Language Modelling using POS tags

CS 635: Web Search and Mining

Vaibhav Bhosale - 130050007
Aditya Kusupati - 130050054

Introduction:

Language Modelling has been keen to many applications like speech recognition, machine translation, or image captioning. It is in core fining **i^{th}** word when previous **i-1** words in a sentence are already known or seen.

This is the main algorithm in our suggestions on our keyboard while typing on smartphones. We used a new method as such to get a more robust version of predictive suggestions for the sentence completion in smartphone keyboards.

Problem Statement:

Given a part of the sentence, predict the **'k'** best next words as part of the suggestion to go ahead with the sentence. A sparse corpus might cause the standard language model to predict a word which might correspond to the sparse co-occurrences. Eg "I am ravishing" is the only sentence having a sentence structure of "I am _____" thus the prediction will likely be ravishing. But we know that it is not true most of the times so we shall use the predictive POS tag like the predicted word and incorporate POS signal along with the word signal to give out more fitting suggestions.

Corpus under study: Brown Corpus[1] (Train: 'News', Test: We are using 'Reviews and Editorial')

Proposed Solution:

We propose a novel idea where the predicted POS tag for the next word will compensate for the Language modelling shortcomings due to non-exhaustive corpus. The proposed solution has two parts:

1) Basic Language Modelling from Corpus
2) POS tag prediction

Basic Language Modelling from Corpus:

The core of the model consists of an LSTM cell that processes one word at a time and computes probabilities of the possible continuations of the sentence. It basically mimics the Language Modelling code given as part of TensorFlow[2]. The code was for PTB dataset, which was tweaked to accomodate Brown Corpus.

Given the machine capacity we have trained for a small configuration of hidden layers and other parameters. We shall use this model for the basic word prediction and use the top **'k'** words predicted from the vocabulary of the corpus. These predictions will contribute to the signal from the LM in the final model.

POS tag prediction:

POS tag prediction for the words has been well established, but futuristic prediction ie., POS tag of **'i+1'th** word (or placeholder) given POS tags of previous **'i'** words in the sentence. Surprisingly, this problem has never been looked into by the community(or rather we couldn't find relevant literature).

When looked at closely it reduces to typical language modelling problem where the vocabulary set is all the POS tags and the corpus used is a simple transformation of language corpus where words are replaced by their corresponding POS tags. This was an interesting observation in the project.

We train the LM on POS_transformed corpus and have the final model POS. The POS model will be used to predict the top **'k'** POS tags for the next placeholder(word). These predictions will contribute to the signal from POS in the final model.

Final Model:

Currently, the final model is pretty crude incorporation of the signals (predictions) obtained from LM and POS models. While we get the predictions we also have the scores/probabilities associated with them.

Each of the word given out as a prediction from LM has a POS tag corresponding to the word in the Brown Corpus. Prune out all the words in the predictions which have POS tags other than the ones predicted by the POS model. This will ensure words with weird POS/unrelated POS tags are removed. This works for the following reasoning: The number of distinct POS tags are 218 whereas number of distinct words in the corpus chosen 14394 => word>>POS tags. Ideally this implies a stronger learning on POS model when compared to the language model, thus relying on POS tags for pruning can be justified.

Given the pruned word predictions and existing POS predictions, we shall calculate a modified score of each word predicted with an influence of its POS tag. All the words have POS tag belonging to POS prediction so

New score of each word = (old score of word)*(score of its POS tag (score taken from the POS predictions))

If $\mathbf{W_i}$ is one of the words predicted and $\mathbf{P_i}$ is its POS tag from the corpus and $\mathbf{P_i}$ belongs to Predicted POS tags which have **PosScore** for each of them.

WordScore[$\mathbf{W_i}$] = WordScore[$\mathbf{W_i}$] * PosScore[$\mathbf{P_i}$]

Finally we have the updated score for each word prediction and we shall order them accordingly (descending order) and give out the predictions with the ranking obtained.

Entire implementation was in TensorFlow.

Problems Faced:

Being new to TensorFlow it took us decent amount of time to grasp it and get the things right. We are still not able to figure out a way to score a online stream of words ie., typing based input. We are hoping to get things right within next few days.

Results:

We have both LM and POS models trained and are trying to get better models by having deeper networks and optimal hyperparameters. For the basic small model of LM and POS we have perplexity scores as follows:

On using **News** as Train, **Editorial** as Validation and **Reviews** as Test we had perplexity of LM model ~ 500 (test)

Perplexity of POS model ~ 11 (test)

This clearly shows that POS model has more deterministic say over the prediction given the stability.

Larger models (are under training) are expected to get perplexities to around 200 and 5 respectively even lesser as claimed for LM by various reports online.

The final leg of evaluation is pending given the problem in handling online word stream as an input. But we have run the experiments on batch corpus and found that Ordering changes slightly and might have helped the predictions to be better. We are

not yet sure about exact metric on showing it to be better, but for the cases like Eg in the problem statement it started showing promising results by pushing up the scores of the other words predicted.

We need to do extensive experiments once the stream platform is ready. Given the constraint of vocabulary being from Brown Corpus is a bit tricky to handle manually so we are yet to automate it in the stream model.

By the time of hard deadline we intend to show promising results and given BTP presentation of Vaibhav on Friday, we shall be delighted if you can provide us with a chance of demo on Saturday (We shall inform when it is done).

Time Permits?

All we need is time.

References:

[1] http://clu.uni.no/icame/brown/bcm.html
[2] https://www.tensorflow.org/versions/r0.11/tutorials/recurrent/index.html