MagNet Challenge Final Submission Protocol

Jacob Reynvaan, Martin Stoiber

***Index Terms*—Machine Learning, Magnetic loss, Neural Networks, MagNet, Steinmetz.**

I. Introduction

THE MagNet Challenge, a practical initiative for engineering and science students, emphasizes the development of predictive models through machine learning. This challenge not only seeks to refine the accuracy of algorithms but also serves as a valuable platform for students to apply computational methods to real-world problems. In our participation, we initially faced the necessity to discard our original idea and shift our focus, adapting our strategy to the data. Our team's efforts have spanned various aspects of machine learning, from data analysis to model refinement. In our final presentation, we will share the insights we've gathered throughout this process.

II. Data Visualization and Feature Extraction

The initial phase involved visualizing the dataset to understand its features and identify trends or irregularities. This was achieved by generating various plots to examine the data's distribution. After this analysis, attention was redirected towards developing feature extraction methods. The plotting of these features was then employed to evaluate their contribution to improving data separability, a critical factor in enhancing the subsequent training phase's efficiency and accuracy.

III. The Original Idea

The initial proposal focused on utilizing Graph Neural Networks (GNN) in tandem with Symbolic Regression (SR) to accurately model magnetic core losses. (see figure 1) This approach involved training a GNN and applying SR to the edges of the graph, breaking down the complex problem into smaller, more manageable units. The concept was grounded in using SR, based on genetic algorithms, to find solutions through a novel integration with GNN.

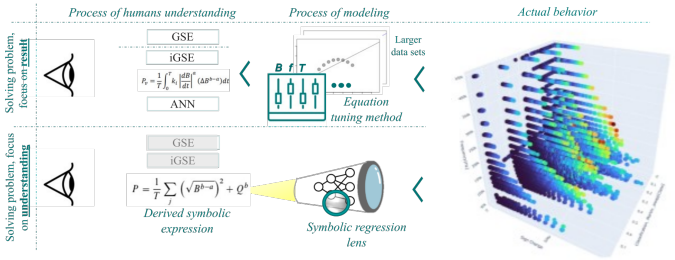

Fig. 1. From Data to Equation: A visual representation of the originally proposed method. The image depicts the transformation from raw data to a fully connected neural network (NN) model, and finally, the derived analytical expressions, general and easily perceived by humans.

A. Reasons Behind the Original Concept's Failure

The initial strategy entailed the utilization of GNN and SR as described in [1]. SR primarily relies on genetic algorithms, which do not guarantee an optimal solution. Hence, the proposed concept was to train a GNN and apply SR to the graph's edges, thereby breaking down the problem into smaller segments. However, due to the unsuitability of the given data for graph representation, we had to modify this approach. Figure 2 illustrates the arrangement of multiple waveforms, describing the magnetic flux, for each material grouped according to frequency and temperature. To determine the average core loss, polynomial regression was employed to obtain coefficients for these waveforms. Subsequently, SR was utilized for each coefficient to establish a formula that is dependent on both frequency and temperature. Unfortunately, the SR process did not yield any usable result. There are numerous factors contributing to the failure of this approach. The polynomial regression model was either underfitted or performed poorly. Additionally, it is likely that there does not exist a general description for each coefficient. Moreover, the coefficients may not solely depend on frequency and temperature.

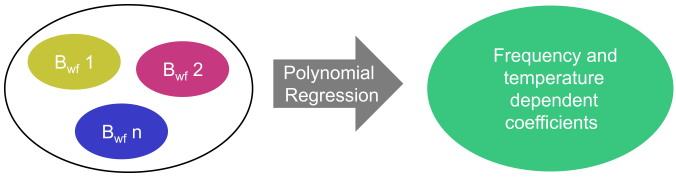

Fig. 2. The image portrays a process flow where multiple magnetic flux waveforms (B_{wf}), each with 1024 data points, are processed through polynomial regression to extract frequency and temperature-dependent coefficients, effectively simplifying the complex waveform data into a more manageable form for further analysis.

IV. Data Augmentation Strategies for Improved Model Performance

A collection of 10 training materials was initially provided, followed by an additional set of 5 training materials, resulting in a total of 15 materials. Each material is assigned a unique material ID ranging from 0 to 14. The temperature is measured in degrees Celsius, while the frequency is measured in Hertz. Furthermore, the waveform of the magnetic flux is represented in Tesla and consists of 1024 data points. To minimize the impact of discontinuities, we employed linear interpolation to decrease the waveform from 1024 to a total of 256 data points. Following this, we performed Fast Fourier Transform (FFT) on the waveform sent in the respective frequency, resulting in 256 frequency points and 256 data points. These data points accurately reflect the energy levels at their corresponding

frequencies. A visual representation of this entire procedure is illustrated in figure 3.

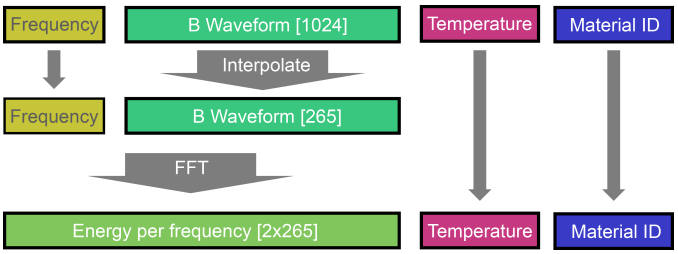

Fig. 3. The image illustrates the data preprocessing pipeline for a magnetic flux waveform analysis. Starting with the raw waveform at 1024 data points, it undergoes linear interpolation to reduce complexity, resulting in 256 points. A Fast Fourier Transform (FFT) is then applied, transforming the waveform into a frequency domain representation, which is subsequently normalized along with temperature and material ID for neural network training.

To effectively utilize this dataset in a neural network, it is necessary to normalize it. Prior to normalizing the entire dataset, each row is modified. The frequencies obtained from the FFT are divided by the respective maximum value, which is subsequently stored within the same row. Similarly, the energies derived from the FFT undergo the same normalization process. Henceforth the dataset has two more columns. Following this, the maximum energy, maximum frequency, temperature and material ID were normalized across all rows.

V. Optimization of Neural Network Architecture

In our pursuit to develop an effective neural network (NN) for modeling electronic components, we tailored the architecture by employing a fully connected neural network (FCNN). This approach has previously shown promise in the domain of electronic component modeling [2]. The initial network configuration, which featured an input layer with 2056 nodes, was systematically reduced through an iterative process, ultimately bringing the Input node count down to 516. Simultaneously, the corresponding hidden layers of the network were also methodically reduced in size, ultimately finding that a reduction in both size and depth significantly boosted performance. For the final Network size see Table I.

TABLE I
Neural Network Layer Configuration

Layer	Number of Neurons
Input Layer	516
Hidden Layer 1	512
Hidden Layer 2	128
Output Layer	1

These modifications, alongside the inherent benefits of FCNNs in capturing intricate data relationships, contributed to a refined model with reduced overfitting and enhanced generalization capabilities. The final, optimized architecture is depicted in figure 4. Structural adjustments to the network were complemented by experimentation with various activation functions, further tailoring the model's performance. After extensive testing, LeakyReLU emerged as the most effective choice, balancing non-linearity and computational efficiency, thereby contributing to the overall performance of the network.

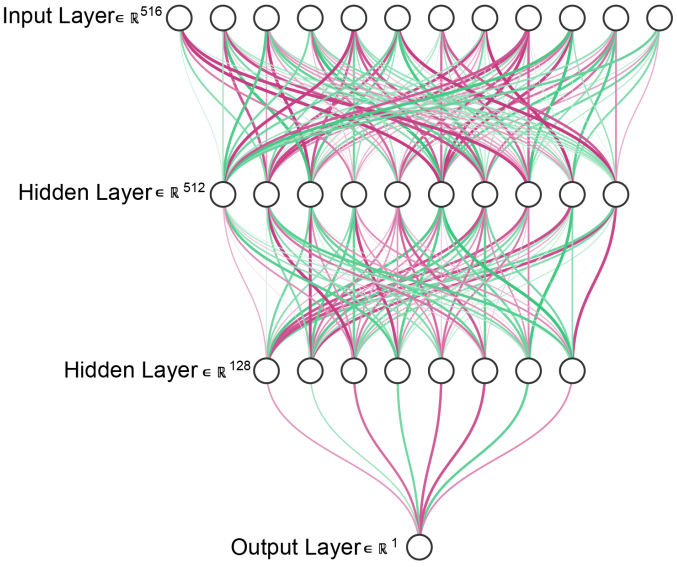

Fig. 4. The image visualizes the optimized neural network architecture with multiple input layers progressively reducing in size, from 516 nodes in the first layer to 512 in the second, and finally 128 nodes in the third layer, before converging to a single output node.

A. Refinement and Monitoring of NN Training

During the training phase, the model's progress was meticulously monitored using interactive visualization tools developed with Plotly. This approach was pivotal for real-time assessment and early detection of potential overfitting. The effectiveness of these tools is exemplified in the training and testing loss trends over time, as well as the adjustments to the learning rate, which are visually depicted in Figure 5 Our optimization strategy was anchored around the AdamW optimizer, noted for its effectiveness with sparse gradients and inclusion of weight decay for regularization. In our implementation, the AdamW optimizer was set with an initial learning rate of 1×10^{-3}. In our final model configuration, weight decay was used, adhering to the default setting of 1×10^{-2}. Additionally, a a learning rate scheduler was incorporated to optimize the training process further. The ReduceLROnPlateau scheduler from PyTorch's optimization suite was selected for its ability

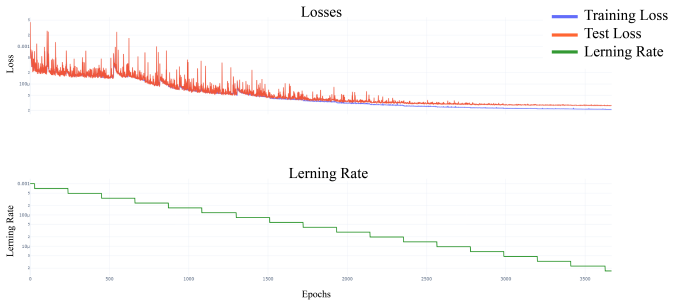

Fig. 5. The upper graph shows the training and test losses, both decreasing over time, indicative of the network's learning. The lower graph illustrates the learning rate schedule, which steps down at specific epochs, a technique often used to refine learning as the network's weights approach optimization. This combined visual representation aids in monitoring and adjusting the training process.

to adjust the learning rate based on the model's performance. Specifically, the scheduler was configured with a cooldown period of 200 epochs and a learning rate reduction factor of 0.7. These combined strategies of real-time monitoring, advanced optimization techniques, and adaptive learning rate management significantly contributed to the effective and efficient training of our neural network model.

In the subsequent phase of the training process, a refining strategy was employed by reintroducing the previously trained model to a new training cycle, this time with an adjusted learning rate. Contrary to the initial higher learning rate used in the first phase, a slightly lower learning rate was used in this secondary phase. The aim was to fine-tune the model's weights and biases, allowing for more precise adjustments that could lead to enhanced performance. This approach is based on the principle that a lower learning rate in later stages of training can help the model converge more smoothly to a potentially better local minimum. An opportunity for the model to refine its learning in a more focused and detailed manner was provided by carefully reducing the learning rate for the already trained network. The outcome of this secondary training phase was encouraging. A discernible improvement in the model's performance metrics was observed, affirming the efficacy of gradually lowering the learning rate as part of the training regimen. This method underscores the importance of adaptive learning rate strategies in the iterative process of neural network optimization.

VI. Comparative Analysis of Neural Network Prediction Accuracy.

A. Accuracy Benchmarking on Original Data

Tabel II is segmented into four key metrics: average error, 95th percentile error, 99th percentile error, and maximum error. These metrics collectively offer a holistic view of the prediction accuracy and the distribution of errors within each dataset. The Average Error metric represents the mean prediction error for each dataset. The analysis revealed a spectrum of accuracy, with some datasets showing high precision, while others indicated lower predictive accuracy. The 95th Percentile Error provides insight into the upper range of the error distribution, indicating that 95% of the predictions fall below a certain error rate. This measure varied significantly across datasets, reflecting the differences in reliability. Further, the 99th Percentile Error extends our understanding to nearly the entire range of the data, revealing the error rate below which 99% of predictions fall. This metric is crucial as it highlights the presence of outliers or specific instances where prediction accuracy drastically decreases. Lastly, the Maximum Error observed in each dataset underscores the potential extent of inaccuracies, offering a perspective on the worst-case scenario in prediction accuracy for each dataset. The neural network's average prediction error is notably lower than the measurement error inherent in the data generation process. This implies that the predictions will consistently align within the accuracy range of the original measurements, effectively 'hitting' the correct values as per the measured data's accuracy. This highlights the model's precision in capturing the true nature of the datasets, despite the limitations of the measurement accuracy.

TABLE II
Error Distribution Across Datasets

Material	Average	95th Percentile	99th Percentile	Maximum
3C90	0.50%	2.30%	5.96%	16.24%
3C94	0.29%	1.04%	2.27%	7.28%
3E6	0.23%	0.64%	1.70%	17.95%
3F4	0.75%	2.86%	10.18%	33.59%
77	0.53%	2.03%	6.11%	80.56%
78	0.52%	2.16%	5.67%	41.26%
N27	0.29%	0.97%	2.66%	10.89%
N30	0.36%	1.47%	3.10%	9.84%
N49	0.71%	3.00%	9.33%	39.32%
N87	0.49%	2.14%	4.77%	21.50%

B. Adaptation and Prediction on Recent Datasets

The training of the neural network on newer datasets, despite initial challenges due to their smaller sizes, demonstrated considerable success. A diverse range of data, using both new and previously used datasets, was integrated, effectively enhancing the model's learning process. The results, as shown in Table III, illustrate the model's capability to accurately predict.

TABLE III
Error Distribution for Materials A-E

Material	Average	95th Percentile	99th Percentile	Maximum
Material A	0.83%	3.45%	6.76%	18.50%
Material B	0.38%	1.28%	3.09%	17.17%
Material C	1.01%	4.22%	11.07%	62.35%
Material D	1.44%	6.50%	20.74%	44.50%
Material E	0.29%	1.22%	3.61%	8.93%

VII. Conclusion

Our approach in the MagNet Challenge, from data visualization and feature extraction to meticulous neural network architecture optimization and training, has yielded a good predictive model. The model's performance, demonstrated by the error metrics across various datasets, underscores its robustness and accuracy. As the initial concept was discarded, strategies were adapted to the complexities of the data. The successful handling of both original and recent datasets underscores the adaptability and potential of our methods specifically for loss prediction in magnetic cores. It is noteworthy, however, that high-frequency materials seem to exhibit a slightly higher error margin. This observation suggests the need for further refinement in our model, particularly in its application to high-frequency materials.

Acknowledgment

We extend our thanks to the entire MagNet Team and the Jurors for their pivotal role in facilitating this remarkable challenge. We are grateful to SAL for providing the opportunity to contribute to the MagNet challenge. Special thanks to Milan Pajnic, Hèlios Sanchis-Alepuz, Monika Stipsitz and Heinz Krenn for their invaluable discussions and encouragement that greatly enriched our experience.

REFERENCES

[1] M. D. Cranmer, A. Sanchez-Gonzalez, P. W. Battaglia, R. Xu, K. Cranmer, D. N. Spergel, and S. Ho, "Discovering symbolic models from deep learning with inductive biases," *CoRR*, vol. abs/2006.11287, 2020. [Online]. Available: https://arxiv.org/abs/2006.11287

[2] J. Reynvaan, M. Stipsitz, P. Skoff, T. Langbauer, and A. Connaughton, "A preliminary investigation into approximating power transistor switching behavior using a multilayer perceptron," in *2022 IEEE Design Methodologies Conference (DMC)*, 2022, pp. 1–5.