Modern Approaches to the Sentence-Completion Task

CS224U Project Milestone

Bonnie Nortz (bnortz@stanford.edu)

Tyler Romero (tromero1@stanford.edu)

Abstract—**We provide an update on the state of our final project.**

Keywords—sentence completion, near-synonyms, SAT, GRE, fill-in-the-blank, PMI, LSA, RNNs, context, connotation

I. Goals of the Project

Sentence completion questions are designed to evaluate skilled reading ability. Skilled reading is a challenging problem that requires accurate parsing of the structure of sentences and the connotations of words. High school and college aged students are evaluated on their verbal reasoning ability through sentence completion questions on the Scholastic Aptitude Test (SAT) and Graduate Readiness Exam (GRE), respectively. For artificial semantic models, these types of questions can also serve a purpose, perhaps as a more accurate measure of their performance than current methods of evaluation such as perplexity, which does not always correlate with downstream success. And besides measuring overall progress, high performance on sentence completion tasks can prove a system's ability in subtasks of machine translation and natural language generation, where a system must choose the most appropriate words while paying attention to lexical nuances.

We have two major goals for our contributions to this field. First, to present a model that combines some of the best features from the papers we have reviewed; and second, to introduce novel features that we hypothesize will address some of the main downfalls of models evaluated on standardized test questions, and which might improve performance on more mainstream sentence-completion tasks.

II. Previous Approaches

Previous work in this area focused on three distinct but related tasks. All tasks involve sentence completion, meaning that the system is presented with a sentence that is missing one (or more) words or phrases, and then a set of candidates to fill in that missing space.

Some papers focus on a "near synonym" task, which means that all sentences to be completed are missing one single word, and all candidates presented are nearly synonymous with each other. These papers generally have a test set drawn from Wall Street Journal (WSJ) articles written in 1989, that has around 30,000 examples, with 7 different sets of near synonyms. These include sets like "mistake, error, fault", and "job, task, chore" [3], and for each test example, only one of the set of near synonyms is considered a correct answer.

Recently, Microsoft Research (MSR) released a sentence completion dataset and an associated competition [8], which includes 1,040 sentences with a single word removed, taken from *Sherlock Holmes* novels. For each of these, the system is presented with the sentence and a set of 5 candidate words, which are unrelated, and again it must choose the single word that correctly completes the sentence.

Finally, for any paper that involves completing SAT questions, these will usually be of the same format as the MSR sentences, with one word missing and 5 unrelated candidate words provided. However, the SAT also has sentence completion questions involving two blanks that must be filled at once; some papers explicitly say that they take on these types of questions, and for others it is unclear whether they do or do not. To our knowledge, there is no standard SAT question dataset, and each team that evaluated their model with one gathered their own set of questions.

A. A Statistical Model for Near-Synonym Choice (Inkpen, 2007) [4]

This paper explores three approaches to automatic sentence completion. The first, pointwise mutual information (PMI), is an information theoretic take on the problem. This paper also proposes using anti-collocations to eliminate incorrect answer choices. An anti-collocation is a combination of words that a native speaker would not use. This method ranks the strongest collocations higher than the anti collocations, and answers based on these rankings. Finally, the paper puts forth an approach using Naive Bayes and Boosted Decision Trees, where sentences are featurized using the 500 most frequently co-located words with the missing word.

Results were trained on the Waterloo Multitext System, and evaluated on the WSJ 1989 corpus.

B. Near-Synonym Choice Using a 5-gram Language Model (Islam and Inkpen, 2010) [2]

The researchers in this paper use the Google Web 1T n-gram corpus and a 5-gram language model, calculating the likelihood of a sentence using the number of times its constituent n-grams appear in the dataset. They perform some slightly more detailed statistical modeling with the n-gram counts to handle the interaction between different sized n-grams.

To evaluate the model, this paper uses the WSJ dataset of near-synonym sentence completion examples.

C. Near-Synonym Lexical Choice in Latent Semantic Space (Wang and Hirst, 2010) [3]

In this paper, a latent semantic space is built by using SVD to perform dimensionality reduction on a word-document co-occurrence matrix. This results in a vector representation for each word in the corpus. Then, using cosine distance, the nearest neighbor algorithm is run to identify the best word for the context, representing the context as a weighted centroid of its word vectors. They additionally build a supervised model: a support vector machine transforming the weighted context centroid into a classification over the candidate words.

These researchers train and evaluate on the same datasets as Islam and Inkpen, 2010.

D. Computational Approaches to Sentence Completion (Zweig et al., 2010) [7]

Working with both SAT questions and the Microsoft Research sentence completion dataset, this paper tests three types of models: n-gram language models, a recurrent neural network (RNN) language model, and latent semantic analysis (LSA). They also use an ensemble of these models to get their final results, as they hoped to make use of both the local information captured by the n-gram and RNN models and the global context captured by the LSA model.

To train their models for the SAT task, the researchers use data from the LA Times. As discussed above, the MSR challenge comes with its own training data.

E. Dependency Recurrent Neural Language Models for Sentence Completion (Mirowski and Vlachos, 2015) [6]

Looking to improve on previous deep learning models, this team combines RNNs with dependency information about the word being predicted and the surrounding sentence. In addition to using dependency structure information on its own, the model utilizes dependency label features as well.

The researchers also add a maximum-entropy model to the RNNs by creating a matrix that connects the n-gram context of an input word to the output word probabilities.

This paper used the data from the MSR sentence-completion challenge.

F. Exploiting Linguistic Features for Sentence Completion (Woods, 2016) [5]

This paper presents a model that builds upon the PMI model put forth in previous papers, using positive PMI (PPMI), and an extended feature set. In this model, context sentences are preprocessed by removing determiners, coordinating conjunctions, pronouns, and proper nouns. The remaining words were then added to a feature set. Words in the context that share a syntactic dependency with a candidate word are included in another feature set. Finally, sentence keywords were identified and placed into a feature set. The candidate word that achieves the highest similarity score is selected as the most likely answer.

Bigrams and trigrams were also incorporated into the feature sets described above.

This paper evaluates on both MSR and SAT data. For the SAT task, the training data was the New York Times portion of the Gigaword corpus.

G. Taking the SAT Exam: Sentence Completion (Busch et al., 2012) [1]

A former Stanford CS224U paper, the authors propose three models for answering SAT questions, generally using the New York Times section of the Gigaword corpus as training data. They build a trigram and 5-gram language model as well as one using PPMI, with the score of a candidate being the sum of all pairwise PPMI scores between words in the candidate answer and the rest of the sentence, assigning a pairwise PPMI score of 0 if they encountered any word not in their vocabulary.

Finally, they implement a dictionary method, where they use information from WordNet to help inform their choice. For each candidate word, they find its WordNet gloss, the words in its synset, and the words in any direct relation to it in WordNet. After stemming this set of words and those in the question sentence, they calculate the number of words that overlap in these two sets, and choose the candidate answer with the highest score.

The team also tried to combine the results from all of their methods by various means; however, none of these results out-performed their individual PPMI model.

H. Summary of Previous Methods and Results

Many papers we reviewed incorporated information from n-grams around the candidate word, realize that the local context of the missing word in the sentence contains a wealth of information about the word itself. That being said, four of these five groups attempted to combine this highly localized information with more global context, with PPMI, LSA, RNNs, and/or syntactic dependency information. PMI methods represent the current state of the art on the MSR and SAT datasets, both achieved by Woods (2016), who notably outperformed Busch et al. on SAT questions, presumably due to her additional feature engineering. It should be noted that the two test sets of SAT questions used by both papers were not identical, but they are similar in size, so we feel comfortable comparing the results.

One of the main challenges faced by sentence-completion models is dealing with obscure words. As noted by Busch et al. and Woods, answers to SAT questions tend to not appear frequently in the training corpus, making them difficult to deal with. To counter this, Zweig et al. had a specific feature in their LSA model that counts the number of terms in the answer that aren't in the vocabulary. We hope to address precisely this problem in our project, hypothesizing that it will improve our performance on standardized test questions, and curious to see how it will affect performance on the more mainstream MSR task.

III. Current Approach

A. Models and Features

As stated, we plan to combine many previously attempted approaches, as well as add some of our own twists to deal with issues that others faced.

Since PPMI previously produced state-of-the-art results, we plan for the base structure of one of our models to use that method. As for features, we plan to imitate Woods (2016) in that we will use the bag of words of the sentence, the words that share direct syntactic dependencies with the candidate word, and some bigram and trigram information. We will also try her method of preprocessing, namely removing certain parts of speech as stopwords. Additionally, we might add in larger n-grams, as other teams found that that worked to their advantage.

As for our own additions to the PPMI model, one major issue with answering SAT (and we expect, GRE) problems was that candidate answers were not in the model vocabulary. To address this (and to potentially increase performance on the MSR task), we want to incorporate Wordnet information. Busch et al. (2012) did this to some extent, but we propose approaching it a different way. While they calculated overlap between a set of words gathered from Wordnet and the words in an example sentence, we want to bring these types of features into a PPMI model. Particularly, we think that given a candidate word ***w***, the words in ***w***'s Wordnet synset, hypernym synsets, and potentially the words in its gloss might be useful features in the PPMI model.

We think this will improve models answering standardized test questions where candidate words do not exist in the training vocabulary and therefore have no co-occurrence information. And, even if a word exists in a vocabulary (e.g. Gigaword, which is enormous), it could be so rare that its co-occurrence values are very low, so we can potentially define some threshold that if a word is rare enough, we incorporate Wordnet synonym and/or hypernym features in order to give the system a better idea of what contexts this rare candidate word might belong in

In addition to incorporating these Wordnet features, we also plan to add features to make the PPMI model more sophisticated than using a simple bag of words scoring approach. First, we could weight the value of each word in the sentence by the number of words they are away from the candidate word. We could also use syntactic dependencies to weight the bag of words, instead measuring distance of a word to the candidate word by its distance when traversing the dependency tree starting at the candidate word.

Finally, we would like to incorporate the labels from dependency parsing somehow in the style of Mirowski and Vlachos (2015), but since those features do not fit neatly into a PPMI model, we have to work on how to leverage those. Also, we are writing baselines for LSA and GloVe vectors, and, based on their performance, we might try to incorporate our features into those models, and/or combine these models in an ensemble. These two types of models may capture more global sentence context and richer semantic information, respectively, than our PPMI model can capture on its own, so they might all complement each other well in an ensemble.

B. Evaluation

In terms of evaluating our models, simple accuracy is actually a faithful metric for single-blank sentence completion questions. For multiple-blank sentence completion questions, we can measure accuracy in multiple ways. We can count an answer to a question as correct if they get every part of the question right, or we can give partial credit. If we choose to compare multiple-blank questions with single-blank questions, we will have to decide which metric makes that comparison the most faithful and the fairest.

We are also considering looking at GRE sentence equivalence questions, which present a system with six candidate words instead of five, and ask the system to choose exactly two candidates that correctly complete the sentence and that make the sentence mean the same thing when completed. If we gather enough of these questions to have a significant dataset, it would be interesting to try different strategies on this type of problem. But if not, they could easily be converted to single-blank sentence completion questions by removing one of the correct candidate words.

Finally, for standardized test questions we can also evaluate performance based on how the actual test is evaluated, where questions are only correct if all blanks are correct, and incorrect answers deduct some fraction of points. To compare this type of result to real human performance on these tests, we would need to make sure the number of questions was comparable to actual standardized tests.

IV. Progress

A. Data Collection

There is no generally accepted dataset for this task; previous papers have used a variety of corpora.

We were able to obtain the MSR corpora, which consists of 500 books from the Gutenberg project as training data, and 1040 multiple-choice sentence completion questions as testing data. This will be particularly useful as a point of comparison, as many previous papers have used these exact datasets for training and testing.

In addition, we have collected questions from GRE practice tests, which tend to be more challenging as the vocabulary choices are more complex and questions can have multiple blanks. We currently have 25 questions that are a variety of single-blank, multiple-blank, and sentence equivalence questions, and plan to collect up to 100 for testing. Along the same lines, we recently obtained access to several old SAT exams, and we plan to scrape those exams to create another dataset.

For training data on the standardized test tasks, we have access to the New York Times portion of the Gigaword corpus that other teams have used, as well as other newspaper data in that corpus, including the LA Times. A slightly different dataset from the LA Times was used for training by one paper, so we think this one could be useful for us.

Finally, we have access to a corpus of Wall Street Journal articles as well and could feasibly create a similar dataset to the papers focused on near-synonymy if we choose to, but this particular task is less interesting to us than the standardized test task.

We are collecting a wide variety of corpora in order to give ourselves a basis for comparison with many different previous approaches. In addition, different corpora offer different challenges, even for the same task. For example, some corpora use more obscure vocab words, and others offer answer choices that are more difficult to distinguish from one another.

B. Baselines

We have implemented three different baselines: one based on PPMI, one based on LSA, and one based on GloVe. We applied these baselines to the MSR and GRE datasets described above. All results reported are evaluated with the strictest accuracy measures possible; so, for the GRE multiple-blank questions, all blanks must be correct for the question to be considered correct.

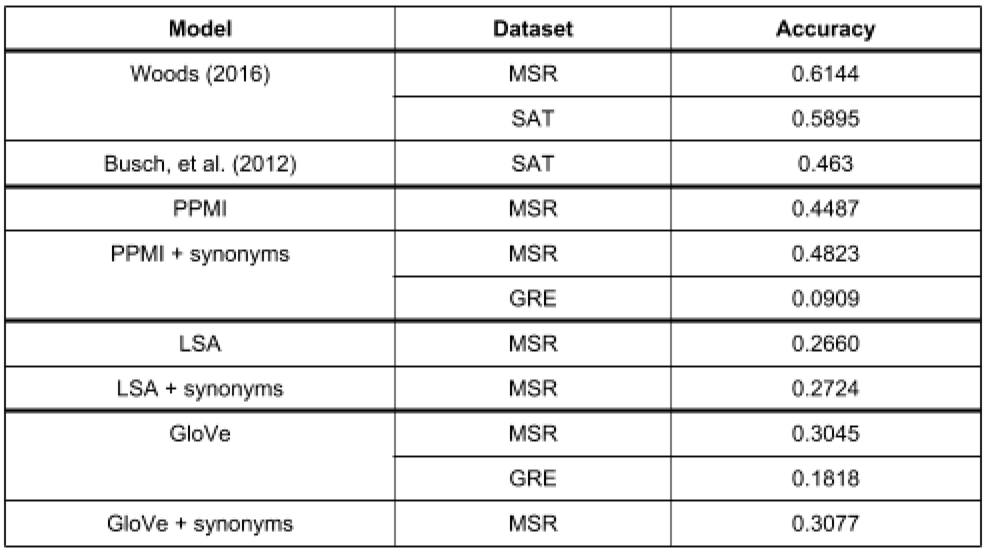
Our PPMI baseline calculates PPMI from a word-word co-occurrence matrix computed from the 500 Gutenberg books. A score is computed for each potential answer by summing the PPMI between the answer choice and every other word in the sentence.

Our LSA baseline computes word vectors from a word-document co-occurrence matrix. Frequency counts are scaled by log(1+n), after which SVD is applied. Scores for each potential answer are calculated using the sum of cosine similarities between each answer choice and the rest of the words in the sentence.

Finally, our GloVe baseline uses the same scoring approach as the LSA baseline, but using precomputed GloVe vectors instead.

C. Our Model

Pushing past our baselines, we have implemented a naive synonym finder that is used when the answer choice is not in the PPMI matrix's vocabulary. Specifically, we gather all the words in all of the candidate word's Wordnet synsets, without incorporating any part-of-speech (POS) information. We used this synonym finding algorithm in conjunction with the LSA and GloVe models as well.



Model	Dataset	Accuracy
Woods (2016)	MSR	0.6144
	SAT	0.5895
Busch, et al. (2012)	SAT	0.463
PPMI	MSR	0.4487
PPMI + synonyms	MSR	0.4823
	GRE	0.0909
LSA	MSR	0.2660
LSA + synonyms	MSR	0.2724
GloVe	MSR	0.3045
	GRE	0.1818
GloVe + synonyms	MSR	0.3077

Table 1. Results of Preliminary Models
The first two models listed are from papers that we reviewed; Woods holds the state-of-the-art scores for the MSR and SAT task, and Busch, et al. is a previous 224U paper that worked with SAT sentence completion data. All other models are our own work. The PPMI, LSA, and GloVe models on their own are baselines; the models including synonyms are our first steps towards contributing novel feature engineering to this problem.

We have also fully built tools to improve our synonym-finding by incorporating POS tagging, and tools to gather hypernyms, dependency features, but have not yet fully incorporated these into any model.

D. Discussion of Preliminary Results

As shown in Table 1, the addition of synonyms improves performance in every model we tried, even though it is naive and even though we only looked for synonyms when the word did not exist in the vocabulary at all. It had the least effect for GloVe, which we would expect because GloVe is pretrained on Gigaword, which has an enormous vocabulary, so if synonyms are only applied when a word does not exist, it would probably only even affect a tiny number of text examples. This is a promising result because we did not even try synonyms on GRE data yet, just MSR data, where we predicted its effect would be more limited.

The preliminary results on the GRE data are unsurprisingly very poor. This is probably for a number of reasons - our dataset is tiny (25 questions), and unlike the MSR dataset which only has one blank per question, the GRE dataset has a variety of single-blank, multiple-blank, and sentence equivalence questions. The latter two types of questions have much stricter accuracy measures, so we would expect any naive baseline to have lower scores when compared to the MSR task. Finally, while models needing to be trained with co-occurrence data were all trained with Gutenberg (MSR) data, we plan to eventually train the GRE models with data from newspapers like the New York Times and LA Times portions of the Gigaword corpus, as other papers have had more promising results on SAT questions while training on those datasets.

E. Steps Forward

Currently, our biggest obstacle is data collection, but we are confident that even in the worst case we can get some reasonable and interesting standardized test question set, even if it does not include multiple-blank or sentence equivalence questions. Other than that, we continue to implement the models described above, and we do not foresee any major issues on that front.

V. References

[1] Busch, Katherine, et al. *Taking the SAT Exam: Sentence Completion*. 2012.

[2] Islam, and Inkpen. *Near-Synonym Choice Using a 5-Gram Language Model.* University of Ottawa. 2010.

[3] Wang, and Hirst. *Near-Synonym Lexical Choice in Latent Semantic Space.* 2010.

[4] Inkpen. *A Statistical Model for Near-Synonym Choice.* 2007.

[5] Woods. *Exploring Linguistic Features for Sentence Completion.* 2016.

[6] Mirowski, and Vlachos. *Dependency Recurrent Neural Language Models for Sentence Completion.* 2015.

[7] Zweig, et al. *Computational Approaches to Sentence Completion.* 2010.

[8] Zweig, and Burges. “The Microsoft Research Sentence Completion Challenge.” Microsoft Research Technical Report, 20 Feb. 2011.