Assessment 3: Writing Neutral Networks – Language processing

Introduction

For assessment 3, the task is to write a Pytorch program that learns to read business reviews in text format and predict a rating (1=positive or 0=negative) and a business category (0=Restaurants, 1=Shopping, 2=Home Services, 3=Health & Medical, 4=Automotive) associated with each review. This report outlines how my program works and my design decision.

Selection of architecture, algorithms, and enhancements

My architecture contains implementation for Bidirectional LSTM (BiLSTM) followed by a fully connected layer with Rectified Linear Unit (ReLu) activation function and an input Dropout rate of 0.2.

Due to the limitations of Simple Recurrent Networks like may struggle with long range dependencies, I used Long Short Term Memory networks (LSTMs) which can learn long range dependencies using a combination of input, output and forget gates. Moreover, the BiLSTM can improve model performance on sequence classification problems by propagating the input forwards and backwards through the LSTM layer and then concatenating the outputs. Therefore, the network can keep information in both directions.

For activation function, I used ReLu activation function. Compared to TanH and Sigmoid activation function, the ReLu activation function has no issue of Vanishing Gradient problem therefore prediction accuracy and efficiency is maximum. A Dropout rate of 0.2 is used with input to avoid network overfitting.

Selection of cost function and optimiser

Since the network contains two predictions, we have a loss function combines rating loss function and category loss function. For category loss function, I used Cross-Entropy Loss function. The category prediction has 5 targeted outputs which is a multi-class classification problem. Cross-Entropy is the default loss function to use for multi-class classification problem because it can summarise the average difference between the actual and predicted probability distributions for all classes. The rating prediction has output of 0 or 1, so it is a binary classification therefore I used Binary Cross-Entropy Loss function.

Optimiser Adam and SGD were tried for the network and optimiser Adam gave much better performance. The performance of optimiser Adam gave around 84% weighted score while optimiser SGD only achieved 43% weighted score with all other metaparameters fixed.

Selection of dimension for GloVe word vectors and other metaparameters, and data preprocessing

GloVe word vectors dimension of 50, 100, 200 and 300 were tried for the network. The best result is 100, however sizes of 200 and 300 obtain a very close result.

When number of epochs was fixed at 10, different batch size of 32, 64 128, 256 and 512 were tried for the network. Batch size of 32 leads to a higher accuracy overall.

I selected number of hidden layers as 2, which gives us a stack of two LSTM layers which can performs deeper learning.

Some data preprocessing was processed for original data:

- ✧ remove non ascii characters
- ✧ remove punctuation and numbers
- ✧ convert letters to lower case
- ✧ remove stopwords (list from nltk)

Use of validation set and any additional steps taken to avoid overfitting and improve generalisation

When I tune parameter in the training phase, the training set and validation set were split into 80% and 20%. This can help to check and avoid overfitting and eliminate errors in future predictions.

Dropout as regularization is useful and it can improve the network accuracy on the test set. A Dropout rate of 0.2 is used with input to avoid network overfitting.