SUBMITTED BY Z-5349878
19-04-2022

WRITING NEURAL NETWORKS – LANGUAGE PROCESSING

ASSESSMENT 3

ZZEN9444-NEURAL NETWORKS, DEEP LEARNING (H222 ONLINE)

Introduction

This report will discuss the application of neural networks (NN) to the task of text classification in Pytorch. We will train the network to undertake predictive text classification of business reviews into categories and ratings by using GloVe vectors applied to Simple Recurrent Networks (SRN) such as Long Short Term Memory (LSTM); and the various considerations in building out such a network.

The data

The data consists of a collection of business reviews with columns mapped to labelFields and textFields. Our dataset consists of two label fields: one for business category (0-4) and a second for a positive or negative rating (0-1). The classification being undertaken is logistic in nature with multi-label and multiclass classifiers for business category and rating.

Dependence vs independence

In multi-label classification, the architecture of the net is built around correlation or independence of the labels (rating & category). For example, if we believe that the category is correlated with rating, then the model should incorporate connections between the two outputs in classifier chains.

If instead, we assume independence between review rating and business category, then the architecture would be based on Binary Relevance. Here we treat each label as separate classification tasks and the weights are backpropagated in separate LSTM layers.

We will consider the simple case of independence and assume that ratings are not dependent correlated with business category. This will inform the structure of our model and network layers.

Tokenisation and pre-processing

1. Replace punctuation with whitespace (however apostrophe was simply removed to preserve meaning, for example in “don’t” or “didn’t”).
2. Tokenisation achieved through split() function to separate words and remove whitespace.
3. Stop-words removing high frequency words that did not contribute to predicting a rating or business category such as: pronouns, connective clauses, common dates, etc.
4. Removal of low-frequency words (freq <1) did not contribute significantly to the overall performance of the model.

Cost function and optimiser

As with most NNs, classification requires a cost function and optimiser to minimise the difference between prediction (training data) and target (label data). The ratings classifier can be conceived as a simple binary classifier with two classes (positive or negative) determined from the activation of a single node (BCEWithLogitsLoss). Alternatively, we can conceive the binary case as a multi-class classifier with two outputs: one node representing positive rating and the other representing negative (nn.CrossEntropyLoss). Between the two methods, nn.CrossEntropyLoss() was found to generate a smaller loss for the rating classifier and yielded higher accuracy.

Category is a multi-class label arising from the five business categories in the dataset. The appropriate multi-class loss function is negative log likelihood loss or NLLLoss() (plus a LogSoftmax layer). In Pytorch, the result is equivalent to applying nn.CrossEntropyLoss(), however nn.CrossEntropyLoss() does not required an additional LogSoftmax activation layer to be applied and is considered more efficient, thus was chosen to be the loss function.

The testing for highest accuracy optimiser considered Stochastic Gradient Descent (SGD) and Adaptive Movement Estimation (ADAM). It was found that SGD was more effective in minimising the loss function achieving minimum at similar speed; thus, SGD was chosen as the optimiser.

Model and dimensionality of layers

Using Binary relevance, separate LSTM network layers were built for rating and business categories. The dimensionality of the model was determined using .size() on the input tensors, which had the following dimensions: [batch size, review length (padded), dimension of GloVe vectors]. The review length varied between batches based on longest review in each batch and dynamic padding applied. The chosen batch size of 16 and with GloVe vector dimension of 200 provided the highest accuracy in testing. Compared to 50- and 100-dimensional GloVe vectors, the use of 200-dimensional GloVe vectors appeared to capture more contextual and semantic information for our classification, generating higher accuracy during testing. However, the increase in accuracy was offset by longer training times. 300-dimensional vectors did not yield increased accuracy.

For each classification task, the input was fed into a LSTM layer with 20% drop out to avoid overfitting. The initial LSTM layer utilised input dimensions of 200, to match the GloVe vector input and output dimensions of 60 and 90, respectively for rating and business category classifiers. This was fed into a fully connected output layer with 2 and 5 output nodes respectively for the (binary) rating classifier and (multiclass) category classifier with 5 outputs (one for each category: Restaurants, Shopping, Home Services, Health & Medical, Automotive).

An alternative to the LSTM layer would be a Convolutional Neural Net (CNN) which would utilise convolutional and subsampling layers with stride and padding, in conjunction with linear layers. It is likely that effective utilisation of CNN architecture for text classification would involve a deeper network that would require greater complexity to train, thus we did not utilise CNN in this scenario.

Validation set and metaparameters

To avoid overfitting, a validation set was utilised. Splitting the data into a training set and a validation set prevented the network from hyper-specifying to the data and allowed us to test the performance of the model at the end of each epoch. It was found that a split of 80% training and 20% validation data (plus 20% dropout) to be optimal. Higher (0.9-0.99) splits appeared to overfit to the training data and lower splits (0.5-0.6) appeared to over generalise, resulting in a decrease in accuracy in both cases.

Through multiple testings, a batch size of ~16 and ~8 epochs appeared to perform optimally during training, achieving high accuracy with reasonable speed. A learning rate of 0.05 and momentum of 0.8 with weight decay were other parameters used in learning this dataset.

Results

```
Rating incorrect, business category incorrect: 0.70%
Rating correct, business category incorrect: 13.05%
Rating incorrect, business category correct: 4.26%
Rating correct, business category correct: 81.99%

Weighted score: 85.42
```

Conclusion & other considerations

Neural networks utilising LSTM and GloVe vectors have sufficient complexity to decipher meaning and predictively classify unstructured business reviews. Our results show that multilabel and multiclass text classification can be achieved with relatively simple recurrent nets.

Further opportunity to improve the accuracy of the text classification exist with the use of lemmatization, however this was not utilised to avoid the code downloading from external sources.