MATH 4995 Mini Project 2
Limitations of Translation: How much translation affect the analysis of Chinese text in different models?
By Ho Chuen Ho, Chun Lok Him Brian3`QW

Introduction

Sentiment Analysis is one of the topic that are developing in the Natural Language Processing field. In this project, we would like to look at different models that would be used to tackle such problem. We would use both customized model and pretrained model from Chinese-BERT-wwm to train models. Moreover, we would use model on the same type to train original Chinese text and translated text to see the difference in prediction accuracy. Our hypothesis is that Chinese text prediction would be noticeably higher than translated text on each of the model.

About the Data

We would be using the weibo_senti_100k data, which contains 100k reviews from Weibo. The data contains 2 column: review and label. Among those data, roughly half of the data contains positive review (label = 1) and half of them are negative (label = 0). All reviews are primarily Chinese text.

Preprocessing

The first step of the preprocessing is to create machine translation of the reviews. In our project, we used Google Translate as the tools on translation.

Moreover, in the dataset, 80% of each of the positive and negative review would be used as the training data and the remaining 20% would be used as the testing data. For customized model, non-translated text would be treated character by character, while translated text would be processed according to tokenization of nltk.

In text preprocessing, we have another technic called term frequency–inverse document frequency (TF-IDF) which is a numerical statistic that is intended to reflect how important a word in the dataset.

Term frequency, tf(t,d), is the frequency of term t.

$$\mathrm{tf}(t,d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}},$$

the number of times that term t occurs in dataset d.

DIY Model – Naïve Bayes

Naïve Bayes is constantly used as an example on predicting if a mail is a scam, which could be considered as an tool related to NLP. Therefore, it might be worth using the model to train the model. Unlike traditional computation where the probability is calculated, to speed up computation given label is balanced, we choose

$$S_+ = \prod_{i=1}^{\#unique\ char\ in\ test} count(c_i|train, label = 1)$$

$$S_- = \prod_{i=1}^{\#unique\ char\ in\ test} count(c_i|train, label = 0)$$

$$\underset{sign}{\operatorname{argmax}}(S_+, S_-)$$

It can be done and still be considered as bayes algorithm because:

$$P(x|features) = \frac{\left(\prod_i^{\#feature} \frac{P(feature,x)}{P(x)}\right)}{P(features)}$$

With $P(x = 1) = P(x = 0) = 0.5, P(x)$, $P(features)$ will be the same regardless of class and will not affect the outcome in argumentative max function.

For English text, token is used instead of character. The calculation is almost identical, except that instead of using character as feature, each individual token is feature instead. While the prediction score is not the best, it is worth noting that the prediction score using English translation is better than the Chinese text contrary to our hypothesis and is the only model we discuss in this report to have such attribute.

	0	1
[	113253	107147
鼓	1092	12346
掌	1037	12245
]	113218	107125
/	110077	101009
T	2	1
嬲	2	1

Score of some character in KNN

	0	1
[	107958	105508
Applause	720	10116
]	107341	105072
//	30309	27491
@	76230	88879
illogically	2	1
ventilated	2	1

Score of characters in Bayes

True\Pred	0	1
0	11660	339
1	4669	7330

Confusion Matrix (Chinese)

True\Pred	0	1
0	11769	293
1	3800	8199

Confusion Matrix (English)

DIY Model – KNN

KNN is also used on the prediction. Traditionally, KNN in NLP will try to find closest similarity of one review and compare with all review in training set. However, instead of calculating the closest distance between review, we tried to score the coordinate of label and coordinate of the testing review and to catch if the review is closer to positive or negative side. One noticeable advantage is that it will increase computational speed, as the complexity of prediction is $O(nf(x))$ per each review, with $f(x)$ is the similarity function.

The scoring of each character is done by the following formula:

$$S_{i,x} = \sum^{reviews} \frac{count(i\ in\ review|x)}{length(review)}$$

The coordinate would be $count(S_i)$ dimension. In the testing phase, the distance would be scored in binary for each dimension, i.e.

$$D = \sum (I(char\ in\ review) \times S_i)^{norm}$$

where I is indicator function.

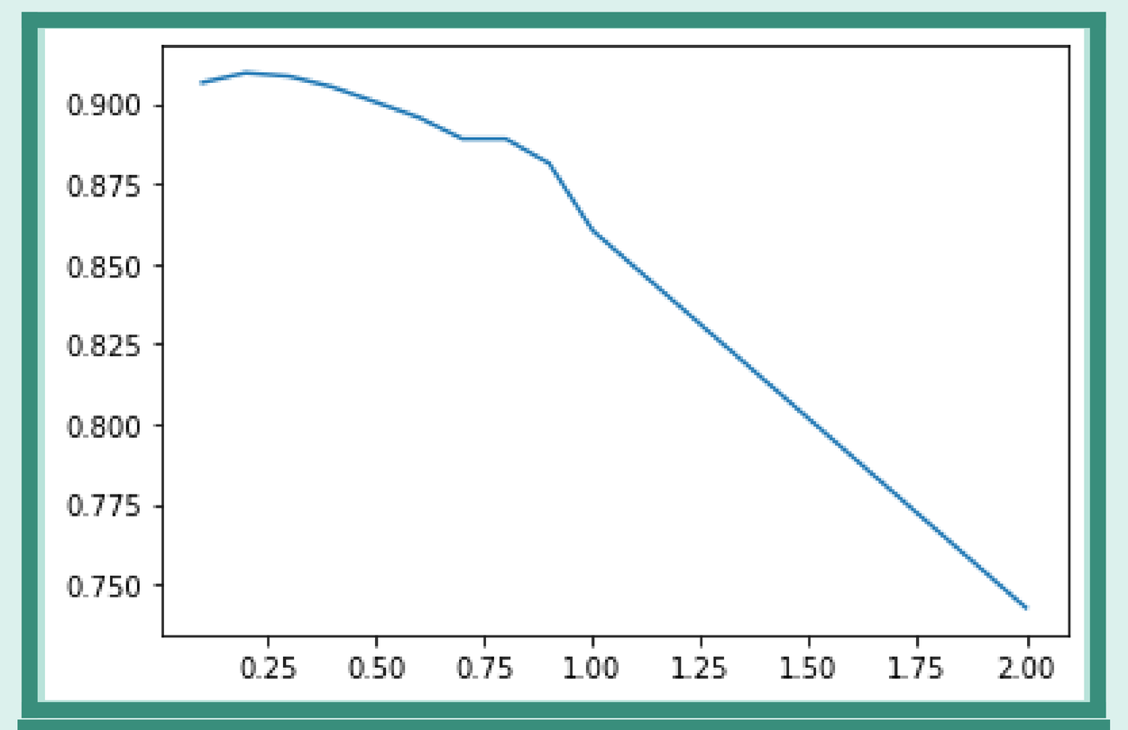
Moreover, different norm is tried on this formula (From 0.1 to 1, as well as Euclidean norm). It is found that a 0.2-norm works best for Chinese character.

Below are some of the result of testing accuracy of different norms.

norm	0.1	0.2	0.3	0.4	0.6	0.8	1.0
Chinese	.907	.910	.908	.905	.896	.881	.860
English	.682	.660	.630	.616	.585	.564	.650

	0	1
[	2432	2357
鼓	14	296
掌	14	295
]	2432	2357
/	1541	1372
T	0.008	0
嬲	0.008	0

Score of some character in KNN



Testing accuracy against norm

BERT Model

Bert is a state-of-the-art pre-trained model in Language model, which has 110M trainable parameter, 12 layers of transformer block

Bert is trained to fill word in sentence.

```
Input tokens : ['[CLS]', '今', '晚', '的', '月', '色', '真', '[MASK]', '。']
Translate: The moon tonight is so beautiful (BY google)
-----------------------------------------------------
Top 1 (79%) : ['[CLS]', '今', '晚', '的', '月', '色', '真', '美', '。'] ...
Top 2 (18%) : ['[CLS]', '今', '晚', '的', '月', '色', '真', '好', '。'] ...
Top 3 ( 0%) : ['[CLS]', '今', '晚', '的', '月', '色', '真', '佳', '。'] ...
```

Example of Masking, training BERT to learn representation subspaces

Attention between words in BERT

Bert has a multi-head Self-attention Mechanism which allows words in sentence focus on different context, different position and different layer. Then, contextual word representation could be learnt.

```
Dictionary Size 21128
token                 index
----------------------------
##蒽                   18977
##波                   16854
##钿                   20243
亻                       590
jj                    11095
鉻                      7064
评                      6397
##顽                   20616
##克                   14103
##遁                   19932
```

Bert Dictionary

```
name              module
--------------------
bert:embeddings
bert:encoder
bert:pooler
dropout           Dropout(p=0.1, inplace=False)
classifier        Linear(in_features=768, out_features=2, bias=True)
```

Modules in BERT Model

3 types of Bert variant models in Chinese and English

Full – Bert Model with full dataset

Reduced – Bert model with text less than 30 words (Chinese) or 150 words(English)

Emoji – Bert model with extracted emotional text

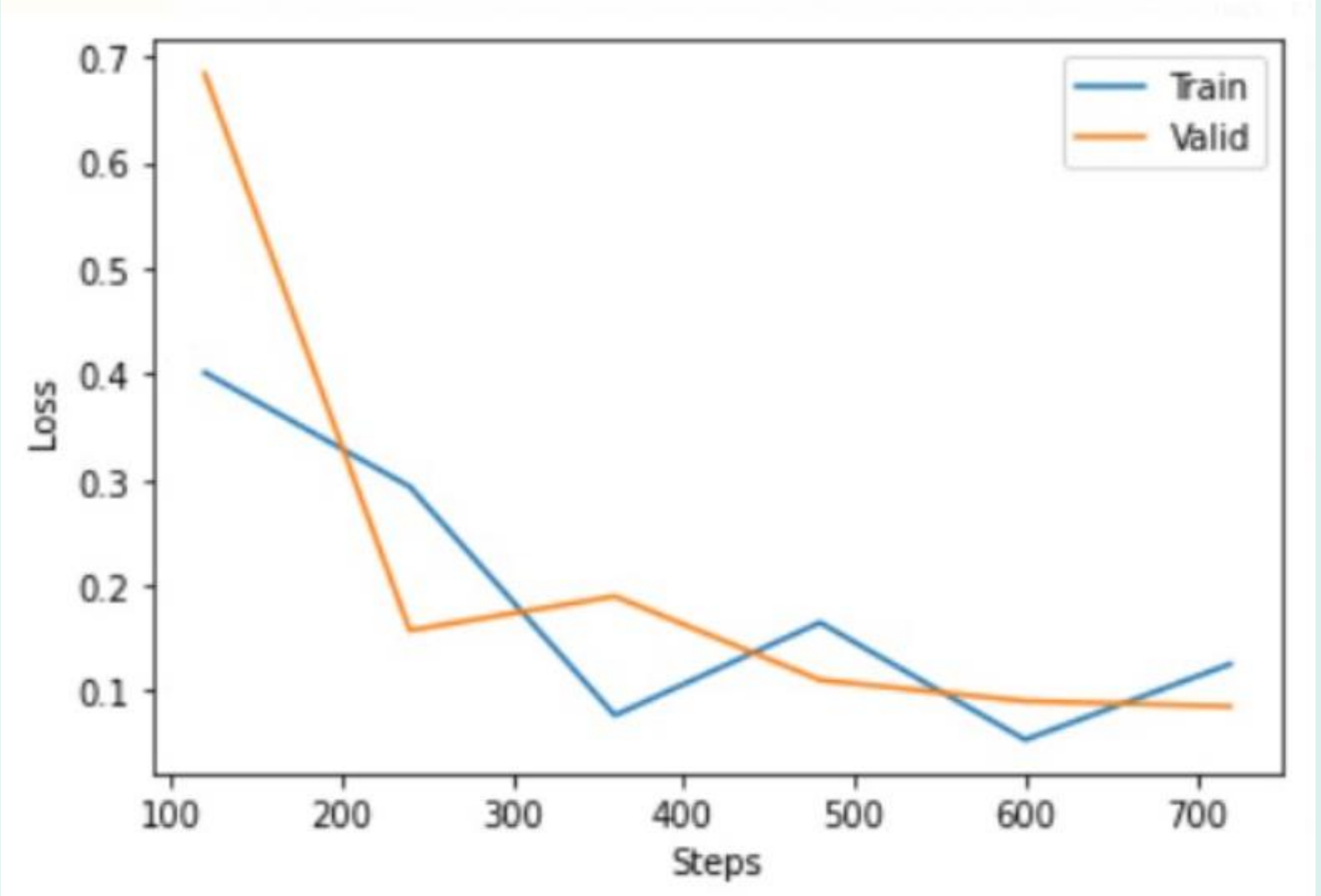


Figure: Loss Graph of BERT Model of Reduced Dataset

Conclusion

1. From processing of customized model, tagging infer positive reviews.
2. All models perform better in Chinese, while Naïve Bayes perform better when words are translated. (Could be related to tokenization.)
3. KNN is perform best in 0.2-norm among all tested norms.
4. From the bert model result, we can con concluded that emoji text is infomative data

Result

Model	Bayes	KNN(1)	KNN(2)	Full	Reduce	Emoji
test	0.7941	0.8624	0.7462	0.829	0.836	0.980
train	0.7913	0.8604	0.7426	0.943	0.946	0.983
engTest	0.8692	0.6523	0.5271	0.852	0.966	0.971
engTrain	0.8294	0.6497	0.5272	0.918	0.974	0.981

Contributions

Ho Cheun Ho: BERT

Chun Lok Him Brian: Translation, Naïve Bayes, KNN

References

Cui, Y. (2019). Chinese-Bert-WWM: Pre-training with whole word masking for Chinese bert（中文bert-wwm系列模型）. GitHub. Retrieved November 14, 2021, from https://github.com/ymcui/Chinese-BERT-wwm. jinhuakst. (2018, April 2). Chinesenlpcorpus/intro.ipynb at master · sophonplus/chinesenlpcorpus. GitHub. Retrieved November 14, 2021, from https://github.com/SophonPlus/ChineseNlpCorpus/blob/master/datasets/weibo_senti_100k/intro.ipynb.