Context-Sensitive Word Selection For Single-Tap Text Entry

Nick Hawes and John Kelleher
Common Sense Group, Media Lab Europe
{nick.hawes, john.kelleher}@medialabeurope.org

Abstract. Predictive text input using a single-tap entry method is currently the standard for text entry in the mobile domain. One problem facing this approach is which word to present to the user when more than one word matches an input sequence. The standard single-tap approach selects words using corpus occurrence frequencies, ignoring linguistic context. This can lead to the selection of words that are unrelated to the current input. In this paper we present an implementation of a context-aware predictive text framework that overrides the frequency-based selection model for words related to the current context.

1 Introduction

Mobile devices are playing an increasingly important role in day to day life, and most require text input (e.g. sending a text message on a mobile phone). The nature of these devices constrains the interface available for text input. This has lead to the development of *ambiguous keyboards* (first discussed in [1]). Such keyboards use one key to represent multiple inputs, and some method is required to select between them. On these keyboards, characters are typically grouped into sets and the sets are bound to a particular key. A popular (mobile phone) layout, and the one assumed for the remainder of this paper, is ABC bound to key 2, DEF to 3, GHI to 4, JKL to 5, MNO to 6, PQRS to 7, TUV to 8 and WXYZ to 9. Although we assume this mapping, it should be noted that the work presented in the following sections is relevant and applicable in any situation where an ambiguous keyboard is used to enter text.

There are currently two ambiguous keyboard input methods used on mobile devices: *single-tap* and *multi-tap*. In the multi-tap mode the input character cycles with every press of its related key. Using the above layout, the letter H can be entered by pressing key 4 twice. When the user presses a different key, or after a delay, the previous character is fixed in the input. In the single-tap method each key-press can represent any of its associated characters. Sequences of key-presses can then represent any word that can be constructed from the characters in the order they were entered. If more than one word is associated with a key-sequence (e.g. 7468 produces “pint” and “riot” amongst others) the user can cycle through the available set using another key on the keyboard. In terms of word-entering efficiency, single-tap is a considerable improvement over multi-tap, although this efficiency is dependant on: (i) whether the word wanted by the user is in the single-tap database and (ii) the order in which the words tied to a key-sequence are presented to the user. The first of these issues can be tackled through a careful choice of the corpus used to build the single-tap database. If we

assume that the database contains most of the words required by the user, then the issue of word selection order provides the biggest source of potential inefficiency for the user.

2 Predictive Text

The basic model for predictive text is based on a database of key-sequences each mapped to a set of words. We will refer to these sets of words as *sequence sets*. For example, using the previously introduced mapping, the sequence set for 227 will include "bap" and "bar". In single-tap prediction, sequence sets are sorted using corpus-based word frequencies. For example, the sequence set 843 would have "the" as its first member, followed by "tie" and "vie". When key-presses are input into the standard single-tap model, the most frequent word from the associated sequence set is displayed. Data presented in [2] suggests that in 95% of cases the desired word is presented to the user.

Ironically, the success rate of single-tap word prediction leads to one of its problems. When a system is regularly correct, users tend to rely on its correctness. When a word is incorrectly predicted, experienced users often fail to notice. In contrast, if an inexperienced user notices that the wrong word has been suggested, they can often be puzzled that the system got it wrong (due to them positing single-tap prediction with more intelligence than it actually possesses). The general problem faced by the single-tap approach, and addressed by this work, can be illustrated with an example. The key-sequence 228 produces the frequency-ordered sequence set: "act", "cat", and "bat". If the user entered the text "Yesterday a stray dog chased my " followed by the key-sequence 228, they may be surprised to see "act" as the suggested word. Admittedly it seems incongruous to criticise a system for functioning correctly, but there is a certain (common) sense in expecting the system to pick the correct word based on contextual clues. It is this problem that the work in this paper addresses.

3 A Common Sense Approach To Predictive Text

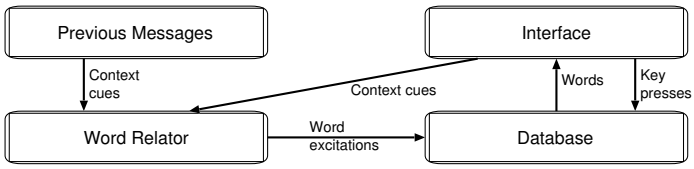

Figure 1: A framework for a context-sensitive predictive text system.

Given a *linguistic context* humans have the ability to select the most relevant word from a set of possible choices, because they use *common sense knowledge* to identify which of the possible words best fits the context. The standard single-tap approach to text entry has no notion of context. As such, it has no idea that certain words are related to a particular context. The thesis of this paper is that it is possible to use common sense knowledge to augment the standard single-tap approach in order to improve the percentage of words that are correctly presented first time from sequence sets. The remainder of this section will present a framework for constructing such a system to do this. The framework is depicted in Figure 1.

As with the standard approach, a database of sequence sets is central to the context-sensitive approach. In this approach the database is linked to a *word relator*. The word relator determines which words are related to the current context. Using the above example, it is the

word relator that informs the database that “cat” is related to “dog” and “chased”. To support the idea of certain words being related to a particular context, the standard single-tap database must be altered to support *word excitation*. When a word is related to the current context the word relator informs the database that it should be excited. To excite a word, the database should disregard word frequency data and push the word to the start of its sequence set. This will cause it to be returned first if its key-sequence is entered. Because the context will not remain constant, the database must gradually return the word to its original position in its sequence set after excitation has occurred. The final piece of the framework is the extraction of words to be used to determine linguistic context. There are two sources for these: the text being entered into the interface; and previous interactions with the framework (e.g. messages sent and received). One interesting question is which words from these bodies of text should be used as input to the word relator? Using every word will certainly lead to the database exciting words that are not particularly relevant to the current context. Using too few words may prevent the context being accurately represented by the words excited in the database.

4 Context-Sensitive Word Selection

To investigate this new approach to single-tap text input, we have implemented a prototype of the above framework. The following sections discuss word excitation in the database, recognition of linguistic context, and determining word relationships.

4.1 Sequence Set Excitation

In the traditional single-tap approach the sequence of words in a sequence set is fixed. In the context-sensitive approach the order can be changed through word excitation. When a word is excited in the database it is moved to the beginning of its sequence set. Once a suitable excitation period has passed, the word is returned to its frequency dependent position. In the prototype implementation, the word spends a fixed amount of time at each possible position above its original position in the sequence. This is intended to capture the fact that the word retains some level of excitation outside of its immediate context. The period of excitation used is currently one minute. This may appear short, but words that are strongly related to the current context get re-excited as the user enters text.

4.2 Context Cues

The first issue to investigate when attempting to determine the linguistic context of a piece of text is which parts of the text provide the most contextual information. It is desirable to do this for two reasons. First, if the word relator is queried with every input word it will have a lot of work to do. The computational overhead of determining the words related to an input word is not large (see Section 4.3), but if done regularly it could impact upon the interactivity of the system. Second, if words related to every possible input word are excited in the database, the system may lose the beneficial effects of exciting strongly related words by swamping the database with weakly related words. We will refer to the parts of text that we use to determine context as *context cues*. Talmy [3] notes that closed class words are mostly structural, whereas open class are mostly content-carrying. Consequently, we only use open class words as contextual cues, although this may be refined in future iterations of the system.

Build	Avg	Max	Min	Bst	Wrst	Err	Bst	Wrst	Eff
standard approach	83.16	91.67	73.59	-	-	-	-	-	-
0 words excluded	81.21	91.67	73.33	0	-6.31	0.47	-0.5	1.08	15
20 words excluded	82.41	91.67	73.59	0.79	-4.13	0.27	-0.5	1.12	15
50 words excluded	82.68	91.67	73.59	1.58	-3.04	0.21	-1.5	1.14	15
100 words excluded	83.15	91.67	73.59	1.48	-1.95	0.09	-1.67	1.33	13
150 words excluded	83.21	91.67	73.59	1.56	-1.56	-0.04	-1.67	1.33	13

Table 1: Results from the weblog corpus.

4.3 Word Relationships

Critical to our approach to single-tap text input is determining which words are related to an input word. In the prototype this is done using ConceptNet [4]. ConceptNet is a semantic network of common sense knowledge. Nodes in ConceptNet are concepts (e.g. “bottle” and “opening a door”) and are connected by relationships (e.g. “involves part” and “is a”). To get the words related to a particular input word, we query ConceptNet for the concept it describes and then add all of the concepts connected to this to the result set.

A critical issue with this approach is which words should be returned from the word relator. This concerns the interaction between the standard single-tap approach to text entry and the context-sensitive extension. In the standard approach, words that commonly occur are always suggested before words that occur less commonly. The context-sensitive approach may override this frequency ordering using word excitation. If the context-sensitive approach excites a word in a sequence set that also contains a very common word, then it is quite probable that the excited word may actually hinder text input. Overcoming this problem leads to a trade-off in the database: finding the appropriate number of commonly used words which should have their sequence sets excluded from excitation. Exclude too many and the context may not be adequately represented by the words excited in the database. Exclude too few and common words may be obscured from suggestion. A number of versions of the prototype that exclude different numbers of common words from excitation have been tested.

4.4 First Results

We evaluated the prototype implementation in terms of the percentage of words it correctly predicts as the first member of a sequence set in unconstrained text. This metric is augmented by the notion of selection error. This gives the average number of sequence set positions between the desired word using the standard approach and the context-sensitive approach.

The ideal test corpus for a text entry system is a collection of the type of texts that users will be entering with it. There is no widely available corpus of typical mobile device text, so we have developed some corpora with similar features. The first corpus used contained the latest entries from the first twenty weblogs reporting the top story on blogdex.net. The second contained the twenty most recent posts on boards.ie. Also, following [5], we used two corpora containing articles taken from cooking and wedding websites[1]. These topics were chosen because ConceptNet contains a great deal of knowledge about them.

The results from the system can be seen in Tables 1 to 4. The first column shows the version of the implementation used. The first row contains data for the standard approach and

[1] http://www.cooking.com and http://www.weddingchannel.com

Build	Avg	Max	Min	Bst	Wrst	Err	Bst	Wrst	Eff
standard approach	78.90	88.65	65.31	-	-	-	-	-	-
0 words excluded	76.56	85.59	60	0	-6.67	0.63	0	1.67	13
20 words excluded	78.14	88.46	66.67	2.04	-4.55	0.44	-0.17	2	10
50 words excluded	78.51	88.46	66.67	2.04	-2.26	0.38	-0.17	2	8
100 words excluded	78.79	88.46	66.67	4.08	-2.22	0.36	-0.75	2	7
150 words excluded	78.52	88.46	64.71	2.04	-5.88	0.01	-3	1	6

Table 2: Results from the message-board corpus.

Build	Avg	Max	Min	Bst	Wrst	Err	Bst	Wrst	Eff
standard approach	85.6	92.61	68.28	-	-	-	-	-	-
0 words excluded	84.26	90.21	66.94	2.86	-6.04	0.24	-0.43	0.98	20
20 words excluded	86.66	92.31	68.07	4.76	-2.96	-0.13	-1	0.97	20
50 words excluded	86.71	92.31	68.17	4.76	-2.71	-0.15	-1	0.97	19
100 words excluded	86.91	92.39	68.07	3.81	-1.36	-0.26	-1	0.89	20
150 words excluded	86.9	92.59	68.28	3.81	-1.23	-0.42	-1.11	0.75	19

Table 3: Results from the wedding corpus.

the subsequent rows contain data for versions of the context-sensitive approach, with each row presenting results with a different number of common words excluded from excitation. The second to fourth columns show the average, maximum and minimum percentages of words suggested correctly first time. This is followed by columns reporting the best and worst improvement of this percentage. The next three columns report the average extra selection key-presses required over the standard approach when the context-sensitive approach differs in word suggestion position (e.g. if the standard approach places a word in third position and the context-sensitive approach places it in second, then minus one extra key-press would be required). The final column reports the number of documents from the corpus on which the results of the context-sensitive approach differ from the standard approach.

In terms of average number of words suggested first across the data set, our context-sensitive approach generally offers a small improvement over the standard approach. The system performed better in the domains where ConceptNet has more knowledge. The weblog and message-board datasets are on various topics and are typically less consistent about a particular domain, so the word relator has a harder task finding related words using ConceptNet. This is reflected in the results from these datasets, where the context-sensitive approach has a negative effect on performance. Although the average improvements over the entire corpora were marginal, specific cases demonstrate a significant improvement over the standard approach. For example, the best results in Table 4 show improvements of almost ten percent.

When neither the standard or context-sensitive approaches place the desired word in the first position, the context-sensitive approach places it closer to the start of the list in the domains where ConceptNet is more knowledgeable. In the other domains the words are generally a little further down the list.

It is important to note that in the weblog domain, the performance of the system improves as more sequence sets containing commonly occurring words are excluded from excitation. This is because the exclusions prevent words that are not strongly related to the current context from obscuring the commonly occurring words.

To summarise, while ConceptNet has a lot of information about cooking and weddings, there are a great deal of subjects that it knows nothing about. This leads to the word relator

Build	Avg	Max	Min	Bst	Wrst	Err	Bst	Wrst	Eff
standard approach	78.78	88.16	60.32	-	-	-	-	-	-
0 words excluded	79.22	89.66	55.56	8.62	-5.74	-0.24	-1.67	1	18
20 words excluded	80.57	91.38	58.73	10.35	-2.94	-0.47	-2.2	1	16
50 words excluded	80.19	89.66	58.73	8.62	-2.45	-0.43	-2.2	1	15
100 words excluded	79.88	88.11	58.73	6.90	-1.96	-0.49	-2.5	1	16
150 words excluded	79.97	88.11	58.73	6.90	-1.96	-0.57	-2.5	1	15

Table 4: Results from the cooking corpus.

suggesting weakly related words that end up degrading the performance of the system (by overriding the successful word frequency data). Also, when the word relator is knowledgeable about a domain, it runs the risk of hampering the performance of the system by exciting weakly related words along with the strongly related words. We feel that the problem of ensuring the system applies the available knowledge in a focussed and specific way is related to our choice of contextual cues and the types of words we extract from ConceptNet. In the future we aim to integrate a part-of-speech tagger into the system to aid the exploration of this problem. We believe that restricting context cues and word excitation to particular classes of words (e.g. nouns) will potentially improve the system's performance. We also aim to use part-of-speech information, in addition to common sense knowledge, to dynamically order sequence sets.

5 Conclusion

We have developed an approach to applying common sense knowledge to the problem of selecting a word from a sequence set for a predictive text system. The approach extends the standard single-tap approach to make it context-sensitive. It works by overriding frequency-based sequence set orderings to excite words that are related to the current context. We presented some results from a prototype of this system that showed some success. However, when it had little knowledge about the domain the extension tended to interfere with the successful behaviour of the standard algorithm. When it had a wealth of common sense knowledge it offered improved performance over the standard approach. We feel the system's performance in these domains justifies the approach we have taken. In the analysis of the results we identified possible causes for the failure of the system and suggested possible ways in which they could be overcome in future iterations.

References

[1] I. H. Witten. *Principles of Computer Speech*. Academic Press, London, UK, 1982.

[2] M. Silfverberg, I. S. MacKenzie, and P Korhonen. Predicting text entry speed on mobile phones. In *Proceedings of the ACM Conference on Human Factors in Computing Systems, CHI 2000*, pages 9–16. ACM, 2000.

[3] L Talmy. *Towards a Cognitive Semantics - Vol 1*. The MIT Press, 2000.

[4] H. Liu and P. Singh. Commonsense reasoning in and over natural language. In *Proceedings of the 8th International Conference on Knowledge-Based Intelligent Information & Engineering Systems (KES'2004)*, 2004.

[5] Tom Stocky, Alexander Faaborg, and Henry Lieberman. A commonsense approach to predictive text entry. In *Proceedings of Conference on Human Factors in Computing Systems, CHI 2004*, Vienna, Austria, 2004.