

B. Experimental setup

Because only the top of the hairpins is of importance for the welding quality, stripped copper wire pieces with a length of $100mm$ were used. These wire pieces were inserted into a test carrier, where two pins were welded together.

A 3D camera, XR-HT40M from Keyence, was used to capture 3D data of the welding seams along with grayscale images. The advantage of a 3D camera over a classic 2D camera is a higher stability of the inspection because the height information provides important insights and features for the inspection process. However, a disadvantage is the price,

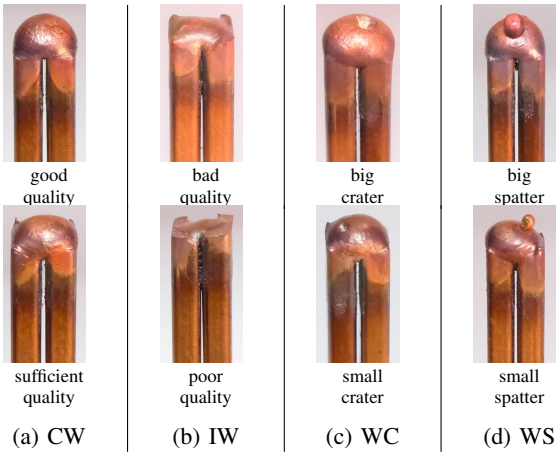

Fig. 1: Representation of the four quality classes that result from the welding process of hairpins.

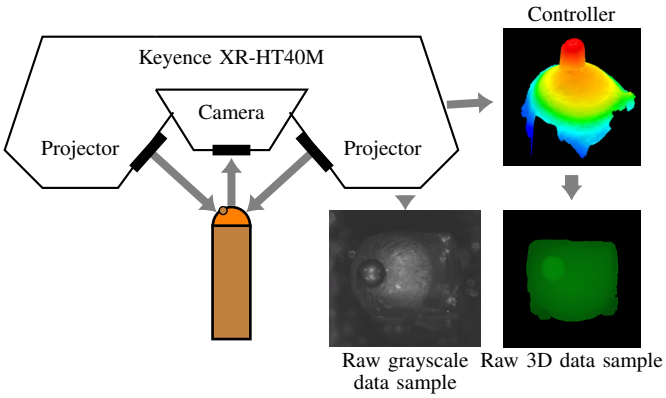

Fig. 2: Experimental setup.

which is significantly higher than the price of a conventional 2D camera for industrial applications. Fig. 2 schematically illustrates the process of recording the height information and the resulting RGB image of a welding spatter (WS) and the grayscale image. The images were recorded by a grayscale camera with a resolution of $2048x2048$ pixels covering an area containing four hairpins. Based on information from the 3D scanner, hairpin detection was executed, resulting in images of hairpins with a resolution of $300x300$ pixels. Object detection in grayscale images can be achieved using sate-of-the-art image processing methods and is not further investigated throughout this work.

C. Data generation

The aim is to generate a dataset covering all required classes. This is a challenging task to execute in an industrial production line. As a result, 550 to 600 images of hairpin welds could be generated for each class with different degrees of defect severity. To further increase the number of samples in the training set, data augmentation using rotations, shifts and mirroring was implemented. Using these techniques, a more realistic training set is generated. For example, in serial production, it cannot be guaranteed that the hairpin image will always be centered and rotated similarly. The data augmentation ensures that the network learns these variations. However, it is important that the synthetically generated images are not included in the test set to guarantee unbiased results. A detailed breakdown of the dataset including data augmentation is shown in Table I. The datasets for 3D and graysale

TABLE I: Division of the dataset for the failure classes

Class	Training set	Test set	Sum
IW	456	104	560
WS	455	102	557
WC	438	125	563
CW	478	126	604
Sum	1,827	457	2,284
augmented	91,350	-	-

images were split into training and validation datasets with 80% training data and 20% validation data.

D. 3D Data Analysis

1) Preprocessing: We preprocessed the 3D scans to increase the resulting performance of a classifier and its accuracy. This preprocessing pipeline is divided in five parts.

- The images are cropped around the center of gravity with the size $450x450$ pixels, as shown in Fig. 2, for a raw 3D data sample.
- The colour of the 3D scan is converted into height information. The values are in a range of $0 - 16mm$. The result is shown in Fig. 3.
- The 3D data was reduced to a size range of $30x30$ pixels in the x and y directions. In addition, the altitude range was scaled to a value range between 0 and 255.
- The preprocessed 3D information was saved as a grayscale image.
- The image was normalized based on the subtraction of the mean of one pixel over the entire dataset, dividing by the standard deviation of these pixels.

2) 3D data analysis based on own network design: The basic structure of the model is based on convolutional blocks (Conv-Block). The used Conv-Blocks consist of consecutive convolutional layers including batch normalization and rectified linear unit (ReLU) activation function followed by a pooling layer. The proposed structure uses four Conv-Blocks, where the number of filters is doubled from one Conv-Block to the next. This means that the first Conv-Block consists of two convolutional layers, each with eight kernels with a size of $3x3$ pixels. The second, third, and fourth layers have the same structure; however, the number of kernels is doubled after each repetition. The fourth block applies global average pooling instead of max pooling.

This is followed by a fully connected layer with 32 neurons, a batch normalization layer, a ReLU activation function and a dropout layer with a dropout rate of 0.5. Finally, there is an output layer with one neuron for each class and a softmax activation function to create a probability output for each class.

To find the optimal training parameters, a stochastic gradient descent algorithm based on a batch size of 150 was used. An adaptive step size of $1e-3$ was defined, which is reduced to 0.9 times every five training epochs. Furthermore, a Nesterov momentum of 0.9 was applied to this optimization method.

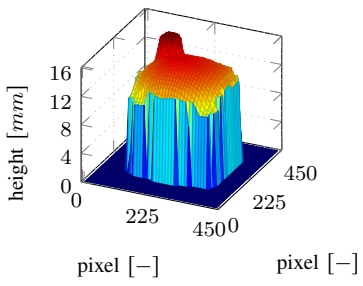

Fig. 3: Height transformation of the 3D data.

The cost function, with which the gradients for training the model can be calculated, is based on the categorical cross entropy in this application.

To assign a special importance to certain classes while training a model or to compensate for an uneven amount of training data per class, a class weighting can be integrated into the cost function. Because the detection of faulty welding needs to be particularly reliable, the costs for the predictions of these classes are weighted more strongly. This leads to higher costs if these welds are incorrectly classified, and therefore the gradients are more strongly influenced. In this application, the weighting of the different classes was set to

$$\begin{bmatrix} w_{CW}, & w_{IW}, & w_{WS}, & w_{WC} \end{bmatrix} = \begin{bmatrix} 1, & 10, & 10, & 10 \end{bmatrix}$$

This weighting means that an incorrect classification of the incorrect classes is weighted 10 times as strongly in the costs as an incorrect classification of the correct class.

E. Grayscale Data Analysis

The grayscale image data was recorded simultaneously to the 3D data, which allows for a direct comparison of the approaches. The hairpin crops were normalized and re-sized to the required input resolution of the used network. To identify the most suitable network architecture, two different networks based on the grayscale data were implemented. Inception V3 was chosen as a network architecture because it has been proven to work for multiple use cases when using transfer learning, as presented and executed for example in Ref. [14]. The feature extraction process was adapted to the requirements of our use case using an input size of $300x300$ pixels. In addition, our own network architecture was based on the same images but with a reduced resolution of $30x30$ pixels. This network addresses our specific needs and was trained from scratch.

1) Grayscale analysis based on Inception V3: Inception V3 [15] was used as a base network architecture. This architecture was chosen primarily due to its increased efficiency, which is achieved by breaking large convolution kernels down into multiple smaller convolution kernels, resulting in a reduced number of parameters. In addition, Inception weights pretrained on ImageNet data are available. These weights are used as a starting point even though the present problem is different from classifying the regular images found in ImageNet. The feature extraction process for images is similar in functionality, which allows us to tailor the weights and the feature extraction to our specific problem during training. Our network uses all layers of Inception V3 until mixed 10 layer, which is followed by three fully connected layers and a softmax layer. The two first fully connected layers have 1024 neurons, followed by a layer with 512 neurons, followed by the softmax layer with four output classes. Each of the fully connected layers is followed by ReLU activation and a dropout layer with a ratio of 0.1. All layers are re-trained because the available weights based on ImageNet do not provide the required feature extracting capabilities but serve as an initialization for solving our specific problem. This decreases the training effort

in comparison to an initialization with randomly distributed weights.

We chose a categorical focal loss with $\alpha = 0.25$ and $\gamma = 2$ as our loss function. An Adam optimizer was used for the training with a learning rate of $lr = 5 \cdot 10^{-6}$ in the first training phase and a learning rate of $lr = 1 \cdot 10^{-6}$ in the second training phase, using the accuracy as a metric. The regularizers $\beta 1$ and $\beta 2$ were chosen to be $\beta 1 = 0.9$ and $\beta 2 = 0.999$. A 5-fold cross-validation was used to obtain the results. The training process was split into two phases to leverage the existing weights in the first phase and to address the specifics of our use case. The first training phase lasted 100 epochs and treated all classes equally; phase one used a higher learning rate than phase two, which lasted 50 epochs. In both phases, a batch size of 32 and 100 steps per epoch were used. To address the imbalance in the classes and more importantly the cost of errors, the classes were weighted such that

$$\begin{bmatrix} w_{CW}, & w_{IW}, & w_{WS}, & w_{WC} \end{bmatrix} = \begin{bmatrix} 1, & 0.4, & 0.75, & 0.35 \end{bmatrix}$$

2) Grayscale analysis based on own network design: Our own network architecture uses $30x30$ grayscale images as input. The network is separated into three main parts. The first stage of the network is formed by five convolutional layers, each followed by batch normalization, ReLU activation, and a maximum pooling layer. The used kernel sizes in this stage are $1x1$, $3x1$, $1x3$, $1x1$, and $3x3$ using a stride of 1. Consequently, the first stage serves as a feature extractor, which can be compared to a dense block as used in Ref. [16]. Stage one preserves the available input information by avoiding down sampling or by adding a maximum pooling layer before the discriminating features are extracted. The max pooling uses a stride of $2x2$. The second stage of the network is a convolutional layer with a $3x3$ kernel followed by batch normalization, activation, and maximum pooling layers. The purpose of this stage is to further refine and reduce the feature space. The final classification stage of the classifier consists of two fully connected layers, one with 1024 neurons and the other with 512 neurons, and a softmax activation layer. In addition to batch normalization and ReLU activation, a dropout of 0.1 is used after the first two layers. To train the network, the weights are initialized using the Glorot initializer [17]. All convolutional layers use 32 filters. In addition to a learning rate of $lr = 10^{-4}$, the training process uses focal loss and an Adam optimizer with a similar parameterization to that used for re-training the Inception V3-based architecture. Training was executed in two phases of 100 epochs each, with batch sizes of 32 and 100 steps per epoch. The first phase treated all classes equally, whereas in the second phase, the class weights as shown in Section III-E1 were used.

IV. Evaluation of experimental results

The results of all network architectures are compared based on the same metrics in this section. The proposed models was implemented in keras (version 2.2.4) using the tensorflow backend (version 1.14.0).

A. Training process

The training process follows the 5-fold cross-validation methodology. The result is a very stable and nearly ideal training process with a high accuracy was achieved with the 3D data as input. There are no fluctuations or other signs of overfitting or underfitting during the training.

As described in Section III, the training process for gray scale images follows a two phased training approach. Similar to the training process for 3D scans, 5-fold cross-validation is used. Neither for the inception based network, nor for our own design an indication for over- or underfitting could be identified. The validation accuracy curve observed over all training epoch in both phases for both networks, is constantly following the training accuracy curve closely.

B. Confusion Matrix

The confusion matrix (CM) resulting from the 5-fold cross-validation is shown in Fig. 4.

The results of the CMs indicate that the trained models using 3D data and grayscale images are suitable to separate the classes. This is illustrated by the high values on the CM diagonals, which are marked highlighted using a darker gray color. All entries that are not on this diagonal are incorrectly classified by the model.

C. Evaluation metrics

Numerous evaluation metrics can be calculated depending based on CM. These metrics quantify the classification capability of models. Therefore, we calculate the classification accuracy, precision, recall, and F_β-score. For most image classification challenges, the Top-1 and Top-5 accuracies are chosen as a comparison metric. However, this is not a suitable metric for critical image classification applications. The F1-score weighs the precision and recall equally, which implicitly assumes that the costs of all types of errors are equal. In our use case, the cost of incorrectly classifying a faultless sample as faulty is far less expensive than missing a faulty sample. Consequently, an F_β with a $\beta = 3$ is chosen to allow

		3D data				30x30 grayscale				300x300 grayscale			
		predicted class				predicted class				predicted class			
		CW'	IW'	WS'	WC'	CW'	IW'	WS'	WC'	CW'	IW'	WS'	WC'
true class	CW	0.99	0.11	0	0	0.91	0.09	0	0	0.93	0.07	0	0
	IW	0.01	0.99	0	0	0	1	0	0	0	1	0	0
	WS	0	0	1	0	0	0.06	0.94	0	0	0.01	0.97	0.02
	WC	0	0	0	1	0	0	0.15	0.85	0	0	0	1

Fig. 4: Visualization of the classification results based on relative values. Left: CM with 3D data. Middle: CM with $30x30$ grayscale data using our own network architecture. Right: CM with $300x300$ data using the Inception V3-based architecture.

a meaningful comparison and emphasize the importance of avoiding missing faulty samples.

$$F_\beta = (1+\beta)^2 \cdot \frac{precision \cdot recall}{\beta^2 \cdot precision + recall} \quad (1)$$

TABLE II lists the classification accuracy, precision, recall, and F_β-score as described in Equation 1 of the different networks.

As shown in TABLE II, the accuracy with 3D data and grayscale images of $300x300$ pixels is over 99%. Using only grayscale images with a size of $30x30$ pixels reduces the accuracy by approximately 5%. These results demonstrate that the proposed CNN architectures are capable of reliably deciding whether a welding process is faulty. A high recall rate is a particularly important requirement for a reliable production systems. The proposed models with 3D data and $300x300$ pixel grayscale images achieved a high average recall rate of over 99%, which means that, on average, only 1% of failures were not detected or classified as other failures. The grayscale images of $300x300$ pixels result in a worse value for the recall.

V. Conclusion and Outlook

Our results show that addressing the quality control problem of hairpin welding is feasible and shows satisfying results. Throughout this work the input data from a 3D scanner and gray scale images were used to train different neural networks. In this study, input data from a 3D scanner and grayscale images were used to train different neural networks. The classification based on the 3D scanner and large grayscale images achieved comparable accuracies and F_β-scores, for $\beta = 3$. Our own grayscale image-based neural network design performed slightly worse than the other two networks. However, with a drastically reduced network and image input size compared to the Inception V3-based network, the results are very encouraging. Both the 3D image-based feature space and the grayscale-based feature space allow for a straightforward class separation due to the height differences of the classes. The most crucial aspect is the price of a camera-based system compared to a 3D scanner, which is significantly more expensive. In addition, camera-based systems can be easily adapted to a new use case whenever necessary. Especially in an industrial environment, where lower quantities or more frequent adjustments to the manufacturing process are necessary, this is of high importance. In such cases, only software changes need to be applied, assuming that the new problem can be solved using a camera.

TABLE II: Accuracy, Precision, Recall and F_β-score

Type	Overall Accuracy (%)	Average Precision (%)	Average Recall (%)	Average $F_{\beta=3}$-score (-)
3D Scan	99	99	99	1.59
Image (300x300)	99	99	99	1.58
Image (30x30)	94	96	92	1.48

In future work, the developed solutions will be integrated into the manufacturing process and the results will be evaluated with an available state-of-the-art 3D scanner. Depending on the results, it might be necessary to improve the developed networks. In addition, more training data will allow for a better generalization of the networks, which could be done prior to integration into production.

References

[1] P. Burggräf, A. Kampker, C. Deutskens, and C. Niebuhr, *Competitive Strategies for Value Creation During Disruptive Innovations*, 03 2013, pp. 469–477.

[2] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Fu, and A. Berg, "Ssd: Single shot multibox detector," in *European conference on computer vision*. Springer, 2016, pp. 21–37.

[3] J. Redmon and A. Farhadi, "YOLO9000: better, faster, stronger," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7263–7271.

[4] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[5] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.

[6] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99.

[7] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[8] M. Weigelt, A. Mayr, J. Seefried, P. Heisler, and J. Franke, "Conceptual design of an intelligent ultrasonic crimping process using machine learning algorithms," *Procedia Manufacturing*, vol. 17, pp. 78–85, 2018.

[9] A. Mayr, A. Meyer, J. Seefried, M. Weigelt, B. Lutz, D. Sultani, M. Hampl, and J. Franke, "Potentials of machine learning in electric drives production using the example of contacting processes and selective magnet assembly," in *2017 7th International Electric Drives Production Conference (EDPC)*. IEEE, 2017, pp. 1–8.

[10] A. Mayr, M. Weigelt, J. von Lindenfels, J. Seefried, M. Ziegler, A. Mahr, N. Urban, A. Kuhl, F. Huttel, and J. Franke, "Electric Motor Production 4.0 – Application Potentials of Industry 4.0 Technologies in the Manufacturing of Electric Motors," in *2018 8th International Electric Drives Production Conference (EDPC)*. IEEE, 04.12.2018 - 05.12.2018, pp. 1–13.

[11] A. Mayr, B. Lutz, M. Weigelt, T. Gläßel, D. Kißkalt, M. Masuch, A. Riedel, and J. Franke, "Evaluation of Machine Learning for Quality Monitoring of Laser Welding Using the Example of the Contacting of Hairpin Windings," in *2018 8th International Electric Drives Production Conference (EDPC)*. IEEE, 2018, pp. 1–7.

[12] A. Riedl, M. Manusch, M. Weigelt, T. Gläßel, A. Kühl, S. Reinstein, and J. Franke, "Challenges of the Hairpin Technology for Production Techniques," in *2018 21st International Conference on Electrical Machines and Systems (ICEMS)*, 2018, pp. 2471–2476.

[13] P. Schmidt, "Laser beam welding of electrical contacts of Lithium-ion batteries for electric- and hybrid-electric vehicles," Dissertation, Technical University of Munich, Munich, 2015.

[14] H. Shin, H. Roth, M. Gao, L. Lu, Z. Xu, I. Nogues, J. Yao, D. Mollura, and R. Summers, "Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning," *IEEE transactions on medical imaging*, vol. 35, no. 5, pp. 1285–1298, 2016.

[15] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.

[16] G. Huang, Z. Liu, L. Van Der Maaten, and K. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[17] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 249–256.