Various Methods for Extracting Spatio-Temporal Features from EEG Data

Arhison Bharathan
UCLA ECE
4th Year Undergraduate
Los Angeles, CA, USA
arhisonb@gmail.com

Grace Tang
UCLA ECE
1st Year Masters
Los Angeles, CA, USA
gtang12@g.ucla.edu

Robert Ozturk
UCLA ECE
4th Year Undergraduate
Los Angeles, CA, USA
robertcanberkozturk@gmail.com

Winston Wang
UCLA ECE
4th Year Undergraduate
Los Angeles, CA, USA
winston.mw@outlook.com

***Abstract*—EEG data exhibits both spatial and temporal characteristics. To capture both the spatial and temporal characteristics of the data, we formulated EEG movement classification as a spatio-temporal sequence classification problem. We explored 3D convolutional networks, spatial and temporal convolutions, and convolutional LSTM approaches to capturing the temporal nature of the data while retaining the spatial information. We found that both spatial and temporal convolutions struggled to learn and overfit the data severely, while convLSTM underfit the data and struggled to learn altogether.**

***Index Terms*—LSTM, STCN, EEG decoding, 3D-CNN, RNN**

I. Introduction

In class, we viewed neural network design from an image processing perspective solely focused on capturing spatial features of the data. In contrast, EEG data not only contains both spatial data from each of the probes, but also temporal data. To capture this information, we extend the convolutional approaches of image processing to also capture the time characteristics of the data. We used three primary approaches to design the neural network classifier: 3D convolutional networks that convolve through both space and time, sequential layers of spatial and temporal convolutions, and spatial convolutions which feed into convolutional LSTM layers.

II. Methods

All models were trained with data processed by the steps described in section A. The approaches described in sections B and C attempt to extract more information in the time-frequency and spatial domains, and they were applied in the cases detailed in Appendix B.

A. Initial Pre-processing

The raw EEG signals were first truncated to the first 500 time steps for each sample. The latter half of the signals were observed to contain primarily noise compared to the first half across all four classes. For data augmentation, the data was downsampled by 2, once with a maxpooling strategy and once with an averaging strategy. These two modified datasets were then concatenated together, along with a subsampling of the original data with Gaussian noise added to form the final dataset on which classification was performed. These preprocessing steps attempt to extract the most relevant features in the data by effectively high pass and low pass filtering in the maxpooling and average steps, and impose regularization via additive noise, while also scaling up the dataset size.

B. Spatial Reshaping

The original EEG data is taken from 22 channels which are positioned on various points on the transverse plane of the head as shown in Fig. 1 [5]. To extract spatial data, we reshaped and padded the data into a 6x7 array so that the height corresponded to increasing anterior location and width corresponded to increasing lateral distance. The depth of the tensor remains as the time axis. Our exploratory analysis in Fig. 2 shows that there are spatial variations in neural activity with regard to different stimuli, particularly between tongue and foot since they are quite different motions (compared to moving an arm to the left or right). This indicates that there is some spatial information that could be of use for classification.

The neural activity is highest in the posterior part of the head, right above the occipital lobe responsible for visual processing, rather than the motor cortex associated with movement. This may be due to the nature of the experiment, in which the subject is given a visual indication for which movement to perform. This leads to the possibility that our a classifier may be detecting and classifying the visual stimuli rather than the actual neural motor signals. However, this is just conjecture, as factors such as how electric dipoles are read may play a factor that should be further explored.

C. Wavelet Decomposition

Wavelet decomposition was tested as a method of feature extraction. The discrete wavelet transform (DWT) is a common technique for analysis of EEG signals, as it resolves

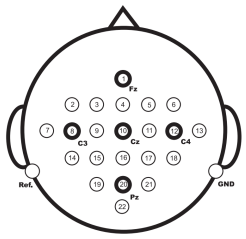

Fig. 1. EEG Channel Layout

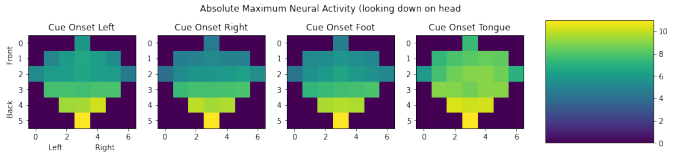

Fig. 2. Spatial Activity

a signal well in both time and frequency [1]. The DWT decomposition of a signal successively divides the frequency domain into lowpass and highpass sub-bands. At the first level, a lowpass filter produces approximation coefficients (A1) and the complementary highpass filter produces detailed coefficients (D1). At each subsequent level, the approximation coefficients are further decomposed. In this study, the number of decomposition levels L was treated as a hyperparameter, and a range of levels from 3 to 50 were tested. The extracted coefficients are denoted D1, D2, ..., D_L, and A_L.

Six statistical features—mean absolute value (MAV), average power (AVP), standard deviation (SD), variance, mean, and skewness—of the DWT coefficients were computed. These statistics have been shown to produce relevant wavelet features for EEG signal processing [2]. Their functions are given below.

$$\text{MAV} = \frac{1}{s}\sum_{i=1}^{s}|x_i| \tag{1}$$

$$\text{AVP} = \frac{1}{s}\sum_{i=1}^{s}|x_i|^2 \tag{2}$$

$$\text{SD} = \sqrt{\frac{1}{s-1}\sum_{i=1}^{s}(x_i-\mu)^2} \tag{3}$$

$$\text{Var} = \frac{1}{s-1}\sum_{i=1}^{s}(x_i-\mu)^2 \tag{4}$$

$$\text{Mean} = \frac{1}{s}\sum_{i=1}^{s}x_i \tag{5}$$

$$\text{Skew} = \frac{\sum_{i=1}^{s}(x_i-\mu)^3}{s*\text{SD}^3} \tag{6}$$

where x_i composes the set of wavelet decomposition coefficients for a given sub-band, channel, and sample, μ is the mean of the coefficients, and s is the number of coefficients.

After this computation, the wavelet feature matrix was a 4-dimensional tensor, with dimensions (N,L+1,6,22), where N is the number of samples.

D. 3D Convolutional Neural Network

Another way we wish to test the classification ability of the dataset's spatial dimension is to incorporate convolution over the 6x7 array as well as through time. This necessitates the creation of a 3D convolutional neural network (3D-CNN). In this architecture, data in the time dimension and in the 6x7 spatial domain are convolved with filter tensors at each NN layer.

In our architecture, we opt to test variations on the deepest networks we can construct given this dataset. To this end, the channel dimensions ultimately allow us to construct a network at depth 6 with filters: **5 x (T timesteps, 2 channel, 2 channel), 1 x (T timesteps, 1 channel, 2 channel)**. Pooling was applied to every other layer in order to reduce noisy backpropagation and maximize depth.

We modulated the number of filters in each layer to create three architectures. Two maintained a constant number of filters, while one increased deeper into the network, all limited as to not have the network blow up in size. The models were analyzed sans initial pre-processing for two separate dropout values and multiple L2 regularization constants to determine which model performed the best given optimized regularization.

Along with the initial pre-processing, we also believed that low-pass filtering the data by removing frequencies below 50 Hz would remove extraneous data from the EEG signals. This is because useful human brainwaves are thought to only exist below this threshold [6]. To this end, we tested our best 3D-CNN using only low-pass filter preprocessing.

E. Occipital Lobe Dependency

To test the idea that the Posterior EEG data is more important, we utilized a hybrid 1D convolutional fully connected LSTM neural network structure. When trained on the full data, it achieved a test accuracy of 0.696. Omitting the front 6 channels and training the model, gives an accuracy of 0.657, while omitting the last 4 channels leads to a test accuracy of 0.637, despite having more channels and more data than the prior experiment. This indicates that the 4 channels on the posterior part of the head are more important.

F. Convolutional LSTM

The fully connected LSTM discards the spatial component of data by containing recurrent layers to every other unit in the layer. To avoid this, Shi et al. [4] devised the ConvLSTM network, which replaces the input and recurrence relationships between units with convolutions, preserving spatial data. Our 2-layer convolutional LSTM yielded an accuracy of 0.36. It was particularly difficult to train this model as recurrent neural networks train much faster on CPU. This might be because on GPU, Keras moves data back and forth between RAM and GPU, leading to speed decreasing dramatically. This makes it extremely difficult to train complex convLSTMs. Furthermore, the convLSTM model significantly reduces the number of parameters in the model. Our models had 43,000 weights and took 1 minute per epoch. On the otherhand, hybrid convolutional LSTM networks that performed well on the dataset had about 300,000 parameters, and took 10 seconds per epoch with GPU accelerations. This indicates that alone, convLSTM's struggle to process input data from long timeseries.

G. Spatial-Temporal LSTM (STCN)

To alleviate some of the training problems with convLSTMs, Guo and Chao [3] placed frame-wise convolutional layers in front of the convLSTM layer. The hope here was that the convolutional layers would handle spatial processing much more efficient than convLSTMs. This was indeed the case as the STCN performed better with an accuracy of 0.413, despite having only 33,456 parameters. To decrease cost of training, we grouped the time steps so that the second convolution layer would take 4 time samples per filter. This downsampling made the convLSTM much easier to train due to the shallower unrolling.

H. Hybrid Convolutional LSTM

Inspired by Tonmoy's discussion, we then flattened the spatial dimensions, resulting in data with a height corresponding to time and channels corresponding to each EEG channel. We then implemented 4 convolutional layers and an LSTM layer. To increase the number of features available for the softmax classifier, we fed all sequences from the LSTM. We observed a 1 percent increase in performance over his model (0.696).

III. Results & Discussion

A. Using LSTM

We found that both implementations of convolutional LSTM struggled to learn the data, with the 2-Layer convLSTM achieving an accuracy of 0.36 while the STCN achieved an accuracy of 0.413. Discarding the spatial characteristics of the data and utilizing a hybrid convolutional model acheived an accuracy of 0.696. The convolutional LSTM architectures incorporated spatial and temporal features, but struggled to obtain a high accuracy. This may be because the spatial features are arranged in a 6x7 block, which is extremely small compared to CIFAR images and other use cases. As a result, the minimum filter size, 2x2, was not able to filter to find meaningful features. It is also important to note while there are 22 channels, there are 250 time steps per sample. The data is predominantly temporal, so reshaping and focusing on the spatial aspect of the data (where there is less information) hindered learning. Indeed, the STCN used by Guo and Chao was trained on a series of 31 consectutive 41x41 pixel frames [3], indicating that such networks perform much better on primarily spatial data. Furthermore, it may not have been necessary to preserve this spatial information anymore, given that the convolutional network had already been applied to the data. It seems that this rearrangement of the data made it harder for the neural network to learn relevant features to the model in a way that does not overfit the data.

B. Wavelet Feature Classifier

When classifying on wavelet decomposition features, L=4 produced highest accuracy, and the optimal architecture found was a hybrid network with one convolutional layer followed by an LSTM layer. Increasing the number of convolutional layers appeared to cause the model to overfit, even with more stringent regularization applied by increasing dropout in all layers. This effect can be understood as the wavelet decomposition replacing the role of convolutional layers in extracting features in the data, rendering additional convolution layers ineffective in extracting new and generalizable features.

The optimal model achieved a test accuracy of 0.3815, which is much lower than accuracy of classifying on the original pre-processed EEG signals. These results suggest that wavelet features may not be well suited for classification with a neural network in this context, and that a CNN can better extract features directly from EEG signals.

A possible explanation is that the mother wavelet chosen for the DWT (Daubechies wavelet of order 4, as in [2]) may not be optimal for this specific problem and should be optimized as a hyperparameter if given more time. A neural network might also perform poorly given that the height and width of the wavelet feature matrix, 5 (in the optimal case) and 6, respectively, are small, so the DWT produces significantly less data per channel per sample compared to the original pre-processed data with 250 time steps. However, increasing the number of levels was found to decrease the accuracy. At L=50, for instance, the test accuracy was 0.2310 even after optimizing the architecture. Ultimately, wavelet decomposition may be ill-suited for classification with a neural network on this particular dataset.

C. 3D Convolutional Neural Network

Optimization over regularization was done on three architectures until loss saturated. The results show that an increasing number of filters outperformed a constant number of filters. A constant 25 filter per layer led to a best validation accuracy of .57. A constant 50 performed similar at .56. Increasing the number of filters led to a best accuracy .62.

Implementing low-pass filtering with our best 3D-CNN resulted in no performance improvement. While this failed to improve our NN, this does suggest that our network previously learned from only the low-frequency data anyway. Therefore in future trials, we may remove high-frequency characteristics of the data to reduce overall processing workload.

When using the best model with the initial pre-processing method, we found that the model led to underfitting, producing a poor accuracy of .53, due to regularization from the added noise in the pre-processing step itself. After adjusting the L2 regularization of the 3D-CNN, adding the initial pre-processing produced an accuracy of .65, the best results produced in our 3D-CNN trials.

Throughout all our trials, we found that regularization was the key hyperparameter to optimize for good 3D-CNN performance, and often this had to be rather high. This is likely due to the overall noisy nature of EEG signals. Since large amounts of noise are present in the training data, high regularization reduces the effect of these aberrations.

As a whole, the 3D-CNN did not perform extraordinarily well even as it incorporated spatial characteristics. This would suggest EEG signals only need be computed on a per-channel basis for good performance.

REFERENCES

[1] Chen, D, Wan, S, Xiang, J, et al. A high-performance seizure detection algorithm based on Discrete Wavelet Transform (DWT) and EEG. PLoS ONE 2017; 12(3): e0173138.

[2] A. al-Qerem, F. Kharbat, S. Nashwan, S. Ashraf, and Khairi Blaou, "General model for best feature extraction of EEG using discrete wavelet transform wavelet family and differential evolution," International Journal of Distributed Sensor Networks, vol. 16, no. 3, p. 1550147720911009, Mar. 2020, doi: 10.1177/1550147720911009.

[3] J. Guo and H. Chao, "Building an end-to-end spatial-temporal convolutional network for video super-resolution," in Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, San Francisco, California, USA, Feb. 2017, pp. 4053–4060.

[4] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W. Wong, and W. Woo, "Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting," arXiv:1506.04214 [cs], Jun. 2015 [Online]. Available: http://arxiv.org/abs/1506.04214. [Accessed: Mar. 14, 2022]

[5] Brunner, Clemens, et al. "BCI Competition 2008–Graz data set A." Institute for Knowledge Discovery (Laboratory of Brain-Computer Interfaces), Graz University of Technology 16 (2008): 1-6.

[6] P. A. Abhang, B. W. Gawali, and S. C. Mehrotra, "Chapter 2 - Technological Basics of EEG Recording and Operation of Apparatus," in Introduction to EEG- and Speech-Based Emotion Recognition, P. A. Abhang, B. W. Gawali, and S. C. Mehrotra, Eds. Academic Press, 2016, pp. 19–50.

APPENDIX A
PERFORMANCE OF ALGORITHMS

Algorithm	**Test Accuracy**
Hybrid Convolutional LSTM	0.6964
Spatio-temporal LSTM (STCN)	0.4125
ConvLSTM2D	0.3600
CNN+LSTM for wavelet features	0.3815
3D-CNN, N = [25,25,25,25,25,25] w/ no initial pre-processing	0.57 (D = .5, L =.005)
3D-CNN, N = [50,50,50,50,50,50] w/ no initial pre-processing	0.555 (D = .5, L = .01)
3D-CNN, N = [25,25,50,50,50,100] w/ no initial pre-processing	0.6225 (D = .5, L = .05)
3D-CNN, N = [25,25,50,50,50,100] w/ LPF pre-processing	0.62 (D = .5, L = .05)
3D-CNN, N = [25,25,50,50,50,100] w/ initial pre-processing	0.5294 (D = .5, L = .05)
3D-CNN, N = [25,25,50,50,50,100] w/ initial pre-processing	0.6513 (D = .5, L = .001)

APPENDIX B
ARCHITECTURE OF ALGORITHMS

Algorithm	**Specifications**		
	Pre-processing	***Layers (Output Size)***	***Training Parameters***
Hybrid Convolutional LSTM	initial pre-processing	Conv2D (None, 250, 1, 25) MaxPooling2D (None, 84, 1, 25) BatchNormalization, Dropout Conv2D (None, 84, 1, 50) MaxPooling2D (None, 28, 1, 50) BatchNormalization, Dropout Conv2D (None, 28, 1, 100) MaxPooling2D (None, 10, 1, 100) BatchNormalization, Dropout Conv2D (None, 10, 1, 200) MaxPooling2D (None, 4, 1, 200) BatchNormalization, Dropout Flatten (None, 800), Dense (None, 100), Reshape (None, 100, 1) LSTM (None, 100, 20), Conv1D (None, 100, 100) MaxPooling2D (None, 34, 100) BatchNormalization, Dropout Flatten (None, 3400), Dense (None, 4)	learning rate = 1e-3 epochs = 50 optimizer = Adam batch size = 64
STCN	initial pre-processing	Conv2D (None, 256, 6, 7) MaxPooling2D (None, 256, 6, 7) BatchNormalization, Dropout Conv2D (None, 64, 6, 7) MaxPooling2D (None, 64, 6, 7) BatchNormalization, Dropout Conv2D (None, 16, 6, 7) MaxPooling2D (None, 16, 6, 7) BatchNormalization, Dropout (None, 64, 6, 7) Conv2D (None, 16, 6, 7) MaxPooling2D (None, 16, 6, 7) BatchNormalization, Dropout (None, 16, 6, 7) Reshape (None, 16, 1, 6, 7) ConvLSTM2D (None, 16, 10, 6, 7) Flatten (None, 3400), Dense (None, 4)	learning rate = 1e-3 epochs = 50 optimizer = Adam batch size = 64
ConvLSTM2D	omitted for space	-	-
CNN+LSTM for wavelet features	initial pre-processing, wavelet decomposition	Conv2D (None, 5, 6, 25) MaxPooling2D (None, 1, 2, 25) BatchNormalization Dropout Reshape (None, 2, 25) LSTM (None, 10) Dense (None, 4)	learning rate = 1e-3 epochs = 50 optimizer = Adam batch size = 64
3D-CNN	See Below	**5x** Conv3D (N Filters, (11, 2, 2) dimension) **5x** ReLU **2nd, 4th** MaxPooling3D ((2, 1, 1) dimension) **5x** BatchNorm3D **5x** Dropout Conv3D (N Filters, (22, 1, 2) dimension) ReLU MaxPooling3D ((2, 1, 1) dimension) BatchNorm3D Dropout Reshape (None, 2, 25) Dense (None, 4)	LR: 1e-2 with ReduceLROnPlateau epochs = 150 optimizer = Adam batch size = 50 Dropout = D L2 Reg = L

3D-CNN Variations - We ran all combinations of the following. We only report the test accuracies of the best combination for each N for brevity.	
Pre-Processing (PP)	No PP was applied in the initial architechtural comparisons. We compared results of LPF PP and the initial PP on the best model.
N	[25,25,25,25,25,25], [50,50,50,50,50,50], [25,25,50,50,50,100]
D	.5, .7
L	.001, .005, .01, .05
ReduceLROnPlatueau	This setting decays the LR by by 0.2 once the validation loss has not decreased by more than 0.01 after 10 epochs.