Word predictor effectiveness: an evaluation of word predictor functionality

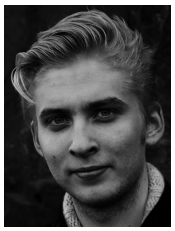
GROUP41

Jonathan Kindfält
1993-08-03
kindfalt@kth.se

Johannes Olsson
1989-07-28
johanneo@kth.se

Mikael Sjöberg
1993-10-20
miksj@kth.se

Alexander Östman
1990-08-27
aostm@kth.se



Abstract

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

1 Introduction

Word predictors are programs that can estimate the sequence or part of the sequence of words for a given text sentences.Traditionally these kind of programs have been used in the the context of health care in aiding people with disabilities to communicate [1]. During the last decades these programs have been a part in the field of augmentative communication but this time directed towards the border mass. It have become a tool for increasing efficiency of typing in general [1].

There are several ways of making word predictors. Most of the word prediction make use of N-gram language models to estimate the probability of the following word in a phrase [2]. N-grams are a way to group and sample words that occur next to each other in text. It grasps the probability of sequences of words rather than the single word alone. I the sampling of word sequences also captures part of the grammar in a language due to it keeps the order of the words within the sequences. The N i N-grams stand for the number of words in the word sequence sampled.

There are several methods to even further refine the nature of natural language. One way of the possible ways is to implement grammar constraints that take in account that words can/can not follow in any arbitrary order. In this way it's possible to limit the number of possible words that might follow the most recent word predicted solly by probability but are not correct according to the semantics.

Yet another way might also be to take in account the context of the text. By identifying different parts of the sentence construction, e.g. the acknowledgement of earlier written subjects and objects in main respectively dependent clauses, the next word can be predicted further. This could also add to the constraints of words possible due to need of matching with the already written words.

How to make word predictors more efficient is the main question of this project and it's going to be looked into by the means of experiments.

1.1 Problem statement/formulation

Word prediction is an important part in Natural Language Processing (NLP) since in many tasks it is necessary to determine the next word [3]. Predicting words correctly can be a difficult task since there are a lot of factors that have to be taken into account. Grammar constraints and context of the input are often complex matters which need to be analyzed in order to make accurate predictions.

Gregory W. Lesher et.al [1] show that the performance has been seen to

rely on training text size. It is mentioned in the report that the performance of a word predictor could be improved with syntax-based prediction which should be investigated. This is also mentioned by Keith Trnka et.al.[4] that syntactic knowledge can help improve word prediction.

One goal of word predictors is to be able to give accurate results to sentences in different contexts [3]. This requires a way for the word predictor to recognize the context in the sentence.

Both syntax-based prediction (grammatical analysis) and context recognition require to be implemented along with a model. This work uses N-gram modelling since it is a common tool in NLP [2]. A problem with N-gram moddeling is that it uses training data which makes it possible that not all combinations of words occur, which requires a smoothing technique to give unseen N-grams some probability [5]. Smoothing techniques are a requirement for word prediction to be usable for different sentences than those in the training data.

Combining these techniques together: smoothing, grammatical rules and context recognition, is the problem this report focuses on, and how they can improve a word predictors' performance.

1.2 Hypothesis

In this section the hypothesis for this work is explained in the different categories: Smoothing techniques, grammatical constraints and context recognition.

1.2.1 Smoothing techniques

The smoothing techniques evaluated are kneser-ney smoothing and absolute discount smoothing. Both these smoothing techniques rely on similar ideas. They use a constant discount value to distribute probability from known n-grams to unknown n-grams. This idea can be seen from Good-Turing smoothing where the discounted value follow a similar pattern with a near constant discount [6].

Both the smoothing techniques interpolate their value from their lower n-grams but with different approaches. Absolute discounting uses the probability of the unigrams to interpolate the probability of a word, while Kneser-ney instead uses the continuation count of the word [6].

Our hypothesis regarding how the different smoothing techniques will perform in comparison to each other is that for a well-balanced corpus, they may provide similar predictions to each other. For instance for predicting the word after "it was", the most popular words that follow this sentence can

already be used in many different contexts and have a high unigram probability. If on the other hand the corpus is unbalanced and there is an unusual word which is overrepresented, it may have a high unigram probability. In absolute discounting this will give the unusual word a high prediction as an unseen word. For kneser-ney on the other hand it will still pick the words that are usable in many different contexts.

So for a balanced corpus, they may be quite similar, while with an unbalanced corpus Kneser-ney will provide better results.

1.2.2 Grammatical constraints

Grammatical constraints tries to apply the grammatical rules in a language to help the predictor to get more reasonable results. Our hypothesis is that these constraints will help in many cases to remove words that is unusable in a certain sentence. Since they can look on the complete sentence regardless of the n-gram sizes, they may provide valuable information on which n-grams that are usable in that current sentence. So the result using grammatical constraints should be better than without it.

1.2.3 Context recognition

By context recognition, it is meant that the word predictor tries to understand the context that a sentence is about and will try to provide suggestions based on that context. Context recognition may be both good and bad. If the context is misinterpreted the suggestions may be worse than without context recognition. Example of sentences which can be hard to understand the context in:

- My dog and I went to the beach and it was happy
- My dog and I went to the beach and it was nice

These sentences are almost exactly the same but talk about different things. The first sentence talk about the dog, while the second sentence more talk about the experience. Since the English language is a Subject-verb-object language, our hypothesis is that the probability to get the correct context should be higher than misinterpreting it.

1.3 Contribution

The results of this work could guide anyone who wants to implement a good word predictor. It may also be used to draw conclusions regarding questions

that are not the main focus of this work. An example of such questions are "when does a word predictor have to be efficient?". Functionality that is bad in theory because it should perform very badly for some situations may prove practically good in a statistical research. Another question is "what functionality is worth implementing?". Some functionality may provide minimal improvement while being extremely resource hungry. Hopefully, this report will contribute to the field by providing results and discussions that could help give answers such questions.

1.4 Outline

2 Related work

3 Method

This section explains how the implementation was done and how the experiments were conducted. After that follows an outline of the experiment setup and a motivation for the designs of the experiments.

3.1 Implementation

The implemented word predictor had three basic components: N-grams with probability smoothing, grammar constraints and context recognition. The implementation was done using two NLP-libraries and several books to make a corpus. KYLM[1] was a library capable of producing N-grams with different smoothing techniques. We used it to process our corpus and make an ARPA-file, consisting of N-grams of different sizes and their probability derived from the corpus and chosen smoothing technique. We also used OpenNLP for part-of-speech-tagging that was used in the implementation of grammar constraints in the word predictor. The rest of the program: searching the ARPA-files, grammar constraints, context recognition and user interface was implemented by the authors.

To test our hypotheses, a qualitative inductive method was used. Evaluation and comparison of the different functionalities was made by measuring keystroke counts. This was a widely used method[4], and suited our project as it is an objective performance evaluation, and user interface/experience was of no interest to the study.

In a keystroke count experiment, for a given word, the number of keystrokes are the number of keys that the participant has to press in order for the word

[1] A full list of resources can be found in section 8

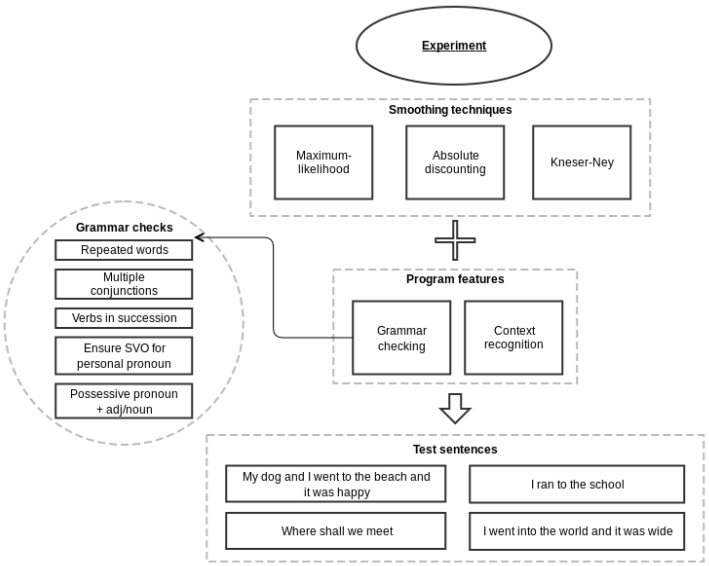


Figure 1: Diagram showing how experiments were generated

predictor to suggest the word the participant meant to write. In this experiment, given previous words, the word predictor may be able to suggest the thought of word without any strokes in which case the keystroke count for that word is zero. Spaces between words did not count as keystrokes.

3.2 Evaluation

The process of generating experiments is shown in figure 1. Four sentences that were thought to bring out the differences in the different functionalities were constructed. Tests were then performed for each sentence, for each combination of functionalities and for each smoothing technique, which made a total of 48 different tests. After that, some tests were rerun with case-insensitivity. OpenNLP did not work well with case insensitivity. For the reruns, grammar constraints were therefore always set to *off*, resulting in an additional 24 tests.

3.2.1 Experimental setup

Corpus The corpus used in the experiments consisted of 10 books[2] of varying themes and content, downloaded from `http://www.gutenberg.org/`.

[2]The full list of books can be found in section 8

Choosing the books was done at random but from different categories such as children's books, science fiction and philosophy.

The corpus was manipulated by removing citation marks, underscores and new lines. Periods, commas and question marks were made into separate words by adding a space before them.

The reasoning for first using a case-sensitive corpus was that the POS tagging required for grammar constraints was case-sensitive. However, after our first test round it was clear that the grammar constraints did not help much. A lowercase corpus was considered to produce better probabilities since there is no differentiating between the start of a sentence and the start of a bi-sentence. Therefore another corpus was created where all words were converted to lowercase to see if more keystrokes could be saved, even though this corpus could not be used together with grammar constraints.

N-gram size A maximum N-gram size of 4 was chosen for the experiments. Having a larger N-gram size was deemed to not give a significant increase in performance compared to the exponentially increasing number of N-grams and computation time. Grams of a higher size than 5 was also considered being too dependent on the corpus which could result in the word predictor generating very unlikely predictions.

Grammar constraints Several grammar constraints were implemented using POS (Part-of-Speech) tagging. These grammar constraints were supposed to remove some predictions that could be considered very unlikely to be correct.

The implemented grammar constraints with ideas of why they should work were:

- Avoid repeated words - When the user has entered a word it should be highly unlikely that the same word would appear directly after it.
- Avoid multiple conjunctions - A user rarely wants to enter several conjunctions into a single sentence, for example several "and"s. Therefore only a few conjunctions are allowed.
- Avoid verbs in succession - An English sentence rarely has several verbs in succession. By using POS tagging, even verbs not in the corpus can be identified and more accurate predictions can be made after that verb.
- Ensure SVO - Ensures SVO (Subject-Verb-Object) order for personal pronouns, meaning that only appropriate word types are predicted after

a personal pronoun. The allowed word types are verbs, modal verbs, adverbs and conjunctions.

- Possessive pronoun + adjective/noun - A possessive pronoun (my/her/his/its) is often followed by either an adjective or a noun, so any other word type is ignored.

Context recognition A simple context recognition was implemented by exchanging the word "it" with the first noun of the sentence, together with any possessive pronoun or determiner. For example, the sentence "my dog was nice and it was..." would be changed to "my dog was nice and my dog was...". This exploits the fact that English sentences often follow the SVO order which means that the subject of a sentence is often the first occurring noun in a sentence. The word "it" will most likely point to the subject of the sentence.

For some sentences the first occurring noun is not the subject but rather the object, for example in the sentence "I liked the dog because it was nice", where the subject is a personal pronoun . However, the implemented context recognition will work well even for sentences like this, because the first noun was "dog", which in fact was what "it" referred to.

The word "it" is not replaced when predicting the word directly after "it", since it could sometimes make predictions grammatically incorrect.

Test sentences This chapter explains the reasoning for coming up with the test sentences and their anticipated results.

My dog and I went to the beach and it was happy This sentence was constructed to test the context functionality of the word predictor. It contains the noun "dog" which is the subject of the sentence. With context recognition, the word "it" will be replaced by "my dog" which should make the predicted words more accurate and thus save more keystrokes.

I ran to the school This sentence was created to test the grammar functionality. The sentence contains "I", a personal pronoun, for which some grammar constraints are implemented. Some improbable suggestions could therefore be removed which should result in a higher probability of showing the correct word, and thus less keystrokes should be required.

I went into the world and it was wide This sentence was fabricated to show the context recognition for when the word "it" refers to the object of the sentence, "the world". Since "the world was wide" exists as an N-gram in the corpus, and "it was wide" is not a very common statement compared to other "it was"-combinations, this should result in more accurate word predictions.

The sentence also tests grammar constraints in a similar manner as the sentence "I ran to the school".

Where shall we meet This sentence was constructed as an everyday language sentence, for showing how the word predictor could perform for text messaging. Context recognition will not matter for this sentence, but grammar constraints could make the predictions more accurate.

4 Results and analysis

This chapter shows the different results from the experiments. Both on how the different smoothing techniques, grammar analysis and context can change the number of keystrokes needed to complete a sentence, and a qualitative analysis on the results from using context recognition.

4.1 Keystroke experiment

This experiment focused on testing the different sentences explained in *test sentences* and how the different techniques applied in the work can change the number of keystrokes needed to complete a sentence. Every column represent if grammar or context was used when counting the number of keystrokes needed.

Table 1: Kneser-ney and case-sensitive n-grams

Kneser-Nay				
Test sentence	Grammar	Context	Both	None
My dog and I went to the beach and it was happy	16/36	15/36	16/36	15/36
I ran to the school	3/15	3/15	3/15	3/15
Where shall we meet	8/16	6/16	8/16	6/16
I went in to the world and it was wide	7/29	5/29	6/29	6/29

Table 2: Absolute discounting and case-sensitive n-grams

Absolute discount				
Test sentence	Grammar	Context	Both	None
My dog and I went to the beach and it was happy	14/36	13/36	14/36	13/36
I ran to the school	3/15	3/15	3/15	3/15
Where shall we meet	7/16	6/16	7/16	6/16
I went in to the world and it was wide	8/29	7/29	7/29	8/29

Table 3: Maximum likelihood and case-sensitive n-grams

Maximum likelihood				
Test sentence	Grammar	Context	Both	None
My dog and I went to the beach and it was happy	14/36	14/36	14/36	14/36
I ran to the school	4/15	4/15	4/15	4/15
Where shall we meet	5/16	5/16	5/16	5/16
I went in to the world and it was wide	8/29	7/29	7/29	8/29

Table 4: Kneser-ney and case-insensitive n-grams

Kneser-Nay				
Test sentence	Grammar	Context	Both	None
My dog and I went to the beach and it was happy	0	13/36	0	13/36
I ran to the school	0	3/15	0	3/15
Where shall we meet	0	7/16	0	7/16
I went in to the world and it was wide	0	4/29	0	6/29

Table 5: Absolute discount and case-insensitive n-grams

Absolute discount				
Test sentence	Grammar	Context	Both	None
My dog and I went to the beach and it was happy	0	13/36	0	13/36
I ran to the school	0	3/15	0	3/15
Where shall we meet	0	6/16	0	6/16
I went in to the world and it was wide	0	5/29	0	7/29

Table 6: Maximum likelihood and case-insensitive n-grams

Maximum likelihood				
Test sentence	Grammar	Context	Both	None
My dog and I went to the beach and it was happy	0	13/36	0	13/36
I ran to the school	0	4/15	0	4/15
Where shall we meet	0	5/16	0	5/16
I went in to the world and it was wide	0	7/29	0	8/29

4.2 Context recognition

These tables show different predictions given as the next word for a given sentence with and without context recognition using a case insensitive.

Table 7: Predictions of kneser-ney without context recognition

Context recognition					
Test sentence	First	Second	Third	Fourth	Fifth
I went into the world and it was	the	a	not	as	only
my dog and i went to the beach and it was	the	a	not	as	only

Table 8: Predictions of kneser-ney with context recognition

Context recognition					
Test sentence	First	Second	Third	Fourth	Fifth
I went into the world and it was	being	very	more	full	wide
my dog and i went to the beach and it was	not	apparently	a	the	in

Table 9: Predictions of maximum likelihood with context recognition

Context recognition					
Test sentence	First	Second	Third	Fourth	Fifth
I went into the world and it was	being	frightened	full	itself	more

4.3 Keystrokes analysis

From the tables with case sensitive n-grams, it is seen that absolute discounting show better results than kneser-ney in two of the sentences. This differs

slightly from our hypothesis that kneser-ney should present better results. In the case insensitive tables it can be seen that the difference is smaller. Both smoothing techniques now take the same amount of keystrokes for the first sentence. This difference can be that kneser-ney uses a continuation count to calculate its probabilities. In the case sensitive case "The car" and "the car" becomes two different n-grams, the continuation count of "the" will be wrong in that case. This could explain the result from the case sensitive result, since the continuation count gets distorted.

Usage of grammar constraints show that the results is worse than without it. This can be due to many reasons: badly implemented grammar constraints, grammar constraints conflicting with each other. Since the english language is complex there are many rules which do not always apply in certain situations. Aurora-systems which developed a word predictor writes that grammatical rules only apply if the writer does not know how to formulate correct sentences, otherwise it is more of a hindrance[7].

In the context recognition results it is seen that the fabricated sentence explained in Test sentences performs better than without context recognition. This is most probably the case just because the sentence is fabricated and the corpus contains the n-gram "the world was wide". In the first sentence the context recognition does not provide any improvements. This shows that for the context recognition to work, there actually needs to be an N-gram which contains the actual sentence. In regards to the hypothesis this is quite correct, as long as there are n-grams which contain the contextual word there will be a better prediction. This could mean that as found by Gregory W. Lesher et.al that the performance scales with the training set size[1], this implementation of context recognition may scale depending on training set size aswell.

4.4 Analysing predictions with context recognition

In table 7 it can be seen that the two results are identical and solely rely on the n-gram "it was". Comparing that with table 8 with context when "it" has been replaced with the first appearing noun the suggested words slightly more predicts to the context.

Table 9 shows the prediction instead with maximum-likelihood, the suggested words can be seen to not be as general as with kneser-ney.This is visible in the keystrokes result as well where KN needed three less keystrokes to complete the sentence.

5 Discussion

Method Using KSR was chosen as it would produce objective results for this qualitative study. It may, however, give a somewhat misleading result as the word predictor may seem more efficient than it actually is. The reason is that filtering is done on the first few letters of the last words. In a small corpus, that could mean that there are few, or maybe even just one option of words when a few strokes have been made which makes prediction easy. For the same reason, it makes sense to make experiments with a large corpus so prediction still can differ when a large part of a word is given. There are also subjective parts of this work. People write differently and our fabricated sentences may not be relevant to some. Perception testing could produce interesting results, where subjects can rate how relevant words are to what they wanted to write. Perhaps the word predictor manages to predict a synonym that could substitute for the planned word.

Quantitative perception testing should also have be a better method than KSR to test functionality that make trade offs in the way that they should work well in certain situations, such as grammar constraints. These situation may occur at different frequency for different test subjects, meaning that some experience more of the positive effect while others experience more of the negative.

Implementation Writing grammar constraints for the English language proved to be hard. Nevertheless, the way they were implemented in the word predictor for this study made the word predictor unable to predict words that were constituted exceptions to the grammar constraint. Instead of forbidding these words, it should have been better to let the constraints decrease the probability of words, rather than ruling them out completely.

Furthermore, grammar constraints can be assumed to have greater effects when a desired word occurs scarcely, or not at all, in the corpus. To enhance the grammar constraints' effect on predictability, grammar constraints for proper nouns (names) could have been implemented since those words often do not occur in corpora. The experiments show that context recognition helped produce better word predictions. However, due to the fact that the authors both implemented the word predictor and came up with the test sentences, the sentences were made to fit the implementation. If strangers to the implementation were to come up with sentences the context recognition might have made incorrect assumptions as to what the word "it" referred to. This would make the word predictor produce words for a different context than what the user is thinking about, which will most likely be way off.

A possible way to protect against such sentences could be to interpolate

the probabilities from the word “it” and the word found by the context recognition to produce more even results. Another possible way would be to implement a more advanced context recogniser, since the one implemented is rather simple.

(...These problems further points out the need to make more thorough experiments to test the implemented functions. Further tests could include more test sentences and a larger corpus.)

A decision was made to not model unknown words when producing the N-grams. This was because a large amount of trust was given to the corpus being good and having a lot of common combination of words. Having a word predictor that finds out that an uncommon word is most probably also felt counterproductive, since it is better to give any prediction than to give none.

Modelling unknown words could however help in some cases. If a word unseen in the corpus would appear in a sentence, it could be labeled as an unknown word. It would thereafter be possible to find N-grams corresponding to this wildcard sentence. For example, the sentence “My dog Fido is” would produce the N-gram “My dog ¡UNK¿ is” with unknown word modelling and only “is” without. With unknown word modelling, context can be conserved in a way that is not possible otherwise. It would have been interesting to implement unknown word modelling as well to see if and how the results differ.

The implemented word predictor always tries to predict the most probable word, no matter how short it might be. It might however be better to omit words which are a very short. For example, successfully predicting the word “a” only saves one keystroke, while predicting a word such as “another” would save seven keystrokes.

6 Summary

Wrap it up!

7 Contribution

We, the members of group 41, unanimously declare that we have all equally contributed toward the completion of this project.

References

[1] G. W. Lesher, B. J. Moulton, and D. J. Higginbotham, "Effects of ngram order and training text size on word prediction," 1999.

[2] T. Wandmacher and J. Antoine, "Methods to integrate a language model with semantic information for a word prediction component," *CoRR*, vol. abs/0801.4716, 2008.

[3] Y. Even-Zohar and D. Roth, "A classification approach to word prediction," *CoRR*, vol. cs.CL/0009027, 2000.

[4] K. Trnka, D. Yarrington, and K. Mccoy, "The keystroke savings limit in word prediction for aac," tech. rep., 2005.

[5] S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*. Pearson Education, 2 ed., 2003.

[6] C. M. Dan Jurafsky, "Kneser-ney smoothing."

[7] I. Aurora Systems, "Frequently asked questions - word prediction." `http://www.aurora-systems.com/pages/faqpred.html#GRAMMAR`. Accessed: 2015-10-15.

8 List of resources

NLP libraries

OpenNLP Description

KYLM Description

Books for corpora

- Around the World in 80 days by Jules Verne
- Motor Matt Makes Good by Stanley R. Matthews
- People Minus X by Raymond Zinke Gallun
- Philosophical Studies by George Edward Moore
- The Errand Boy; Or, How Phil Brent Won Success by Horatio Alger

- The Little Nugget by P. G. Wodehouse
- The Wailing Asteroid by Murray Leinster
- The War of the Worlds by H. G. Wells
- The White Feather by P. G. Wodehouse
- Try and Trust by Horatio Alger