Q6 Data Augmentation in Natural Language Processing

Author: Jiyao Wang 20797324

Date: 2021.12.13

Task1, as I know, there are these possible ways to do augmentation:

- Easy Data Augmentation (EDA) operations: synonym replacement, word insertion, word swap and word deletion. Some easy-implement ways to do augmentation, for example, word swap is randomly swap some words in one sentence and generate label as its original one, which is also what method I opt.
- Back translation, it means that, for each sentence with label we want to generate, we translate it into another language and then translate it back. So there are some difference between two sentences.
- GANs, it utilize some generative adversarial networks to generate samples. It firstly learns distribution of the class we want to do augmentation, then generate sentences by it.

Task2

I use both structed data and text by prepocessing on them. For text, I use word2vec model in gensim to do pre-train on all of text to get word dictionary to transfer each word in sentence into digits and their initial embedding by CBOW model. For structed data, I transfer discrete features into unique id for following model. For the detail embedding matrix, can refer to my notebook.

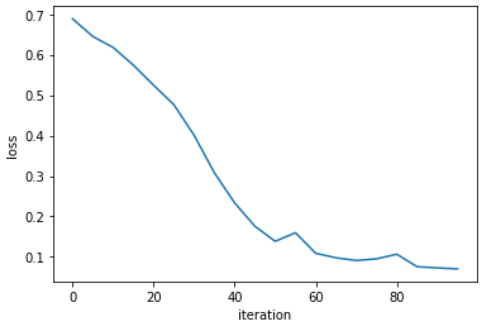
For augmentation, I randomly replace 3 locations in sentence with stop word id 0 until I get enough generated samples.

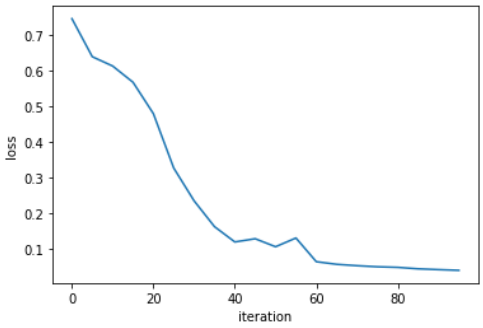
```
                                        -------Deal with struct feature
                                        -------Word2vec begin
                                        -------Finish Transfer, Saving
Number of positive samples = 3271       Number of positive samples = 3571
Number of negative samples = 4342       Number of negative samples = 4342
```

Following is the structure of my model, it is a NNs model based on BiLSTM. It plus a extra linear embedding layer and linear layer to combine features from structed data.

```
BiLSTMPlus(
  (emb): Embedding(5889, 128, padding_idx=0)
  (emb_struct): Embedding(4681, 128)
  (lstm): LSTM(128, 128, num_layers=2, batch_first=True, dropout=0.2, bidirectional=True)
  (L1): Linear(in_features=256, out_features=128, bias=True)
  (L2): Linear(in_features=384, out_features=128, bias=True)
  (L3): Linear(in_features=128, out_features=2, bias=True)
)
```

I set optimizer as Adam, and loss function is CrossEntropy and I split train data into train and validation data. To compare the performance with or without augmentation, I set two independent experiments, following is the learning curves of them:



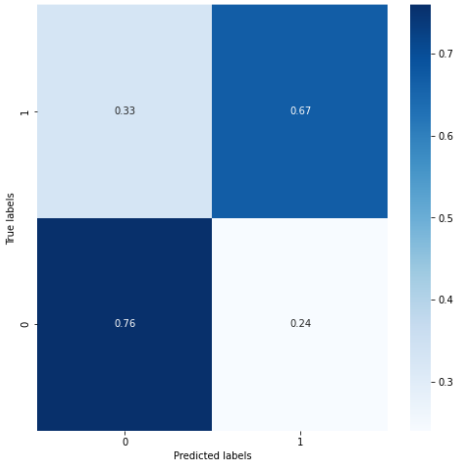


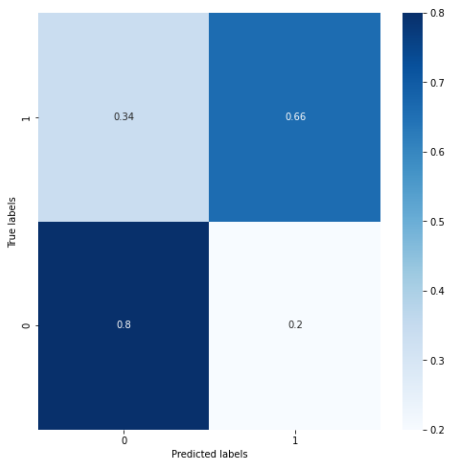
with augmentation without augmentation

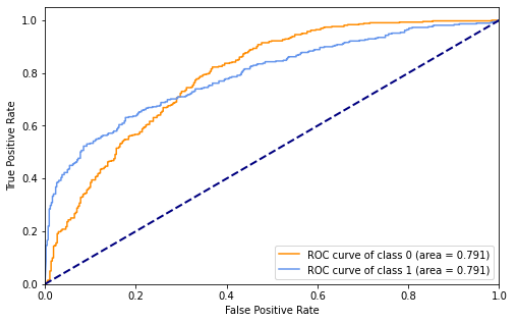
Following are results of them on two classes, on Accuracy, Recall, F1 Score and AUC Curve:

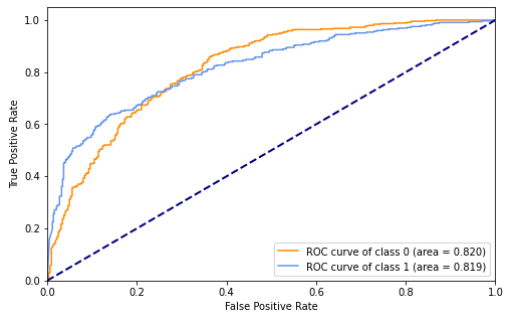
	precision	recall	f1-score	support
0	0.73	0.76	0.74	423
1	0.71	0.67	0.69	369
accuracy			0.72	792
macro avg	0.72	0.71	0.71	792
weighted avg	0.72	0.72	0.72	792

	precision	recall	f1-score	support
0	0.76	0.80	0.78	438
1	0.71	0.66	0.69	324
accuracy			0.74	762
macro avg	0.74	0.73	0.74	762
weighted avg	0.74	0.74	0.74	762









with augmentation without augmentation

From above results, we can conclude that augmentation is not always good for performance. It should depend on label distribution and feature data type.

Reference:

[1]https://research.aimultiple.com/data-augmentation/#:~:text=Data%20augmentation%20is%20not%20as%20popular%20in%20the,replacement%2C%20word%20insertion%2C%20word%20swap%20and%20wor