A Topological Improvement of the overall performance of SET

Motif-based Structural Optimization of Sparse MLPs Project

submitted in partial fulfillment for the degree of master of science

Xiaotian Chen
14984466

Master Information Studies
data science
Faculty of Science
University of Amsterdam

Submitted on 29.06.2024

	UvA Supervisor
Title, Name	Hongyun Liu
Affiliation	Graduate School of Informatics
Email	h.liu@uva.nl

ABSTRACT

The Deep Neural Networks(DNNs) have been proved a great success and have been applied in most areas related to deep learning. However, as the models applied with DNNs are getting more and more complex. The demand for reducing computational cost and memory requirements is becoming more and more urgent. Therefore, the sparsity is now developing as a leading approach. Robustness of sparse Multi-layer Perceptrons(MLPs) for supervised feature selection and the application of Sparse Evolutionary Training(SET) showed the possibility of reducing of computation cost while maintaining the original accuracy. What is more, it is believed that the SET algorithm can still be largely improved(with over 40% improvement in efficiency and less than 4% drop in performance) by applying a structural optimization which is called motif-based optimization. This research will investigate whether the structural optimization of Sparse Evolutionary Training applied to Multi-layer Perceptrons (SET-MLP) can bring performance improvement and how much the improvement will be.

KEYWORDS

DNN, Sparse Neural Networks, Topology, Motif-based Optimization

GITHUB REPOSITORY

https://github.com/cxtverygood/Thesis_Project

1 INTRODUCTION

The emergence of the neural network enables the possibility of artificial intelligence (AI) which refers to the capability of machines to simulate human cognitive processes, facilitated by technologies such as neural networks. As the development of the neural network, the task that the neural networks are dealing with is becoming more and more complex which typically involve handling high-dimensional data to meet specific requirements. To achieve better performance, deep neural networks (DNNs) are becoming increasingly advanced in mimicking human brain functions, resulting in a significant increase in computational cost and training time [2]. Typically, DNNs have many layers with fully-connected neurons, which contain most of the network parameters (i.e. the weighted connections), leading to a quadratic number of connections with respect to their number of neurons[5].

To address this issue, the concept of sparse connected Multi-layer Perceptrons with evolutionary training was introduced. This algorithm was used to compare with fully connected DNNs, the sparse-connected network with evolutionary training can significantly reduce the computational cost on a large scale while on the other hand, with feature extraction, the sparsely connected DNNs can maintain the same performance as the fully connected DNNs models did.

However, such development was still not satisfying enough because it still requires quite large computational power and time. What is more, as the concept of motif based DNNs, which can retrain the neurons based on a small structural group (e.g.,3neurons as a small group) might outperform the sparse connected DNNs, was introduced, it was believed that this can largely improve the performance of the deep neural network. In this paper, the structural based DNNs will be analyzed and tested to compare the performance with the benchmark model.

To provide a deeper understanding, the following sections will delve into the foundational aspects of these approaches:

1.1 Multi-layer Perceptrons

The Multi-layer Perceptrons (MLPs) is a fundamental type of artificial neural network composed of an input layer, one or more hidden layers, and an output layer . Additionally, the widespread use of MLPs with fully connected dense layers has resulted in over-parametrization, contributing to higher computational power and efficiency requirement[16].

1.2 Sparse Evolutionary Training (SET)

As mentioned before, traditional neural networks are usually densely connected, meaning that each neuron is connected to every other neuron in the previous layer, resulting in a large number of parameters in the network. Unlike normal DNNs models, SET helps introduce sparsity, reduce redundant parameters in the network, and improve computational efficiency. Through the evolutionary algorithm, SET can gradually optimize the weights so that many connections become unimportant or zero[5]. Therefore, the SET is applied to improving training efficiency by optimization of the sparse structure of the model and reduction of the redundant parameters, which eventually can end up with lowering the computation cost[16].

1.3 Feature Selection with SET

To further improve the performance of the Deep Neural Network, feature engineering is considered as a critical step in the development of machine learning models, involving the selection, extraction, and transformation of raw data into meaningful features that enhance model performance [30]. By enforcing sparsity in the neural network, SET effectively prunes less important connections, thereby implicitly selecting the most relevant features. As the evolutionary algorithm optimizes the network, connections that contribute insignificantly to the model's performance are gradually set to zero, allowing the network to focus on the most informative features. If feature selection can be applied in this process, with some important features selected and remaining features dropped, the complexity of the network would be largely decreased and the quantified the features would keep the original accuracy. Consequently, feature selection with SET results in a streamlined model that is both computationally efficient and more accurate, leading to better overall performance[16], and Neil Kichler has demonstrated the effectiveness and robustness of this algorithm in his studies, further validating its practical application and benefits[12].

1.4 Motif

Network motifs are recurring, significant patterns of connections within complex networks. They reveal fundamental structural and

functional insights in systems like gene regulation, ecological food webs, neural networks, and engineering designs. By comparing the occurrence of these motifs in real versus randomized networks, researchers can identify key patterns that help understand and optimize various natural and engineered systems.

1.5 Motif-based Structural Optimization

As mentioned before, the SET updates new random weights when the weights of the connections are negative or insignificant (close to or equal to zero) which to some extent lead to some computation burden[1]. Based on the concept of motif and SET, a structurally sparse MLPs is proposed. The motif-based structural optimization gave an idea of renewing the weights by establishing a topology which can largely improve the efficiency (shown in Figure 1)[14, 15].

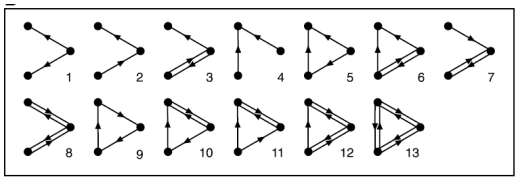

Figure 1: Motif-based Structural Sparse MLPs [15]

1.6 Research Questions

The key research question is posed in this thesis is:

To what extent can the efficiency and accuracy of sparse MLPs get improved by optimizing the structure of the Sparse MLPs and fine-tuning of the parameters of the network?

To answer the question above, there are some **sub research questions**:

1. To achieve the final goal of the structural optimization of sparse MLPs:
   - How can the structure of the topology get improved?
   - How can the model get optimized by fine-tuning the parameters?
   - To what extent can the accuracy and efficiency of the model get improved?
2. To find the optimized model with the best overall performance:
   - In which way the overall performance should be calculated and compared?
   - In what scenario should the motif-based models be applied?

1.7 Thesis Structure

To give the readers a better understanding of this paper, the structure of following sections in this thesis will be introduced in this subsection. Section 2 will give a comprehensive overview of neural network which includes the emergence of the neural network, covering seminal concepts like network pruning and recent advancements such as Sparse Evolutionary Training (SET) which sets the stage for the subsequent sections by providing context for the research presented here. Followed by section 3 in which the construction method of motif-based structural neural network will be proposed in a very detailed way. It follows the sequence of weights initialization, forward propagation, backward propagation and evolution process. Other vital hyper-parameters are introduced in this section as well. Furthermore, in the section 4, the whole experiment plan will be given, including the datasets, hyper-parameter setting and evaluation method of overall performance. Next, section 5 will show the results for each motif-sized model and each type of dataset and a very basic analysis on the efficiency and performance. The detailed analysis will be given in the section 6, the overall performance will be calculated and the best motif-based model will be given. After that, the conclusion will be given in section 7. Reflection and future work on this thesis topic will be discussed in the section 8.

2 RELATED WORK

The sparse MLPs models have been proved having great potential in reducing computation cost (hardware requirement, computation time requirement, etc) while improving the accuracy with feature extraction and sparse training. This research will take the work of Mocanu et al. as a benchmark model to compare with[16]. In this section, the history of the development of sparse neural network will be reviewed. Subsequently, the key idea and algorithm of SET will be discussed. Lastly, the basic idea of structural optimization of Sparse MLPs will be introduced.

Y. LeCun et al.[13] introduced the concept of network pruning in the paper "Optimal Brain Damage" in 1990. This approach calculated the contribution of each connection to the general network error and selectively removed some of the nodes which are considered less important. By using second-order derivatives, this method effectively reduced the model's complexity while maintaining the previous performance. Therefore, this seminal work provided and was regarded as a theoretical basis for later pruning techniques. Based on the seminal paper, B. Hassibi et al. proposed the "Optimal Brain Surgeon" method in 1993, in which second-order derivatives was also used but offered a more precise method for pruning the weights by considering the Hessian matrix's structure [10]. And it was proved that the Optimal Brain Surgeon method achieved better performance by identifying and removing redundant connections more accurately. This refinement was regarded as an important step forward in the efficiency of network pruning. In 2016, Han et al. introduced the concept of "Deep Compression", a concept which combines pruning, quantization, and Huffman encoding[9]. This three-step method significantly reduces the storage requirements and computational cost of neural networks while maintaining the accuracy. Except from normal sparsely connected network training steps like first pruning the network, then retraining the weights, the Deep Compression adds Huffman Encoding as the final step. This work demonstrated the practical benefits of combining multiple techniques. In 2018, Mocanu et al. introduced the idea of Sparse Evolutionary Training in their paper "Scalable Training of Artificial Neural Networks with Adaptive Sparse Connectivity Inspired by Network Science" [16].In this research, Erdős-Rényi graph initialization is used to create the initial sparse network and initialized the weights. By selectively adding, removing connections based on the deviation and performance, SET aims to maintain a high proportion of zero-valued connections while optimizing the model's

accuracy. The process of the SET algorithm is shown as Figure 2. As mentioned above, this model will be taken as the benchmark mark to compare with the one in this research.

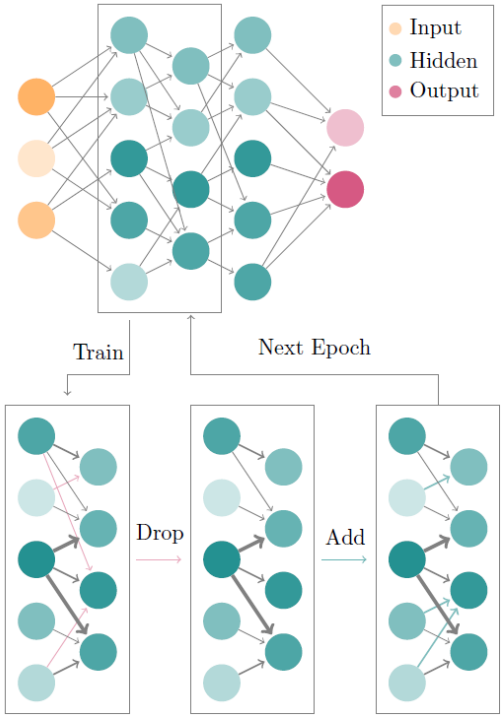

Figure 2: Process of Training, Pruning and Retraining in SET [12]

Based on the previous research work, the concept of "Dynamic Sparse Reparameterization" was given by J. Frankle et al which is a method continually adjusting the sparsity of network connections during training[17]. By dynamically reparameterizing the network, this approach helps maintain a balance between performance and efficiency. This method stands out for its ability to adaptively optimize the network structure dynamically, leading to more efficient training processes. Based on the research of Mocanu et al., the paper 'Robustness of sparse MLPs for supervised feature selection' by Kichler [12] in 2021 combined with feature extraction with sparse multi-layer training method which promoted the development of research. With a certain proportion of features dropped, the performance of the neural network can still maintain as densely connected network. The motif-base is concept referring to certain type of structural topology or network structure, which is usually used to describe a sub-network or a network pattern(shown in Figure 3). This diagram illustrates the general difference between the traditional SET algorithm and motif-based optimized models. The DNNs model on the left are trained and retrained by individual nodes[28]. Therefore, the weights between nodes are different from each other. However, for the motif-based model, the weights will be initialized and retrained by small groups with a specific size of nodes[6]. By applying the motif-base structure, the efficiency of the network can be improved while maintaining the accuracy.

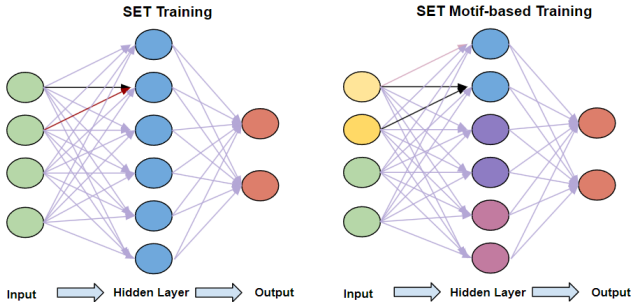

Figure 3: Comparison between SET and motif-based SET training

3 METHODOLOGY

To address the research questions regarding the motif-base structural optimization of sparsely connected neural network, this section illustrates the approaches in details. In this section, topological optimization method will be discussed which answers the first sub research question in 1.6. Subsequently, the process of training will be discussed. Lastly, the evaluation environment of this research will be given.

The core of the motif-based structural optimization of SET is that in each training process, the weights between neurons are assigned by motifs and then the weights will be delivered to each node. The following pseudo code is the general training process of the motif-based SET model:

```
1. Initialize network with motif_size:
    - For each layer:
        - Initialize sparse weights and biases with
     motifs.

2. Forward pass:
    - For each hidden layer:
        - Process each motif with weights and biases.
        - Apply activation function.

3. Backward pass:
    - For each hidden layer in reverse:
        - Calculate delta.
        - Update weights and biases for each motif.
```

Listing 1: General Motif-Based Sparse Neural Network Process

This pseudo code shows the process of network initialization, forward propagation and backward propagation in motif-based DNNs model. As mentioned before, almost all the steps are done by a certain size of motif whose size can be customized, the detailed version of network construction and training process will be given in the following subsections

3.1 Network Construction

The general idea of using structural optimization based on motif is to group nodes with a certain size and train them together. Unlike simply decreasing the number of neurons, each node of the motif-base optimized network will still get involved in training and retraining. The difference lies in the process of assigning new

weights to the nodes, this assignment is done according to a specific topology, which can improve the network's efficiency [11, 14].

3.1.1 Parameter Initialization. Before initializing the weights of neural networks, some parameters need to be defined. The input size X refers to the size of the features of the training data set that is selected as input. For instance, for the Fashion MNIST dataset, the number of pixels in the graph is 784 (28X28); therefore, the input size is 784. The motif size m refers to the size of a topology or sub-network. Hidden size refers to the number of neurons for each hidden layer [24]. Epsilon ϵ is a parameter controlling the sparsity level of the network. The activation function σ is used to introduce non-linearity into the model. The loss function L measures the difference between the predicted outputs and the actual outputs[23].

3.1.2 Weights and Bias Initialization. The random uniform distribution is utilized for generating the initial weights. The weights are assigned to each motif instead each node which will be more efficient. The He function is used here to set limitations for the weights [4]. Erdős-Rényi is used for setting sparse weights masks which can remove or keep the initial weights. And for each layer, it creates a random bias vector b_i. The motif size also needs to be chosen such that the input size is divisible by the motif size. [17]The pseudo code for network construction is shown as follows:

```
class MotifBasedSparseNN:
    Initialize(input_size, motif_size, hidden_sizes,
     output_size, init_network, epsilon, activation_fn,
     loss_fn):
        Set motif_size, epsilon, activation_fn, loss_fn
        Set create_network based on init_network (uniform
     /normal)
        Ensure input_size is divisible by motif_size
        Initialize weights (W) and biases (b)
        prev_size = input_size
        For each hidden_size in hidden_sizes:
            Ensure hidden_size is divisible by motif_size
            Create and append weight matrix to W using
     create_network
            Create and append bias to b
            Update prev_size to hidden_size
        Create and append final weight matrix and bias
     for output_size
```

Listing 2: Network initialization

Comparing to the normal SET model, instead of initializing the weights and bias individually, the weights and bias initialization in this project are done by motifs, for instance 3 nodes as motif. The weights and bias are first assigned to each motif then each motif will assign the parameters to each node which are the same within each motif. Additionally the motif-size of the model can be customized. In this way, the efficiency of the network initialization will be largely improved.

3.2 Training Process

This subsection will show the process of the motif-based model training which is the core of the whole project.

3.2.1 Forward Propagation. The forward function is similar to normal forward propagation in other models; the only difference in this project is that the nodes involved in the forward propagation process are trained in small groups[8] which will improve the total efficiency. The process is shown as following equations, $Z^{(i)}$ representing the linear combination of i layer, $A^{(i-1)}$ for the activation function for each layer(ReLu, Softmax, Sigmoid), $W^{(i)}$ for weight matrix for i layer, $b^{(i)}$ for bias[7]. For each motif in layer i:

$$\mathbf{Z}^{(i)} = \mathbf{A}^{(i-1)}\mathbf{W}^{(i)} + \mathbf{b}^{(i)} \tag{1}$$

$$\mathbf{A}^{(i)} = f_{\text{activation}}(\mathbf{Z}^{(i)}) \tag{2}$$

For the output layer, the Softmax is chosen as the activation function because it converts raw model outputs into probabilities, ensuring they sum to 1. This is crucial for multi-class classification, providing clear and interpretable probabilities for each class.

3.2.2 Backward Propagation. In the process of backward propagation, the output layer error $\delta^{(L)}$ needs to be calculated first, where $\mathbf{A}^{(L)}$ is the output of the network and $\mathbf{Y}$ is the true label:

$$\delta^{(L)} = \mathbf{A}^{(L)} - \mathbf{Y} \tag{3}$$

Then the gradient for the output layer and each hidden layer needs to be computed in a reversed order:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{W}^{(L)}} = \frac{1}{m}(\mathbf{A}^{(L-1)})^\top \delta^{(L)} \tag{4}$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{b}^{(L)}} = \frac{1}{m}\sum_{i=1}^{m} \delta^{(L)} \tag{5}$$

For each hidden layer i from $L-1$ to 1:

$$\delta^{(i)} = (\delta^{(i+1)}\mathbf{W}^{(i+1)}) \odot f'(\mathbf{Z}^{(i)}) \tag{6}$$

Motif-based computation for each hidden layer, a zero matrix δ is initialized. And for each motif j in layer i, each submatrix $W_{\text{submatrix}}$ from weight matrix $W^{(i)}$ is extracted:

$$\mathbf{W}_{\text{submatrix}} = \mathbf{W}^{(i)}[j_{\text{start}} : j_{\text{end}}, :] \tag{7}$$

The sub-delta and sub-activation for each motif are calculated:

$$\delta_{\text{sub}} = (\delta^{(i+1)}\mathbf{W}_{\text{submatrix}}) \odot f'(\mathbf{Z}_{\text{sub}}) \tag{8}$$

Update gradients for each motif and node individually:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{W}_{\text{sub}}} = \frac{1}{m}(\mathbf{A}^{(i-1)})^\top \delta_{\text{sub}} \tag{9}$$

$$\delta^{(i)}[j_{\text{start}} : j_{\text{end}}, :] = \delta_{\text{sub}} \tag{10}$$

In the end of backward propagation, the weights and bias are updated for each layer.The pseudo code of back propagation is shown as following:

```

Function backward(X, y_true, Z_list, A_list):
    m = number of samples
    Calculate initial delta from loss gradient
    Calculate dW, db for final layer, Update final
     weights and biases
    For i in reverse order of hidden layers:
        Calculate delta, Initialize dW
        For each motif in current layer:
            Calculate sub_delta and sub_A
            Update dW and biases for current motif
        Update weights and bias for current layer
```

Listing 3: MotifBasedSparseNN Training

Followed by forward propagation, the backward propagation training will also be trained by motifs in each layer, each motif will assign the retrained results to each node. Therefore, the

3.3 Process of Evolution

The core of the SET algorithm involves evolution, where weights close to zero are pruned and new weights are assigned[16]. The pseudo-code for this process is provided below:

```
    for weight_matrix in weights:
        for i in range(weight_matrix.shape[0]):
            for j in range(weight_matrix.shape[1]):
                if random_uniform() < epsilon:
                    weight_matrix[i, j] = 0.0  # Prune
    weight if random value < epsilon
                weight_matrix[i, j] += random_normal() *
    init_density

  return weights
```

Listing 4: Process of Evolution

3.4 Evaluation Environment

The evaluation is done by testing the model's runtime and accuracy. Here, the result in Mocanu et al. [16] is taken as a benchmark to be compared with and check if the model has any improvement. What's more, checking the model's performance by different types of datasets can also test the robustness of the optimized model. All the code will be executed on a desktop computer under Windows OS 10 with 32GB DRAM memory, intel 15 13th generation CPU, and a RTX4070ti with 12g VRAM.

4 EXPERIMENT

In this section, the whole experiment process of this project will be introduced by first explaining the data preparation of this experiment, and then the experiment design.

4.1 Data Preparation

In this research, Fashion MNIST(FMNIST) and Lung datasets are taken as the benchmark datasets to test the performance and efficiency of the model [27]. The FMNIST dataset(shown in Figure 4) is widely recognized as a benchmark for testing deep learning architectures. It is a dataset of Zalando's article images—consisting of a training set of 60,000 examples and a test set of 10,000 examples. Each example is a 28x28 grayscale image, associated with a label from 10 classes [12]. This study loads images using TensorFlow's FMNIST module, normalizes pixel values to $[0, 1]$, and performs one-hot encoding on categorical labels. Optionally, it standardizes data using scikit-learn's StandardScaler and stores preprocessed data in compressed .npz files for efficient access and reuse. This pipeline ensures the dataset is ready for machine learning tasks, facilitating robust model training and evaluation.

The lung dataset contain also grayscale x-ray medical scan of lung which indicate 5 lung conditions which is 5 labels. A sample of Lung data is shown in Figure 5. The dataset is initially loaded and its labels are converted into one-hot encoded format. Subsequently, the dataset is split into training and testing sets, reserving one-third for testing purposes. Feature normalization is then performed using StandardScaler library, ensuring consistent scaling across the dataset.

The properties of two datasets are shown in table 1.

Figure 4: Sample from FMNIST Dataset

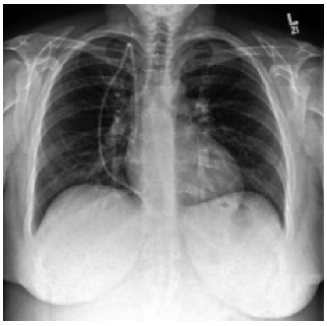

Figure 5: Sample from Lung Dataset

Table 1: Datasets Properties

Name	Features	Type	Samples	Classes
FMNIST	784	Image	70000	10
Lung	3312	Microarray	203	5

4.2 Experiment Design

As mentioned above, in this experiment Fashion MNIST and Lung are the two benchmark datasets used to test the accuracy and performance of the motif-base sparse neural network model. The benchmark model is set as the SET model [16]. To evaluate the performance and resource utilization of the model on the datasets, a comprehensive experimental setup was implemented. This setup includes data preprocessing steps such as standardization and one-hot encoding, as well as model initialization and training procedures. Functions were incorporated to retrieve CPU and GPU information to monitor hardware usage. The start and end times of the training process were recorded to calculate the total execution time[19]. Finally, the results, including test accuracy and resource details, were saved to a file for thorough analysis. This approach ensures a detailed and reproducible evaluation of the model's performance and resource efficiency.

4.2.1 Design of FMNIST Testing . The Fashion MNIST has 784 features which is the input size for the neural network, therefore, the motif size is set as 1 for benchmark model, 2 and 4 for model testing. The number of neurons for each hidden layer is 3000[24, 25]. To further test the efficiency and show the differences of each motif-size model, a simpler version of DNN model is developed with 2 hidden layers and each one with 1000 neurons.

4.2.2 Design of Lung Testing . The Lung dataset has 3312 features which is the input size for the neural network and divisible by 1, 2 and 4. Therefore, 4 motif sizes are designed for testing. Like the FMNIST dataset, a simpler version of model with 2 hidden layers(1000 neurons for each) is designed and will be tested.

4.2.3 Hyperparameter Setting. In order to ensure the rigor of the experiment, the control variable method was applied here the number of epochs is set to 300, the learning rate is set to 0.05, sparsity is set to 0.1.

4.2.4 Comprehensive Score. To better evaluate the performance of each motif-size model, a comprehensive score (S) that combines the percentage reduction in running time (R_r) and the percentage reduction in accuracy (A_r) is designed[18]. Typically, for a DNN model, accuracy is significantly more important than efficiency[20, 21]. Therefore, the accuracy reduction is weighted at 90% and the running time reduction is weighted at 10%. The comprehensive score is calculated using the following formula:

$$S = 0.1 \times R_r + 0.9 \times (1 - A_r) \tag{11}$$

Where:

$$R_r = \frac{T_{\text{base}} - T}{T_{\text{base}}} \tag{12}$$

$$A_r = \frac{A_{\text{base}} - A}{A_{\text{base}}} \tag{13}$$

In these equations:

- S is the comprehensive score.
- R_r is the percentage of running time reduction.
- A_r is the percentage of accuracy reduction.
- T_{base} is the running time observed for benchmark model.
- T is the running time for the specific motif-size model.
- A_{base} is thebenchmark model accuracy.
- A is the accuracy for the specific motif-size model.

5 RESULTS

In this section, the final results for each dataset and each motif size model will be shown first. What is more, the model with the best overall performance will be given for each dataset which will answer the research question in 1.6 by giving the exact numbers of accuracy and efficiency difference.

5.1 Experiment Results

In this subsection, the results of both FMNIST and Lung datasets will be given and analyzed.

5.1.1 Results of FMNIST. In this test, 300 epochs were run with 3 hidden layers (3000 neurons for each). The result is shown in Table 2, when the motif size is set to 1, the total running time is 25236.2 seconds with an accuracy of 0.761. When the motif size is set to 2, the total running time is 14307.5 and the accuracy is 0.723. Comparing to the benchmark model, when motif size is set to 2, the running time is 43.3% less and the accuracy is 3.7% lower. When the motif size is set to 4, the total running time is 9209.3 seconds and the accuracy is 0.692. Comparing to the benchmark model, the efficiency is improved by 73.7% with 9.7% drop in accuracy.

To further validate the efficiency of each motif size, a simpler model is established for testing the running time which has 2 hidden layers and each one of them has 1000 neurons. The average running time for each epoch is shown in Table 3. The running time of first 30 epochs is recorded and shown in Figure 6 which gives same results as the ones mentioned above.

Table 2: Result of each motif size(FMNIST)

Motif Size	Running Time(s)	Accuracy
Motif Size:1 (SET)	25236.2	0.7610
Motif Size:2	14307.5	0.7330
Motif Size:4	9209.3	0.6920

Table 3: Efficiency for each motif size(FMNIST)

Motif Size	Average Running Time (seconds)
Motif Size:1 (SET)	17.73
Motif Size:2	9.14
Motif Size:4	6.74

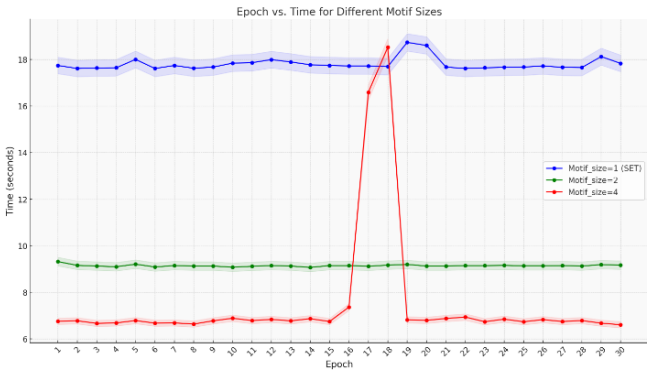

Figure 6: FMNIST Efficiency Test

Taking both efficiency and performance into consideration, the comprehensive score for each motif size is then given by:

For motif size 1:

$$S_1 = 0.1\times\left(\frac{25236.2 - 25236.2}{25236.2}\right)+0.9\times\left(1 - \frac{0.7610 - 0.7610}{0.7610}\right) = 0.9000 \tag{14}$$

For motif size 2:

$$S_2 = 0.1\times\left(\frac{25236.2 - 14307.5}{25236.2}\right)+0.9\times\left(1 - \frac{0.7610 - 0.7330}{0.7610}\right) = 0.9100 \tag{15}$$

For motif size 4:

$$S_4 = 0.1\times\left(\frac{25236.2 - 9209.3}{25236.2}\right)+0.9\times\left(1 - \frac{0.7610 - 0.6920}{0.7610}\right) = 0.8864 \tag{16}$$

According to the result, When the motif size is set to 2, the model has the best overall performance with 43.3% efficiency improvement and 3.7% accuracy drop which outperforms the benchmark model 1.1% in overall performance.

5.1.2 Result of Lung. For the Lung dataset, 300 epochs were also conducted with 3 hidden layers, each containing 3000 neurons. The results are presented in Table 4. When the motif size is set to 1, the total running time is 4953.2 seconds with an accuracy of 0.937. When the motif size is set to 2, the total running time is 3448.7 seconds, and the accuracy is 0.926. Compared to the benchmark model, with a motif size of 2, the running time is reduced by 30.4%, and the accuracy is 1.2% lower. When the motif size is set to 4, the total running time is 3417.3 seconds, and the accuracy is 0.914. This configuration improves efficiency by 31.0% compared to the benchmark model, with a 2.5% decrease in accuracy.

Table 4: Result of each motif size(Lung)

Motif Size	Running Time(s)	Accuracy
Motif Size:1 (SET)	4953.2	0.937
Motif Size:2	3448.7	0.926
Motif Size:4	3417.3	0.914

To further test the efficiency, a simpler model with 2 hidden layers, each containing 1000 neurons, was established to test the running time. The average running time per epoch is presented in Table 5. The running times for the first 30 epochs are recorded and displayed in Figure 6, which corroborate the results mentioned previously. The data in Table 5 and Figure 7 illustrate the efficiency results for each motif-size model. Both the motif-size 2 and motif-size 4 models demonstrate significant and comparable improvements in efficiency. However, the motif-size 2 model exhibits better accuracy performance and lower standard error, indicating a more stable evolution process. Consequently, the motif-size 2 model outperforms others when considering both efficiency and accuracy factors.

Table 5: Standard error for each motif size(Lung)

Motif Size	Average Running Time	Standard Error
Motif Size:1 (SET)	16.5s	0.07996
Motif Size:2	11.50s	0.09613
Motif Size:4	11.42s	0.11471

From the results, it can be referred that the motif-based model has a very significant improvement in efficiency but with some accuracy loss as well. Here, the comprehensive score equation is applied to calculate the overall performance score for each motif size model:

For motif size 1:

$$R_{r1} = \frac{4953.2 - 4953.2}{4953.2} = 0 \tag{17}$$

$$A_{r1} = \frac{0.937 - 0.937}{0.937} = 0 \tag{18}$$

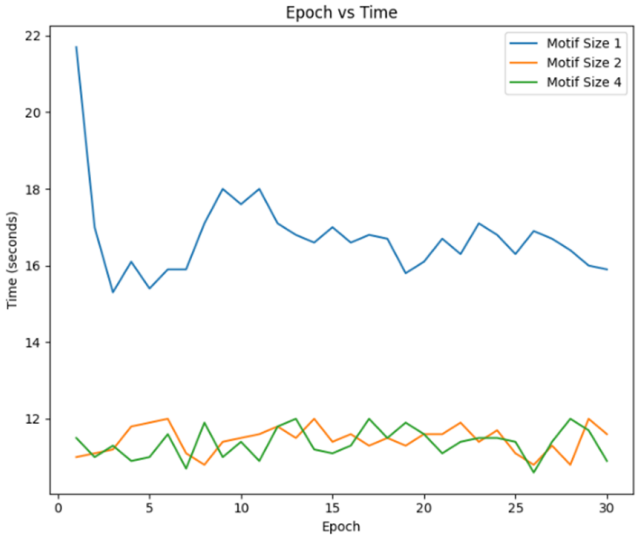

Figure 7: Efficiency of Lung dataset for each for each motif-size model

$$S_1 = 0.1 \times 0 + 0.9 \times (1 - 0) = 0.9000 \tag{19}$$

For motif size 2:

$$R_{r2} = \frac{4953.2 - 3448.7}{4953.2} = 0.3039 \tag{20}$$

$$A_{r2} = \frac{0.937 - 0.926}{0.937} = 0.0117 \tag{21}$$

$$S_2 = 0.1 \times 0.3039 + 0.9 \times (1 - 0.0117) = 0.9199 \tag{22}$$

For motif size 4:

$$R_{r4} = \frac{4953.2 - 3417.3}{4953.2} = 0.3103 \tag{23}$$

$$A_{r4} = \frac{0.937 - 0.914}{0.937} = 0.0246 \tag{24}$$

$$S_4 = 0.1 \times 0.3103 + 0.9 \times (1 - 0.0246) = 0.9089 \tag{25}$$

According to the results, the model with motif size as 2 has the best overall performance. The motif-base optimized model has a improvement of 30.4% in efficiency and a drop of 2.5% percent in accuracy and this model 2.2% outperform the benchmark model(SET) in the overall performance.

From the results given above, it can be told that for both FMNIST and Lung dataset, the model with motif size as 2 has a better overall performance in both efficiency, stability and accuracy. Further analysis on the results will be given in the next section.

6 DISCUSSION

In this paper, the concept of motif-base structural optimization is proposed. Based on SET-MLP, feature engineering benchmark model, the motif-based models were designed, tested and the one with the best performance was selected. According to the results found above, the motif-based model with feature engineering indeed has a very significant improvement in reduction of required computation cost and a noticeable drop in accuracy. However the concrete performance for each type of motif-based model depends

on datasets which means the results may differ based on different datasets and other factors. Therefore, this section will also discuss these variations. Additionally, the trade-off relationship between efficiency and accuracy is a very crucial part in this project which will be discussed in detail.

6.1 Result Analysis

In the process of training and testing the performance of the model, it was found that the precision of the Lung dataset is much higher than that of FMNIST in the very first phase (first 30 epochs). This is mainly due to the number of features (3312 features) in Lung being almost four times as many as the number in FMNIST (784 features). And the Lung dataset only has 5 output labels, the FMNIST on the other hand has 10 output labels[27].

According to the results, for both Lung and FMNIST datasets, the model with a motif size set to 2 had the best overall performance: high efficiency, high accuracy maintained and relatively stable evolution processes. It is observed that the efficiency improvement from the reference models to the models with motif size of 2 is larger than the improvement from the motif 2 models to the motif 4 models. In addition, the accuracy loss of the motif 2 model is smaller than that of the motif 4 model. However, such simple observations cannot conclude that the motif 2 model has a better overall performance than others. Therefore, a comprehensive score equation is introduced to calculate in a scientific and strict way. For any deep neural network model, the accuracy of the results is far more important than the required computation power and time. Therefore, the weight of efficiency in the comprehensive score is set as 0.1 and the weight of accuracy is set to 0.9. However, the results and analysis indicate that the overall performance of a motif-based DNNs model may differ when it comes to different types of datasets, projects and research purposes.

6.2 Trade-off Relationship

Therefore, in this section,"trade-off relationship between the efficiency-accuracy weight ratio and the comprehensive score for each motif size will be discussed. Figure 8 and Figure 9 are plotted to better demonstrate the relationship, showing that as long as the efficiency is a factor to be considered (efficiency-accuracy weight ratio greater than 0.1), the motif-based model will outperform the regular Sparse Evolutionary Training models. In the common sense, the accuracy of a deep neural network is more important than model efficiency, however in the real life, efficiency can also be a crucial factor that determines if a DNN model is good [9]. Balancing efficiency and accuracy is crucial in practical applications, as efficient models can provide real-time performance and resource optimization, which are essential in various scenarios such as mobile devices, real-time video analysis, and autonomous driving[3, 26].

What is more, it can be judged based on the results and analysis that the overall performance may differ by the motif size, datasets, structure of the neural network. This illustrates the significance of motif-based models when efficiency needs to be taken into consideration and the overall performance of different motif-size models may vary when the purposes or scenarios of the project change.

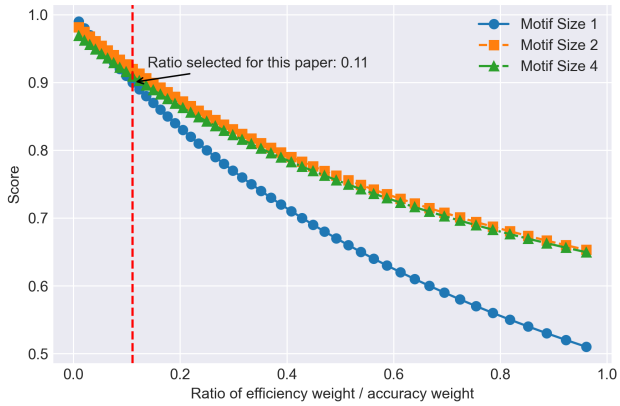

Figure 8: Relationship between efficiency accuracy weight ratio and Comprehensive Score for each motif size in Lung

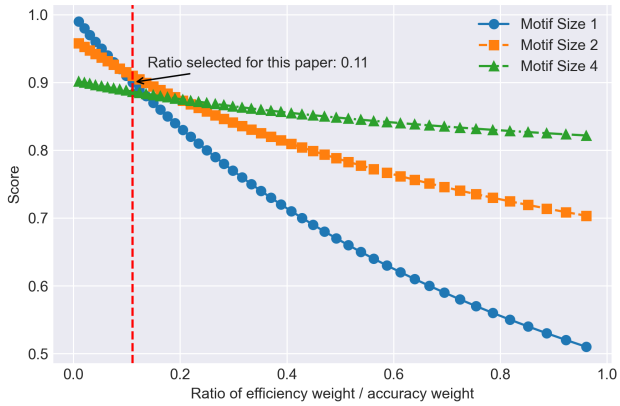

Figure 9: Relationship between efficiency accuracy weight ratio and Comprehensive Score for each motif size in FMNIST

6.3 Application Scenarios

To answer the last sub-research questions in 1.6, the application scenarios will be introduced in this subsection. As mentioned above, the motif-based models can be applied in some specific scenarios which require high efficiency and immediate feedback[22]. Therefore, based on the advantages of motif-based models, they can be applied in the following scenarios:

- **Mobile Devices**: On mobile devices, computational resources and battery life are critically constrained. Motif-based models, due to their efficiency, can provide high-quality predictions without significantly consuming device resources, making them suitable for mobile applications and embedded systems[29].
- **Autonomous Driving**: Autonomous vehicles need to process vast amounts of sensor data in a very short time to make driving decisions. Motif-based models can significantly reduce computation time while maintaining accuracy.

- **Financial Trading**: In high-frequency trading and financial market prediction, motif-based models can provide rapid market predictions.
- **Smart Home**: Smart home devices need to quickly respond to user commands and environmental changes. Motif-based models can efficiently process sensor data and users' commands.

7 CONCLUSION

This paper introduced the concept of motif-based structural optimization and demonstrated its application to the SET-MLP feature engineering benchmark model. Through extensive testing, it was shown that motif-based models significantly reduce computational costs while experiencing a slight decrease in accuracy. The analysis reveals that the Lung dataset, with more features and fewer output labels, achieved higher accuracy faster compared to the FMNIST dataset.

Among all the tested models, the motif size of 2 appeared to be the most optimal choice, offering a balance between efficiency and accuracy. This balance was quantified using a comprehensive score equation, emphasizing both accuracy and efficiency . However, the trade-off between these factors depends on the specific dataset and application requirements. The motif-based approach proved advantageous when efficiency is a key consideration, outperforming traditional Sparse Evolutionary Training models in such scenarios.

In conclusion, the models with motif size as 2 have the best overall performance(with a 43.3% improvement in efficiency and 3.7% drop in accuracy for FMNIST dataset testing, 30.4% improvement in efficiency and 1.2% decrease in accuracy for Lung dataset testing) and the motif-based structural optimization approach is highly effective for applications where computational efficiency is critical. The study highlights the need to tailor the motif size to the specific characteristics of the dataset and the intended application, ensuring an optimal balance between accuracy and efficiency.

8 REFLECTION AND FUTURE WORK

This paper investigates the structural optimization of SET using a rather simple motif-based method . From the experiment results on 2 different datasets and 6 types of models with different motif size models, conclusions are drawn in 7 that the models with motif size as 2 have the best overall performance and motif-based models usually have better overall performances than normal SET models when efficiency is an important factor. However, it still cannot be excluded that better structural optimization strategies exist. A dynamic motif size selection mechanism during training process may help improve the overall performance. The qualitative aspect of the results is, however, likely to not change and is the reason for choosing this simple mechanism. As with any experimental study, one should analyze the behaviour on many more datasets. What is more, more datasets and scenarios should be analyzed to check if such model has enough robustness and can be applied to various datasets and scenarios Last but not least, a comprehensive score equation is given in this paper to calculate the overall performance of a model. However, there is not a specific threshold to balance the efficiency and accuracy for a machine learning model at present which is a direction for the future work of this project.

9 ACKNOWLEDGEMENT

The author would like to express sincere gratitude to the advisor, Hongyun Liu, for invaluable guidance and support throughout the research.The author is deeply thankful to the parents for their unconditional love and encouragement, and to friends for their unwavering support. Lastly, a special thanks to the video games that provided much-needed relaxation and balance during this journey.

REFERENCES

[1] N. S. Altman. [n. d.]. An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression. 46, 3 ([n. d.]), 175–185. https://doi.org/10.1080/00031305.1992.10475879

[2] Arash Ardakani, Carlo Condo, and Warren J. Gross. [n. d.]. Sparsely-Connected Neural Networks: Towards Efficient VLSI Implementation of Deep Neural Networks. https://doi.org/10.48550/arXiv.1611.01427 arXiv:1611.01427 [cs]

[3] Zahra Atashgahi, Ghada Sokar, Tim van der Lee, Elena Mocanu, Decebal Constantin Mocanu, Raymond Veldhuis, and Mykola Pechenizkiy. [n. d.]. Quick and Robust Feature Selection: the Strength of Energy-efficient Sparse Training for Autoencoders. ([n. d.]). https://doi.org/10.48550/ARXIV.2012.00560

[4] Albert-Laszlo Barabasi and Reka Albert. [n. d.]. Emergence of Scaling in Random Networks. 286, 5439 ([n. d.]), 509–512. https://doi.org/10.1126/science.286.5439.509

[5] Guillaume Bellec, David Kappel, Wolfgang Maass, and Robert Legenstein. [n. d.]. Deep Rewiring: Training very sparse deep networks. https://doi.org/10.48550/arXiv.1711.05136 arXiv:1711.05136 [cs, stat]

[6] Zhang Bin, Gan Zhi-chun, Chen Wen, Hu Qiang-qiang, and Hou Jian-feng. 2019. Topology Optimization of Complex Network based on NSGA-II. In *2019 IEEE 4th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*. IEEE, Chengdu, China, 1680–1685. https://doi.org/10.1109/IAEAC47372.2019.8997597

[7] Ed Bullmore and Olaf Sporns. 2009. Complex brain networks: graph theoretical analysis of structural and functional systems. *Nature Reviews Neuroscience* 10, 3 (March 2009), 186–198. https://doi.org/10.1038/nrn2575

[8] Soravit Changpinyo, Mark Sandler, and Andrey Zhmoginov. 2017. The Power of Sparsity in Convolutional Neural Networks. https://doi.org/10.48550/arXiv.1702.06257 arXiv:1702.06257 [cs].

[9] Song Han, Huizi Mao, and William J. Dally. 2016. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding. https://doi.org/10.48550/arXiv.1510.00149 arXiv:1510.00149 [cs].

[10] B. Hassibi, D.G. Stork, and G.J. Wolff. 1993. Optimal Brain Surgeon and general network pruning. In *IEEE International Conference on Neural Networks*. 293–299 vol.1. https://doi.org/10.1109/ICNN.1993.298572

[11] Tomohiro Hayase and Ryo Karakida. [n. d.]. MLP-Mixer as a Wide and Sparse MLP. ([n. d.]). https://doi.org/10.48550/ARXIV.2306.01470

[12] Neil Kichler. [n. d.]. *Robustness of sparse MLPs for supervised feature selection*. https://essay.utwente.nl/86886/

[13] Yann LeCun, John Denker, and Sara Solla. 1989. Optimal Brain Damage. In *Advances in Neural Information Processing Systems*, Vol. 2. Morgan-Kaufmann. https://proceedings.neurips.cc/paper/1989/hash/6c9882bbac1c7093bd25041881277658-Abstract.html

[14] Shiwei Liu, Tim Van der Lee, Anil Yaman, Zahra Atashgahi, Davide Ferraro, Ghada Sokar, Mykola Pechenizkiy, and Decebal Constantin Mocanu. [n. d.]. Topological Insights into Sparse Neural Networks. https://doi.org/10.48550/arXiv.2006.14085 arXiv:2006.14085 [cs, stat]

[15] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon. [n. d.]. Network Motifs: Simple Building Blocks of Complex Networks. 298, 5594 ([n. d.]), 824–827. https://doi.org/10.1126/science.298.5594.824

[16] Decebal Constantin Mocanu, Elena Mocanu, Peter Stone, Phuong H. Nguyen, Madeleine Gibescu, and Antonio Liotta. [n. d.]. Scalable training of artificial neural networks with adaptive sparse connectivity inspired by network science. 9, 1 ([n. d.]), 2383. https://doi.org/10.1038/s41467-018-04316-3

[17] Hesham Mostafa and Xin Wang. [n. d.]. Parameter efficient training of deep convolutional neural networks by dynamic sparse reparameterization. ([n. d.]). https://openreview.net/forum?id=S1xBioR5KX

[18] J. Jinu Sophia and T. Prem Jacob. 2024. A Comprehensive Analysis of Exploring the Efficacy of Machine Learning Algorithms in Text, Image, and Speech Analysis. *Journal of Electrical Systems* 20, 2s (April 2024), 910–921. https://doi.org/10.52783/jes.1688

[19] Youcheng Sun, Xiaowei Huang, Daniel Kroening, James Sharp, Matthew Hill, and Rob Ashmore. 2019. Testing Deep Neural Networks. https://doi.org/10.48550/arXiv.1803.04792 arXiv:1803.04792 [cs].

[20] Morgan Swink, Srinivas Talluri, and Temyos Pandejpong. 2006. Faster, better, cheaper: A study of NPD project efficiency and performance tradeoffs. *Journal*

of Operations Management 24, 5 (Sept. 2006), 542–562. https://doi.org/10.1016/j.jom.2005.09.004

[21] Justin Tan and Liang Wang. 2010. Flexibility–efficiency tradeoff and performance implications among Chinese SOEs. *Journal of Business Research* 63, 4 (April 2010), 356–362. https://doi.org/10.1016/j.jbusres.2009.04.016

[22] The Theano Development Team, Rami Al-Rfou, Guillaume Alain, Amjad Almahairi, Christof Angermueller, Dzmitry Bahdanau, Nicolas Ballas, Frédéric Bastien, Justin Bayer, Anatoly Belikov, Alexander Belopolsky, Yoshua Bengio, Arnaud Bergeron, James Bergstra, Valentin Bisson, Josh Bleecher Snyder, Nicolas Bouchard, Nicolas Boulanger-Lewandowski, Xavier Bouthillier, Alexandre de Brébisson, Olivier Breuleux, Pierre-Luc Carrier, Kyunghyun Cho, Jan Chorowski, Paul Christiano, Tim Cooijmans, Marc-Alexandre Côté, Myriam Côté, Aaron Courville, Yann N. Dauphin, Olivier Delalleau, Julien Demouth, Guillaume Desjardins, Sander Dieleman, Laurent Dinh, Mélanie Ducoffe, Vincent Dumoulin, Samira Ebrahimi Kahou, Dumitru Erhan, Ziye Fan, Orhan Firat, Mathieu Germain, Xavier Glorot, Ian Goodfellow, Matt Graham, Caglar Gulcehre, Philippe Hamel, Iban Harlouchet, Jean-Philippe Heng, Balázs Hidasi, Sina Honari, Arjun Jain, Sébastien Jean, Kai Jia, Mikhail Korobov, Vivek Kulkarni, Alex Lamb, Pascal Lamblin, Eric Larsen, César Laurent, Sean Lee, Simon Lefrancois, Simon Lemieux, Nicholas Léonard, Zhouhan Lin, Jesse A. Livezey, Cory Lorenz, Jeremiah Lowin, Qianli Ma, Pierre-Antoine Manzagol, Olivier Mastropietro, Robert T. McGibbon, Roland Memisevic, Bart van Merriënboer, Vincent Michalski, Mehdi Mirza, Alberto Orlandi, Christopher Pal, Razvan Pascanu, Mohammad Pezeshki, Colin Raffel, Daniel Renshaw, Matthew Rocklin, Adriana Romero, Markus Roth, Peter Sadowski, John Salvatier, François Savard, Jan Schlüter, John Schulman, Gabriel Schwartz, Iulian Vlad Serban, Dmitriy Serdyuk, Samira Shabanian, Étienne Simon, Sigurd Spieckermann, S. Ramana Subramanyam, Jakub Sygnowski, Jérémie Tanguay, Gijs van Tulder, Joseph Turian, Sebastian Urban, Pascal Vincent, Francesco Visin, Harm de Vries, David Warde-Farley, Dustin J. Webb, Matthew Willson, Kelvin Xu, Lijun Xue, Li Yao, Saizheng Zhang, and Ying Zhang. 2016. Theano: A Python framework for fast computation of mathematical expressions. https://doi.org/10.48550/arXiv.1605.02688 arXiv:1605.02688 [cs].

[23] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention Is All You Need. https://doi.org/10.48550/arXiv.1706.03762 arXiv:1706.03762 [cs].

[24] Mowei Wang, Yong Cui, Shihan Xiao, Xin Wang, Dan Yang, Kai Chen, and Jun Zhu. 2018. Neural Network Meets DCN: Traffic-driven Topology Adaptation with Deep Learning. *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 2, 2 (June 2018), 1–25. https://doi.org/10.1145/3224421

[25] Zi Wang, Jihye Choi, Ke Wang, and Somesh Jha. 2024. Rethinking Diversity in Deep Neural Network Testing. https://doi.org/10.48550/arXiv.2305.15698 arXiv:2305.15698 [cs].

[26] Felix Wu, Kwangyoun Kim, Jing Pan, Kyu J. Han, Kilian Q. Weinberger, and Yoav Artzi. [n. d.]. Performance-Efficiency Trade-Offs in Unsupervised Pre-Training for Speech Recognition. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2022-05). 7667–7671. https://doi.org/10.1109/ICASSP43922.2022.9747432 ISSN: 2379-190X.

[27] Han Xiao, Kashif Rasul, and Roland Vollgraf. 2017. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. https://doi.org/10.48550/arXiv.1708.07747 arXiv:1708.07747 [cs, stat].

[28] Zi Yin and Yuanyuan Shen. 2018. On the Dimensionality of Word Embedding. https://doi.org/10.48550/arXiv.1812.04224 arXiv:1812.04224 [cs, stat].

[29] Shijin Zhang, Zidong Du, Lei Zhang, Huiying Lan, Shaoli Liu, Ling Li, Qi Guo, Tianshi Chen, and Yunji Chen. 2016. Cambricon-X: An accelerator for sparse neural networks. In *2016 49th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*. 1–12. https://doi.org/10.1109/MICRO.2016.7783723

[30] Shuai Zhang, Bo Yin, Weiyi Zhang, and Yu Cheng. 2022. Topology Aware Deep Learning for Wireless Network Optimization. *IEEE Transactions on Wireless Communications* 21, 11 (Nov. 2022), 9791–9805. https://doi.org/10.1109/TWC.2022.3179352