Deep Learning for Partial Discharge Detection

Tim von Hahn

***Abstract*—In the field of predictive maintenance, the use of machine learning methods in is often hampered by two problems the lack of available data (and in particular, imbalanced data sets) and the need for specific domain knowledge for signal processing (feature engineering). This paper seeks to tackle these two problems in the context of detecting partial discharge faults on medium voltage power lines, specifically through the use of deep learning. Unsupervised pre-training, through a Convolutional Autoencoder (CAE), is used to generalize a network on a large set of unlabeled data. The weights from the encoder side of the CAE are then transferred to a Convolutional Neural Network (CNN), whereby the network is fine-tuned to detect partial discharge faults (minority class). The described methodology is compared to the performance of a simple Convolutional Neural Network without unsupervised pre-training. The shortcomings of the methodology, and further improvements, are discussed.**

I. Introduction

DEFECTS in the insulation of electrical transmission cables can lead to a phenomenon known as partial discharges (PD). During this phenomenon, an electrical discharge partially bridges the insulation of the transmission cable, and over time, this repeated process can lead to the failure of the cable. As such, regular testing of transmission cables is performed to detect signs of incipient failures.

Detecting partial discharge faults in electrical transmission cables lines is a subset of the field of predictive maintenance. Traditional machine learning techniques have been used in the domain for many years. Typically, this process starts with an individual, possessing strong domain knowledge, selecting features (often with advanced signal processing). Following feature engineering, a machine learning algorithm, such as an SVM, is then used to classify the signal.

However, with the rise of deep learning, the paradigm of designing features with specific domain knowledge has shifted to that of designing network architectures. Recent research within predictive maintenance has focused on designing effective end-to-end networks (networks with minimal feature engineering). In addition, as with many real-world applications of machine learning, the data available is often highly unbalanced (that is, there are many more samples of one class than the other).

The research presented here explores both the use of an end-to-end network, and a method to tackle highly imbalanced data. These are explored in the context of predicting partial discharge faults on medium voltage power lines, with the data provided by the Technical University of Ostrava. First, a Convolutional Autoencoder (CAE) is trained in an unsupervised manner. The weights from the encoder side of the CAE are then transferred to a convolutional neural network (CNN), which is fine-tuned with a balanced selection of partial discharge faults and clean signals.

II. Related Work

Partial discharge signals "are usually very weak and superimposed on noise, interferences, disturbances of various kinds, and they are time dependent." [1] As such, much effort in the past has gone into isolating the partial discharge patterns through signal processing methods. Of these methods, frequency analysis and waveform decomposition have been most prevalent. Ahmed et al., for the first time used digital frequency spectrum analysis to detect partial discharges in electrical cables in 1998. [2] Later Ma et al. used waveform analysis to isolate partial discharge pulses from noisy signals. [3]

Neural networks have been used for many years to detect PD faults in electrical equipment. Hozumi used a neural network to detect partial discharge patterns in epoxy resin. A small three-layer network was used, consisting of one hidden layer, for a total of 3,232 parameters. [4] Suzuki et al. used a similarly small neural network to detect partial discharge faults in high voltage power lines. In that experiment, PD signals were induced in the lines and the discharge magnitude, number of PD pulses, and phase angle were recorded to create a basket of features to be used by a neural network with one hidden layer. [5]

With recent advances in computing power, deep learning methods have been employed extensively in the field of predictive maintenance. Specific to partial discharge detection, Gaoyang et al. used a convolutional neural network to classify PD signals in gas insulated switchgear. They first preprocessed the data with the Short Time Fourier Transform (STFT) to produce spectrograms. The spectrogram images were then fed into a 3 layer CNN, with fully connected layers, to classify the signals. [6] The methodology employed by Gaoyang is similar many methods of speech recognition using CNNs. Li et al. expand on Gaoyangs technique by using the Gabor representations – a sub-set of the short-time Fourier transform [7]. They also use a CNN, except that they apply time, frequency, and texture filters to the spectrograms of varying resolution.

In a recent paper, Khan et al. use a single layer 1D CNN, with a fully-connected layer, to detect partial discharge faults on experimental data. The single convolutional layer (with 32 filter maps, a receptive field of 10, and stride of 1) is followed by a max-pooling layer of size 2 and stride of 2. The output from the max-pooling is flattened and sent through a fully connected layer of 8 neurons. They use synthetically generated data to train the network. Their experiment is the first time a 1D CNN is used, in literature, to detect PD faults from raw waveforms. [8]

A. Convolutional Neural Networks and Raw Waveforms

The Convolutional Neural Network (CNN) was a concept refined by Yann LeCun et al. in 1989. [9] In a CNN, filters (or kernels) are passed over the input space. At the core of a CNN is the convolution. Each filter is "convolved" (that is, a dot product is computed) between the filter and the input. Consequently, the network is able to learn features – for example, finding the beak of a bird in an image. The CNN is trained through backpropagation.

Convolutional neural networks have been used, effectively, in many applications. In particular, much research has been done with computer vision using CNNs. AlexNet, from 2012, is one of the most famous examples after significantly beating the previous best score in the ImageNet competition. [10]

CNNs were also being used at this time for recognition of human speech. However, speech recognition with CNNs were still using signal processing techniques, such as using spectrograms, rather than working on raw waveforms. This changed in 2013 when speech recognition was performed on raw waveforms using 1D CNNs. [11] Now most speech recognition is performed directly on the raw waveforms.

Several researchers postulate why CNNs can learn directly from raw waveforms. It appears that CNNs learns from raw waveforms by learning representation of the frequencies; that is, each convolutional layer is a stack of frequency components. [12] [13] [14] As raw audio waveforms are nothing more than time-series data, the principle of CNNs learning frequency components also applies to electrical signals (also time series data) in partial discharge detection tasks.

B. Convolutional Autoencoders

Hinton proposed the "autoencoder" as a means to reduce the dimensionality of data with neural networks in 2006. [15] Several years later, Masci et al. used a convolutional autoencoder (CAE) to pre-train a CNN on the MNIST data set. They obtained superior results to only using a CNN. The CAE architecture is like that of an autoencoder, except that – as in a CNN – at each layer the filter is convolved with the input. In Mascis instance, the CAE also inherited the property of weight sharing. [16] In our experiment, we use a CAE to pre-train our network.

C. Imbalanced Data

Much deep learning research is performed using balanced data sets. However, within the field of predictive maintenance – as with many disciplines – much of the real-world data is highly unbalanced. This is the case for the PD data set being explored in this paper. Working with unbalanced data sets is still a practical area of research and there are several ways to approach the problem. [17]

Increasing the data set size through data augmentation is one such method to work with imbalanced data. Li explores this in the context of detecting machinery failures through raw vibration waveforms using CNNs. Li uses Gaussian noise; masking noise; signal translation; amplitude shifting; and time stretching to augment the data set. Overall, it was found that when dealing with raw waveforms, the signal translation provided the most benefit. [18]

Bootstrapping is another method that can be used to tackle imbalanced data. In bootstrapping, a balanced training set is created by randomly sampling from the minority class data. Yan et al. use bootstrapping to help a CNN learn while trying to classify media. [19]

This paper will use transfer learning to work with the unbalanced data. In transfer learning, the weights gained from learning one problem are "transferred" to a similar, but different, problem. In our case, the weights from the CAE (only the encoder side), that was used to learn a sparse representation of the partial discharge signals, will be transferred to a CNN with fully connected layers. The weights of the CNN, excluding the fully connected layers, are frozen and then the network is fine-tuned. This allows the network to learn on far less data than would be possible if the transfer learning had not been completed.

III. Data and Model

A. Data Description

The medium voltage partial discharge data set is provided by the Technical University of Ostrava, in the Czech Republic. A total of 8712 labeled signals are provided, with an additional 20,130 unlabeled signals.

The labeled data set is unbalanced. Of the 8712 labeled signals, only 525 (6%) have partial discharge faults. The remainder of the signals are clean; that is, having no indication of a partial discharge fault. Each signal is taken over 20 milliseconds, or one cycle of the 50 Hz electrical grid. Thus, each signal consists of 800,000 measurements. During the collection of the data, simultaneous measurements were taken on each of the three phases of the medium voltage power lines, as demonstrated in Figure 1. Finally, the partial discharge faults were identified by experts.

B. Preprocessing

Downsampling was performed on each signal in the data set to reduce temporal resolution. Each signal was broken up into 800 sequential windows of time with each window containing 1000 data-points. The mean was then calculated for each window to create the downsampled signal of 800 data-points. Normalization, between -1 and 1, was then performed. A visualization of a downsampled signal is shown in Figure 2. Note that each signal is fed into the models as a vector, not as an image.

C. Models

The primary model was built in a two-step training method. First, a Convolutional Autencoder (CAE) was trained on all the data (28,842 signals) in an unsupervised manner.The CAE consisted of an encoder and decoder portion containing one-dimensional (1D) convolutional layers, with a general architecture similar to that in Dai et al. [20] The first 1D convolutional layer filtered the 800x1 downsampled signal into 64 filters (or feature maps) with a receptive field (kernel

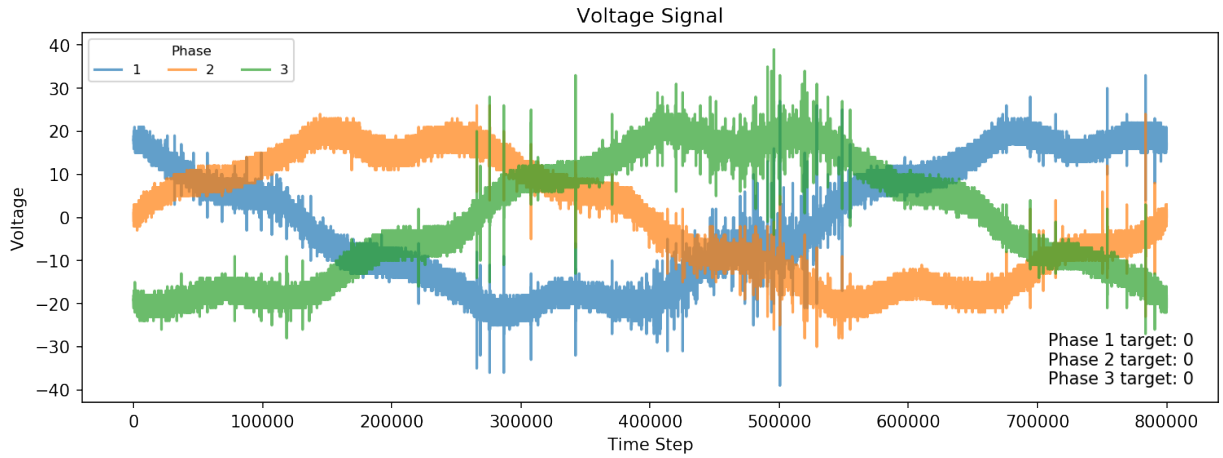

Fig. 1. Three phases of a voltage signal. None of these signals have a partial discharge signal present, however, these signal are representative of the noise that is generally present in the data set.

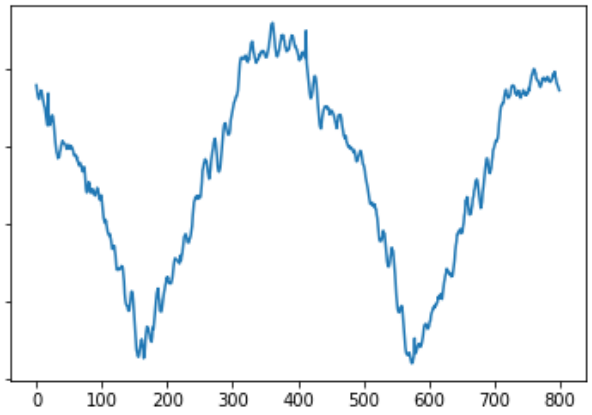

Fig. 2. Visualization of a downsampled and normalized signal. Note, data is fed into model as a vector, not as an image.

size) of 3. The second layer took the output from the first convolution and applied 128 filters (again, with a receptive field of 3). Following that was a max-pooling layer of 16. The third and fourth layers followed each other consecutively, and both contained 256 filters with a receptive field of 3. Batch normalization was used after the second and fourth convolutions.

The decoder mirrors the encoder. First, a 1D convolutional layer, with 256 filters and a receptive field of 3, takes the output from the encoder. This was followed by an up-sampling layer of size 16. A second convolution layer followed, with 128 filters and a receptive field of 3. Batch normalization occurred, with the final convolutional layer – with one filter and a receptive field of 3 – creating the reconstituted signal of 800x1.

After training of the CAE, the decoder portion was discarded. The weights from the encoder were frozen in place. A global average pooling layer was attached, followed by two fully-connected layers (with 20 neurons in each layer), and a single output neuron. This "new" network – the primary model – was fine-tuned on the training set to recognize the partial discharges. The network architecture is illustrated in Figure 3.

A benchmark model was created to get a baseline to compare the primary model against. The benchmark model is a simple CNN with two 1D convolutional layers and a fully connected layer. The first one-dimensional convolutional layer took the input (800x1) and applied 30 filters with a receptive field of 5. A max-pooling layer of size 16 followed. A second convolutional layer (applying 10 filters with a receptive field of 3) is then followed by a global average pooling layer. A fully connected layer of 12 neurons was next. Finally, there was a single neuron with a sigmoid activation to create the output. The network architecture is also illustrated in Figure 3.

IV. Experiment

The process of splitting the training, validation, and testing data was done as per Table I, below. The minority class (signals with partial discharges) was small at 525 signals total. Consequently, this limited the size of the training/testing set for the fine-tuning of the model. However, the training of the CAE was completed on all the labeled and unlabeled data of 28,842 signals (with 20% being held back for validation of the CAE).

TABLE I
Data Splits for Training, Validation, and Testing

Data Split	Signal Count	Description
Test Set	210	50/50 split of PD and Clean
Validation Set	168	50/50 split of PD and Clean
Train Set	672	50/50 split of PD and Clean
CAE Set	28,842	All signals, including unlabeled

A. Convolutional Autoencoder

ReLu activations were used throughout the CAE, and each convolutional layer utilized "same" padding. The "same" padding results in an output that has the same length as the input. A stride of 1 was used throughout. A glorot uniform schema was used to initialize the weights.

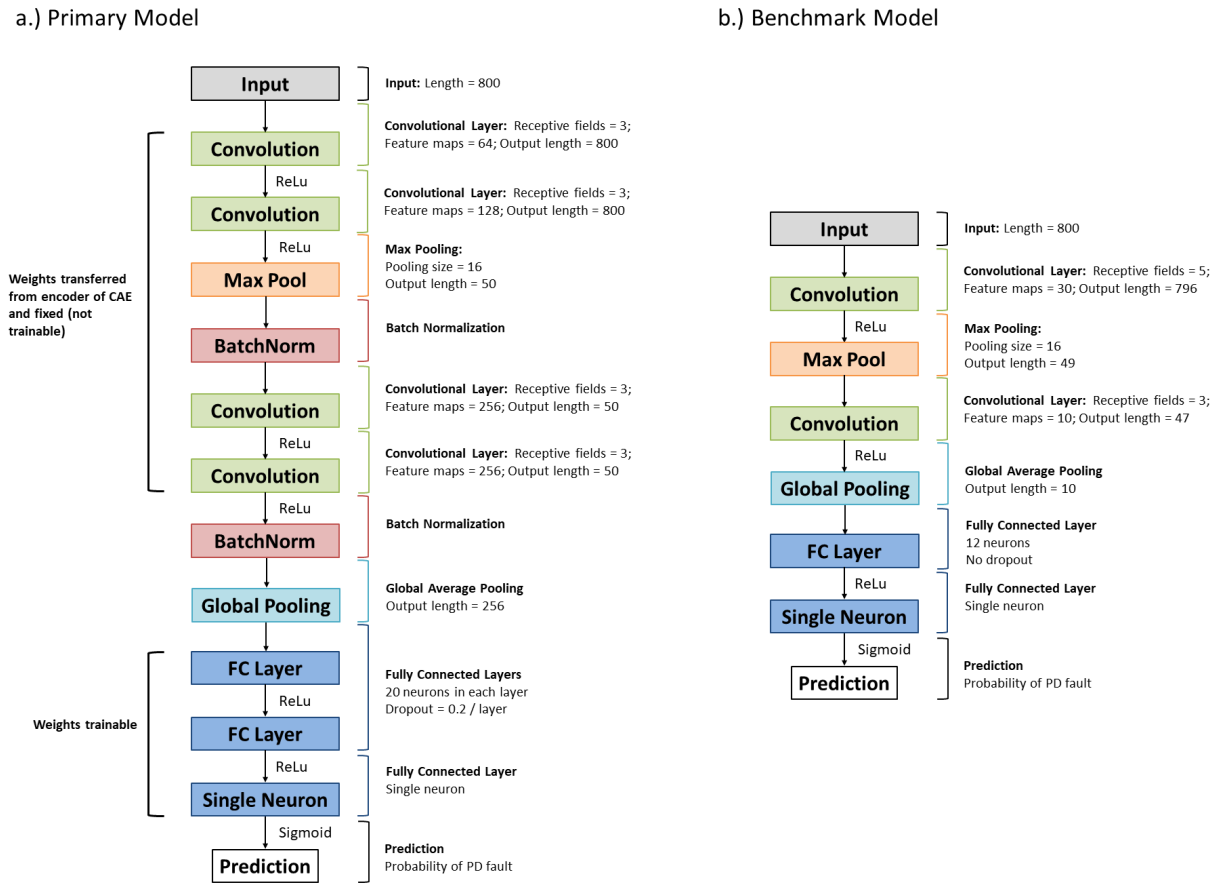

Fig. 3. The architecture of a.) the Primary Model, and b.) the Benchmark Model. The primary model has the weights from the encoder side of the CAE transferred to it. Only the fully connected layers in the Primary model are trainable.

The model was trained for 200 epochs, with a mini-batch size of 1024, and the data shuffled between epochs. The Adam optimizer was used with mean-squared-error as the loss function.

The implementation of the CAE, and all subsequent models, was completed with TensorFlow and Keras. The code can be found on the author's github page. [21]

B. Primary Model

After training the CAE, the decoder portion of the CAE was discarded. The weights of the encoder were then fixed in place, with several fully-connected layers attached, and one output neuron. The network utilized ReLu activations throughout, however, the final output neuron utilized a sigmoid activation. Dropout was performed on each fully-connected layer.

Grid search was used to select the optimum network architecture for the primary model. The number of filters and size of the receptive field within the convolutions; the max-pooling size; the number of fully-connected layers; the number of neurons in each fully-connected layer; and the size of the dropout, were all varied to select the best performing model. The final model contained 327,501 parameters.

With the weights on the "encoder" portion of the model fixed, training only occurred on the fully-connected layers (5,581 parameters) with the glorot uniform schema being used for weight initialization. A mini-batch size of 32 was used with the data being shuffled after each epoch. The Adam optimizer was used with binary cross-entropy being the loss function. Early stopping, based on validation loss, was used to select the model with the highest accuracy.

C. Benchmark Model

A relatively simple architecture was selected for the benchmark model. The benchmark model employed convolutions with a stride of 1 and no padding. ReLu activations were present throughout the model, except where the output went through a sigmoid activation for the final prediction.

Grid search was used to optimize the network structure. The number of filters and size of the receptive field within the convolutions; the number of convolutions; the max-pooling size; the number of fully-connected layers; the number of neurons in each fully-connected layer; and the size of the dropout, were all varied. The final model contained 1,235 parameters.

Early stopping on validation loss was used to select the best model based on validation accuracy. Training occurred with a mini-batch size of 32 and glorot uniform to initialize the weights. The data was shuffled at each epoch. The Adam

optimizer was used with binary cross-entropy being the loss function.

V. Analysis

The primary model achieved a test accuracy of 74.68%, as shown in Table II. Surprisingly, the benchmark model achieved a slightly better test accuracy of 75.31%. The results speak to the fact that, despite utilizing transfer learning and a deeper and more complex network, the primary model was unable to learn features that were superior to the much simpler benchmark model. However, the models did demonstrate divergence in the training time. The primary model took 236 epochs to train, whereas the benchmark model took 6869 epochs. This shows the effectiveness of transfer learning to reduce training time.

TABLE II
Validation and Testing Accuracy

Model	Validation Accuracy	Test Accuracy
Primary	74.52%	74.68%
Benchmark	80.25%	75.31%

Of special note in the results, for the primary model, is that the testing accuracy improved when compared to the validation accuracy. This could be due to the random split of the testing data such that the test data has samples which are better suited for the primary model. Given more time, cross-validation should have been used.

From the experimental results it is concluded that the primary model, pre-trained using transfer learning from the CAE, did not outperform the much simpler CNN (the benchmark model). Aggressive downsampling is the primary reason suggested for the marginal performance of the primary model. Partial discharge events are often masked by noise, and thus, with the downsampling calculated via the mean over a large window, the indication of any partial discharge was likely lost. Consequently, the primary model was not able to leverage the larger data set from the training of the CAE since the partial discharge "information" had been removed by downsampling. Thus, its results are similar to that of the benchmark model.

A. Further Work

There are several potential methods to improve the results. The first is to reduce the downsampling so as to preserve important partial discharge information. The simplest method would be to decrease the window size in downsampling from 1000 data-points to, say, 500. However, decreasing the window size can significantly increase the number of parameters in the model. Nonetheless, this should be explored further.

Creating a basket of features – as with variance, frequency spectrum data, or wavelet energy, as an example – is commonly used in predictive maintenance applications of deep learning. [22], [23] In the research presented here, only the mean value was used in preprocessing. Thus, for further improvement, additional features could be tested, either separately or in an ensemble together. However, doing this would negate much of the benefit of having an end-to-end model in that it would require human domain knowledge to create these features. Thus, this is not the preferred method to pursue.

The preferred method is to keep the end-to-end architecture. Convolutional layers could be used to effectively "downsample" the data early in the model. Dai et al. use large strides in their first couple convolution and max-pooling layers to achieve a reduced temporal resolution. [20] In addition – and because the data is temporal in nature – causal convolutions could be explored as a means to reduce the number of parameters in the model, as was done with Van Den Oord et al. [24] Fewer parameters with the use of causal convolutions would allow for smaller strides and max-pooling sizes, thus preserving relevant partial discharge information.

Given additional time, further network architectures – particularly the use deeper networks and different hyper-parameters – should be explored. Random search, as suggested by Bergstra et al., should be used in lieu of the simple grid search. [25]

Finally, methodological methods should be explored to further manage the unbalanced data set. The use of bootstrapping and data augmentation, using time stretching, should be explored.

VI. Conclusion

In this research, deep learning was used to tackle the problems of imbalanced data sets, and the need for specific domain knowledge for feature engineering. The problems were explored in the context of detecting partial discharge faults on medium voltage power lines. To address the problem of specific domain knowledge, an end-to-end architecture was employed that utilized minimal preprocessing. To mitigate the problem of unbalanced data, unsupervised pre-training, through a Convolutional Autoencoder (CAE), was used. The CAE was generalized to a large set of unlabeled data. The weights from the encoder side of the CAE were then transferred to a Convolutional Neural Network (CNN), whereby the network was fine-tuned to detect partial discharge faults (the minority class). This CNN – the primary network – was then compared to a benchmark network, a simple CNN.

The results show that the pre-training from the CAE improved training time on the subsequent primary network. However, the results do not demonstrate that the primary network is superior to the simple benchmark CNN. This is largely attributed to the aggressive downsampling of the data that occurs in preprocessing. Subsequent studies should work to reduce the downsampling. Preferably, the downsampling can be eliminated altogether with the use of larger strides in convolutional and max-pooling layers, along with the use of causal convolutions.

References

[1] B. Fruth and L. Niemeyer, "The importance of statistical characteristics of partial discharge data," *IEEE Transactions on Electrical Insulation*, vol. 27, no. 1, pp. 60–69, Feb 1992.

[2] N. H. Ahmed and N. N. Srinivas, "On-line partial discharge detection in cables," *IEEE Transactions on Dielectrics and Electrical Insulation*, vol. 5, no. 2, pp. 181–188, 1998. [Online]. Available: https://ieeexplore.ieee.org/document/671927/

[3] X. Ma, C. Zhou, and I. J. Kemp, "Interpretation of wavelet analysis and its application in partial discharge detection," *IEEE Transactions on Dielectrics and Electrical Insulation*, vol. 9, no. 3, pp. 446–457, 2002. [Online]. Available: https://ieeexplore.ieee.org/document/1007709/

[4] N. Hozumi, T. Okamoto, and T. Imajo, "Discrimination of partial discharge patterns using a neural network," *IEEE Transactions on Electrical Insulation*, vol. 27, no. 3, pp. 550–556, 1992. [Online]. Available: https://ieeexplore.ieee.org/document/142718/

[5] H. Suzuki and T. Endoh, "Pattern recognition of partial discharge in xlpe cables using a neural network," *IEEE Transactions on Electrical Insulation*, vol. 27, no. 3, pp. 543–549, 1992.

[6] G. Li, M. Rong, X. Wang, X. Li, and Y. Li, "Partial discharge patterns recognition with deep convolutional neural networks," in *2016 International Conference on Condition Monitoring and Diagnosis (CMD)*. IEEE, Conference Proceedings, pp. 324–327.

[7] G. Li, X. Wang, X. Li, A. Yang, and M. Rong, *Partial Discharge Recognition with a Multi-Resolution Convolutional Neural Network*, 2018, vol. 18.

[8] M. A. Khan, J. Choo, and Y.-H. Kim, "End-to-end partial discharge detection in power cables via time-domain convolutional neural networks," *Journal of Electrical Engineering & Technology*, pp. 1–11, 2019.

[9] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, Conference Proceedings, pp. 1097–1105.

[11] D. Palaz, R. Collobert, and M. M. Doss, "Estimating phoneme class conditional probabilities from raw speech signal using convolutional neural networks," *arXiv preprint arXiv:1304.1018*, 2013.

[12] Y. Hoshen, R. J. Weiss, and K. W. Wilson, "Speech acoustic modeling from raw multichannel waveforms," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Conference Proceedings, pp. 4624–4628.

[13] T. N. Sainath, R. J. Weiss, A. Senior, K. W. Wilson, and O. Vinyals, "Learning the speech front-end with raw waveform cldnns," in *Sixteenth Annual Conference of the International Speech Communication Association*, Conference Proceedings.

[14] S. Dieleman and B. Schrauwen, "End-to-end learning for music audio," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Conference Proceedings, pp. 6964–6968.

[15] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, p. 504, 2006.

[16] J. Masci, U. Meier, D. Cirean, and J. Schmidhuber, "Stacked convolutional auto-encoders for hierarchical feature extraction," in *International Conference on Artificial Neural Networks*. Springer, 2011, Conference Proceedings, pp. 52–59.

[17] R. Zhao, R. Yan, Z. Chen, K. Mao, P. Wang, and R. X. Gao, "Deep learning and its applications to machine health monitoring," *Mechanical Systems and Signal Processing*, vol. 115, no. Complete, pp. 213–237, 2019.

[18] X. Li, W. Zhang, Q. Ding, and J.-Q. J. J. o. I. M. Sun, "Intelligent rotating machinery fault diagnosis based on deep learning using data augmentation," 2018.

[19] Y. Yan, M. Chen, M.-L. Shyu, and S.-C. Chen, "Deep learning for imbalanced multimedia data classification," in *2015 IEEE International Symposium on Multimedia (ISM)*. IEEE, 2015, pp. 483–488.

[20] W. Dai, C. Dai, S. Qu, J. Li, and S. Das, "Very deep convolutional neural networks for raw waveforms," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 421–425.

[21] T. von Hahn, "Partial discharge detection with deep learning," https://github.com/tvhahn/PD-Deep, 2019.

[22] L. Jing, M. Zhao, P. Li, and X. Xu, "A convolutional neural network based feature learning and fault diagnosis method for the condition monitoring of gearbox," *Measurement*, vol. 111, pp. 1–10, 2017.

[23] F. Gu, H. Chang, F. Chen, C. Kuo, and C. Hsu, "Application of the hilbert-huang transform with fractal feature enhancement on partial discharge recognition of power cable joints," *IET Science, Measurement Technology*, vol. 6, no. 6, pp. 440–448, 2012. [Online]. Available: https://www.jove.com/video/58233/quantitative-analysis-thermogravimetry-mass-spectrum-analysis-for

[24] A. Van Den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. W. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio." *SSW*, vol. 125, 2016.

[25] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol. 13, no. Feb, pp. 281–305, 2012.