Data Classification Methods

Daniel Bici
Department of Electrical and Computer Engineering, Temple University
danbici@temple.edu

Introduction: For our final, we participated in a class competition to get the best performance when classifying datasets. This project classifies datasets provided using a neural network approach, in this case I chose to use a convolutional neural network. Ten sets of data were provided which was 1D training data, development data and evaluation data. For a non-neural network-based approach, I chose to use simple thresholding.

The detailed analysis of the two algorithms are provided below in detail.

Algorithm No. 1 Description (CNN): A convolutional neural network is a simple algorithm where the ability to learn many filters in parallel that are specific to a training dataset under the constraints of a specific predictive modeling problem. It is designed to automatically and adaptively learn spatial hierarchies of features through backpropagation. It was a great choice for this project because it was quicker to implement and there was no need to pre-process the data, and we had a decent amount of data to work with. These were more desirable than an RNN due to their cheaper computational complexity, and with this, we were able to work with data. Figure 1 below demonstrates that we could select a window size that is close to the minimum duration of the event.

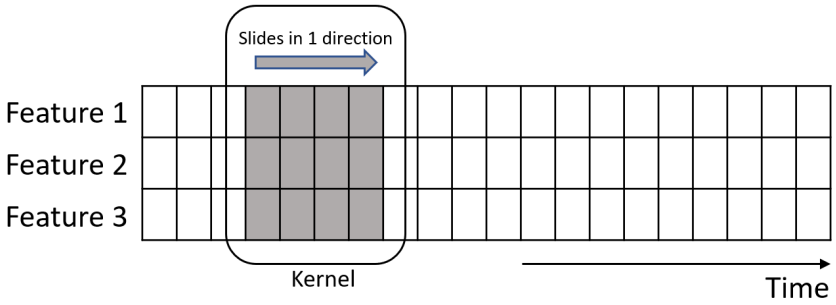

Figure 1: Convolving on Time Dimension

For the CNN, I attempted to overlap the windows because the model wasn't training. I used the PyTorch function Adam to optimize the data and tried to tweak the learning rate to adjust the weights based on the training data, but didn't have enough time to produce meaningful results utilizing the scoring package. This optimization is appropriate because of its application for noisy gradients, and even though it's not a large network with only a single convolution layer and a single fully connected layer for each output totaling in 3 separate fully connected layers, the networks gradients were not able to accumulate in order to reach a minima because of the limited time I allotted for this project.

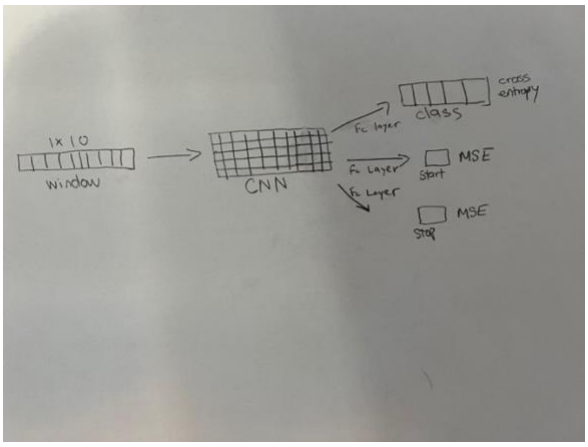

Figure 2: A sketch of the CNN

Algorithm No. 2 Description (Simple Thresholding): Simple thresholding is a type of evaluation method that determines a threshold for an evaluation score and treats the predictions differently depending on whether or not they score above the threshold. It looks at the difference between two points and compares the difference in amplitude between them; if the threshold is > 6, it isn't background noise, then the classes are defined by taking modulo 5 (since there are 5 classes) of the second amplitude, setting low amplitudes as background noise. An important part of choosing a threshold is to determine how much you'll suffer for making a mistake.

Postprocessing and tuning the system:

Because the output of both algorithms are formatted in the same manner, I was able to use a single postprocessing function for both the CNN and thresholding models. The format of the output is class, start, stop. The class output ranges from 0 to 4 for each class, including background; start is an integer that tracks the possible starting index within a sequence of inputs. If the sequence is not predicted to have a starting index, then start will be assigned a value of -1. The stop output follows the same formatting scheme as start. In post processing, I iterate over all values of class, start, stop for each non overlapping sequence. Then, during this iteration, I assign a start index variable on our first seen positive start position. I then wait for a positive stop index which then becomes the start and stop indices of our event. Class assignment is done by taking the first non-zero background class if one exists, otherwise the event will be a background class. This is done for all sequences of size 10 to assess larger event windows.

Results:

Right away, it was easy to notice the heavy class imbalance. I was unable to run the scoring package and produce meaningful results on the dataset error rates.

Conclusions:

I presume that the CNN will be significantly better over time in terms of error rate, but that the thresholding would be the better model as this system currently is. An oversight of many in this project is that the thresholding does not consider the individual classes, more-so if there is background noise or not, and it wasn't a great algorithm to use. Training the CNN model for longer would produce far better results.

	Data Set		
Algorithm	**Train**	**Dev Test**	**Eval**
CNN			
Simple Thresholding			

Table 1. Error rate for different datasets and algorithms