Context Based Word Prediction for Texting[1] Language

Sachin Agarwal & Shilpa Arora
Language Technologies Institute
School of Computer Science
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213
{sachina, shilpaa}@cs.cmu.edu

Abstract

The use of digital mobile phones has led to a tremendous increase in communication using SMS. On a phone keypad, multiple words are mapped to same numeric code. We propose a Context Based Word Prediction system for SMS messaging in which context is used to predict the most appropriate word for a given code. We extend this system to allow informal words (short forms for proper English words). The mapping from informal word to its proper English words is done using Double Metaphone Encoding based on their phonetic similarity. The results show 31% improvement over the traditional frequency based word estimation.

Introduction

The growth of wireless technology has provided us with many new ways of communication such as SMS (Short Message Service). SMS messaging can also be used to interact with automated systems or participating in contests. With tremendous increase in Mobile Text Messaging, there is a need for an efficient text input system. With limited keys on the mobile phone, multiple letters are mapped to same number (8 keys, 2 to 9, for 26 alphabets). The many to one mapping of alphabets to numbers gives us same numeric code for multiple words.

Predictive text systems in place use the frequency-based disambiguation method and predict the most commonly used word above other possible words. T-9 (Text on 9-keys), developed by Tegic Communications, is one such predictive text technology used by LG, Siemens, Nokia Sony Ericson and others in their phones. iTap is another similar system developed and used by Motorola in their phones.

T-9 system predicts the correct word for a given numeric code based on frequency. This may not give us the correct result most of the time. For example, for code '63', two possible words are 'me' and 'of'. Based on a frequency list where 'of' is more likely than 'me', T-9 system will always predict 'of' for code '63'. So, for a sentence like *'Give me a box of chocolate'*, the prediction would be *'Give of a box of chocolate'*.

The sentence itself indeed gives us information about what should be the correct word for a given code. Consider the above sentence with blanks, "Give _ a box _ chocolate". According to the English grammar, it is more likely that 'of' comes after a noun 'box' than 'me' i.e. it is more likely to see the phrase "box of" than "box me". The algorithm proposed is an online method that uses this knowledge to correctly predict the word for a given code considering its previous context.

[1] SMS Text language

An extension of T-9 system called T-12 was proposed by (UzZaman, 2005). They extend the idea of T-9 to what we call informal language which is used in text messaging a lot. This includes abbreviations, acronyms, short forms of the words based on phonetic similarity (e.g. 'gr8' for 'great'). They use the Metaphone Encoding (Lawrence Philip's Metaphone Algorithm, 1990) technique to find phonetically similar words. And from among those phonetically similar words, they choose the appropriate word using string matching algorithms such as edit distance between the word and its normalized form. However, the edit distance measure also suggests some incorrect words such as 'create' for informal word 'gr8'. In the proposed method, the context information is used to choose the appropriate word.

Although the method has been proposed for a text messaging system, it is applicable in a number of other domains as well. The informal and formal (English vocabulary words) language mixture discussed here is also used in instant messaging and emails. The proposed method can also be used to convert a group of documents in informal language into formal language. These days, even (non-personal) discussions over emails/IM between friends, colleagues, students is done in a more informal language but if someone were to make use of these discussions formally, then the system can automatically do the conversion or suggest appropriate conversions.

Proposed Method

The proposed method uses machine learning algorithms to predict the current word given its code and previous word's Part of Speech (POS). The workflow of the system is as shown in Figure 1. The algorithm predicts the current word after training a Markov Model on Enron email corpus since short emails resemble SMS messages closely.

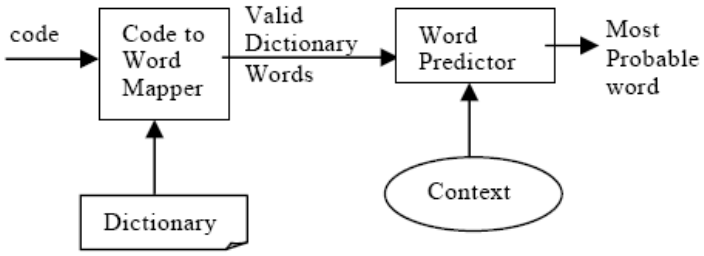


Figure 1: Workflow for Context Based Word Prediction System for formal lanaguge

The code, word and its POS are three random variables in the model. The dependency relationship between these variables can be modeled in different ways and we analyse and present a discussion of pros and cons of each modeling approach. The appropriate modeling of a given problem is a design issue and we present our detailed design approach in this paper for the given problem at hand. The first-order markov model with different representations of this relationship is discussed below. The bi-gram language model (Manning and Schütze, 1999) is used to predict the most probable word and POS pair given its code and previous word's POS.

Markov Model-I

In this first order Markov model (Figure 2), word is dependent on its code and the part of speech is dependent on the word and part of speech of previous word. Here, in a sentence, C_t refers to the numeric code for t^{th} word, W_t refers to t^{th} word and S_t refers to the part-of-speech of t^{th} word. Let W_{t+1} W_t be a sequence of words where W_{t+1} is to be predicted and W_t is known. Also, C_{t+1} and S_t are known. We need to learn the $P(W_{t+1}S_{t+1}/C_{t+1}S_t) = \frac{P(W_{t+1}C_{t+1}S_{t+1}S_t)}{P(C_{t+1}S_t)}$

The joint probability distribution using factorization theorem is given as, $P(W_{t+1}C_{t+1}S_{t+1}S_t) = P(S_{t+1}/W_{t+1}S_t)P(W_{t+1}/C_{t+1})P(C_{t+1})P(S_t)$

Hence, $P(W_{t+1}S_{t+1}/C_{t+1}S_t) = \frac{P(S_{t+1}/W_{t+1}S_t)P(W_{t+1}/C_{t+1})P(C_{t+1})P(S_t)}{P(C_{t+1}S_t)}$,

where $P(C_{t+1}S_t) = \sum_{W_{t+1},S_{t+1}} P(W_{t+1}C_{t+1}S_{t+1}S_t)$

$(W_{t+1}S_{t+1}) = \arg\max_{(W_{t+1}S_{t+1})} P(W_{t+1}S_{t+1}/C_{t+1}S_t)$

The word for which the above joint probability (word and its part of speech) is highest given its numeric code and previous word's part of speech is chosen. In order to predict first word of the sentence, we assume a null word preceding it, which denotes the beginning of the sentence. The null word also represents the context of the word as not every word can start a sentence.

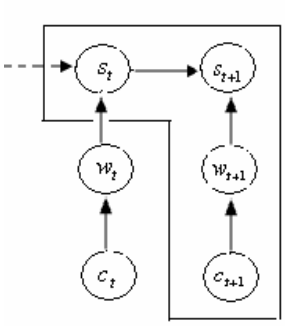


Figure 2: Markov Model-I for Context based word prediction

Markov Model-II

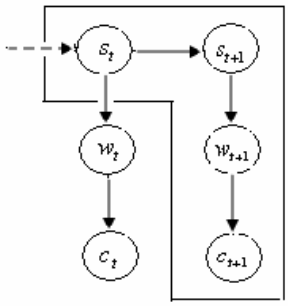


Figure 3: Markov Model-II for Context based word prediction

Here the code is dependent on its corresponding word and the word is dependent on its part of speech. This appears to be a more intuitive way of expressing the relationship from the user's perspective as when the user enters a code; he/she has the word in mind and not the code. The POS of consecutive words have a causal relationship which encodes the grammar of the sentence.

The joint probability distribution as calculated from the graphical model using factorization theorem is given as $P(W_{t+1}C_{t+1}S_{t+1}S_t) = P(S_{t+1}/S_t)P(W_{t+1}/S_{t+1})P(C_{t+1}/W_{t+1})P(S_t)$

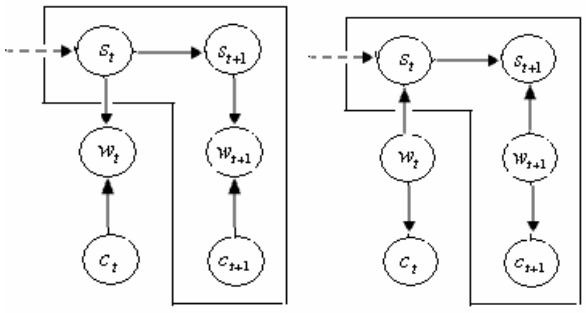
Hence, $P(W_{t+1}S_{t+1}/C_{t+1}S_t) = \frac{P(S_{t+1}/S_t)P(W_{t+1}/S_{t+1})P(C_{t+1}/W_{t+1})P(S_t)}{P(C_{t+1}S_t)}$

Where $P(C_{t+1}S_t) = \sum_{W_{t+1},S_{t+1}} P(W_{t+1}C_{t+1}S_{t+1}S_t)$ and $(W_{t+1}S_{t+1}) = \arg\max_{(W_{t+1}S_{t+1})} P(W_{t+1}S_{t+1}/C_{t+1}S_t)$

Variations of Markov Model-II:

The model above may be counter-intuitive because of the following reasons:

1. In the Markov Model-II, word determines the code but for the prediction system, code is given and that determines the possible words, so looking from the prediction system's perspective, it is more intuitive to have the code determine the possible word. This is depicted in the second model in Figure 4 (Markov Model-III)
2. In the Markov Model-II, part of speech determines the word; where as normally the word determines the part of speech. A variation of the above model in which the dependency between current word and its part of speech is reversed is as shown in Figure 4 (Markov Model-IV).



Markov Model-III Markov Model-IV

Figure 4: Variations in Markov Model-II for Context Based Word Prediction

For Markov Model-III., the joint Probability distribution from the graphical model would be
$P(W_{t+1}C_{t+1}S_{t+1}S_t) = P(S_{t+1}/S_t)P(W_{t+1}/S_{t+1},C_{t+1})P(C_{t+1})P(S_t)$

Hence, $P(W_{t+1}S_{t+1}/C_{t+1}S_t) = \frac{P(S_{t+1}/S_t)P(W_{t+1}/S_{t+1},C_{t+1})P(C_{t+1})P(S_t)}{P(C_{t+1}S_t)}$

Where $P(C_{t+1}S_t) = \sum_{W_{t+1},S_{t+1}} P(W_{t+1}C_{t+1}S_{t+1}S_t)$ and $(W_{t+1}S_{t+1}) = \arg\max_{(W_{t+1}S_{t+1})} P(W_{t+1}S_{t+1}/C_{t+1}S_t)$

For Markov Model-IV, the joint Probability distribution from the graphical model would be
$P(W_{t+1}C_{t+1}S_{t+1}S_t) = P(S_{t+1}/S_t,W_{t+1})P(W_{t+1})P(C_{t+1}/W_{t+1})P(S_t)$

Hence, $P(W_{t+1}S_{t+1}/C_{t+1}S_t) = \frac{P(S_{t+1}/S_t,W_{t+1})P(W_{t+1})P(C_{t+1}/W_{t+1})P(S_t)}{P(C_{t+1}S_t)}$

Where $P(C_{t+1}S_t) = \sum_{W_{t+1},S_{t+1}} P(W_{t+1}C_{t+1}S_{t+1}S_t)$ and $(W_{t+1}S_{t+1}) = \arg\max_{(W_{t+1}S_{t+1})} P(W_{t+1}S_{t+1}/C_{t+1}S_t)$

Support Vector Machines (SVMs)

SVM has been used in sequence tagging for predicting the POS sequence for a given word sequence. Hidden Markov Support Vector Machines (Altun, 2003) uses a combination of SVM

and Hidden Markov Model (Rabiner, 1989) for sequence tagging. SVM^{HMM} is implemented as a specialization of the SVM^{struct} package for sequence tagging.

In the given problem, the correct word is to be predicted. Using SVM for this purpose would require as many classes as number of words in the dictionary. The English dictionary has roughly around 100,000 words; SVM would need to learn classification for these many classes. To learn a good SVM classifier for 100,000 classes, sufficiently large number of examples is required for all the classes i.e. a large training dataset which covers words from all these classes but the training time for SVM grows exponentially with the number of training examples.

However, for the given problem of predicting the correct word for a given code, one classifier per code is really what we need to learn. But the number of codes can be very large as well (# of digits in code = #of letters in word). Hence, to use SVM for this problem, the number of codes needs to be limited. The features used for SVM are similar to parameters used in the above graphical models i.e. the POS tag of previous word and the given code. SVM^{HMM} was used for implementation.

Informal Language Model

The workflow for the Informal Language Model is as follows:

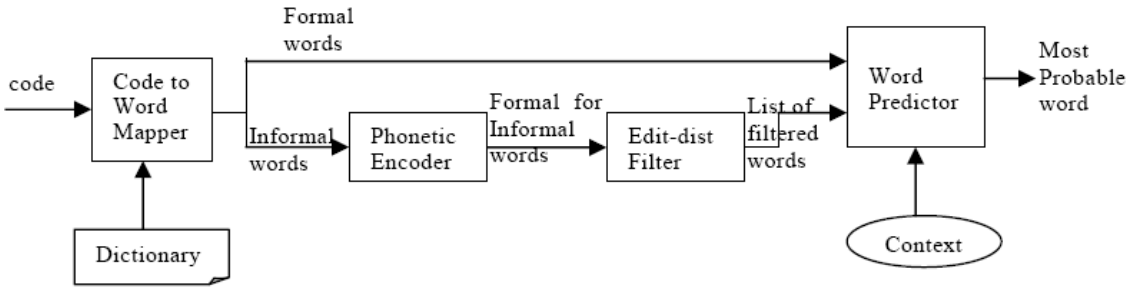


Figure 5: Workflow for Context Based Word Prediction System for Informal language

The input code is processed to generate all possible words corresponding to that code. These words are split into formal and informal words based on the dictionary look up. The phonetically similar formal words are generated for the informal words using Double Metaphone encoding. This gives a big list of possible formal words. The words in this list are filtered out based on their edit distance from the informal word. Levenshtein Distance is used as the edit distance measure. The final set of possible words for the given code includes its formal words and this filtered list. The trained model for the formal language discussed above is then used to predict the most probable word from this list. Preference is given to the formal words over this list of filtered words as normally the user would enter formal text with some commonly used informal words.

Even after filtering the words based on edit distance, the list still contains the words which are not commonly used in the Texting language. Threshold of two is used for edit distance. If the threshold is reduced further, some of the legitimate formal words are filtered out. For example, edit distance between ‘you’ and its commonly used short form ‘u’ is 2. Hence, reducing threshold further would filter out ‘you’. Therefore, an encoding scheme with better precision in the given domain is required. With the given encoding scheme, a lookup list was used to filter out the informal words that are not commonly used in Texting language. This lookup list was

generated from the SMS message corpus used and a lexicon of SMS available online called "MobiLingo".

NOTE: The informal words considered here are phonetic short forms of a formal word. We do not handle short forms that are abbreviation of a set of formal words, e.g. 'lol' is commonly used short form for 'laugh out loud', but they are not phonetically similar.

Experiments

The training was done on about 19,000 emails (Enron Email Data) and the testing was done on about 1900 emails, with each email consisting of 300 words on average. The English dictionary available on Linux system was used. Results were compared with frequency based estimation method using the frequency list from Wikipedia. The results are documented in Table 1. As can be seen the error (% of words incorrectly predicted) reduces by approximately 31% for Markov Model-I and 16.8% for Markov Model-II. The error for Markov Model-IV is similar to frequency based method and for Markov Model-III the error is more than frequency based.

# Training examples	# Test examples	Avg. % error with Markov Model				Avg. % error Freq. based prediction
		I	II	III	IV	
19000	1900	5.54%	6.69%	11.97%	8.05%	8.04%

Table 1: Test Results for Context Based Word Predicton System for formal language

Analysis of performance of Markov models

In Markov Model-II and Markov Model-III, the Part Of Speech (POS) of the current word is determined only by the POS of the previous word. However, the current word also plays an important role in determining the POS. As observed in the training data and is intuitive as well, the POS 'IN' (preposition) is more likely to have a POS 'CD' (Cardinal number) following it than a 'PRP' (Personal pronoun). E.g. CD follows IN – "About 20% increase in sales was observed this year" and PRP following IN – "They were concerned about me". But given a code "63", which maps to the number "63"and word "me", it is more likely that "me" comes after a preposition (like about) than a number "63". Thus, current word and previous POS together determine the current POS. This is modeled in Markov Model-I and Markov Model-IV.

Markov Model-III has a V-structure between current POS, word and code (Figure 6). According to this model, given the word, code and POS become dependent. However, in the given problem, once we know the word, code doesn't give us any additional information about POS i.e. POS becomes independent of the code given the word. And if the word is not observed, knowing the code increases the probability of POS tags corresponding to the words of that code. Thus, the causal relationship between code and POS is not modeled correctly in Markov Model-III and hence it performs worse than other models.

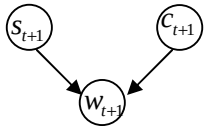


Figure 6: V-structure relationship for code, word and POS

In models Markov Model-II and Markov Model-IV, the word determines the code. However, given the word, code is deterministic i.e. there is only one possible code for a given word. But given a code, word corresponding to it is determined probabilistically based on the context. Also, for the predictive system, code is known and we need to find the most probable word for it. Thus, Markov Model-II and Markov Model -IV do not model the causal relationship between word and code appropriately and hence they perform worse than Markov Model -I. Given all the three analysis above, Markov Model-I models the given problem the best and as also observed it gives the best performance.

Our analysis of different ways of modeling the problem shows that it's very important that the causal relationship between different variables is modeled correctly to develop an efficient system. And this analysis may require the domain knowledge of the problem at hand.

Note: While calculating the error, we ignore the special characters and non-dictionary words. Also, we do not handle the date-time format, email address and hyperlinks. Non-dictionary words like Proper nouns, email addresses and hyperlinks can be unlimited and hence would not be handled by the system as is the case with the current mobile text messaging systems.

Performance of SVM

To assess how SVM performs in classifying the words for a given code, it was tested on 10 codes corresponding to a few very frequent English words. Comparison of SVM, graphical model and frequency method on these words is shown in Table 2.

SVM performs better than frequency method and reduces the average error (% of words incorrectly identified) by 18.62%. However, Markov Model-I outperforms SVM by reducing the error further by 35.75%. Markov model performs better than SVM because causal relationships between variables can be better modeled in a Markov model.

Words	SVM (c^2=0.1,e^3=0.5)	Markov Model-I	Frequency Method
63: (of, me)	27.43%	27.48%	27.48%
46: (in,go,io)	4.74%	5%	5%
43: (he,if)	24.99%	29.2%	76.42%
64448: (night,might)	36.84%	37.59%	63.15%
843: (the,tie,vie)	0.08%	0.08%	0.08%
84373: (there,these)	51.74%	46.95%	49.12%
8436: (then,them)	63.68%	18.24%	63.68%
66: (no,on)	90.21%	11.94%	88.89%
4283: (have, hate, gave,gate)	1.58%	1.57%	1.57%
87: (up,us)	33.11%	36.76%	35.57%
Average Error	33.44%	21.48%	41.10%

Table 2: Test Results for comparison of graphical model and SVM

[2] Weight for slack term
[3] Precision to which constraints are required to be satisfied by the solution

Performance of Model for Informal Language

For testing Informal Model, data set of about 850 SMS messages with informal language from (SMS test Data, IIT kharagpur) was used. Markov Models discussed above were tested on the informal data and their results were compared with frequency based model. Results are as shown in Table 3.

# Training examples	# Test examples	Avg. % error with Markov Model				Avg. % error Freq. based prediction
		I	II	III	IV	
19000	850	25.24%	25.94%	29.75%	26.88%	33.4%

Table 3: Test Results for Context Based Word Predicton System for formal language

Markov Model-I performs the best for informal language and it reduces the error by 22.33% over the frequency based method.

Conclusion and Future work

The Context Based Word Prediction system performs better than the traditional frequency based method. Different Markov models were analyzed to judge what best models the causal relationship between parameters. SVM^{HMM} model used for sequence tagging was found to be inappropriate for the given problem due to the large number of classes. The bi-gram model used can be extended to tri-gram or more but since SMS text messages are normally short sentences, a higher N-gram model wouldn't be useful. Phonetic encoding scheme with more precision in the given domain would help improve performance of the Informal Model.

Currently, we model the first order Markov dependency only between the POS of the consecutive words. Modeling this dependency among the consecutive words themselves might give an improvement in performance because certain word bi-grams are more likely than others. This might be a good extension to the current system.

In the current model, error made at the t^{th} word is propagated further in the sequence and hence the error for the current word also reflects the error made on the previous word (on the basis of which it was predicted). However, in a mobile messaging system, a user can actually correct the word if the word proposed by the system is wrong and hence it would be better to predict the current word based on the actual (correct) previous word.

For unseen words, a very low probability is assigned to them and probabilities of all the words for a given code are normalized. Applying a better smoothing method like the Good Turing estimates (Good, 1953 and Nadas, 1985) as in the Katz back-off model (Katz, 1987) should give better results.

Acknowledgements

We are thankful to Tom Mitchell, Eric Xing and Yifen Huang whose precious guidance helped us in proceeding with our ideas to their actual implementation.

References

UzZaman, N., Khan, M. (2005). T12: An Advanced Text Input System with Phonetic Support for Mobile Devices. In *2nd International Conference on Mobile Technology, Applications and Systems*, (pp.1-7)

Altun, Y., Tsochantaridis, I., & Hofmann, T. (2003). Hidden Markov Support Vector Machines. In *Proceedings of 20th International Conference on Machine Learning*.

Rabiner, L. R., (1989) A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. In *Proceedings of the IEEE*, 77 (2), p. 257–286.

Manning, C. D., Hinrich Schütze, (1999) Foundations of Statistical Natural Language Processing, *MIT Press*:. ISBN 0-262-13360-1.

Katz, S.M. (1987) Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer. In *IEEE Transactions on Acoustic, Speech and Signal Processing*, 35(3):400-401.

Good, I. J. (1953) The population frequencies of species and the estimation of population parameters. In *Biometrika*, vol. 40, no. 3 and 4, pp. 237- 264.

Nadas, A., (1985) On Turing's formula for word probabilities. In *IEEE Trans. Acoustic, Speech, Signal Processing*, vol. ASSP-33, pp. 1414-1416.

T-9 System. http://www.t9.com/

Lawrence Philip's Metaphone (1990). Algorithm (http://aspell.net/metaphone/)

Enron Email Data. http://www.cs.cmu.edu/%7Eeinat/datasets.html

SVM^{HMM}. http://www.cs.cornell.edu/People/tj/svm_light/svm_hmm.html

SVM^{STRUCT} . http://svmlight.joachims.org/

Double Metaphone, http://en.wikipedia.org/wiki/Double_Metaphone

Levenshtein Distance.http://www.merriampark.com/ld.htm

MobiLingo. http://www.mobimarketing.com/downloads/mlingo.pdf

Wikipedia Frequency List. http://en.wiktionary.org/wiki/Wiktionary:Frequency_lists

SMS test data, IIT Kharagpur. http://www.mla.iitkgp.ernet.in/~monojit/sms.html