Review: Bag of Tricks for Efficient Text Classification

Chen Yuan

Introduction

Text classification is a classic topic in NLP where predefined categories are assigned to free-text documents. However, the challenges remain for large data sets. Deep learning models are very popular and perform well but tend to be slow in training. Therefore, the authors of the paper proposed a simple text classifier that can perform as well as deep learning models, but significantly faster.

Model Archetecture

A baseline classifer for text classification is to use the bag of words model to represent sentences and train a linear classifier, where each word is considered as a feature. However, this does not allow parameter sharing among features and classes. Common solutions includes facorize the linear classifier into low rank matrices or using multilayer neural network.

Fasttext provides a simple solution. The model architecture of fasttext is very similar to CBOW model, where both models have three layers: the input layer, the hidden layer and the output layer. While instead of the middle word, fast text output the probability distribution of the labels.

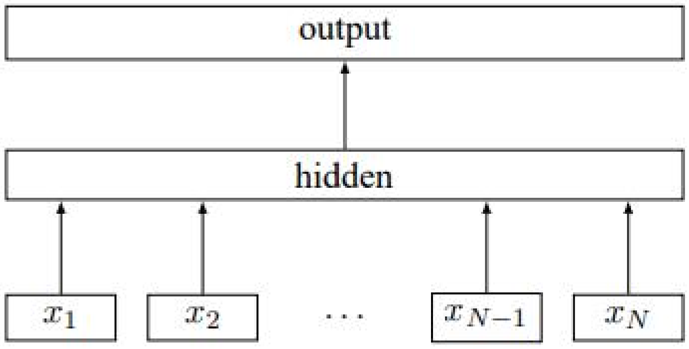

Figure 1: Model architecture of `fastText` for a sentence with N ngram features $x_1, \ldots, x_N$. The features are embedded and averaged to form the hidden variable.

Figure 1: Model Architecture of Fasttext (Joulin, 2017)

Firstly, it uses a weight matrix, which is a look-up table of word representation as the input layer. Then the input layer is input into one simple neural network model, which is the hidden layer. The output of the neural network is an averaged word representation, which can be considered as a text representation. Afterwards, the text representation is input into a linear classifier to calculate the probability distribution of the labels, which gives the output layer.

However, when the number of labels is huge, softmax can be computationally expensive. Therefore, fasttext uses the hierarchical softmax, which is based on the Huffman coding tree. The computational complexity dropped to $O(h\log_2(k))$ from $O(kh)$, where k is the number of classes and h is the dimesion of the text representation.

Hierarchical Softmax

A hierarchical softmax uses a tree structure in place of the softmax function, where each class is represented as a leave of the tree. A Huffman coding tree is built based on word probability or frequency. Each node contains the probability of travelling from the root through inner nodes to that node. The probability of travelling to the left or the right node is calculated using a sigmoid function:

$$P(node, left) = \sigma(v_n^T h) = \frac{1}{1+e^{-v_n^T h}}$$
$$P(node, right) = \sigma(-v_n^T h) = \frac{1}{1+e^{v_n^T h}}$$

where h is the output value from the hidden layer and v_n is the output vector of each node.

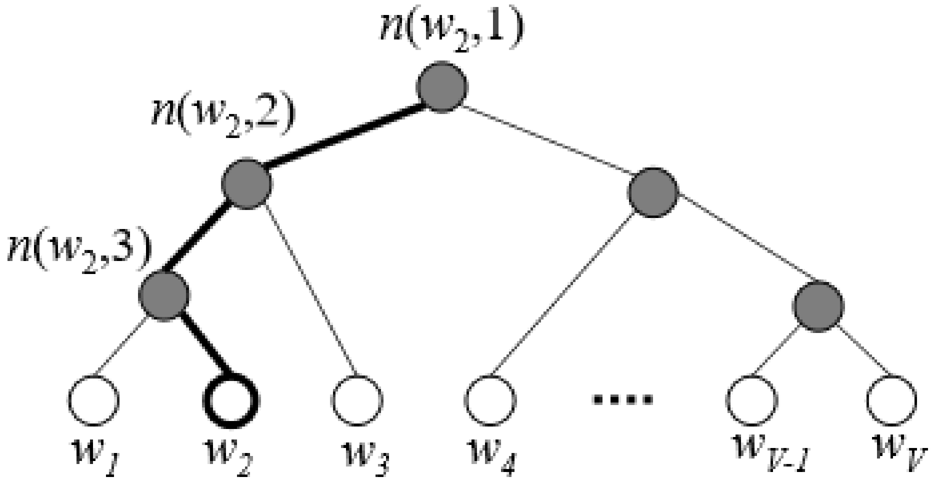

Figure 2: Hierarchical Softmax (Oliinyk, 2017)

To calculate the probablity of label w_2, we need to calculate the product of probabilities of intermediate nodes from the root node to w_2, which is

$$\begin{aligned} p(w_2) &= p(n(w_2,1), left) \cdot p(n(w_2,2), left) \cdot p(n(w_3,3), right) \\ &= \sigma(v_{n(w_2,1)}^T h) \cdot \sigma(v_{n(w_2,2)}^T h) \cdot \sigma(-v_{n(w_2,3)}^T h) \end{aligned}$$

N-gram features

A simple Bag of Words model does not take into account the word order which can be very useful information to text classification. Therefore, fasttext uses two different types of n-gram algorithm(zjrn, 2019), which are word-based and character-based as feature representation. However, the matrix can be sparse and therefore computationally inefficient.

In order to address this issue, fasttext imposes a hash trick on the n-gram representation to reduce the dimension of n-gram features.

Conclusion

Experiments have shown that fasttext can achieve a compariable accuracy in sentiment analysis and tag prediction compared with deep neural network, while being much more efficient. Fasttext is very suitable for training large datasets, while small datasets may cause overfitting.

Reference

[1] Oliinyk, H. (2017, December 11). Hierarchical softmax and negative sampling: short notes worth telling. Retrieved from https://towardsdatascience.com/hierarchical-softmax-and-negative-sampling-short-notes-worth-telling-2672010db
[2] Joulin, Armand & Grave, Edouard & Bojanowski, Piotr & Mikolov, Tomas. (2017). Bag of Tricks for Efficient Text Classification. 427-431. 10.18653/v1/E17-2068. [3] zjrn (2019, August 04). url:https://blog.csdn.net/ZJRN1027/article/details/98340304