Andrew Washington

COGS 118A

Professor Tu

December 15, 2017

Exploration of Neural Net Architectures for End-to-End Electroencephalogram Classification

Abstract:

With the recent revolution in neural network architectures, it is surprising to see such little research into neural net performance on EEG waveform classification. In this project, I explore multilayer perceptrons, recurrent neural nets, and convolutional neural nets as end-to-end classifiers of raw EEG data. I find that out of the architectures tested, convolutional neural nets perform best on a binary classification of whether a subject's eyes are open or closed given a single channel of EEG recording from the medial occipital lobe.

Introduction:

Despite the large amount of research in electroencephalogram (EEG) waveforms and revolution in neural network architectures, little work has been done in the field to connect the two together. Traditional digital signal processing techniques are widely used, despite the fact that neural nets can outperform, or at least perform at the same level as the traditional techniques (Lawhern et al, Schirrmeister et al). Similarly, deep-belief nets, convolutional neural nets, recurrent neural nets, and recurrent-convolutional nets, have all outperformed several traditional machine learning techniques, including support vector machines, logistic regression, and random forests (Bashivan et al).

Most of the progress made in creating neural nets for EEG analysis has been in multi-channel EEG recordings. In this project, I will focus on single channel recordings to focus on the

time-series aspect of EEG waveforms. Preserving the spatial dimensions of EEG recordings can help with understanding the neuroanatomical features used by neural nets. Furthermore, neural nets might even be able to learn features neuroscientists today do not yet understand (Schirrmeister et al); but, this is not the goal of this project. Despite the avoidance of spatial dimensions in my project, it is possible that neural nets might learn typical neurobiological features, such as power spectra and power in specific ranges, e.g. alpha (8-12 Hz), beta (12-40 Hz), and theta (4-8 Hz) bands. Specific to this experiment, alpha band power is the most likely neurobiological feature to be learned, as it is known to be associated with visual tasks, such as the one performed by the subjects in this experiment, as well as with the occipital lobe, from which the EEG data used in this project came from.

In "Learning representations from EEG with deep recurrent-convolutional neural networks," Bashivan et. al. test several neural net architectures, including recurrent neural nets, convolutional neural nets, and combinations of both on a four-class classification task. They find that a recurrent-convolutional network outperforms other neural net architectures. Moreover, all the neural networks they trained performed better than other classical machine learning models, including support vector machines (RBF kernel), L1 logistic regression, and random forests.

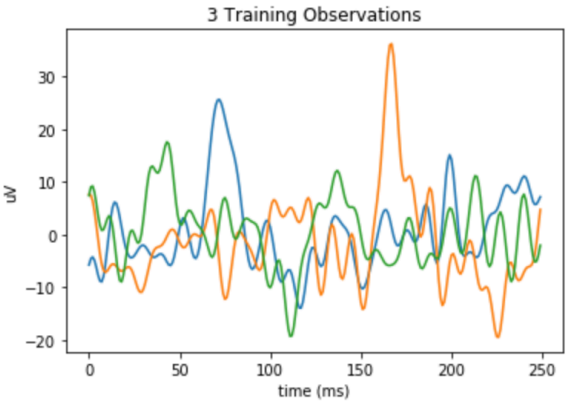
In "EEGNet: A Compact Convolutional Network for EEG-based Brain-Computer Interfaces," Lawhern et. al. describe a fully convolutional neural net architecture and use it to perform four different classification tasks. The researchers' main goal was to create a working model with as few parameters as possible. They omitted any fully connected layers and got the number of parameters down to under 2200. EEGNet outperformed the widely used xDAWN Spatial Filtering and Bayesian LDA model, as well as the Brain Computer Interface competition-winning Covariance Shrinkage Regularized CSP model.

One goal in this experiment is to have the included models be accessible to EEG researchers. As such, only models that could be trained in a reasonable amount of time on a typical modern workstation were considered. This ruled out very deep networks (especially when it comes to recurrent neural nets) and various hyper-parameters that would dramatically affect the training time (e.g. stochastic gradient descent as a solver algorithm). Training time was kept under thirty minutes for at least ten epochs of training (i.e. under three minutes on average per epoch) for all models, using the computer setup described in the Methods section.

Methods:

Data was gathered using a 64 channel BrainVision ActiCap EEG system, although only the Oz (central occipital lobe) channel was used in the classification task. A 500 Hz sampling rate was used. Impedances kept below 20 kΩ. Ten subjects were recorded while performing a visual target detection task for two minutes each, then were recorded for another two minutes while at rest with their eyes closed.

Due to the end-to-end nature of this project, the data processing was kept to a minimum. Only a bandpass filter (2-50Hz pass-band) was used to clean the data. The bandpass filter used is defined in NeuroDSP (version 0.2). For classification, the data was split into 500ms intervals, yielding 4720 observations. These were split for testing and training by subject, with nine subjects for training and one for testing, yielding 4248 training observations and 472 testing observations.



All models were made using the Keras API (version 2.1.2) with a Tensorflow (version 1.4.0) backend in the Python programming language (version 2.7.13). All work was done on an Early 2015 MacBook Pro, with 8GB RAM and 2.7 GHz Intel Core i5 processor, running MacOS Sierra 10.12.6. All work was done in a Jupyter Notebook (Jupyter version 4.3.0, IPython version 5.2.2). The Python packages used include: NumPy (1.13.3), SciPy (0.18.1), and NeuroDSP (0.2).

Multilayer-perceptron (MLP) models tested with varying activation functions, numbers of layers, number of neurons in each layer, dropout rate, and with and without batch normalization. The activation functions tested include ReLU, ELU, tanh, and sigmoid. The number of layers varied from two to three, as three layers already showed signs of overfitting. The number of neurons in each layer varied from 50 to 150. Dropout rate ranged from 0 to 0.9 for the final layer.

Several recurrent neural net architectures were tested. All neurons in recurrent layers were Long-Short-Term-Memory (LSTM) neurons. The number of layers ranged from two to four and the number of neurons in each layer ranged from 5 to 250. The last layer was always a fully connected layer. The activation functions tested in the LSTM layers include tanh, ReLU, ELU, and sigmoid.

The first convolutional neural net was based off of the Keras tutorial available at: https://keras.io/getting-started/sequential-model-guide/. This was done initially for simplicity's sake, but the model seemed to perform well, so the general architecture was kept for the first rounds of models. The activation functions test in the convolutional layers include: ReLU, tanh, sigmoid, ELU. The size of filter kernels ranged from 3x1 to 100x1. The number of filter kernels (per layer) ranged from 10 to 128. The activation functions in the convolutional layers tested include: tanh, ReLU, ELU, sigmoid, and linear. The number of fully connected layers ranged

from 1 (single neuron for output layer) to 3. The other hyper-parameters tested for the fully connected layers are the same as those tested for the MLP classifiers.

The baseline classifier I used in this experiment was a simple threshold for alpha power. It was based on the well-known fact that visual alpha power is heightened when the subject is resting with their eyes closed, and is demonstrated by the figure to the left. The classifier classifies a trial as eyes-closed if the power in the alpha range is greater than the median eyes-open alpha power taken from each trial in the training set.

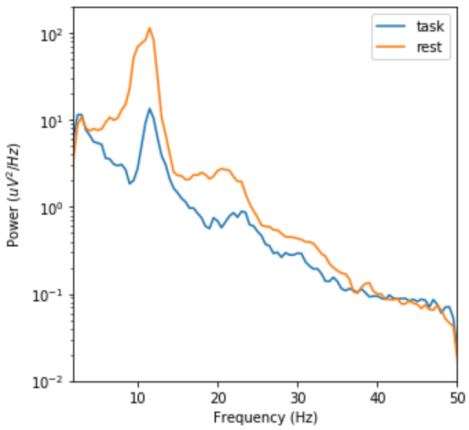
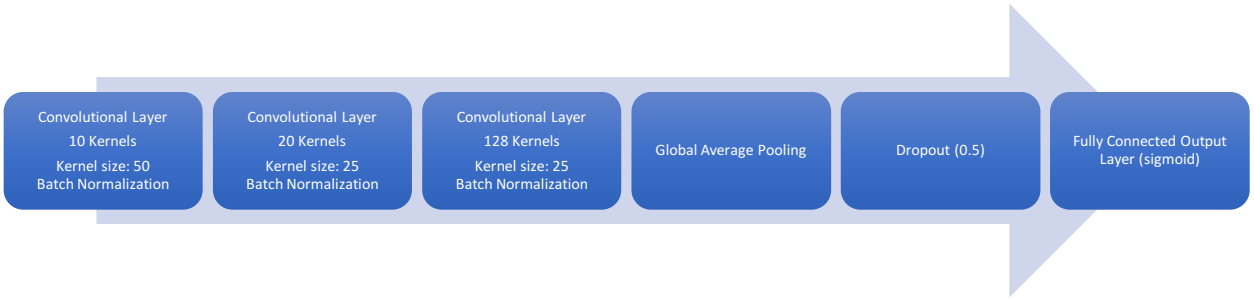


Figure 1: Alpha Power for Subject #1, full four minutes of recorded data

Results:



The best model trained was the convolutional neural net depicted above. This was picked by taking the highest training accuracy where the testing accuracy was within 2% of the training accuracy or higher. The rest of the "best models" from the other architectures are shown in the table to the right.

Architecture Type	Training Accuracy (%)	Testing Accuracy (%)
CNN	80	78
MLP	83	77
LSTM	74	61

The best multilayer perceptron model achieved 83% training accuracy, but only 77% testing accuracy. Increasing the number of layers did not increase training accuracy, but did *decrease* testing accuracy, suggesting that more than two layers only encouraged overfitting. ReLU activation outperformed any other activation function,

with ELU as a close second; others significantly decreased model performance. Changing only the second layer activation to tanh improved training accuracy to over 80%, did not increase testing accuracy, suggesting overfitting.

The best RNN was a four layer network with 250 LSTM neurons in the first two layers, 5 LSTM neurons in the third layer, and a single fully connected neuron in the output layer. This model was suspected of overfitting due to a 74% training error with a 61% testing error. Because of the overfitting problem and long training time, exploration of the recurrent neural net architecture was stopped in favor of convolutional neural nets. Regarding LSTM activation, only tanh produced decent results.

ELU activation in the convolutional layers did not significantly increase testing or training accuracy, and all other convolutional-layer activation functions tested performed significantly worse (except for sigmoid activation when the kernels were small). This is similar to the results of the experiment performed by Schirrmeister et al, where ReLU and ELU outperformed all other test activation functions. In their case, although, ELU significantly outperformed ReLU, whereas there was no significant difference in training or testing accuracy between the two in this experiment. ELU did, although, converge faster than ReLU, as seen in the table to the right. Any difference in training time was not significant.

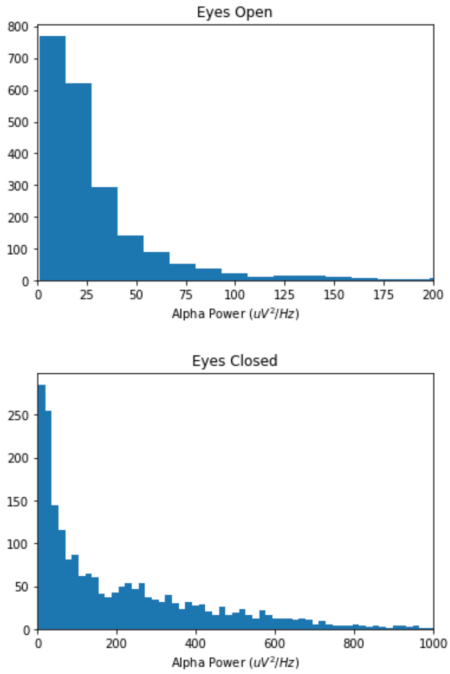
Epoch	ELU activation	ReLU activation
1	69%	65%
2	76%	72%
10	78%	78%

Training Accuracy for Different Convolutional Layer Activation Functions

The inclusion of batch normalization after every convolutional layer did not make a significant difference to training accuracy or testing accuracy. It did, although, improve convergence time. The first epoch of training resulted in 75% training accuracy, as opposed to

69% without batch normalization. These results are similar to, but do not directly match, those in "Deep learning with Convolutional Neural Networks for EEG Decoding and Visualization." Batch normalization and dropout did not significantly improve training or testing error in this experiment (whereas they did in Bashivan et al).

Modifying the size and number of filter kernels showed small improvements in model performance. Larger filters resulted in better training and testing accuracy (up to about 3% improvement), but only when the filter kernel size in the final convolutional layer was enlarged as well. Decreasing the number kernel filters in each convolutional layer did not affect training accuracy, but did reduce the likelihood of overfitting.



The baseline classifier only reached 43% training accuracy and 41% testing accuracy. This is because the alpha power for eyes-open and eyes-closed data is far from linearly separable, as shown in the figure to the right. Still, it represents use of the power spectrum information a researcher would use to determine whether a subject's eyes are open or closed. Typically, one looks for a "rhythmic" oscillation around 8-12 Hz, but the easiest way to represent this is the power in that range. An alternative approach might be to use the ratio of alpha power to the total power across all frequencies, but even this would have its problems.

Discussion:

The classification task used might have given the LSTM network a disadvantage. The way I devised the task took out a lot of the time series nature of the data. The task became more similar to an image classification task than a signal processing task, although this distinction lies

upon a fine line, and is ultimately a question of definitions. My goal in this project was to have an end-to-end neural net, so adding features was out of the question, but a new task might favor LSTM networks a bit better.

It was a surprise to see a standard multilayer perceptron perform as well as the one in this project did. Originally, I had thought that they would be miserable in comparison to LSTM and CNN architectures. As shown earlier, though, they are quite susceptible to overfitting. Thus, I would continue using CNN architectures over MLP architectures.

It might be noted that Bashivan et al found that a recurrent-convolutional neural net was the better than both the recurrent and convolutional neural nets they used. This architecture was not explored in this project for a number of reasons. First, the model they used had the recurrent layers in parallel with the convolutional layers, so the model was effectively an ensemble classifier. Because of the slow training time and poor performance of the LSTM models trained in this project, the architecture described by Bashivan et al was not explored. Furthermore, their model also used a convolutional pre-processing step, which was avoided in this project so that the neural nets would be trained on data as close to raw as possible.

Conclusion:

Convolutional neural nets can be a useful tool in EEG waveform classification, even on relatively small datasets. While classical signal processing techniques are much more common in the field of EEG analysis, the adoption neural nets should be welcomed.

Bonus Points:

Bonus points might be awarded for: using original data; using an end-to-end neural net to get decent performance where other researchers used more engineered features; and/or for citing recent literature in the field.

References

Bashivan, P., Rish, I., Yeasin, M., & Codella, N. (2015). Learning representations from EEG with deep recurrent-convolutional neural networks. *arXiv preprint arXiv:1511.06448*.

Lawhern, V. J., Solon, A. J., Waytowich, N. R., Gordon, S. M., Hung, C. P., & Lance, B. J. (2016). EEGNet: A Compact Convolutional Network for EEG-based Brain-Computer Interfaces. *arXiv preprint arXiv:1611.08024*.

Schirrmeister, R. T., Springenberg, J. T., Fiederer, L. D. J., Glasstetter, M., Eggensperger, K., Tangermann, M., ... & Ball, T. (2017). Deep learning with convolutional neural networks for EEG decoding and visualization. *Human brain mapping, 38*(11), 5391-5420.