Takehome Final Exam

Michael Albert, Jonah Douglas, Ethan Lindell, Archan Rupela

June 9, 2020

1. Machine learning can be understood as finding a line of best fit from data in order to predict the outcome of similar data. You can think of it as an equation and trying to assign appropriate value to different characteristics. For example, if you are trying to identify a chair and you have characteristics such as number of legs, height, color, etc. then the model should learn that the color does not play a role in whether or not the image is of a chair.
   Deep learning is very similar to machine learning in that you are to get the computer to learn an equation in order to predict some outcome. Typically in deep learning, the difference is that the characteristics are not provided to the computer. Instead of providing characteristics such as number of legs, height, color, etc. you would just provide an image and whether or not the image is of a chair. The computer then works to identify what is important and what isn't important to identifying a chair in an image.

2. If you took a trained model and cranked the forget gate bias vector up to contain very large positive values, an LSTM would fail to make any connections between past data and current data.

   If you took a trained model and cranked the forget gate bias vector down to contain very large negative values, an LSTM would act like an RNN because it would be remembering everything. However, since the network is not forgetting anything it would eventually break. This was one of the original problems with LSTMs since it did not have a forget gate, large negative values would have a similar impact.

3. Torch.nn.BCE creates a criterion that measures the binary cross entropy between the target and output, torch.nn.BCEWithLogitsLoss is very similar to nn.BCE, but combines a Sigmoid layer in a single class to calculate the loss since it is numerically more stable than using each function separately. We could use BCEWithLogitsLoss on multiclass classification, and BCELoss we could use on binary classification.

4. (a) The first baseline I would compare the model to would be a majority class baseline. This gives us an idea of the minimum accuracy our model should be achieving, since anything worse than this is performing worse than random guessing. The second baseline I would compare the model to would be a premade model trained on imagenet, perhaps vgg11. This gives us an idea of how a general model performs vs our own and whether or not our choice of system is appropriate.

   (b) I would examine how accurately the model is performing on the training dataset. If it has an extremely high accuracy when training but is subpar when testing, then it is likely overfitting. If the accuracy remains low for training and testing, then it is likely underfitting.

   (c) One choice that may reduce overfitting would be adding dropout layers to the model or modifying the existing dropout layers to be harsher. Another option would be to augment the training dataset through affine transformations and/or color manipulation in order to have a dataset that is more representative of images you may find in the real world. Finally, you could simplify the model by removing layers.

5. The features we chose to use for this task will be the student name and the course name. We decided not to use the essay contents because there is likely a large amount of variation that may lead to the model learning incorrect connection. For example, different instructors may teach the same course and assign similar essay topics. Which instructor does the model then associate the essay contents

towards? Likewise, the same instructor may assign different topics each year, in which case there is little useful information to be gleaned from the essays.

For our model, we believe that a multinomial logistic regression model would work best in this case. As we have a low amount of features, we want to avoid a model with many layers as that would likely lead to overfitting very quickly. Since we are attempting to classify to one of several possible outputs, cross-entropy loss seems like a good fit for the loss. Again, since there are several classes, we chose softmax as the output activation for our model.

6. For our features, we will take a daily average of the wind direction and speed for each location. Our reasoning is two-fold: Firstly, this puts the training data in the same form as the testing data. Secondly, this will speed up the training time of our model since we will have $(365 * 20)$ data points for each cell on the grid instead of $24 * 365 * 20$ data points.

   For our model, it seems reasonable that an LSTM would be most appropriate for our model. This is because we want to learn seasonal patterns as well as times that extreme weather may occur such as hurricanes. We will use Mean Square Error loss since this is a regression problem. For our output activation we could use ReLU or Linear activation since we could take in the wind direction in degrees and all data gathered will be contained inside the range [0..360].

7. For our features we chose to use the transaction amount, where the transaction took place, and the price. For our model, we decided to use a standard deep neural network. For our output activation, we will use sigmoid since the output is either fraudulent or non-fraudulent. For the loss function, we will use binary cross-entropy loss in order to get percentages of how likely it is that a given transaction is fraudulent.