CSE628 Project Update Report
05-20-2014

Sai Chaitanya Mallampati
Paavan Kumar Sirigiri

Title

GRE/SAT style Sentence Completion using Natural Language Processing Techniques

Problem Statement

As a part of this project, our primary goal is to automatically answer SAT/GRE style sentence completion questions using Natural Language Processing techniques. For this we have to build a model that will understand and deal with semantic coherence on a sentence level. Initially, we start out by applying Natural Language Processing Techniques on simple sentences. Later, we will see how these Natural Language Processing Techniques scale to complex analogy-based sentences.

Dataset Collection

Training Dataset - Initially, we obtained our training dataset from `http://research.microsoft.com/en-us/projects/scc`. This dataset consisted of 500 nineteenth century novels. While building the sentence completion models, we found this data to be inadequate and was yielding low prediction accuracies. In an attempt to get a larger corpus to train our model, we requested and got access to the Microsoft web n-gram corpus. This is available as a rest web service. `http://web-ngram.research.microsoft.com/info/` has more details about this corpus.

Testing Dataset - For testing dataset, we obtained a collection of 1040 sentences from the Sherlock Holmes novels - which is the same testing data used in Microsoft Research Sentence Completion Challenge. Since the answers to these 1040 sentences are already available, we use this as our gold standard. The testing dataset can be downloaded from the `https://research.microsoft.com/en-us/projects/scc`

Data Preprocessing

During pre-processing, a sentence completion question of the form

I have seen it on him , and could _______ to it .

A. write
B. migrate
C. climb
D. swear
E. contribute

is converted to the following format.

I have seen it on him , and could [write] to it .
I have seen it on him , and could [migrate] to it .
I have seen it on him , and could [climb] to it .
I have seen it on him , and could [swear] to it .
I have seen it on him , and could [contribute] to it .

All the 1040 testing sentences are converted to the above format and are stored in the questions file and corresponding answers are stored in the answer file.

All our models are run on this preprocessed data. While traversing through the question and answer files, we further pre-process the data by removing unwanted data like delimiters, extra spaces, numerals etc.

Standard Bigram Language Model

In the standard bigram language model, we construct our bigrams set by traversing through the training data. While testing, we read five sentences from the questions file at a time and construct bigrams from each sentence. Then, we calculate the bigram probability of the entire sentence. The option in the sentence with the maximum bigram probability is considered as the answer. Moreover, to deal with previously unseen bigrams, we used laplace smoothing. This model produced a prediction accuracy of **20.77 %** .

We then polished our training bigram set by removing all the bigrams which occured less than 20 times. On doing this, the prediction accuracy increased to **26.05 %**.

Bigrams Type	**Prediction Accuracy**
All Bigrams	20.77 %
Bigrams with frequency greater than 20	26.05 %

Standard Higher Order N-gram Language Model

We extended the above described bigram language model to a language model with trigrams. We also changed the training corpus to microsoft n-gram corpus. Also, we didn't calculate the probability for the entire sentence. We calculated only the probability of the trigram - W_i, W_{i-1}, W_{i-2} where W_i is the option word. Just like above, we calculated this probability with all the options and the option with highest probability was considered as the answer. This model resulted in a prediction accuracy of **44.33 %** .

Similarly, we calculated only the probability of the fourgram - $W_i, W_{i-1}, W_{i-2}, W_{i-3}$ where W_i is the option word. This model resulted in a prediction accuracy of **64.13 %**

Model Type	**Prediction Accuracy**
Trigrams	44.33 %
Fourgrams	64.13 %

Forward-Backward Probability

Backward probability is defined as the conditional probability that the target option word W_i will occur given the earlier word sequences $W_{i-1}, W_{i-2}, ..., W_1$.

Similarly, forward probability is defined as the conditional probability that target option word W_i will occur given the later word sequences $W_{i+1}, W_{i+2}, ..., W_n$.

Forward-Backward Bigram Language Model

Now, we construct a bigram language model using forward and backward bigrams as features. This was done based on the earlier work of Kyusong Lee & Gary Geunbae Lee, 2014 which is the present state of the art in sentence completion task. In this model, the forward and backward probabilities are summed up for each of the 5 anwer sentences and the sentence having the maximum probability among the five sentences for a given question is selected to be having the correct answer. Just like before, we used laplace smoothing for handling unseen bigrams.

When this model is trained on the 500 19^{th} century novels, we got a prediction accuracy of **26.25 %**. On changing the training data to the Microsoft n-gram corpus, the prediction accuracy increased to **30.12 %**

Training Data	**Prediction Accuracy**
500 19^{th} century novels	26.25 %
Microsoft n-gram corpus	33.26 %

Backoff N-gram Language Model

By taking into consideration, the superior prediction accuracy in the case of forward - backward bigram language model using microsoft n-gram corpus as training data, we decided to train all our further models on the microsoft n-gram corpus only.

We now extended the above discuessed forward - backward bigram language model to incorporate n-grams upto 4-grams. And for smoothing , we used a backoff model which backoffs to lower order ngrams when sum of the higher order ngrams probabilities are zero.

Using this model, we obtained a prediction accuracy of **79.42 %**.

On this model, we did several experiments -

Root Word - First of all, we incorporated the concept of root words into the model. For this, we used WordNet tool and the Java Wordnet Interface API to calculate the root word for every actual option word in all of the sentences. Then, the probabilities using the actual option word in the n-gram and the probabilities using the root word in the n-gram are summed up for each of the 5 anwer sentences and the sentence having the maximum probability among the five sentences for a given question is selected to be having the correct answer. In case the root word is same as the actual option word, the latter probability is taken as zero. This hybrid model, has a reduced prediction accuracy by roughly 2 % when compared to the above model. The prediction accuracy was observed to be **77.31 %**

Synonyms - The inclusion of synonyms instead of root words also had a similar impact of the prediction accuracy.

Weighted N-grams - Additionally, we also did an experiment on the backoff n-gram language model by giving a greater weights to higher order and smaller weights to lower order ngrams. We then use the weighted sum as feature for predicting the correct answer. When we gave weights of 4, 3 and 2 to fourgrams, trigrams and bigrams respectively, the prediction accuracy was **74.6 %**. When we altered the weights to 8, 3 and 1 for fourgrams, trigrams and bigrams respectively, the prediction accuracy increased to **76.92 %**.

Similarly, we also used non weighted sum of all order ngrams as features to develop a model. This model was used to predict correct answer just like above. The prediction accuracy was observed as **71.54 %**

Model	**Prediction Accuracy**
Backoff N-Gram Language Model	79.42 %
Backoff N-Gram Language Model + Root Word	77.31 %
Backoff N-Gram LM + Non Weighted Sum	71.54%
Backoff N-Gram LM + Weighted Sum [8-3-1]	76.92 %

Error Analysis

We also did error analysis to observe the specific cases which are failing and why they are failing. We made two observations -

1) We observed that identifying the word sequence occurring with high probability is not sufficient . Capturing the relationship between the target blank and the surrounding words in the sentence may help us to infer about the correct option to be filled in the blank in a better way. As an example we observe the following question which was wrongly answered by the model .

They seize him and use violence towards him in order to make him sign some papers to make over the girl's ______ of which he may be trustee to them.

A. appreciation
B. activity
C. suspicions
D. administration
E. fortune

For the above question the model assumed option d - administration to be correct while in fact e - fortune is the correct option. Although the model takes into account the frequency of the bigram "girls administration" for instance, it fails to take into account the presence of semantically related words in the sentence like"trustee" to give more weightage to the correct bigram "girls fortune" . We plan to take up this task of identifying semantic coherence in future work of the project.

2) Also we have seen that in cases where blank is positioned at first or last index of the sentence, the answer is predicted to be wrong consistently . An example of this kind of question would be

______ by nature, Jones spoke very little even to his own family members.

A. garrulous
B. equivocal
C. taciturn
D. arrogant
E. gregarious

This is because either backward or forward probability features are missing and remaining few probability features are unable to give weightage to the answer sentences with the correct option.

Latent Semantic Analysis

To deal with the cases in the above error analysis, we did latent semantic analysis to incorporate more semantics to our model. In this model, we built a matrix with word frequencies for each document. Since the matrix obtained is sparse, we considered words occuring only in the testing dataset as part of this matrix. We then performed singular valued decomposition on this matrix. After this, for each of the five canditate sentences, we calcualted the total word similarity (by summing up cosine similarities) for the target word with the rest of the words in the sentence.

Unfortunately, we couldn't implement this on the microsoft web corpus, as the rest web service returns only the probability of a word rather than the word frequency per document. We restricted ourself to the 19^{th} century novels dataset.

We obtained a prediction accuracy of 48.12% by the use of trigrams on the novels dataset which is a good 4% improvement compared to the trigrams run on microsoft n-gram web corpus. We believe that by implementing latent semantic analysis on a corpus as large as the microsoft n-gram web corpus, we will obtain a gain of around 3-4%

Summary

To summarize all our results thus far,

Classifier Type	Training Data	Prediction Accuracy
All Bigrams	500 19th century novels	20.77 %
Bigrams with frequency greater than 20	500 19th century novels	26.05 %
Trigrams	Microsoft n-gram corpus	44.33 %
Fourgrams	Microsoft n-gram corpus	64.13 %
Forward - Backward bigram model	500 19th century novels	26.25 %
Forward - Backward bigram model	Microsoft n-gram corpus	30.12 %
Backoff N-Gram Language Model	Microsoft n-gram corpus	**79.42** %
Backoff N-Gram Language Model + Root Word	Microsoft n-gram corpus	77.31 %
Backoff N-Gram LM + Non Weighted Sum	Microsoft n-gram corpus	71.54%
Backoff N-Gram LM + Weighted Sum [8-3-1]	Microsoft n-gram corpus	76.92%
Latent Semantic Analysis	500 19th century novels	48.12%
Latent Semantic Analysis	Microsoft n-gram corpus	> 80%

Observations

During this project we have come across many interesting observations. We listed some of them below -

1) In the standard bigram language model, removing the bigrams with frequency less than 20 increased the overall accuracy. This is because when we remove the not so frequent bigrams, the frequent bigrams have more say on the prediction.

2) In the standard language models, it can be observed that fourgrams gave a better prediction accuracy than trigrams and trigrams gave a better prediction accuracy than bigrams. This is because as the order of the language model increased, the context was captured in an effective manner.

3) In the forward-backward bigram language model, higher prediction accuracy was obtained when the model was trained on Microsoft n-gram corpus when compared to the case when the model was trained on novels dataset. This is because microsoft n-gram corpus had larger data resulting in less unseen n-grams.

4) In the backoff n-gram language model, it can be observed that using weighted sum and non-weighted sum as features has reduced accuracy of predicting the correct answer. This is because, in backoff model, we consider higher order n-gram probabilities and then backoff to lower order n-gram probabilities only if the higher order n-gram probabilities are not available. As the length of the sequence containing the actual option word is more, it is almost certain that the word may occur with same sequence of words. But in models using weighted and non weighted sum as features, giving weights to lower order Ngram probabilities in the initial stage itself can only increase probabilities of predicting wrong answers.

5) Using only standard trigrams or bigram probabilities as features have much reduced performance in terms of accuracy because they take into account very few n-grams when compared to the backoff n-gram language model.

6) To understand why the inclusion of the root word decreased the accuracy, let's take a look at the following example -

The animal has been ______ , and we have the length of its stride .

A. Sultry
B. Achieved
C. Baptized
D. Moving
E. granted

The model will now look for fourgrams like “animals has been move” rather than “animal has been moving”. Since the probability of the latter occuring is more for grammatical reasons, this model fails in such cases.

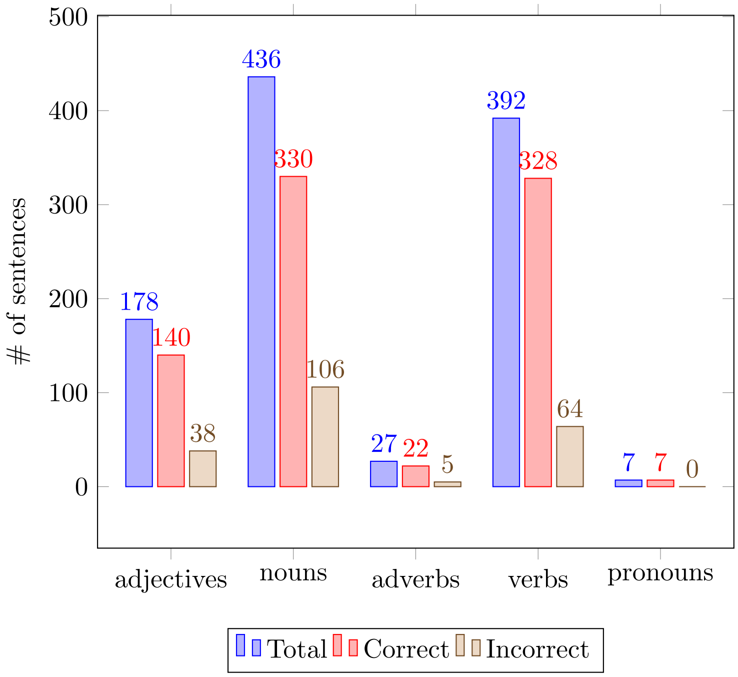
Insight 1

To gain further insights into our sentence completion model, we looked at how the model was working when the target word is a noun, pronoun, verb, adverb and adjective.

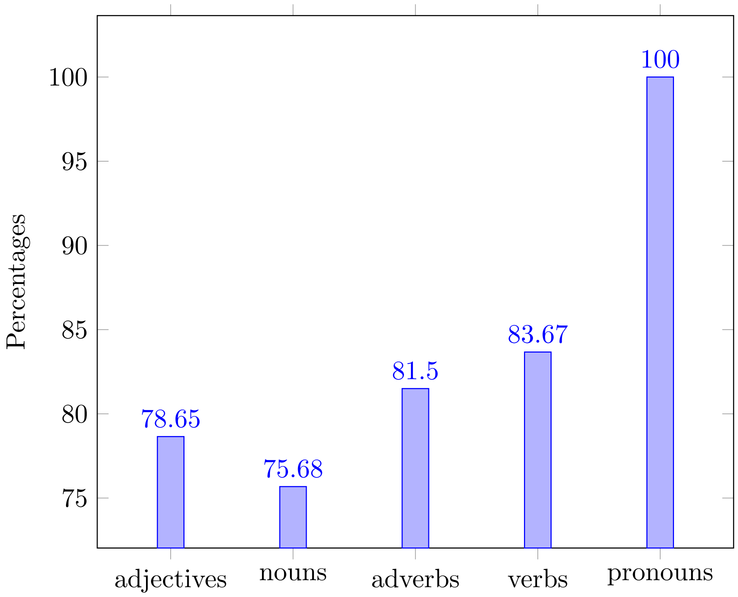
Out of the total 1040 test sentences,

1) 436 had nouns as the target word. 330 of them were correctly filled.
2) 392 had verbs as the target word. 328 of them were correctly filled.
3) 178 had adjectives as the target word. 140 of them were correctly filled.
4) 27 had adverbs as the target word. 22 of them were correctly filled.
5) 7 had pronouns as the target word. All of them were correctly filled.

The following histogram, shows the correctly and incorrectly filled out sentences for each parts of speech tag under consideration -



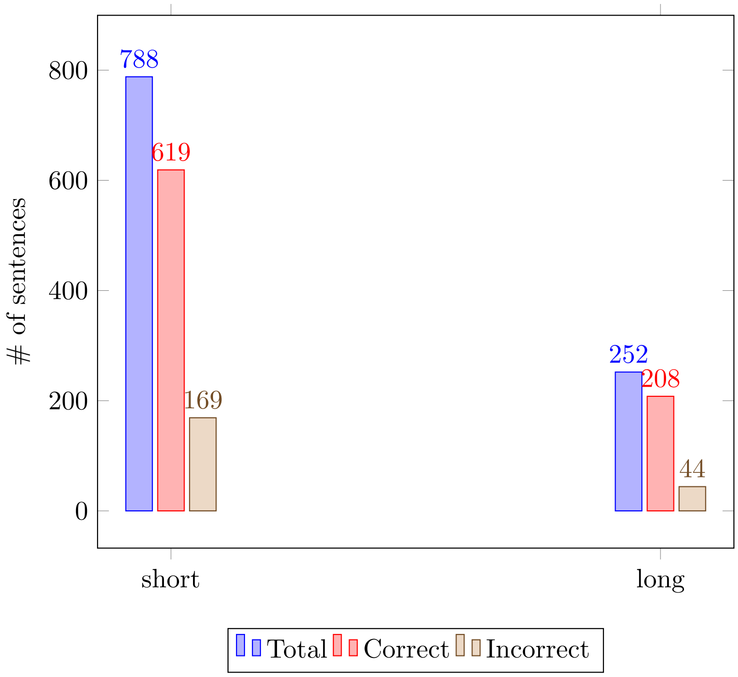
Now, we took a look into the prediction accuracy for each parts of speech tag. The following histogram shows the accuracies -



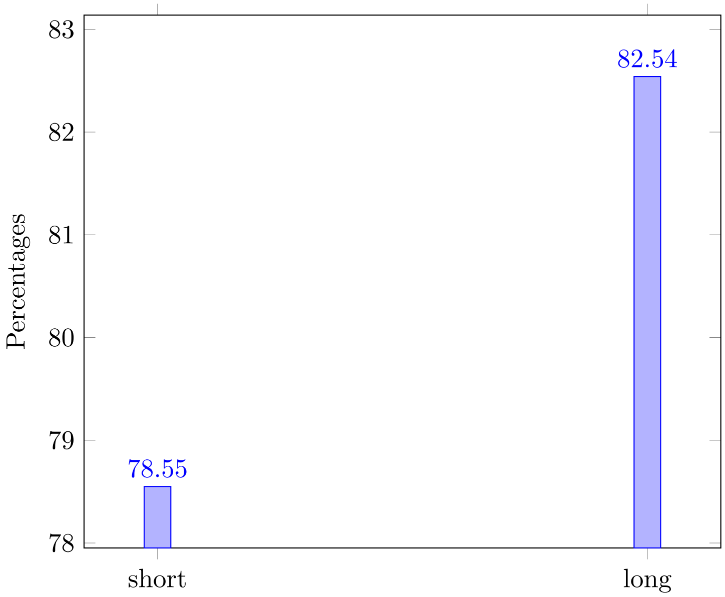
Although the distribution is assymetrical, from the above we can see that our model works best in the order - pronouns, verbs, adverbs, adjectives and nouns. Among the nouns, it was observed that our model is suffering mainly while filling out common nouns. As part of future work, we can build individual models for each parts of speech tag. Dealing with individual models might help towards sentence completion correctly choosing the target word.

Insight 2

Next, we checked how the sentence completion model worked with respect to small and long sentences. The following histogram illustartes this -



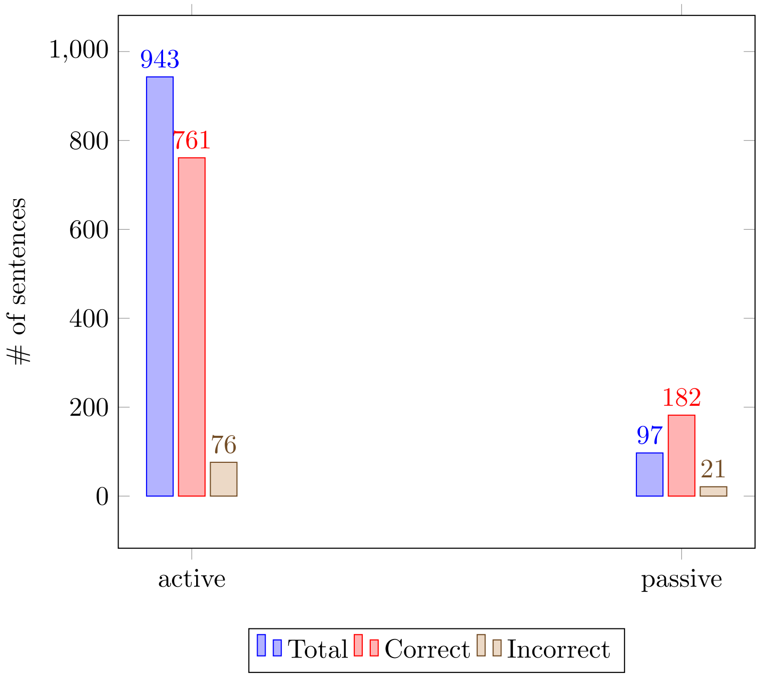
Now, we took a look into the prediction accuracy for each sentence length. The following histogram shows the accuracies -



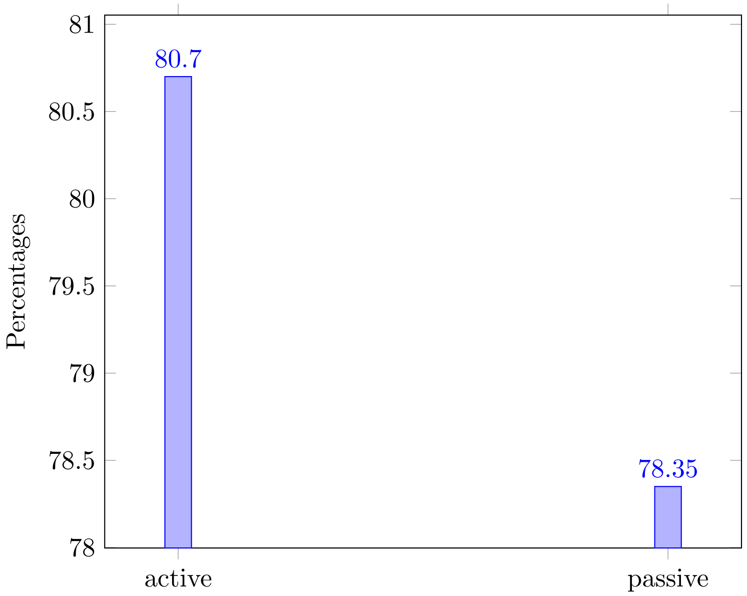
From the above graphs, we can conclude that our model is working better in the case of long sentences than in the case of short sentences. Our fourgram backoff model was able to grasp the context in a good way. We believe that the reason for the model not working so well in the case of short sentences is that the sentences didn't have sufficient context for our model.

Insight 3

Next, we looked at sentence completion task with respect to the subject location in the sentence i.e., based on the voice of the sentence (active or passive). Looking at the results obtained graphically -



Now, we took a look into the prediction accuracy for each sentence length. The following histogram shows the accuracies -



From the above graphs, we can conclude that the model works relatively in the same fashion for both active and passive sentences. We believe this is because of the fact that we considered both forward and backward n-grams while constructing the model.

Insight 4

There are a few sentences in the corpus which have a blank word in between of an idiom. Our model worked very well in completing such sentences. For example,

With these two facts in my possession I felt that either my intelligence or my courage must be deficient if I could not [involve] some further light upon these dark places .
With these two facts in my possession I felt that either my intelligence or my courage must be deficient if I could not [throw] some further light upon these dark places .
With these two facts in my possession I felt that either my intelligence or my courage must be deficient if I could not [cancel] some further light upon these dark places .
With these two facts in my possession I felt that either my intelligence or my courage must be deficient if I could not [earn] some further light upon these dark places .
With these two facts in my possession I felt that either my intelligence or my courage must be deficient if I could not [contain] some further light upon these dark places .

Our model did well to predict the idiom **“throw some light”**

References

[1] The Microsoft Research Sentence Completion Challenge - http://research.microsoft.com/en-us/projects/scc
[2] The Microsoft n-gram corpus - http://web-ngram.research.microsoft.com/info/
[3] Wordnet Tool - Christiane Fellbaum (1998). WordNet: An Electronic Lexical Database. Bradford Books.
[4] JWI API - Finlayson, Mark Alan (2014) Java Libraries for Accessing the Princeton Wordnet: Comparison and Evaluation. Proceedings of the 7th Global Wordnet Conference. Tartu, Estonia.
[5] Sentence Completion Task using Web-scale Data by Kyusong Lee et al. [2014]
[6] The Microsoft Research Sentence Completion Challenge by Geoffry Zweig et al [2011]
[7] Computational Approaches to Sentence Completion by Georey Zweig et al [2012]