Topic Modeling in Word Prediction for AAC

Keith Trnka, Debra Yarrington, Kathleen McCoy
Computer Science Department
University of Delaware
Newark, DE 19716
{trnka,yarringt,mccoy}@cis.udel.edu

ABSTRACT

Word prediction is a method for enhancing the communication ability of persons with speech and language impairments. In this work, we explore one method of adjusting the language model based on the content of a conversation.

Keywords

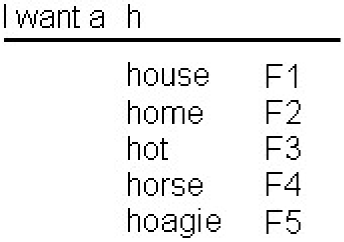
Word prediction, keystroke savings, alternative and augmentative communication (AAC), topic modeling, language modeling

INTRODUCTION

Alternative and Augmentative Communication (AAC) is the field of research concerned with finding ways to help those with speech difficulties communicate more easily and completely. Today there are approximately 2 million people in the United States with some form of communication difficulty. With today's technology, electronic communication devices are widespread. Most communication devices today have speech output and/or displayed text or pictures. One issue in using a communication device is that communication rate is generally slower than the common speaking rate. Whereas speaking rate is estimated at 180 words per minute (wpm) and experienced typists can manage 100 wpm, a disabled user's input rate is estimated at roughly 15 wpm (***Bellegarda, 2000***; Copestake, 1997; Newell et. al., 1998). Thus one goal of developers of AAC devices is to find ways to increase the rate of communication output. Developers cannot increase the dexterity or muscle control of users, so the alternative is to experiment with the user interface for input.

This paper investigates the use of a word prediction system. In word prediction, we assume that the user enters letters using a standard keyboard. The system predicts full words that are likely to be desired, and provides them to the user for selection with one additional keystroke.

A word prediction system predicts the word currently being typed on the basis of what has already been typed. Suppose that the user wants to enter "I want a home in the country." After typing, "I want a h", they might see something like shown below. The system has created a *prediction window* containing the five words which it thinks the current word is most likely to be. In this example, the user can press F2 to complete the word "home" and the system will automatically enter a space afterwards. So in this example, the user needed 3 keystrokes to enter what would normally take 5 keystrokes, when the space is considered.



```
I want a  h
          house    F1
          home     F2
          hot      F3
          horse    F4
          hoagie   F5
```

Figure 1: An example of what a word prediction interface might look like

The prediction list can vary in length, but most systems tend to use lists of length between five and seven. The prediction list can occur in-line (it can appear within the line being entered as it is entered), or it can occur somewhere separately on the interface screen. For row-column scanning devices, the word list often appears in an extra row or column on the scanning grid.

It is difficult to judge how much word prediction can speed communication rate. Much of this determination is dependent on both the characteristics of the user, such as their physical and cognitive abilities, and characteristics of the user interface, such as where the prediction list is displayed and how a word in the list is selected.

It quickly becomes apparent that many factors affect the efficacy of word prediction, and more studies are needed to determine the effect different factors have on the success of word prediction. It is equally apparent, however, that unless the word prediction system is able to successfully predict words and thus decrease the number of keystrokes necessary, other factors are irrelevant. Therefore, before tackling the added issues involved in user interfaces, it is instructive to look at the percentage of keystrokes saved, since this measure provides an upper bound on any communication rate increases from word prediction. Thus our work here concentrates on investigating keystroke savings in word prediction.

Our long-term goal is to investigate methods for increasing keystroke savings in word prediction by taking various amounts of contextual information into account during the prediction process. Of course, in doing this we must have a way of evaluating whether or not our various attempts at capturing contextual information are fruitful - so first we must establish a baseline prediction system and a method for calculating keystroke savings against which our future systems can be tested.

Clearly this paper is not the first to discuss the use of word prediction in AAC (Boggess, 1988; Carlberger et. al., 1997; Copestake, 1997; Fazly and Hirst, 2003; Garay-Vitoria and Gonzalez-Abascal, 1997; Lesher and Rinkus, 2002), and each of these systems has been presented with evaluations. However, as explained in [[TL report]], we find results difficult to interpret due to the lack of a standardized approach in evaluation. Similarly, this paper is not the first to discuss topic modeling (Bellegarda, 2000; Florian and Yarowsky, 1998/1999, Seymore and Rosenfeld, 1997, etc.) However, this is only the second paper to discuss topic modeling in the context of word prediction. (Lesher and Rinkus, 2002) explored this area of research somewhat.

In this paper we first give some background in statistical approaches to word prediction. We also give the details of our baseline word prediction system. We present the full details of our approach to integrating topic modeling into word prediction, then present comparative evaluations between the two systems. Finally, we discuss future improvements to the language model used for word prediction and conclude.

METHODS

Like other approaches, we apply statistical language modeling techniques to word prediction [3,4,5,6,8,13]. Our baseline is a pure n-gram based method. [5,6,8] additionally integrated syntactic knowledge into their language models, which was found to improve prediction somewhat.

Basic word prediction treats each sentence of the user's conversation as independent. At any given point in the sentence, the user has some word that they are typing. At that point, the word prediction system presents a list of words, called a *prediction window*, that the user may be typing. If the desired word appears in the list, the user selects it using a key reserved for that position in the list. If the word doesn't appear in the list, the user must enter another character. Then the word prediction system updates the list and the process repeats.

To present the user with a list of possible words, the word prediction system needs to know all of the words in the language. The vocabulary is constructed by considering all words that occur in some training corpus. If a word being typed isn't in the vocabulary, it can't be predicted.

The second requirement is a statistical language model. The purpose of the language model is to compute the probability P(word | history), where the history is the words that have already been entered. Given a vocabulary and a language model, the list of predictions is generated as follows: The vocabulary is first filtered to remove words that don't match the partially entered word. For example, if 'a' has been entered, the system will only consider words beginning with 'a'. Then this list of candidates is sorted by P(word | history). The top W words are presented to the user, where W is the prediction window size.

The remainder of this section is devoted to the construction of a statistical language model.

Corpus

Statistical approaches require a collection of text to construct a language model. Ideally, our corpus would be a collection of conversations involving one or more people using an AAC system. Such a corpus is unavailable, so we follow [13] in using the Switchboard corpus, which is a collection of telephone conversations and their transcriptions.[1] The corpus is divided into two sections: one section to train the statistical language model and one section to evaluate the word prediction method. The training section contains a randomly selected 2217 conversations and the testing section contains the remaining 221 conversations.

Because the corpus is a collection of transcribed conversations, it contains many speech repairs. Consider the example

is there um an- is there a like a code of dress ...

However, a person in a textual conversation would write

is there a code of dress ...

[9] categorized these sorts of self-corrections and integrated his solution into a parser. In short, his approach identified the words that are being corrected and the correction. The words being corrected are then removed for syntactic processing. Hindle's full set of editing rules was not practical for our purposes, so we implemented a subset of the rules: remove uh/um, exact repetitions, and repetitions in which the last word of the corrected part is abandoned, as in “an-”. These editing rules bring the Switchboard conversations closer to what we envision an AAC user would type. This agrees with the preprocessing that [13] did according to personal communication with Dr. Lesher.

[1] The Switchboard transcriptions were available from http://www.isip.msstate.edu/projects/switchboard/

Baseline Language Model

N-grams have been shown to give better performance with larger amounts of training text and roughly better performance with larger *n*. [14,17] For an introduction to n-grams, refer to [17] or [10]. The main weakness of ngrams is that they require substantial amounts of data. Using higher-order ngrams such as trigrams increases the problem of data sparseness. To combat this, we implement backoff with Good-Turing smoothing, the current best practice in statistical language modeling [17].

Backoff

Backoff is applied here much like in [11]. The idea is that the probability will be determined by trigram probability if the trigram probability is defined, otherwise we'll try the bigram probability. If that's undefined, we'll try the unigram probability. The crux of backoff is adjusting these probabilities so that all probabilities sum to one.

In order to apply backoff, some probability must be removed from trigrams which were seen in the corpus. The process of removing probability from seen trigrams and giving it to unseen trigrams is called smoothing and is described in the following section. For now, appreciate that each conditional trigram distribution will sum to less than one.

$$1-\delta=\sum_{\text{word}\in V} P(\text{word}|\text{word}_{-2}\,\text{word}_{-1})$$

Then the probability obtained by backoff is defined as

$$P'(w|w_{-2}\,w_{-1})=\begin{cases} P(w|w_{-2}\,w_{-1}) & \text{if } P(w|w_{-2}\,w_{-1})>0 \\ \delta\times P'(w|w_{-1}) & \text{otherwise} \end{cases}$$

Similarly, we define the bigram backoff probability as:

$$P'(w|w_{-1})=\begin{cases} P(w|w_{-1}) & \text{if } P(w|w_{-1})>0 \\ \delta\times P'(w) & \text{otherwise} \end{cases}$$

Note that δ is different for bigrams and trigrams.

Smoothing

Good-Turing smoothing can be summarized as taking probability away from low-frequency words and giving it to words that never occurred in the training corpus. For a technical account of Good-Turing smoothing, see [17] or [10]. For the intuitive account, refer to the graph below, which shows the probability of a trigram computed normally and computed after applying Good-Turing smoothing.

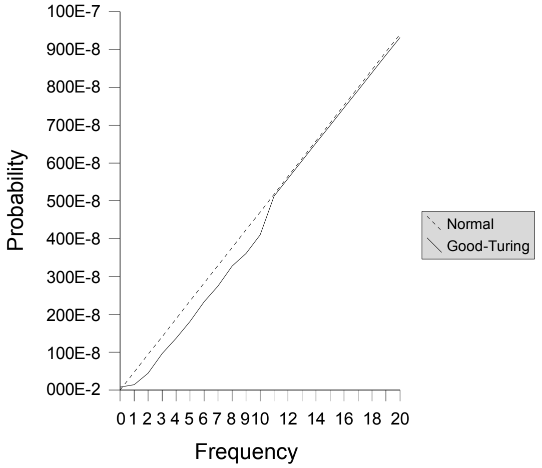


Figure 1: Smoothing reduces the probability of low frequency events and gives it to unseen events.

Sentence-initial Model

The trigram backoff model just described is applied to all words in a sentence but the first word. The first word in the sentence is modeled using a special unigram model to capture the notion that sentences tend to begin with a small set of words such as determiners and discourse markers. If the probability of a given word is undefined, then we backoff to the normal unigram model, using the same type of backoff as for bigrams and trigrams.

Our approach for treating the beginning of a sentence differs slightly from other researchers: The common method employed is to add a fake word at the beginning of a sentence and skip this symbol in training and testing; relying on it as a symbol to signal the particular context of a word. In fact, our approach and the traditional approach will perform identically on the very first word of a sentence. However, if both are trigram approaches, they will differ slightly on the second word of a sentence. As tetragrams (4-grams) are generally considered intractable, the two approaches will be identical for the remainder of the sentence. This difference reflects our intuition that only the second word in a sentence is influenced almost entirely by the preceding word; adding an additional conditioning event (the special symbol) would only serve to make the distribution of possible second words more sparse.

Topic Modeling

The goal of topic modeling is to identify the current topic of conversation, then increase the probability of related words and decrease the probability of unrelated words. The topic identification and topic application steps can be treated somewhat independent of one another, but they both depend on the representation of a topic. Like other researchers, we represent a topic using a collection of text. In this work, we use a collection of conversations to represent a topic, while others have primarily used collections of documents. However, researchers differ on the best way to represent a collection of topics: Some researchers have created a hierarchical collection of documents (Florian and Yarowsky, 1999), while others have created a set of topics (Mahajan, Bellagarda, Seymore and Rosenfeld, etc). The primary advantage of a hierarchical approach is that a more general topic can be selected when the topic of conversation is difficult to identify, and can be very specific when possible. However, as (Seymore and Rosenfeld, I think) point out, a conversation may simultaneously discuss multiple topics. For instance, a discussion of the economic impacts of baseball involves multiple topics. The set approach is more amenable to allowing the selection of multiple topics. We feel that the primary lure of a hierarchical approach, the ability to generalize, can be captured in the set approach as well. Rather than wait to construct a language model until the topic of conversation is known, we precompute a language model for each of the topics in our corpus.

Topic Identification

The current topic of conversation must be identified from the part of the conversation that has taken place so far, and updated periodically in the conversation. Further, we must devise a representation for a partial conversation that is conducive to identifying the topic. We follow other researchers in maintaining something like a unigram model of the current conversation and applying document similarity measures to identify appropriate topics.

In representing the conversation so far, we choose to implement an exponentially decayed cache, like (Bellegarda, 2000), using TF-IDF values rather than raw frequencies. The algorithm proceeds as follows: When a word is added to the cache, first all words that have occurred previously are decayed. This is done by multiplying by λ , a constant between 0 and 1. For our experiment, we followed Bellegarda in using .975. Secondly, the new word is added with it's IDF as the weight instead of 1. As our approach is for topic identification, we ignore words that occur in 85% or more of the topics, with the intuition that such words are function words.

As the step to bridge our model of the current conversation to the application of our topic model, we compute the document similarity between the cache and the unigram model for each topic. We do not feel that the words in the language model for each topic needed to use the TF-IDF method, as incorporating IDF again would be redundant. More importantly, however, is the capacity for variation in computing document similarity. Many different measures of similarity have been used previously in literature, see (***I can't remember***) for a comparison in the similar context of text categorization. However, we chose to use the cosine metric, following (Florian and Yarowsky, 1999).

Topic Application

Given that we have computed a similarity score for each topic to the current conversation, there are two main variations on how to construct a new language model. (Mahajan et. al., 1999) chose to implement a *k*-nearest solution, constructing the topic model from the most similar *k* topics. Each topic's language model was weighted equally for their experiments. We chose to follow Florian and Yarowky's approach. (Florian and Yarowsky, 1999) They expand the probability for a word (w) given a history (h) as follows:

$$P(w|h)=\sum_{t \in topics} P(t|h)\times P(w|t,h)$$

$P(w|t,h)$ is simply the probability of w taken from the language model constructed for topic *t*. The probability of the topic involves a little work to estimate:

$$P(t,h)\approx\frac{S(t,h)}{\sum_{t' \in topics} S(t',h)}$$

where *S(t, h)* is the similarity of the topic to the current part of the conversation.

Discussion

In this approach, all topics will have some influence, albeit minor, over the interpolated language model. This avoids the problem noted by (Lesher and Rinkus, 2002) that selecting the single best topic model leads to data sparseness problems, which nullify the benefit of having a better expectation of the language use. Now that the model is explained in detail, reconsider the activation of multiple topics: in this model, it simply means that multiple topics will have high probabilities. Additionally, we feel that this model has the ability to generalize topics. In the event that the context isn't very specific, we propose that the keywords in the conversation will reflect a more general category of topics. Given that this is the case, the more specific versions of this general topic should all be weighted somewhat higher than unrelated topics. Also, we feel that this model encapsulates the boosting and depressing of probabilities used by (Chen et. al., 1998). Apart from the use of Latent Semantic Analysis, there is one major deviation from Bellegarda's approach. (Bellegarda, 2000) In their work, the topic modeling is used to determine the vocabulary, and then integrated with a standard language model. Our approach, however, allows for stylistic and collocational peculiarities of a topic to be interpolated into the overall language model.

PRACTICAL CONSIDERATIONS

Although we are primarily interested in investigating whether or not topic modeling improves keystroke savings in word prediction, a number of practical considerations have both influenced our approach as well as our evaluation.

Smoothing and Topic Modeling

Smoothing to use for backoff can be applied at two different points in processing with topic modeling. Firstly, each individual topic language model may be smoothed. The alternative is to smooth the interpolated topic model. As there may be many topics, it's likely that each topic model will be rather sparse, whereas the interpolated model will be less sparse. If we apply smoothing before interpolation, bigrams and trigrams may be given less weight than is appropriate for the interpolated model. It is for this reason that we choose to apply smoothing after interpolation. Normally, this would lead to increased time of execution, but our implementation only smooths distributions that are reached in testing, as they are reached. However, smoothing requires frequencies rather than probabilities. To solve this, we approximate the same model by using frequencies in the equation instead of probabilities.[2] However, the interpolated frequencies are not whole numbers do to the interpolation process. We address this problem by creating bins of size one to use for smoothing. This is equivalent to taking the ceiling of each frequency before smoothing.

Number of Topics and k-nearest Topics

We had originally intended to follow (Mahajan et. al., 1999) in abandoning the notion of an explicit topic. Their approach treats each document in the training set as it's own topic, and relies upon the interpolation model to select the relevant topics and make appropriate generalizations. This approach is attractive for two main reasons. Firstly, they achieve and astounding 37.6% reduction in test set perplexity over a baseline trigram model. More importantly, however, their approach doesn't require a document or conversation to be labeled with a topic. This avoids the problem of either finding a corpus annotated for topic or the problem of performing appropriate automatic topic clustering.

Unfortunately, our preliminary tests revealed that interpolation of a topic model from about 2,200 smaller topic language models was too computationally demanding for the resources available to us. We realize now that this is one of the main advantages of using a k-best approach in interpolation. To help alleviate the computational demands, we instead used the topic assignments given in the Switchboard corpus. This reduces the number of topics to roughly 40.

Realistic Dynamic Topic Modeling

As we intend our research to be used for practical word prediction systems, we designed our topic model as practically as possible. That means, unlike Mahajan et. al., we only have access to the parts of the conversation that have been already said. Their approach assumes that the first 100 words of a conversation are available a priori. Also, unlike (Lesher and Rinkus, 2002), we do not assume to know the topic of a conversation ahead of time in testing. Their investigation used Switchboard, like ours, but read the topic number of each conversation to select the appropriate topic in testing.

However, dynamically re-interpolating the topic model is computationally expensive. In our implementation, the topic model is re-interpolated after every second turn.

Memory Requirements

We also discovered that topic modeling requires a substantial amount of memory. This could be alleviated somewhat by using a complex data structure such as a character-based trie, but that would further decrease runtime speed. Instead, we decided to only use bigrams and unigrams for topic modeling. We evaluate the impact of this decision in the following section.

EVALUATION

For a more thorough account of evaluation methods and the impact of user interface decisions on evaluation, see **[[limits]]**. In this work, the speak key is included in all simulations. Although immediate prediction gives much better keystroke savings, we use delayed prediction unless otherwise specified because it runs faster.

Like our other evaluations, the user interface simulates prediction using window sizes of 1-10 and adds a space after the word when prediction is used. All evaluation is done using keystroke savings (KS):

$$KS = \frac{\text{keys}_{\text{normal}} - \text{keys}_{\text{with prediction}}}{\text{keys}_{\text{normal}}}$$

FUTURE WORK

I think that we can describe our plans to continue investigating topic modeling. Also, we should describe some of the other things we intend to investigate, such as style modeling and part-of-speech modeling.

CONCLUSIONS

It doesn't work as well as we'd hoped, so we'll have to try improving the model with further research. However, we feel it makes the predictions more appropriate, etc.

[2] The problem which this causes is that topics are now not only weighted by their topical similarity, but by their size, as topics with higher frequencies will tend to dominate the model. We are looking into the severity of this problem at the moment.

ACKNOWLEDGMENTS

We would like to thank the US Department of Education for funding this research under grant H113G040051, under the National Institute on Disability and Rehabilitation Research program. Also, thanks to Gregory Lesher for correspondence regarding his work.

REFERENCES

1. Bellegarda, Jerome. Large Vocabulary Speech Recognition with Multispan Language Models. *IEEE Trans. On Speech and Audio Processing*, 8(1): 2000.
2. Beukelman, D. R. and Mirenda, P. *Augmentative and alternative communication: Management of severe communication disorders in children and adults*. Baltimore: Paul H. Brookes Publishing Co., 1998.
3. Boggess, Lois. Two simple prediction algorithms to facilitate text production. *Proceedings of Applied Natural Language Processing*, 1988.
4. Carlberger, Alice, John Carlberger, Tina Magnuson, M. Sharon Hunnicutt, Sira Palazuelos-Cagigas, and Santiago Aguilera Navarro. Profet, A New Generation of Word Prediction: An Evaluation Study. *Proceedings of Natural Language Processing for Communication Aids*, 1997.
5. Copestake, Ann. Augmented and alternative NLP techinques for augmentative and alterative communication. *Proceedings of Natural Language Processing for Communication Aids*, 1997.
6. Fazly, Afsaneh and Graeme Hirst. Testing the Efficacy of Part-of-Speech Information in Word Completion. *Proceedings of the 10th Conference of the European Chapter of the Association for Computational Linguistics*, 2003.
7. Florian, Radu and David Yarowsky. Dynamic Nonlocal Language Modeling via Hierarchical Topic-Based Adaptation. *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, 1999.
8. Garay-Vitoria, Nestor and Julio González-Abascal. Intelligent Word-Prediction to Enhance Text Input Rate. *Proceedings of the second international conference on Intelligent User Interfaces*, 1997.
9. Hindle, Donald. Deterministic Parsing of Syntactic Non-fluencies. *Proceedings of the 21st Annual Meeting of the Association for Computational Linguistics*, 1983.
10. Jurafsky, Daniel and James Martin. *Speech and Language Processing*. Prentice Hall, Upper Saddle River NJ, 2000.
11. Katz, Slava. Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer. *IEEE Trans. On Acoustics, Speech, and Signal Processing*, 35(3): 1981.
12. Koester, H.H. and Levine, S.P. The Effect of a Word Prediction Feature on User Performance. *Augmentative and Alternative Communication*, 12(3): 1996, 155-168.
13. Lesher, Gregory and Gerard Rinkus. Domain-specific word prediction for augmentative communication. *Proceedings of the RESNA '02 Annual Conference.*
14. Lesher, Gregory, Bryan Moulton, and Jefferey Higgonbotham. Effects of ngram order and training text size on word prediction. *Proceedings of the RESNA ''99 Annual Conference.*
15. Lesher, Gregory, Bryan Moulton, Jeffery Higginbotham, and Brenna Alsofrom. Limits of human word prediction performance. *Proceedings of California State University Northridge conference, 2002*.
16. Mahajan, Milind, Doug Beeferman, and X. D. Huang. Improved topic-dependent language modeling using information retrieval techniques. *Proceedings. of the International Conference on Acoustics, Speech, and Signal Processing*, 1999.
17. Manning, Christopher and Hinrich Shütze. *Foundations of Statistical Natural Language Processing.* MIT Press, Cambridge MA, 2000.
18. Marcus, Mitchell, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2): 1993.
19. McCoy, Kathleen F. Interface and Language Issues in Intelligent Systems for People with Disabilities. In *Assistive Technology and Artificial Intelligence: Applications in Robotics, User Interfaces and Natural Language Processing*, Vibhu Mittal, Holly Yanco and John Aronis, Editors. Volume 1458, Lecture Notes in AI Series, Springer, 1998.
20. Newell, Alan, John Arnott, Lynda Booth, William Beattie, Bernadette Brophy, and Ian Ricketts. Effect of the "PAL" Word Prediction System on the Quality and quantity of Text Generation. *Augmentative and Alternative Communication*, Volume 8, 1992.
21. Newell, Alan, Stefan Langer and Marianne Kickey. The rôle of natural language processing in alternative and augmentative communication. *Natural Language Engineering,* 4(1): 1998, 1-16.
22. Seymore, Kristie and Ronald Rosenfeld. Using story topics for language model adaptation. *Proceedings of 5th European Conference on Speech Communication and Technology (Eurospeech)*, 1997.
23. Silfverberg, Miika, I. Scott MacKenzie, and Panu Korhonen. Predicting Text Entry Speed on Mobile Phones. *Proceedings of CHI 2000.*

24. Venkatagiri, H. S. Effect of window size on rate of communication in a lexical prediction AAC system. *Augmentative and Alternative Communication*, 10, 1994, 105-112.