DMNN Models for Identifying Seizures Using EEG Data

Joseph Yu Nikhil Mohan Agnibho Chattarji

I. Introduction

Brain-Computer Interfaces (BCIs) are an emerging field aiming to bridge the gap between neural activity and external computational systems. BCIs rely on non-invasive methods like Electroencephalography (EEG) to detect mental or motor tasks by analyzing patterns in brain signals. Classification accuracy is paramount in such systems, and novel techniques are continuously developed to improve performance.

This report focuses on Dendrite Morphological Neural Networks (DMNNs), a biologically inspired classification algorithm. DMNNs integrate dendritic-like structures into artificial neural networks, enabling better decision boundaries for complex tasks. We explore the application of DMNNs for EEG-based motor task recognition and seizure detection, investigating whether or not they have a noticeable advantage over traditional classifiers like Support Vector Machines (SVM) and Linear Discriminant Analysis (LDA) for a seizure detection task.

II. Dendrite Morphological Neural Networks (DMNNs)

DMNNs emulate biological neural circuits by incorporating dendritic computation. Unlike traditional deep neural networks (DNNs), DMNNs use local dendritic operations, such as the *max* function, to process information hierarchically. This structure reduces computational overhead and increases robustness to noise. One limitation is that *max* is non-differentiable so backpropagation is not directly applicable [2]. However, with the introduction of the smooth maximum function and spherical dendrites in DMNNs, the network can be trained using gradient descent.

A. Architecture

We will be primarily focusing on DMNNs with smooth max and spherical dendrites. Traditionally DMNNs use excitatory and inhibitory weights and the *max* function to create straight decision boundaries for classification [2]. However, this approach doesn't allow us to update the network using gradient descent. Instead, we implemented the DMNN with spherical dendrites which have circular decision boundaries and use the smooth maximum function which is differentiable [1]. The DMNN architecture consists of four main components:

- Input neurons connected to multiple dendrites.
- Spherical dendrites that compute the distance between the input and the dendrite center.
- Class nodes that aggregate dendritic activations using smoothmax.
- Fully connected output layer for classification.

B. Training Process

Before training the DMNN with spherical dendrites using gradient descent we need to initialize the dendrite centers and radii. We used a K-Nearest Neighbors (KNN) algorithm to cluster the input data and then calculated the center and radius of each cluster. The error is calculated as the number of samples from the other class that are within the radius of a KNN cluster divided by the total number of samples. We iteratively check different neighboring sizes until the error is below a certain threshold. The cluster centers and radii are then used to initialize the centroids and radii of the spherical dendrites. The DMNN is then trained using stochastic gradient descent with the smooth maximum function and cross-entropy loss. During training we only update the centroid of the spherical dendrites and the weights of the output layer [1].

III. Methodology

A. Dataset

The EEG dataset used consists of recordings from 22 subjects, with a binary classification task of given 1 second of EEG recordings to predict whether or not a seizure follows in the next 30 seconds. Signals are sampled at 256 Hz across 23 channels, capturing key motor-related frequency bands. The dataset contains 8282 training samples which is imbalanced with 6487 negative samples and 1795 positive samples. The dataset also has two test sets one is balanced with 328 negative samples

328 positive samples. The other imbalanced test set has 1134 negative samples and 328 positive samples [3].

B. Data Pre-processing

Given the original dataset was EEG recordings our input dataset shape was three dimensional with the first dimension being the number of samples, the second dimension is the number of channels and then the third dimension is the EEG signals of 1 second which is 256 signals. The preprocessing steps we following from the original DMNN paper for EEG motor task classification. We first applied a Butter-worth band-pass filter with a low-cut of 0.5 and a high-cut of 60. Next we subtracted the common average reference or the average of the signals across all channels from each channel. Then we found 39 spectral power values one for each frequency between 2Hz and 40Hz [2]. Finally, we squeezed across the last time dimension to go from $8282x23x39$ to $8282x897$. We applied these steps to all test set EEG recordings as well. In addition, we also explored 99% explained variance PCA components of the input data to reduce from 897 dimensions to 75. So we ended up with two datasets one containing preprocessed EEG recordings with 897 dimensions and one dataset with 75 dimensions which is the PCA of the input data.

C. Experimental Setup

In order to evaluate the effectiveness of DMNN we focused on 4 main metrics the accuracy, precision, recall, and true negative rate. Precision tells us out of the number of times that we are predicting there is a seizure how many times is there actually a seizure while the recall is mainly focused on out of all the seizures how many are we actually predicting. Both of those metrics are very important especially in a medical setting where this model could be applied. For these 4 metrics we wanted to compare the DMNN performance to traditional classifiers employed in BMI tasks namely SVMR and LDA. SVM with the radial basis kernel function allows SVM to learn non-linear decision boundaries while LDA will only be able to learn linear ones. At a high level the experiment steps are:

- Fit SVMR model and gather metrics for accuracy, precision, recall, and TNR
- Fit LDA model and gather metrics for accuracy, precision, recall, and TNR
- Train DMNN using stochastic gradient descent and then measure accuracy, precision, recall, and TNR.
- Compare metrics to see if DMNN would be a better fit for seizure detection.

D. SVMR

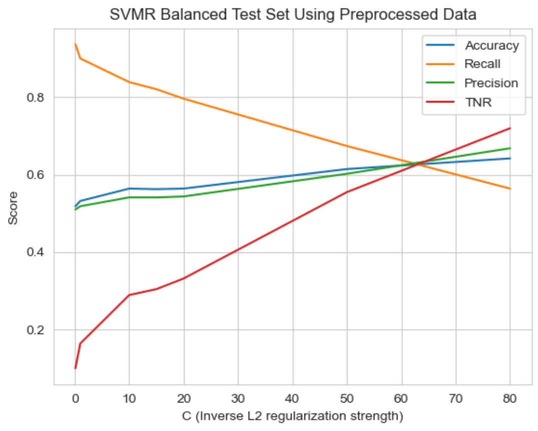
We fit a SVMR model with a radial basis kernel function and the class weight parameter set to balanced. For the different data preprocessing sets, preprocessed and PCA of preprocessed, we wanted to see how C, the inverse of the regularization strength parameter, would affect the model. We used a grid search to find the best C value for each data set. We plotted for the balanced and imbalanced test sets the different metric scores while varying C.

We can see from Fig. 1a using only preprocessing on the EEG signals on the balanced test set that there is a clear tradeoff between recall and TNR. This makes sense as the SVMR model adjusts the boundary to try to capture more of the negative samples it will also capture some positive samples and misclassify them as negative. The best C value is around 61 which is where the recall, TNR, accuracy, and precision intersect at approximately 61%. Depending on the application we would have to decide if we want to have a higher recall or TNR.

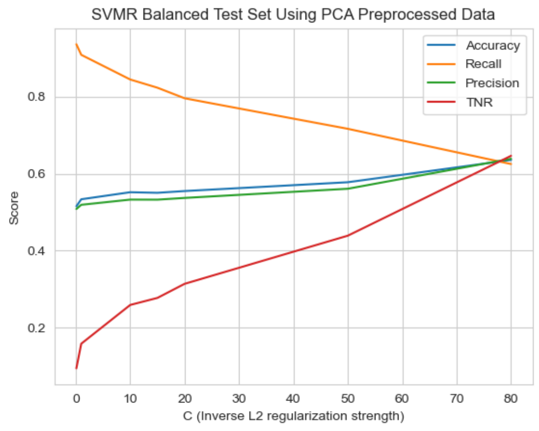
In Fig. 1c we plotted the SVMR model using the imbalanced test set and the preprocessed data. We can see that the main difference is that the precision is much lower than the balanced test set. This is because the model is still predicting that there is close to a balanced number of seizures and non-seizures which ends up lowering the precision since now there are more predictions that are false positives. If we fit the SVMR model without the class weight parameter set to balanced we would see that the model would predict that there are more non-seizures than seizures which would increase the precision but lower the recall. In the end we would have to decide if we want to have a higher recall or precision which for most applications would be recall.

In Fig. 1b we plotted the SVMR model using the balanced test set and the PCA preprocessed data. We see similar scores for accuracy, precision, recall, and TNR as using just the preprocessed data. The best C value however is slightly higher at around 78 which is where the recall, TNR, accuracy, and precision intersect at approximately 61%. This shows that using PCA to reduce the dimensions of the input data does not significantly affect the SVMR model's performance but does reduce the computation and storage costs.

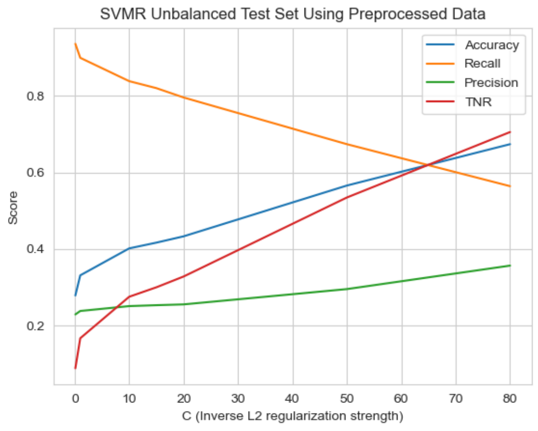
Fig. 1d shows the SVMR model performance using the imbalanced test set and the PCA preprocessed data. We see a similarly lower precision compared to the balanced test set.



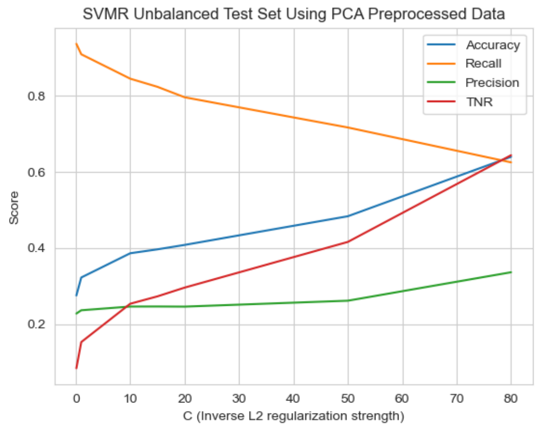
(a) Balanced Test Set Using Preprocessed Data



(b) Balanced Test Set Using PCA Preprocessed Data



(c) Unbalanced Test Set Using Preprocessed Data



(d) Unbalanced Test Set Using PCA Preprocessed Data

Fig. 1: SVMR Regularization Comparison on Test Sets

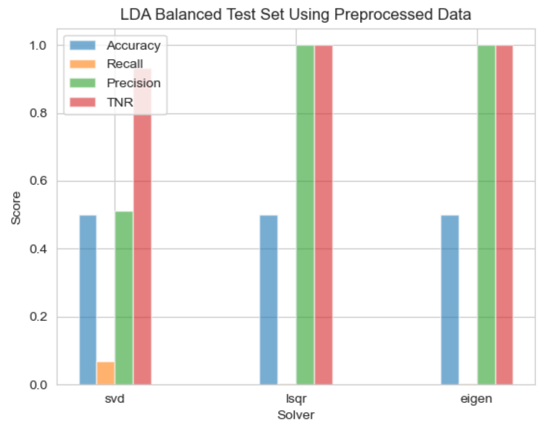
E. LDA

We also evaluated the performance of LDA using the same preprocessed, and PCA of preprocessed datasets with the balanced and imbalanced test sets. Instead of comparing regularization strength we compared different solvers and how well they performed on the 4 metrics.

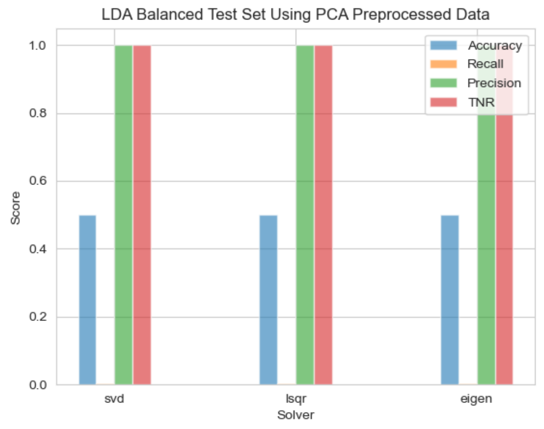
In Fig. 2a we can see that the lsqr and eigen solvers were able to achieve a high TNR and precision, however, the recall was close to 0 which means that the model was not predicting many seizures so while each prediction was precise it was not capturing a significant portion of the positive samples. The svd solver was able to achieve a slightly higher recall but at the cost of a much lower precision. Using the balanced test set and the PCA preprocessed data demonstrated similar results with the lsqr and eigen solvers achieving a high TNR and precision but a low recall, Fig. 2b. Similarly the svd solver was able to achieve a slightly higher recall but at the cost of a lower precision. Next we explored LDA performance on the imbalanced test set using different solvers. In Fig. 2c we investigated the performance of the different solvers on the imbalanced test set using the preprocessed data. We noticed a significant drop in the precision compared to the balanced test set with next to no recall. The PCA preprocessed dataset showed similar test set results except with higher precision in Fig. 2d.

F. DMNN

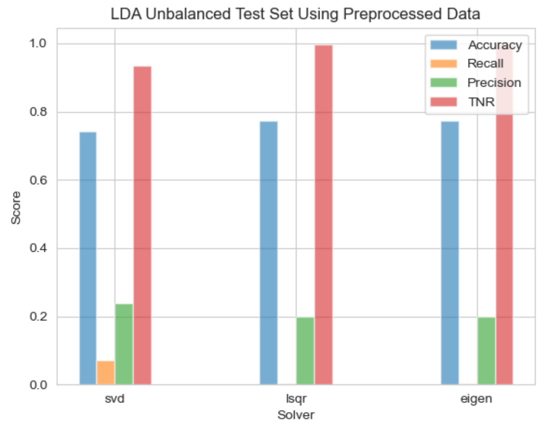
Following the DMNN paper [1] on training using stochastic gradient descent we implemented our own version in Python of the model and training process. First we built a DMNN Spherical Dendrite module which consisted of a learnable centroid and a non-updatable radius parameter. The forward function for the module calculates the euclidean distance between the input and



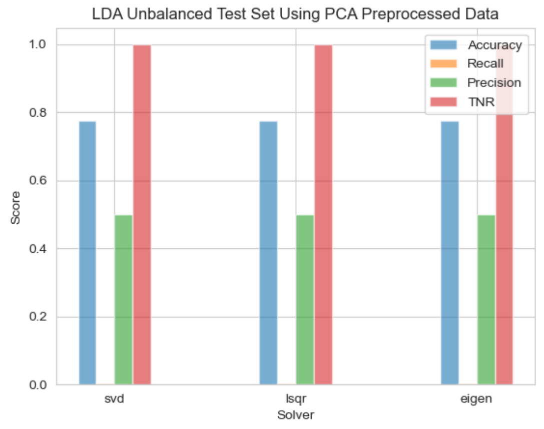
(a) Balanced Test Set Using Preprocessed Data



(b) Balanced Test Set Using PCA Preprocessed Data



(c) Unbalanced Test Set Using Preprocessed Data



(d) Unbalanced Test Set Using PCA Preprocessed Data

Fig. 2: LDA Solver Comparison on Test Sets

the centroid and then subtracts the distance from the radius to get an activation value. We also implemented a smooth maximum module which is used as the activation output for each class node. The definition for smooth max follows the formula $\sum_{i=1}^{n_k} \frac{exp(\beta \cdot s_{i,k})}{\sum_{j=1}^{n_k} exp(\beta \cdot s_{j,k})} \cdot s_{i,k}$ where $s_k = [s_{1,k}, \ldots, s_{n_k,k}]$ is a vector of the dendrite activations for class k. We end up with a $nx1$ vector for each class node where n is the number of input samples. Beta is tunable hyperparameter and as it approaches $\inf$ the smooth maximum approaches the maximum function. We developed the DMNN module which consists of spherical dendrites and output nodes. The output nodes field is a linear layer mapping from the class node activations to a vector used in the cross-entropy loss. The forward function for the DMNN module calculates the dendrite activations for each class node and then calculates the smooth maximum of all dendrite activations for a class. The activations for both classes are fed into the output layer and we get an $nx2$ vector where n is the number of input samples. This vector is used in the cross-entropy loss function. We also added our own implementation for dendrite initialization. We defined functions to calculate the center and radius for each KNN cluster, current error value, and iteratively update the neighbors size [1].

IV. Evaluation

Using the results of our investigation into the SVMR and LDA models we concluded that PCA of the preprocessed data generally shows similar and occasionally even better performance compared to just the preprocessed data but with reduced computation and storage costs. As a result, we only used the PCA preprocessed data to train the DMNN model but still evaluated its performance on the balanced and imbalanced test sets. There are a few hyperparameters that we tuned for the DMNN specifically the beta value for the smooth maximum and the error threshold for KNN initialization. We found the best beta value to be around 1 and the best error threshold to be around 0.4. We plotted the accuracy, precision, recall, and TNR for the DMNN model on the balanced and imbalanced test sets using the PCA preprocessed data.

In Fig. 3a we can see that the model for most of training has a precision, recall, accuracy, and TNR hovering between 0.5 and 0.6 and then the models test recall drops in later epochs potentially showing overfitting to the training distribution. We can also see that on the balanced test set there are sharp changes in all of metrics except accuracy and precision. This is most likely because of the distribution difference between the balanced test set and the imbalanced training set. The DMNN model is learning to move the initialized centroids around to best minimize the loss on training but these shifts could drastically change the classifications on the test set. If we compare this with Fig. 1b we can see that using large C values SVMR outperforms DMNN for the balanced case since the SVMR model has a score between .6 to .7 for all metrics.

When comparing to LDA, DMNN significantly outperforms all of the LDA models. Regardless of the solver for LDA we can see the model still has negligible recall, Fig. 2, which is an important metric for seizure detection. In Fig. 3b we visualized the DMNN model performance on the imbalanced test set. There are many similarities in performance between the DMNN model and the SVMR model on the imbalanced test set in Fig. 1d we noticed that the precision for both are lower on the imbalanced test set hovering between .2 and .4 compared to the precision on the balanced test set. However, for the imbalanced test set we would still prefer the SVMR model over DMNN with the main justification being that by tuning the regularization strength parameter C we can choose between achieving a higher recall or higher TNR which isn't an option with the DMNN model. The DMNN model consistently showed an increasing recall but at the cost of accuracy and TNR at convergence.

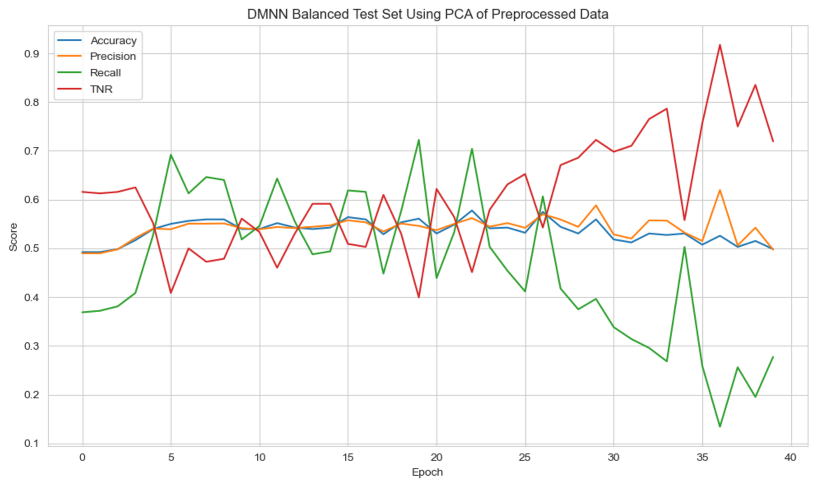
We also wanted to note that the DMNN training process is significantly longer than the SVMR and LDA models. The DMNN model took around 30 minutes to train for 40 epochs while the SVMR and LDA models could be fit in under 1 second essentially negligible amounts of time. The main reason for the DMNN's long training time is the iterative KNN initialization process and during each forward pass needing to calculate spherical dendrite activations for each dendrite which at most can be equal to the number of input samples.

V. Contributions

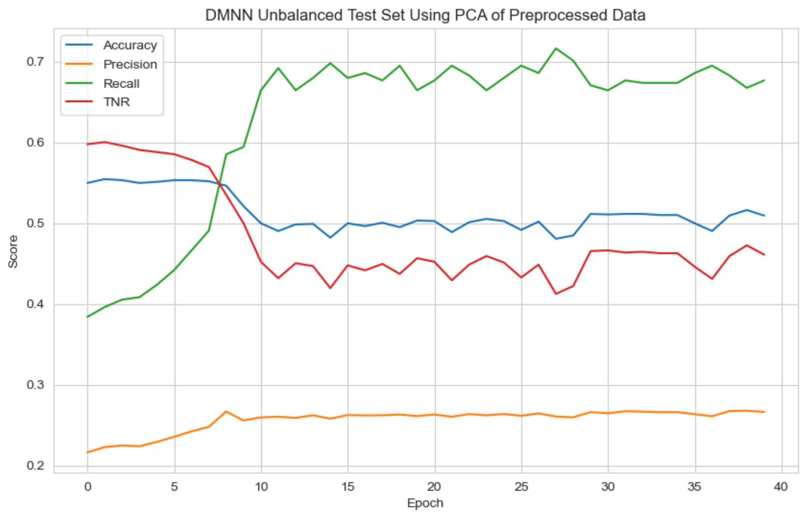
Our main contribution is the evaluation of DMNNs for EEG-based seizure detection. First, we compared DMNNs with traditional BMI classifiers like SVMR and LDA. Our results demonstrate that while DMNNs are able to learn a similar non-linear decision boundary as SVMR the SVMR model shows slightly better performance in terms of our 4 key metrics and the model being more lightweight, tunable, and faster to train still makes it a better choice for the seizure detection task. We also provide a comparison of the SVMR and LDA models, showcasing the limitations in EEG-based seizure detection of the LDA model. The LDA model is only able to generate a linear decision boundary and from the low recall performance we can see that a linear boundary isn't able to capture many of the true positive samples. Our results with DMNN demonstrate that the model is not a one size fits all for motor task classification. While DMNN has strong applications in distinguishing between intentional and unintentional movement our results show that there is not a clear performance benefit in the seizure detection task compared to SVMR. In addition SVMR is much faster to train and run inference with so for a lightweight application on a BMI device SVMR would be a better choice. We also contributed the first, in our knowlege, python implementation of a DMNN model that leverages spherical dendrites and is trainable through gradient descent. Hopefully this can enable further insights into ways that DMNN's can be used for classification tasks.

VI. Conclusion

We found that DMNNs are not the best choice for EEG-based seizure detection. While DMNNs are able to learn non-linear decision boundaries, results show that they are not able to outperform traditional classifiers like SVMR in terms of accuracy, recall, precision, and TNR. We also discovered that depending on the application by tuning C the regularization parameter for SVMR we are able to move along the recall curve opting for higher recall in exchange for lower TNR an option that we did not notice for DMNN models. DMNN models however are able to significantly out perform LDA mainly because the classification of seizures is more likely a non-linear task. The DMNN model is also significantly slower to train and run inference with compared to SVMR and LDA models. Some limitations of our study include the small dataset size and potentially suboptimal data preprocessing. In addition, the DMNN model implementation could be further optimized to reduce training time potentially



(a) Balanced Test Set Using PCA Preprocessed Data



(b) Imbalanced Test Set Using PCA Preprocessed Data

Fig. 3: DMNN Performance on Test Sets

by parallelizing computing the activations for each dendrite. Future work would involve exploring different techniques for preprocessing the EEG data and continuing to optimize the DMNN model to reduce training time.

VII. Author Breakdown

- *Joseph Yu*: Data preprocessing, SVMR and LDA model training, DMNN model implementation, DMNN Model training and evaluation, Report writing, Presentation slides.
- *Nikhil Mohan*: DMNN model implementation, DMNN Model training and evaluation.
- *Agnibho Chattarji*: Report template.

References

[1] W. Gómez-Flores, H. Sossa, "Learning smooth dendrite morphological neurons by stochastic gradient descent for pattern classification," Neural Networks, vol. 168, pp. 665–676, 2023.
[2] J. M. Antelis, B. Gudiño-Mendoza, L. E. Falcón, G. Sanchez-Ante, H. Sossa, "Dendrite morphological neural networks for motor task recognition from electroencephalographic signals," Biomedical Signal Processing and Control, vol. 44, pp. 12–24, 2018.
[3] Badea A., Adam, A. CHB-MIT-Subdataset, EEG Seizure Analysis Dataset, 2021.