Assignment2

-Yixi Rao u6826541

Question 1

1.1

I used the range of c values displayed in the matrix below, where the row represents the different base values ($2^c,\ 3^c,\ 4^c,\ 5^c,\ 6^c,\ 7^c, 8^c,\ 9^c,\ 10^c$), the column represents the different powers ($r^{-5},\ r^{-4},\ r^{-3},\ r^{-2},\ r^{-1},\ r^0,\ r^1,\ r^2,\ r^3,\ r^4,\ r^5$).

$$\begin{bmatrix} 2^{-5} & 2^{-4} & 2^{-3} & \cdots & 2^3 & 2^4 & 2^5 \\ & \vdots & & \ddots & & \vdots & \\ 10^{-5} & 10^{-4} & 10^{-3} & \cdots & 10^3 & 10^4 & 10^5 \end{bmatrix}$$

In general, the range of c is all the value $b^p\ for\ \forall b \forall p\ where\ b \in \{2, 3 \ldots, 9, 10\},\ p \in \{-5, -4, \ldots 0, \ldots 4, 5\}$.

The reasons why I choose this range are that we are using the explicitly constrain parameters as part of the loss function to do regularization, and we do not know what degree the regularization we need, so we cannot decide the magnitude of the λ, which adjusts the Loss function how much attention that should be paid to the model complexity. If we pay more attention to the model complexity, we may get a good accuracy on the validation data, but sometime we get more accuracy when just emphasizing the loss. So, setting the C as $C = Base^{\{-5,-4,\ldots 0,\ldots 4,5\}} = \frac{1}{\lambda}$ can let λ cover a wide range of magnitude (from $Base^{-5}$ to $Base^5$), which can help us find the proper C. And the reason of using the different bases is that different bases have different sizes of the interval in this set ($Base^{\{-5,-4,\ldots 0,\ldots 4,5\}}$), for example, $2^2 - 2^1 = 3$, while $10^2 - 10^1 = 90$. This kind of adjustment of the length of the interval can make the feasible C set more fine-grained, which can help to find the most appropriate C.

The search technique I used is the Grid search. The advantage of Grid search is Grid search is a brutal way of finding the optimal parameters because it trains and test every possible combination. The possible option for base is the $b \in \{2, 3 \ldots, 9, 10\}$ and possible option for power is $p \in \{-5, -4, \ldots 0, \ldots 4, 5\}$. If I using every combination of the settings by using the for loop is very time-consuming, so I decide to first determine the best base value, then using that best base to find the best power. Now we should find a reasonable default value, we analyze that $p = 0$ is apparently useless, and using the $p = 1\ or -1$ will cause the C space spans in a small space (range is not wide) but using $p = 5\ or -5$ will cause the interval of the C set so big. Therefore, using the intermedia value $p = 2, 3, 4\ or -2, -3, -4$ would be appropriate.

1.2

Best C = 32

1.3

Accuracy = 0.8544

Question 2

How your final solution works?

The final solution uses the GRU model, which contains three layers: The Embedding, GRU layer, and the Dense layer using the SoftMax. The embedding layer is used to transform the index sequence to the word embedding, the pre-trained embedding I used is the GloVe 100d vector, this layer will output a (batch_size, 822, 100) tensor. Next, the tensor will go to the GRU layer to produce the history, the reason why using the RNN is that the input is a structured sequence and we care about the meaning of the sentence. The reason why using the GRU is that the simple RNN has the problem of vanishing gradients and exploding gradients.

How you trained and tested your model?

For the training section, I notice that the training data is very imbalanced, for example, it contains 6743 training documents but only 5% of the training documents is horror book genre, 6% of the training documents is the crime fiction book, and 19% of the book is the humor book. The majority (70%) of the book is the science fiction. This kind of imbalance will cause a very low F1 score and the result of the prediction is almost genre 1. To solve this problem, I first try the over-sampling and under-sampling but the accuracy and F1 score is still very low because of the overfitting caused by over-sampling and the information loss caused by under-sampling. Then I try to change the Loss Function weight to make the model more sensitive to a few genres of documents from an algorithmic level. In Keras, I use the class_weight to adjust the weight of the loss of different genres such as I give more weight to genre 0, 2, 3 to make them more sensitive.

I split the training, validation, test set not randomly, I split it based on the document id (book genre). I want the documents from the same book to be distributed regularly such as if we have 10 documents form the same book and the split ratio is 7:2:1, then 7 books will go to training set, 2 books will go to the validation set, and one book will go to the test set. This special split can train the model to better understand any text from the same book should belong to the same genre.

The hyperparameters I select to search are GRU output dimension, number of GRU layer number of epochs, number of batch size, dropout rate. I use the Grid search and early stopping to tune the hyperparameters. For the early stopping, it is used to stop the training if the validation accuracy and loss are not improved, it is a good technique to prevent overfitting. For the Grid search, first I choose a hyperparameter and fix others to tune it, and then select the next one, and so on.

What models you tested, which worked, and which didn't?

I tested the DNN model with different layers, and simple RNN model, LSTM model, GRU model. Multi-layer GRU or LSTM model, and also the logistic regression model. The LSTM model, GRU model work very well but the DNN model and logistic regression model work poorly.

Why you think these other models didn't work?

For the logistic regression model, I think the reason it did not work is that it does not consider the meaning of the text, so some documents may have similar BOW but their meaning is different due to the reading order. For the Multi-layer GRU or LSTM, I think it contains too much parameters to tune and the sample data is limited, so it will overfitting easily. For the DNN network, it does not consider the meaning of the sentence of the order of the word, so its performance will less than RNN model. For the RNN model, it has the problem of using the same neural network at each step of the sequence, and it has the problem of vanishing gradients or exploding gradients which makes it very difficult for the RNN to learn representations of long document.

Question 3

Argmax:
John Borne.

Random:
Harre Eoger.
Poneh Karghull.
Gulers Bidian.
Joss Bugtlerlda.
Frithr vimme.
Dan Ptor.
Alid Feucha.
Arth Leny.
Mickn Sterk.
Daby Jatlvhergfde.