Deep Embedded Clustering for Unsupervised Motor Imagery Feature Learning

Tekin Günaşar
University of California San Diego
San Diego, CA, USA
tgunasar@ucsd.edu

ABSTRACT

This paper presents a study on applying the Deep Embedded Clustering (DEC) algorithm for unsupervised feature learning of motor imagery data. We pinpoint the challenges of collecting labeled EEG data and developing complicated hand-crafted feature extraction that functions for only one BCI paradigm, and how the DEC algorithm can adress these problems. The DEC algorithm is used to learn robust features from motor imagery EEG data, then subsequent clustering of the learned features is performed. By leveraging the clustering results, the proposed method aims to overcome labeling constraints and classify new samples in an unsupervised manner based on cluster membership. The effectiveness of the approach is evaluated using a large motor imagery data set and various performance metrics, including accuracy of cluster assignments, based on minimum distance to a centroid, and normalized mutual information. The results demonstrate promising outcomes in unsupervised motor imagery feature learning and highlight the potential of unsupervised feature learning for BCI applications. Future work could include exploring self-supervised methods and incorporating context-based approaches for more general-purpose EEG feature learning.

CCS CONCEPTS

• **Computing Methodologies** → **Machine learning**.

KEYWORDS

Brain-Computer-Interface,Machine Learning,EEG,Feature Learning,Clustering

ACM Reference Format:
Tekin Günaşar. 2023. Deep Embedded Clustering for Unsupervised Motor Imagery Feature Learning. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Although Brain Computer Interfaces(BCIs) are still rather experimental, they hold promise across various domains including, but not limited to, assistive technologies, neurorehabilitation, and epilepsy detection[19,18,5]. Many different paradigms to implement BCIs exist, such as the steady State Visual Evoked Potential(SSVEP), however, motor-imagery based approaches are especially popular. The fundamental concept behind motor-imagery BCIs is based on individuals mentally simulating specific movements, such as clenching their left fist, while their resulting EEG signals are recorded and classified in real time by machine learning algorithms to decode the intended movement. Thus, two challenges are to train classifiers that can reliably decode motor-imagery signals and finding robust features to train these classifiers with.

Most motor-imagery classifiers are supervised and use complex feature extraction methodologies, such as the Common Spatial Pattern algorithm which optimizes a spatial filter to maximize the variance between class matrices of different mental states[20]. This presents two main problems. The first is that EEG data is expensive and time-consuming to collect, and labeling constraints for supervised classifiers only amplify this. Secondly, significant effort is devoted to developing highly specialized feature extraction algorithms, which can be challenging to implement and require additional research to extend them to multi-class scenarios. This multi-class extension is particularly crucial in the context of any BCI with more than two mental states.

To address these problems, we suggest an automatic feature learning framework using the Deep Embedding Clustering algorithm (DEC), first implemented by Xie et al.[4] for feature learning and subsequent clustering of the MNIST image data set. We address the issue of hyper-specialized feature extraction by aiming for the beginnings of a more general feature learning framework that can be more robust to variance of EEG signals between subjects. As we mentioned, the DEC algorithm also performs subsequent clustering of learned features, thus, we also aim to alleviate labeling constraints by classifying new samples of motor imagery data by cluster membership. Although not with the DEC algorithm, this main idea has been applied to image classification in [3].

The idea of feature learning for motor imagery data, and more general EEG data, is not new. Tabar et al.[15] use the Short-Time Fourier Transform to convert a multi-channel EEG signal into an image format, and use convolutional layers for automatic feature learning before classification. Stober et al.[14] use a convolutional auto-encoder approach to extract relevant features while subjects listened to different types of music. Wen et al. [10] also use convolutional auto-encoders for an unsupervised feature learning approach of epilepsy related EEGs before learned features are passed to a classifier.

In light of these challenges and previous work, our goal is to expand upon feature learning methods for motor imagery data specifically by using the DEC algorithm. Ultimately, we provide an

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

effective introduction for unsupervised feature learning and begin a framework for gradually removing the need for data annotation on EEG data sets.

We structure this paper as follows: first, we present our data-set and our associated processing pipeline, also addressing concerns related to data volume requirements for the effective utilization of neural networks. Next, we delve into the DEC algorithm, which extracts optimal data features for clustering. We then discuss our experimental results, encompassing unsupervised classification and the evaluation of the performance of learned features in conjunction with supervised algorithms. Finally, we conclude with a discussion of possible future work involving self-supervised methods and possibilities of context based approaches for more general purpose EEG feature learning.

2 METHODOLOGY

This paper implements an unsupervised feature learning framework using DEC and performs subsequent clustering and classification through cluster membership. We first review our data set, data processing pipeline, a quality check of our augmented data, and an overview of the DEC algorithm with our modifications to adjust it for EEG data.

2.1 Data set

Given that we are using a Deep Learning based approach to learn the most salient motor imagery features, a sizable data set is important. We use the largest public motor imagery data set available produced by Kaya et al.[6] that contains 75 different recording sessions with four different BCI motor-imagery paradigms. Each recording session is approximately 55 minutes long, split into three interaction segments. Each interaction segment consists of 300 motor-imagery trials. During each recording session participants were comfortably seated while observing a graphical user interface that was used to indicate which mental imagery was to be implemented for the upcoming trial. Each trial consisted of the execution of the mental imagery once, for one second, with a 1.5 to 2.5 seconds pause between trials. When extracting trials, we only consider the first 0.85 seconds of each trial, following the same methods of Kaya et al.[6] in their analysis. We use Min-Max Normalization on the data due to the activation functions we used for our auto-encoder described later.

We make use of the CLA paradigm to create a more feasible introduction to our methods in the BCI domain. Specifically, the CLA paradigm makes use of only motor-imagery involving the left and right hand. For each respective hand, participants executed the associated motor imagery by imagining the clenching of the fist of the respective hand once per trial. Implementation of left-hand v.s right-hand motor imagery with channels C3 and C4 is especially popular and is the approach we take.

2.2 Data Preparation

The original data was down-sampled to 200Hz by the Neurofax recording software, then a band-pass filter of 0.53-70Hz was applied. We do not apply another filter so as to truly test the capabilities of our models to discern between noise and underlying patterns between the class matrices of data. In total we make use of 9897 motor-imagery trials from the CLA paradigm. However, for the purpose of Deep Learning based approaches this is still not a lot of data and we are at risk of over-fitting on this small data set, thus we also investigate possible data augmentation techniques.

Data augmentation is common in Computer Vision applications for image classification with techniques such as geometric transformations, addition of noise, etc. However, in the case of EEG signals, much more care needs to be taken into consideration both in the type of augmentation done so as to not change the characteristics of signals from the original data distribution, and to also make sure that models are not just over fitting on augmented data. Data augmentation is not very widely used in motor-imagery classification. George et al.[8] and Lashgari et al.[9] cover commonly used EEG augmentation techniques, and among these noise addition and trial cropping with a specified window is popular. Parvan et al.[16] actually used noise addition for the OpenBCI Competition IV data set 2b and showed improvements over using no augmentation.

Within this paper, we make use of Gaussian noise addition, but do not perform any cropping to preserve the full temporal dynamics of each motor-imagery trial. Instead, we combine noise addition with time warping of trials, where trials are temporally distorted by compressing or stretching different sections of each trial. The justification for the usage of time warping is that it can create enough variability between augmented and original trials, but at the same time preserve the characteristics of original data since it is randomly applied to different sections of each trial and some of the original data is unchanged. For each trial, we first implement time warping four times, and then on each resulting trial we perform Gaussian noise addition twice, resulting in a data set eight times as large. Before any augmentation is performed, we split the original trials into a validation and training set, ensuring that no validation data is used to create new trials, we did not include an external testing set due to concerns of data set size at this point.

2.3 Validity of Augmented Data

One of the main concerns with augmented data is that the newly generated data might be too similar to the original data set and our complex neural network model will just over-fit on the original data and its similar variations. To evaluate just how similar our augmented data is to the original data we first organize our trials into right and left hand motor imagery trials, within both the original and augmented data. Then, we sampled one thousand random points from each of the organized data sets, flattened them, and calculated the Pearson Product-Moment correlation between 1000^2 possible pairings among both the classes of mental imagery. To visualize this, we use a heat map, and for a quantitative measure, we also report the average magnitude of the correlation.

However, we must also take care that the augmented data is not too different from the original data distribution either so that we do not over fit on unfaithful representations of real world signals, which will result in poor performance on new samples that are not a reflection of what our models were trained on. To get an estimate of how well our augmented data follows the original data distribution we make use of the Kolmogorov-Smirnov(KS) Test[7]. The KS test compares the cumulative frequency distribution(CDF)

for some population and the CDF for a set of observations. For our context,

$$F_N(X) = \frac{k}{N}$$

$$S_N(X) = \frac{l}{N}$$

Where $F_N(X)$ represents the CDF of a signal from the original data, and $S_N(X)$ represents the CDF of a signal from the augmented data; k and l are the number of values in the respective signal less than or equal to X, and then divided by the length of the signal, N. To compare then whether or not the original and augmented signal come from the same data distribution, we compute the test statistic,

$$d = max[F_N(X) - S_N(X)]$$

Which is the maximum value between the original signals CDF and the augmented signals CDF. If this value is too large, then it might suggest that the original and augmented signal do not come from the same distribution. The maximum is not a very robust statistic, and there is already much between subject variability between individual trials in the original data, so to remedy this, and get an interpretable test result,we average the trials into just one signal for both the augmented and original data, for both right and left hand motor imagery trials, to get rid of the between subject variability between individual trials. Additionally, the KS Test is a univariate goodness of fit test, hence, we apply the test to channels C3 and C4 separately. The test results indicate that it is reasonable to conclude that the original and augmented data do not come from different distributions.

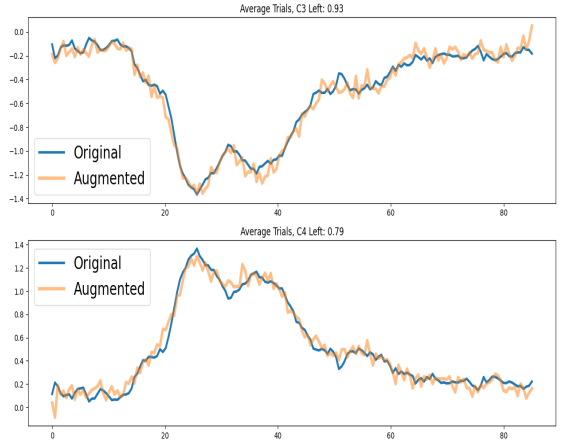

Figure 1: p-values for the KS test results on the right hand motor imagery's across averaged trials. p = 0.93 for channel C3 and p = 0.79 for channel C4. Average real and augmented trials plotted in blue and green respectively

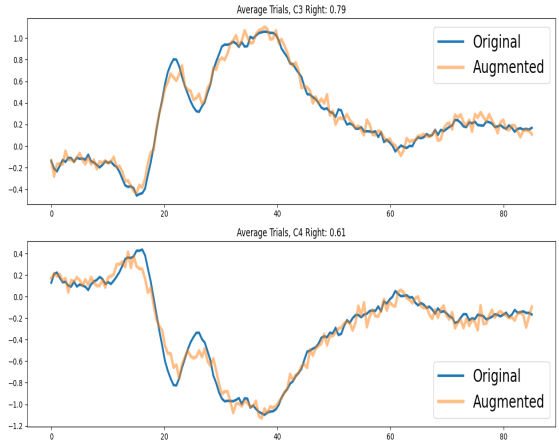

Figure 2: p-values for the KS test results on the right hand motor imagery's across averaged trials. p = 0.79 for channel C3 and p = 0.61 for channel C4. Average real and augmented trials plotted in blue and green respectively

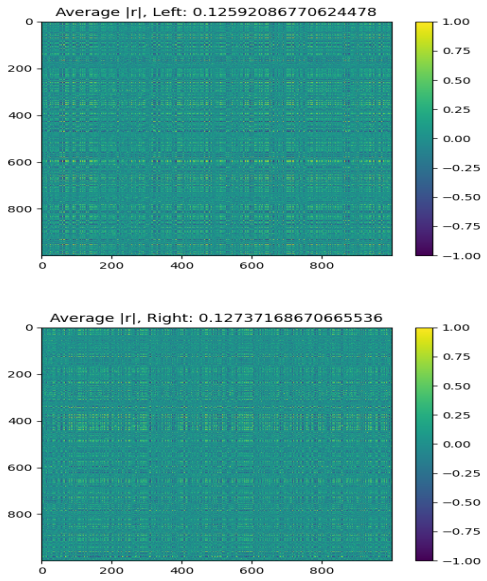

Figure 3: Correlation heat map for right and left trials

When we interpret together the relatively low average magnitude of correlation between the random sample of the original and augmented trials, and the results of the four separate KS test results, there seems to be empirical evidence that we have reached a good balance with our augmented data. It maintains the original data distribution, while not being too similar.

3 DEEP EMBEDDED CLUSTERING

Generally, the goal of clustering algorithms is to partition data into groups of similar points based on underlying patterns. Many clustering algorithms use distance based metrics to group data based on the minimum distance to some set of k cluster centers $\{\mu_j\}_{j=1}^{k}$. However, for high dimensional data these distance based approaches tend to not be very effective due to the curse of dimensionality[17].

The Deep Embedding Clustering algorithms belongs to a class of methods known as Deep Clustering algorithms that aim to learn a non-linear mapping $f_\theta : X \rightarrow Z$ that makes clustering more convenient, where X is the original data space, and Z is the reduced space. The parameters of f_θ are updated iteratively by first creating soft assignments of data points to clusters in the reduced space. A target distribution is then calculated and the loss of the model is defined as the distance between the soft assignments and the target distribution, typically expressed as the Kullback-Leibler Divergence.

To initialize the parameters of f_θ, an auto-encoder is pre-trained on the data, which is a type of neural network that consists of an encoder, which reduces the data to a lower dimensional space, and then a decoder that attempts the reconstruct the input from the reduced state. We first discuss the process of creating soft assignments and optimizing their distance to an auxiliary target distribution.

3.1 Optimizing KL-Divergence

3.1.1 ***Soft Assignments.*** We have a set of initial parameters for f_θ. For some point in the original data space, x_i, we transform it into its latent representation z_i and calculate a similarity score between z_i and all cluster centers $\mu_1, \mu_2, ...\mu_k$ using the Student's t-distribution kernel to deal with the over-crowding problem[12].

$$s_{ij} = \frac{(1 + ||z_i - \mu_j||^2/\alpha)^{\frac{-\alpha+1}{2}}}{\sum_{k \neq j}(1 + ||z_i - \mu_k||^2/\alpha)^{\frac{-\alpha+1}{2}}}$$

Where s_{ij} is the similarity score between point z_i and cluster cluster μ_j, and α is the degrees of freedom of the t-distribution, which is set to $\alpha = 1$ in the DEC paper.

3.1.2 ***Target Distribution.*** The target distribution can then be written as:

$$p_{ij} = \frac{s_{ij}^2/f_j}{\sum_{k \neq j} s_{ik}^2/f_k}$$

where $f_j = \sum_i s_{ij}$. One reason why the target distribution takes on this form is to minimize the distortion of data from the original data space to the reduced space. The objective then is to minimize the KL-Divergence between the soft assignments assigned by f_θ,S, and the auxiliary target distribution, P:

$$L = KL(S||P) = \sum_{i,j} p_{ij} log \frac{p_{ij}}{s_{ij}}$$

3.2 Parameter Initialization

The authors of DEC use a stacked auto-encoder (SAE)[21] to initialize the parameters of f_θ, which is a special type of auto-encoder where each layer feeds into another auto-encoder itself. They have an increased model capacity because of this architecture, which is an issue for our purposes given our previous discussion of lack of real data. For the purposes of this paper then, we just use a standard encoder-decoder structure.

3.2.1 ***Model Architecture.*** We make modifications to the standard DEC model architecture. To be able to encode both spatial and temporal information, we first perform a spatial convolution on signals fed into the model, and then apply temporal convolutions, with multiple kernels per layer, before the data is then projected to its latent representation. The reason for this architecture is to encode both spatial and temporal information, but also to reduce the parameter requirements of our model with the convenient parameter sharing property of the convolutional layers. The data is then reconstructed using convolutional transpose layers, first introduced by Zeiler et al.[22]. This similar architecture with just a convolutional auto-encoder is used by Guo et al[1] to perform feature learning on the MNIST image data set.

Since we used Min-Max normalization on the data, we use the ReLU activation so that the data is projected to be between 0 and 1. Due to this range of data in the reduced space, we then conveniently pair this with the tanh activation for the decoder. We use the Adam optimizer with a learning rate of 10^{-5} and optimize directly the mean square error loss between model inputs and reconstructed outputs. Both the encoder and decoder have a 30 percent drop out in their weights. The auto-encoder is trained for 15 epochs, and we experiment with different dimensions for the reduced space, as is discussed in the results section. For the second part of training where we optimize f_θ, all the training settings are kept the same, except we only train the model for six epochs.

To initialize the cluster centers, $\{\mu_j\}_{j=1}^{k}$, we run k-means clustering once the original data reduced with the encoder parameters after pre-training and use the learned cluster centers initially.

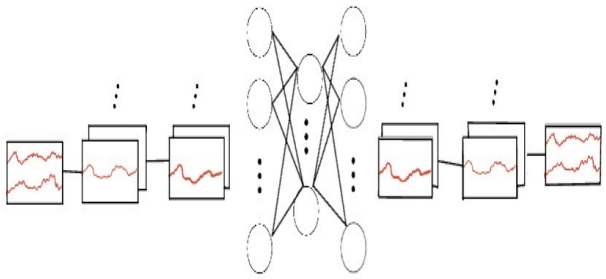

Figure 4: Architecture of our pre-trained auto-encoder. Signals first go through a spatial and temporal convolution before being flattened and projected down to their latent representations and being reconstructed with transposed convolutional layers

4 RESULTS

We use two main metrics that Xie et al.[4] used in the evaluation of the original DEC algorithm. We classify data and calculate these metrics only after transforming the data once more, after features have been learned, with t-SNE[12]. First, we consider cluster membership by assigning points labels based on distances to cluster centroids. For some embedded point z_i, its assigned class is

$$\hat{y}_i = argmin_j ||\mu_j - z_i||_2^2$$

and the accuracy(ACC) defined across all the assigned labels is:

$$Accuracy(y, \hat{y}) = \frac{1}{n} \sum_{i=1}^{n} I(y_i = \hat{y}_i)$$

Next, we also consider the quality of the clusters and how well they assign with the ground truth labels of the embedded points $z_1, z_2, ...z_n$ by using Normalized Mutual Information (NMI).

$$NMI(y, \hat{y}) = \frac{2I(y, \hat{y})}{H(y) + H(\hat{y})}$$

Where y and $\hat{y}$ are the collection of ground truth and assigned labels for all the embedded points $\{z_i\}_{i=1}^n$, $I(y, \hat{y})$ denotes the mutual information between y and $\hat{y}$, and $H(.)$ denotes entropy. For further discussion, we use Z, an integer, as the dimension of the reduced space.

We present a table of results below reporting both the accuracy by assigning predicted labels to embedded points by cluster membership, and with supervised models to compare results. The NMI metric in the below table only applies for the previously mentioned unsupervised method.

Table 1: ACC(%) And NMI

	ACC					
Z,CLF	CM	KNN	RF	LSVM	AVG	NMI
8	56.75	48.68	51.11	56.31	52.36	0.0387
16	59.65	51.06	53.53	58.14	56.09	0.0196
24	64.75	58.33	58.98	64.64	61.37	0.0743
32	51.20	50.05	50.95	49.54	50.63	0.0012
AVG	58.08	52.03	53.64	57.15	55.11	0.0334

Each entry in the above table corresponds to the accuracy of a classifier being applied onto validation data in a different dimensionality latent space, Z. All classification is done on validation data after it has been transformed into its latent representation and then t-SNE is applied. Classifier names have been abbreviated. KNN stands for K-Nearest Neighbors; RF stands for Random Forest; LSVM stands for SVM with linear kernel;and CM stands for 'Cluster Membership', which as we have discussed is our method of classifying points by comparing distances to all cluster centroids and assigning the most likely class as the one that is the shortest distance away.

Our table has an AVG row and an AVG column. The entries of the row AVG correspond to the average accuracy of the same classifiers across different values of Z, while the entries of the column AVG correspond to the average accuracy of different classifiers, but across the same value of Z.

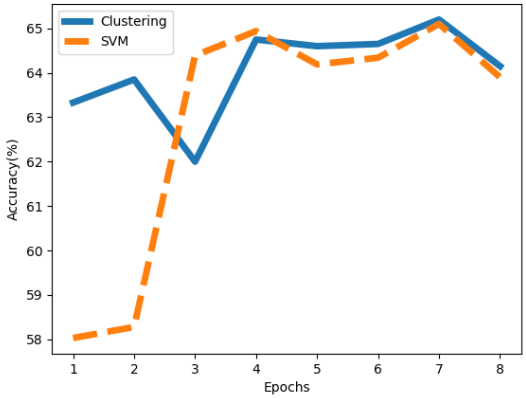

Figure 5: Accuracy of the cluster membership method of classification when the encoder is optimized for a variable number of epochs in blue with Z=24, while the accuracy for SVM shown with the dashed orange line

The highest recorded accuracy is recorded at 65.2% and is achieved by training the encoder, after auto-encoder pre-training, for 7 epochs with Z = 24. Plot achieved with perplexity = 30 and 5000 iterations on the original 24 dimensional space.

5 DISCUSSION

We achieve an optimal accuracy of 65.2% accuracy using no label information by directly clustering feature representations and using cluster membership as labels. Initially, when we reduced the original points to a dimensionality of Z = 8 and gradually increased Z, the accuracy through cluster membership would also increase, but then begin to decrease almost as if in a negative quadratic fashion. Besides our method unsupervised classification, directly applying an SVM classifier with a linear kernel on the t-SNE transformed features yielded similar results.

There seems to be a delicate balance of choosing the dimensionality of the latent representations to preserve important information, but at the same time not including noise. Z should obviously be small compared to the dimensionality of the original data so that only the most salient features are extracted, however, if Z is too small then sufficient information is not being stored in the latent representations of the data that can act as faithful representations of the original signals. We observe an optimal point for accuracy through cluster membership at Z = 24, and after that at Z = 32 it seems some of the extracted features ended up just being noise and adversely affecting cluster quality. On the topic of limiting noise as much as possible, perhaps clustering quality and accuracy could be enhanced if trials were initially filtered to be in the characteristic mu rhythm for motor imagery signals from 8-13Hz[13] to include

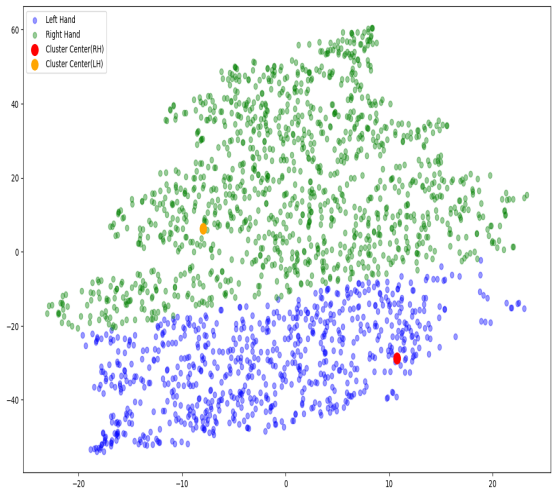

Figure 6: t-SNE visualization of of the transformed space with the highest accuracy classification by cluster membership, where Z = 24. Plot achieved with perplexity = 30, and 5000 iterations. Blue and green points representing left and right hand motor imagery's respectively, with the red and yellow dots representing the associated cluster centroids

maximally relevant information in our signals instead of the entire 0.1 - 70Hz range we use.

There was also a slight effect based on how many epochs the encoder was trained for. Upon running experiments with Z = 24, the accuracy very gradually rose as a more convenient space for clustering was being formed, but it seems it is possible for this space to be distorted too much and cause accuracy by cluster membership to fall around the 7-8 epochs of training mark.

Ultimately, even though we use data augmentation to combat the fact that we are using a complex unsupervised model; for the time being, handcrafted feature extraction seems to yield better results than many of the current unsupervised approaches to learning features in motor imagery signals. However, the goal of this work was to not develop a state of the art method, but rather to introduce discussions and a possible algorithm for how one might go about unsupervised feature extraction in EEG signals. The end goal is to remove data annotation requirements for EEG data sets so that dataset and model development can be greatly accelerated. Another good direction to embark on, might be to consider a contrastive loss function[11] to learn more robust features.

6 CONCLUSION

The key in effective feature learning of general biological signals will not occur in small tweaks of existing saturated methodologies. Rather, it will come from adapting more modern frameworks that take into account more robust considerations such as incorporating data and label context into features rather than considering just raw data instance and label pairings. Unsupervised approaches may not be the answer either, and currently there is a lack of applications of contrastive learning methodologies: methods that can generate pseudo-labels for large quantities of data that can be transferred to a later task, which was motor imagery decoding in our case. The field is very ripe for applications of contrastive learning on creating informative features for EEG signals, and should be considered for subsequent work.

Acknowledgements: I would like to sincerely thank Dr. Zhuowen Tu of the Cognitive Science and Computer Engineering department of University of California San Diego for his invaluable guidance in helping me think critically about the ideas fully developed in this paper and what to focus on next.

REFERENCES

[1] Guo, X., Liu, X., Zhu, E., & Yin, J. (2017). Deep clustering with convolutional autoencoders. In Neural Information Processing: 24th International Conference, ICONIP 2017, Guangzhou, China, November 14-18, 2017, Proceedings, Part II 24 (pp. 373-382). Springer International Publishing.

[2] Li, F., Qiao, H., & Zhang, B. (2018). Discriminatively boosted image clustering with fully convolutional auto-encoders. Pattern Recognition, 83, 161-173.

[3] Caron, M., Bojanowski, P., Joulin, A., & Douze, M. (2018). Deep clustering for unsupervised learning of visual features. In Proceedings of the European conference on computer vision (ECCV) (pp. 132-149).

[4]Xie, J., Girshick, R., & Farhadi, A. (2016, June). Unsupervised deep embedding for clustering analysis. In International conference on machine learning (pp. 478-487). PMLR.

[5] Orhan, U., Hekim, M., & Ozer, M. (2011). EEG signals classification using the K-means clustering and a multilayer perceptron neural network model. Expert Systems with Applications, 38(10), 13475-13481.

[6] Kaya, M., Binli, M. K., Ozbay, E., Yanar, H., & Mishchenko, Y. (2018). A large electroencephalographic motor imagery dataset for electroencephalographic brain computer interfaces. Scientific data, 5(1), 1-16.

[7] Massey Jr, F. J. (1951). The Kolmogorov-Smirnov test for goodness of fit. Journal of the American statistical Association, 46(253), 68-78.

[8] George, O., Smith, R., Madiraju, P., Yahyasoltani, N., & Ahamed, S. I. (2022). Data augmentation strategies for EEG-based motor imagery decoding. Heliyon, 8(8), e10240.

[9] Lashgari, E., Liang, D., Maoz, U. (2020). Data augmentation for deep-learning-based electroencephalography. Journal of Neuroscience Methods, 346, 108885.

[10] Wen, T., & Zhang, Z. (2018). Deep convolution neural network and autoencoders-based unsupervised feature learning of EEG signals. IEEE Access, 6, 25399-25410.

[11] Hadsell, R., Chopra, S., & LeCun, Y. (2006, June). Dimensionality reduction by learning an invariant mapping. In 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06) (Vol. 2, pp. 1735-1742). IEEE.

[12] Van der Maaten, L., & Hinton, G. (2008). Visualizing data using t-SNE. Journal of machine learning research, 9(11).

[13] Pfurtscheller, G., Brunner, C., Schlögl, A., & Da Silva, F. L. (2006). Mu rhythm (de) synchronization and EEG single-trial classification of different motor imagery tasks. NeuroImage, 31(1), 153-159.

[14] Stober, S., Sternin, A., Owen, A. M., & Grahn, J. A. (2015). Deep feature learning for EEG recordings. arXiv preprint arXiv:1511.04306.

[15] Tabar, Y. R., & Halici, U. (2016). A novel deep learning approach for classification of EEG motor imagery signals. Journal of neural engineering, 14(1), 016003.

[16] Parvan, M., Ghiasi, A. R., Rezaii, T. Y., & Farzamnia, A. (2019, April). Transfer learning based motor imagery classification using convolutional neural networks. In 2019 27th Iranian Conference on Electrical Engineering (ICEE) (pp. 1825-1828). IEEE.

[17] Köppen, M. (2000, September). The curse of dimensionality. In 5th online world conference on soft computing in industrial applications (WSC5) (Vol. 1, pp. 4-8).

[18] Krebs, H. I., Hogan, N., Aisen, M. L., & Volpe, B. T. (1998). Robot-aided neurorehabilitation. IEEE transactions on rehabilitation engineering, 6(1), 75-87.

[19] Millán, J. D. R., Rupp, R., Mueller-Putz, G., Murray-Smith, R., Giugliemma, C., Tangermann, M., ... & Mattia, D. (2010). Combining brain–computer interfaces and assistive technologies: state-of-the-art and challenges. Frontiers in neuroscience, 161.

[20] Ramoser, H., Muller-Gerking, J., & Pfurtscheller, G. (2000). Optimal spatial filtering of single trial EEG during imagined hand movement. IEEE transactions on rehabilitation engineering, 8(4), 441-446.

[21] Zeiler, M. D., Krishnan, D., Taylor, G. W., & Fergus, R. (2010, June). Deconvolutional networks. In 2010 IEEE Computer Society Conference on computer vision and pattern recognition (pp. 2528-2535). IEEE.

REPRODUCING OUR RESULTS

To reprododuce our results please visit the GitHub page created for this project that details how data was converted into a usable format for models, how to use our code to train our models, and then how to directly implement them.
The below repository has a Jupyter Notebook that goes into great detail about how we achieved our results, providing a step-by-step analysis from data acquisition, all the way to our results.

Data set: https://www.nature.com/articles/sdata2018211
GitHub Repo: https://github.com/TekinGunasar/KDD-UC-2023