AutoNLP Report

Xinneng XU

23/12/2019

1 Introduction

I chose the challenge of AutoNLP which is *Automated Language Processing for Text Categorization*, click Here to access to the challenge. The object of this challenge is to classifier the texts to classes. In this small report I will explain what I have done in this challenge.

2 Data Processing

In order to well choose the model that fits the dataset, I have visualized the datasets(There are 5 datasets), but here because of limitation of space and time, I only choose the first dataset as an example. The data processing steps are as follow:

- The data has many punctuations, I can replace it by space.
- There is not valid set, so we will split the whole dataset into transet ans valid set to better generate the training and hyper-parameters.
- Each observation is a sentence which is hard to translated into presentation, I will split it to tokens.
- I will convert data to Pytorch tensors so they can be used in a neural network. To do that, you must first create a dictionnary that will map words to integers. Add to the dictionnary only words that are in the training set

I want to see whether the dataset is balanced or not. It is good that figure 1 shows the dataset is balanced.

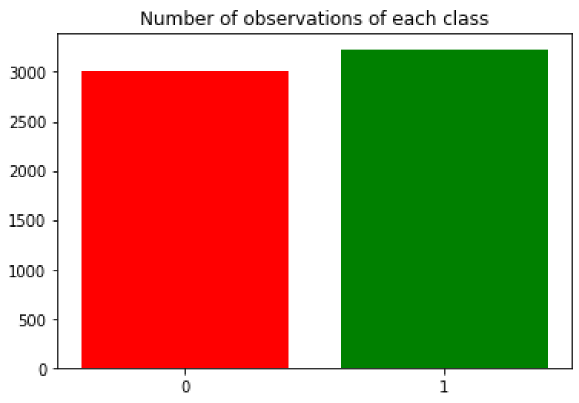

Figure 1: The number of observations of each class.

On way to help us the choose the model is look at the length of sentences of the dataset. Figure 2 shows the dataset has more long sentences which means we can use *LSTM* or *GRU*.

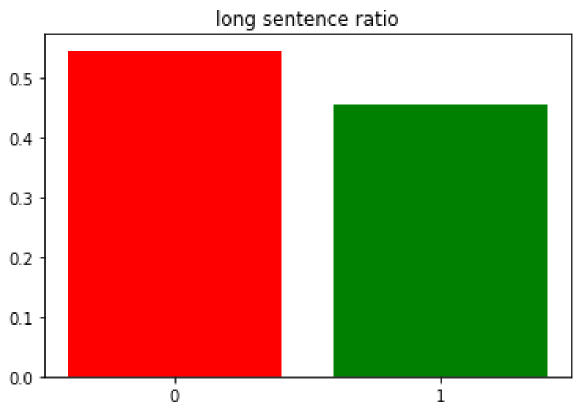

Figure 2: The ratio of sentences of which the length is bigger than 15. In the x-axis, 0 means that the length of sentence is bigger than 15, 1 is inverse. There is 56% sentences of which is bigger than **15**, while 44% is less than 15.

3 The model

Encoding. If I use *One-hot* encoding, explicitly computing the matrix multiplication is expensive, I need to use embedding tables instead.

I have tried 3 models in my head: **N-GRAM MODEL**, **BAG-OF-WORDS MODEL** and **LSTM**. I have given up **N-GRAM MODEL** because it has limitations: 1.The output layer is expensive to compute. 2.It has limited context, can't handle no long range dependencies. Section *Result* talk about their performance.

4 Result

This section shows the performance of **BAG-OF-WORDS MODEL** and **LSTM**. Table 1 shows that **LSTM** is much better than **BAG-OF-WORDS MODEL** on dataset **DEMO**(default test dataset). And it has almost the same performance in long sentences dataset. While **BAG-OF-WORDS MODEL** has much worse performance in long sentences dataset than in short sentences dataset.

Model	BAG-OF-WORDS MODEL	LSTM
Accuracy	58.2%	72.4%
Accuracy on short Sentences	62.4%	72.6%
Accuracy on long Sentences	54.9%	72.2%

Table 1: Result on DEMO dataset. Short sentence means that its length is less than 15. The accuracy is evaluated on the splited dataset(25%) from the whole DEMO dataset.

5 Problems

When I tested the lstm model in local, I got figure 3. It only has a peak at the end of timestamp. I searched in the website of **AutoDL** and I found **the reason** is that: *at each timestamp t, we computethe normalized AUCof the most recent prediction. ALC gives a stronger importance to predictions made at the beginning of the learning curve.* The function test in my model will read the saved model, and the model is saved only after all training epochs is finished. And the model has many parameters which results in very long training. These cause good accuracy but very bad ALC. My **solution** is that reduce the parameters of the network and save model after each training epoch. And save the model which has the best AUC in val set. Figure 4 shows the result with my solution applied. It works, but it is still not good. Perhaps I should change another method to solve the problem(checkpoint). And the training method needs to be optimized, the complexity of network should be considered.

I should also try **Gated Recurrent Units (GRU)** which is less complexity than LSTM, and CNN.

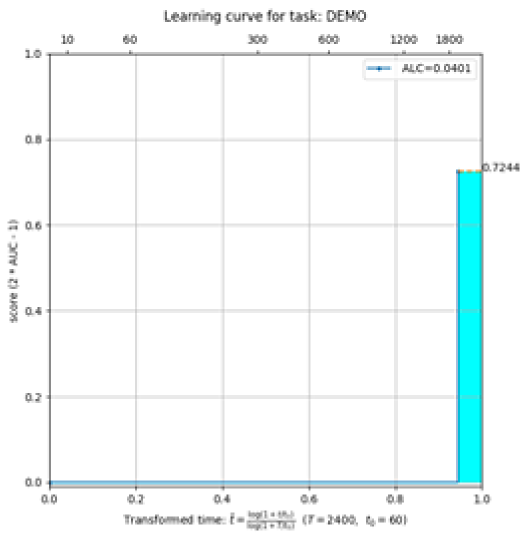

Figure 3: ALC figure on dataset **DEMP**.

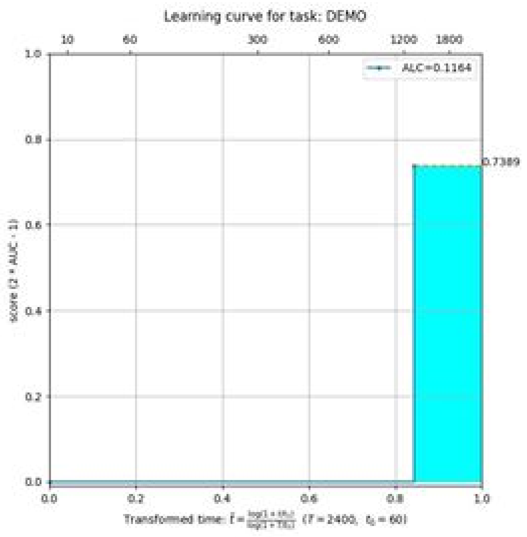

Figure 4: ALC figure on dataset **DEMP**.