NLP Project Report

Detecting Homographic Puns

Arghya Bhattacharya
Sounak Pradhan
Tanmai Khanna

Task Description

In this project, we design a system which can detect homographic puns in a sentence and their location.

Concept

Several people have tried to solve this problem using Machine-Learning based approaches. However, we feel that they are too expensive and might be impractical for this area as puns use lexical-semantic anomalies in language for humour, and are hard to find in regular training sets. So, instead of creating training sets, it might prove fruitful to try knowledge-based and unsupervised methods.

There are two tasks in this project: pun detection, i.e. detecting if a sentence has a pun in it, pun location, i.e., detecting where the pun is in a sentence.

To solve this task, we make several assumptions:

1. A sentence containing a pun usually contains another distinctive word that can be paired with the pun.
2. Following from this, we also assume that this word's association with the pun is considerably greater than any other word in the sentence. For eg., I wanted to be a **banker** but I lost **interest.** Here, interest is the pun as it has two senses, and banker is the word that's associated to it much more than any other word in the sentence.
3. Generally the second word is the pun. Also usually at the end of a sentence.

Approach

1. Convert each sentence into tokens and remove the stop words.
2. We compute Word Association Scores using the Pointwise Mutual Information (PMI) measure.
3. All pairwise word associations are calculated and we determine the most related pair.
4. Now, we need to check if the most correlated for pair has an association score that is **distinctively** higher than the others. To do this, we use a global

threshold value to be used for the whole set of sentences

To get this threshold value, we use Interquartile range (IQR) of the set of sorted PMI scores in each sentence.

5. If an element of this most correlated word pair which is distinctively more associated than the other pairs has a second sense, it's an evidence of being a pun.
6. We assume that the second element of the pair with the highest PMI score in a sentence is the pun based on our assumptions which have been based on empirical observations.

Evaluation Results

1. For Homographic Pun Detection, our evaluation results are as follows:

```
precision 0.8181818181818182
recall 0.35283136278780336
F1-score 0.4930434782608695
```

Observations

1. The precision of the model is pretty high, which means that sentences which we do detect as having puns are mostly correct.
2. The recall however is quite low, which means that we haven't detected a lot of sentences which contain puns.

   Here are some examples of sentences which contained puns but weren't detected by the model:

- They hid from the gunman in a sauna where they could sweat it out .
- WalMart isn't the only saving place !
- Can honeybee abuse lead to a sting operation ?
- A ditch digger was entrenched in his career .
- Did you hear about the new pinata ? It's a huge hit .
- She was suspected of stealing a brooch but they couldn't pin it on her .
- There's room for one more , Tom admitted.
- They threw a party for the inventor of the toaster. And he was toasted .
- If you're a gardener you might call yourself a 'plant manager' .
- My advanced geometry class is full of squares .

There are several reasons why our model failed to detect these puns:

- In a lot of these, the pun word doesn't have to be direct, like the piñata example where the pun is that we usually hit pinatas. Similar reasons for the example with the brooch.
- The threshold may have failed to capture some examples, for eg., plant manager, sting operation, squares, etc.

Approach 2

While the task is mostly unsupervised, in this approach we assume it to be a supervised learning classification problem.

1. We select a part of the dataset and annotate them as pun or non-pun and try to focus on models that try to model sequences of word vectors. These word vectors are extracted from WordNet.
2. Each sentence is a sequence of word vectors which maps to one label at the end and this mapping lends itself nicely to Recurrent Neural Networks.
3. We used a RNN to train the classifier and generated the model.

This model takes care of the subtask 1, i.e. Pun Detection.

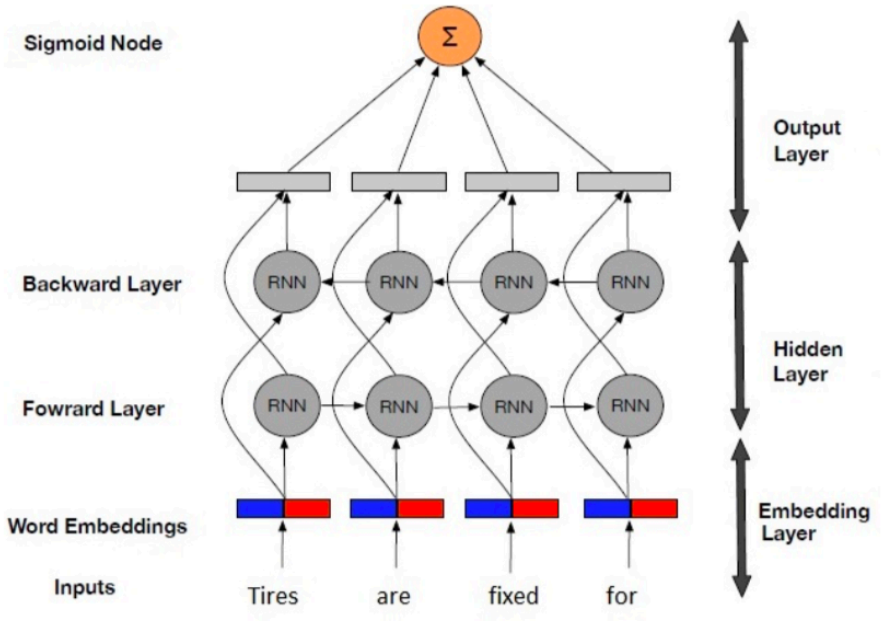

Figure 1: BiDirectional RNN architecture for detecting puns

For Pun Location, we do something similar to approach 1. Now we know that a sentence has a pun and must find the pun word.

1. In the sentence, all words that have less than two senses are removed. If only one word is left, that word is declared to be the pun word.
2. If there is more than one word, then a list of pairs with all combinations of words and the potential pun words is prepared and the pair with the maximum similarity is said to have the pun (same as approach 1).

Evaluation Results

1. For Homographic Pun Detection, our evaluation results are as follows:

```
precision 0.9197
recall 0.7363
F1-score 0.8178
```

2. For Homographic Pun Location, our evaluation results are as follows:

```
precision 0.5215
recall 0.5215
F1-score 0.5215
```

References

Ozge S., Nima G., Selma T. 2017. N-Hance at SemEval-2017 Task 7: A Computational Approach using For Association for Puns

Vijayasaradhi I., Orta Subba R. 2017. Fermi at SemEval-2017 Task 7: Detection and Interpretation of Homographic puns in English Language