Word predictor effectiveness: an evaluation of word predictor functionality

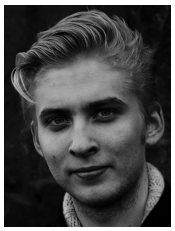
GROUP41

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 have traditionally been used to increase the communication ability of persons with speech and language impairments[1][2][3]. More recently, they have also come to use in increasing efficiency of typing in general [1], such as in-line search suggestions in search engines and the *prediction window* for touch screen keyboards.

Problem statement Presenting word suggestions to a user requires a statistical language model[2] to calculate the probability $P(word|history)$, the most common being N-gram models[4]. The problem and the goal of this work is to find what makes such a word predictor effective. In this work, by effective, it is meant that a more effective word predictor requires a lower KSR (Key Stroke Rate, calculated as $KSR = \frac{\text{keystrokes made}}{\text{keystrokes in sentence}}$) than a less effective one. More specifically, the goal is to test different functionalities that are thought to increase the effectiveness of a word predictor and evaluate the results. The problem is worth addressing because the solution could help making communication with machines more effective in general. Word prediction could also be used as an important component in other areas of NLP, such as speech recognition, why their effectiveness is a crucial part in NLP.

Hypotheses Three different functionalities that were thought to affect the effectiveness of word predictors. For each of these, a hypothesis was come up with to predict how the function would affect effectiveness.

The first function, probability smoothing, is used to assign a non-zero probability to events unseen in the training data[5]. N-gram models as such require smoothing to give unseen N-grams probability[6]. The smoothing techniques tested in this work are *Kneser-Ney* and *absolute discounting*. They both use a constant discount value to distribute probability from known to unknown N-grams, an idea that was derived from *Good-Turing smoothing* where discounted values follow a similar pattern with a near constant discount[7]. The main difference between the two is that absolute discounting uses unigrams to interpolate the probability of a word; Kneser-Ney uses continuation count, that is the number of different contexts in which a given word appears in the training data[7]. The hypothesis regarding these smoothing techniques is that words with high unigram probability are likely to be overrepresented using *absolute discounting*, while *Kneser-Ney* should give a more varied result.

The second function is grammatical constraints, which means using grammatic rules to get more reasonable results. By excluding words that are

known to not be the correct words to guess, the probability of predicting the right word is increased. The second hypothesis is therefore that grammar constraints will decrease the likelihood of predicting impossible words.

The last function, context recognition, involves recognizing the topic of the sentence with the objective to predict more relevant words. Previous studies have shown that topic modeling can be used to improve prediction[8][2]. This work makes the hypothesis that context recognition will help word predictors predicting more relevant words, increasing the likelihood of predicting the word the user meant to write.

1.1 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.

2 Related work

This work is most similar to Trnka, Yarrington, McCoy, and Pennington's study [2] on keystroke savings. The objective of their study was different, but the methods used were well suited to test this work's hypotheses. A problem with the study was that it was focused on obtaining results that are very close to reality. In this work, we are interested in showing the effect of different functionalities of word predictors, thus making experiments that ought to increase the contrast in the results.

The methods used in the previously discussed work used N-gram models to predict words. Wöffel and McDonough's book "Distant Speech Recognition" [9] as well as Hiemstra's "Probability Smoothing" [5] were used to get an in-depth understanding of Markov chains and smoothing used in constructing these models in the implementation of a word predictor.

Lastly, "Corpus Studies in Word Prediction" by Trnka and McFoy [3] was helpful in understanding the effects training data, e.g. the corpus, may have in practice and how a corpus could be generated to produce good results.

As for the firstly mentioned study, the results where heavily reality-oriented, why they could be expected to smooth out some differences in the results of this 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 POS (Part-of-Speech) tagging that was used in the implementation of grammar constraints and context recognition. The rest of the program: searching through the N-grams, grammar constraints, context recognition and user interface was implemented by the authors. The core method of choosing the words to be predicted was to compare large N-grams, choose the grams with highest probabilities, and if more words were required, fall back to smaller grams.

Grammar constraints The implemented grammar constraints worked by removing predictions that could be considered very unlikely to be grammatically 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* - Only a small number of conjunctions are allowed, because a normal sentence rarely contains many conjunctions. *Avoid verbs in succession* - An English sentence rarely has several verbs in succession. Verbs not in the corpus can be identified and more accurate predictions can

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

be made by not allowing other verbs to follow. *Ensure SVO* - Ensures a form of SVO order by ensuring that personal pronouns are only followed by a verb, modal verb, adverb or conjunction. *Possessive pronoun + adjective/noun* - A possessive pronoun is often followed by either an adjective or a noun, so any other word type is ignored.

Context recognition 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 is nice and it is..." would be changed to "my dog is nice and my dog is...". The word "it" will often 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 like the dog because it is...". 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.

3.2 Evaluation

To test our hypotheses, a qualitative inductive method was used. Evaluation and comparison of the different functionalities was made by measuring keystroke rates for typing whole sentences. This was a widely used method for similar projects[2], and suited our project as it is an objective performance evaluation, compared to perception testing which produces more subjective results.

In the constructed keystroke experiment, for a given word, the number of keystrokes were the number of letters that the participant had to enter in order for the word predictor to suggest the word the participant meant to write. In this experiment, the word predictor might suggest the thought of word without any keystrokes, in which case the keystroke count for that word is zero. Spaces between words did not count as keystrokes.

The process of generating experiments is shown in figure 1. Four sentences that were thought to display 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. POS tagging did not work well with case insensitivity so grammar constraints were therefore always set to *off*, resulting in an additional 24 tests.

Test sentences The motivation for the test sentences and their anticipated results were:

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 constructed 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.

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/`. 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 case-insensitive corpus was the same corpus but with only lowercase letter.

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

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

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.

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[10].

In the context recognition results it is seen that the fabricated sentence explained in Test sentences performs better than without context recognition. This is 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. This could also show more relevant results for grammar constraints.

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.

The experiments show that context recognition helped produce better word predictions. However, more complex sentences may produce 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 recognizer, since the one implemented is a proof of concept. More thorough experiments are needed to analyze the probability of finding the correct context.

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

This work have been about implementing and testing a word predictor in the purpose of find out what makes an effective one. The definition of an effective word predictor was defined as one that could reduce the number of key-strokes needed to complete a sentence. This resulted in an implementation based upon parts from previously made projects. The use of KYLM and Open NLP for generate the N-grams probabilities respectively the part of speech tagging was the foundational stones for this work. On top of that development of grammar constraints and context recognition was made as mean of conducting tests to see what could make a word predictor more effective. The tests in this project concluded that grammar constrains didn't seem to benefit the effectiveness of the word predictor while context recognition proved to be.

7 Contribution

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

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

[3] K. Trnka and K. F. McCoy, "Corpus studies in word prediction," 2007.

[4] 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.

[5] D. Hiemstra, L. Liu, and M. T. Özsu, "Probability smoothing," *Encyclopedia of database systems*, vol. 2169, 2009.

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

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

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

[9] M. Wölfel and J. McDonough, *Search: Finding the Best Word Hypothesis*. Chichester, UK: Chichester, UK: John Wiley and Sons, Ltd, 2009.

[10] 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 http://opennlp.apache.org/

KYLM http://www.phontron.com/kylm/

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

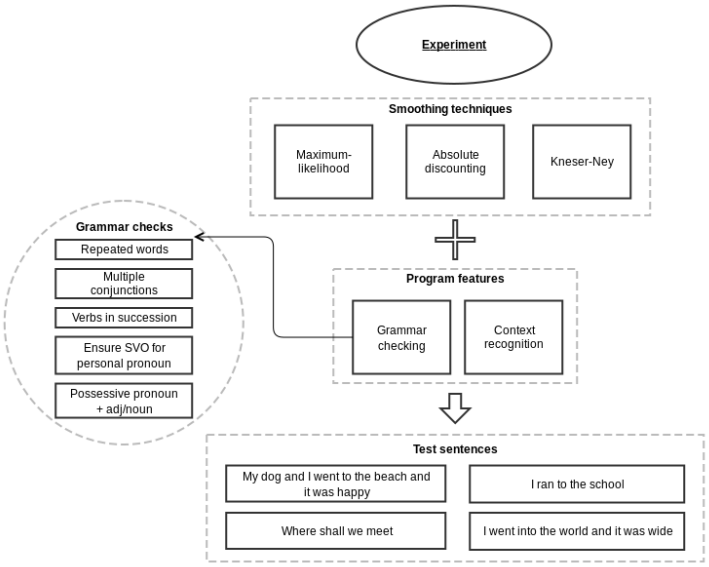


Figure 1: Diagram showing how experiments were generated