Corpus-based stochastic finite-state predictive text entry for reduced keyboards: application to Catalan

Mikel L. Forcada
Departament de Llenguatges i Sistemes Informàtics
Universitat d’Alacant
E-03071 Alacant

*Funded by CICyT through projects TIC97-0941 and TIC2000-1599-C02-02.
Index
Index

Text entry in reduced keyboards
Index

Text entry in reduced keyboards

Predictive text entry in reduced keyboards
Index

Text entry in reduced keyboards

Predictive text entry in reduced keyboards

Stochastic dictionaries from corpora
Index

Text entry in reduced keyboards

Predictive text entry in reduced keyboards

Stochastic dictionaries from corpora

Using stochastic dictionaries for predictive-text entry
Index

Text entry in reduced keyboards

Predictive text entry in reduced keyboards

Stochastic dictionaries from corpora

Using stochastic dictionaries for predictive-text entry

Experimental evaluation
Index

Text entry in reduced keyboards

Predictive text entry in reduced keyboards

Stochastic dictionaries from corpora

Using stochastic dictionaries for predictive-text entry

Experimental evaluation

Concluding remarks
Text entry in reduced keyboards/1

Text services increasingly common for mobile phone users (SMS, chat, WAP).
Text entry in reduced keyboards/1

Text services increasingly common for mobile phone users (SMS, chat, WAP).

... but mobile phones have a very small keyboard!
Text entry in reduced keyboards/2

A typical mobile phone keyboard

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>'</td>
<td>ABC</td>
<td>DEF</td>
</tr>
<tr>
<td>.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>GHI</td>
<td>JKL</td>
<td>MNO</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>PQRS</td>
<td>TUV</td>
<td>WXYZ</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>*</th>
<th>0</th>
<th>#</th>
</tr>
</thead>
<tbody>
<tr>
<td>next</td>
<td>.</td>
<td>space</td>
</tr>
</tbody>
</table>
Text entry in reduced keyboards/3

Traditional text-entry method: multitap


Slow, inconvenient, error-prone

Typically about 2 keystrokes per letter
Predictive text-entry in reduced keyboards/1

The alternative is predictive text entry. It is available for major languages in some mobile phones.
Predictive text-entry in reduced keyboards

The alternative is predictive text entry. It is available for major languages in some mobile phones.

Approaches one keystroke per letter

Predictive text-entry in reduced keyboards/2

Key sequences are highly ambiguous.

[2][2][7][2]: $3 \times 3 \times 4 \times 3 = 108$ combinations:
Predictive text-entry in reduced keyboards/2

Key sequences are highly ambiguous.

[2][2][7][2]: $3 \times 3 \times 4 \times 3 = 108$ combinations:

aapa aapb aapc aaqa aaqb aaqc aara aarb aarc aasa aasb aasc abpa abpb abpc abqa abqb abqc abra abrb abrc absa absb absc acpa acpb acpc acqa acqb acqc acra acrb acrc acsa acsb acsc bapa bapb bapc baqa baqb baqc bara BARP barp BARC basp BASC bbpa bbpb bbpc bbqa bbqb bbqc bbra bbrc bbrc bbsa bbsb bbsc bcpa bcpb bcpc bcqa bcqb bcqc bcra bcrb bcrc bcsa bcsb bcsC CAPA capb capc caqa caqb caqc CARA carb carb CARC CASA casb CASC cbpa cbpb cbpc cbqa cbqb cbqc cbra cbrb cbrC cbsa cbsb cbsC cCpa cCpb cCpc cCqa cCqb cCQC cCra cCrb cCrc cCSa cCSb cCSc
Predictive text-entry in reduced keyboards/3

Approach 1: letter sequence probabilities (e.g., bigrams, trigrams).

\[ p(\text{casa}) > p(\text{barb}) \gg p(\text{caqc}) \]
Predictive text-entry in reduced keyboards/3

Approach 1: letter sequence probabilities (e.g., bigrams, trigrams).

\[ p(\text{casa}) > p(\text{barb}) > p(\text{caqc}) \]

Approach 2: look sequence up in a frequency-annotated dictionary

\[
\begin{array}{ll}
\ldots & \ldots \\
\text{casa} & 389 \\
\ldots & \ldots \\
\text{barb} & 1 \\
\ldots & \ldots \\
\end{array}
\]

*(caqc not in dictionary)*
Predictive text-entry in reduced keyboards/4

Commercial products fall in two groups:
Commercial products fall in two groups:

- Letter sequence probabilities: Eatoni’s Wordwise/Letterwise
Commercial products fall in two groups:

- Letter sequence probabilities: Eatoni’s Wordwise/Letterwise
- Frequency-annotated dictionaries: Tegic’s T9 (e.g., on Nokia phones), Zi’s eZiText.
Commercial products fall in two groups:

- Letter sequence probabilities: Eatoni’s Wordwise/Letterwise
- Frequency-annotated dictionaries: Tegic’s T9 (e.g., on Nokia phones), Zi’s eZiText.

Dictionary strategy used for Catalan here.
A word on “word completion”:

Zi’s eZiText suggests one or more likely completions (for the currently available prefix):

[3][4][2][2][4]: dicci… onari?

Hitting a ”complete” key accepts the suggestion

May improve keystroke savings
Stochastic dictionaries from corpora/1

A stochastic (deterministic) finite-state automaton (SFSA) is

\[(Q, \Sigma, \delta, p, q_I, \pi)\]
Stochastic dictionaries from corpora/1

A stochastic (deterministic) finite-state automaton (SFSA) is

\[(Q, \Sigma, \delta, p, q_I, \pi)\]

\(Q\), set of states
Stochastic dictionaries from corpora/1

A stochastic (deterministic) finite-state automaton (SFSA) is

\[(Q, \Sigma, \delta, p, q_I, \pi)\]

\(Q\), set of states (\(q_I\) : initial state)
A stochastic (deterministic) finite-state automaton (SFSA) is

\[ (Q, \Sigma, \delta, p, q_I, \pi) \]

- \( Q \), set of states (\( q_I \) : initial state)
- \( \Sigma \), alphabet
Stochastic dictionaries from corpora/1

A stochastic (deterministic) finite-state automaton (SFSA) is

$$(Q, \Sigma, \delta, p, q_I, \pi)$$

- $Q$, set of states ($q_I$ : initial state)
- $\Sigma$, alphabet
- $\delta : Q \times \Sigma \rightarrow Q$, next-state function
Stochastic dictionaries from corpora/1

A stochastic (deterministic) finite-state automaton (SFSA) is

\((Q, \Sigma, \delta, p, q_I, \pi)\)

\(Q\), set of states (\(q_I\) : initial state)
\(\Sigma\), alphabet
\(\delta : Q \times \Sigma \rightarrow Q\), next-state function
\(p : Q \times \Sigma \rightarrow [0, 1]\), probability of transition
A stochastic (deterministic) finite-state automaton (SFSA) is

\[(Q, \Sigma, \delta, p, q_I, \pi)\]

- \(Q\), set of states (\(q_I\) : initial state)
- \(\Sigma\), alphabet
- \(\delta : Q \times \Sigma \to Q\), next-state function
- \(p : Q \times \Sigma \to [0, 1]\), probability of transition
- \(\pi : Q \to [0, 1]\), probability of ending word in state
Stochastic dictionaries from corpora/2

A corpus $S$ (vocabulary $W$) may be easily turned into an SFSA.

The skeleton is a trie:
Stochastic dictionaries from corpora/2

A corpus $S$ (vocabulary $W$) may be easily turned into an SFSA.

The skeleton is a trie:

States are the prefixes of words:
Stochastic dictionaries from corpora/2

A corpus $S$ (vocabulary $W$) may be easily turned into an SFSA.

The skeleton is a trie:

States are the prefixes of words: $Q = \text{Pr}(W)$
A corpus $S$ (vocabulary $W$) may be easily turned into an SFSA.

The skeleton is a trie:

States are the prefixes of words: $Q = \Pr(W)$

The initial state:
Stochastic dictionaries from corpora/2

A corpus $S$ (vocabulary $W$) may be easily turned into an SFSA.

The skeleton is a trie:

States are the prefixes of words: $Q = \Pr(W)$
The initial state: $q_I = \epsilon$
Stochastic dictionaries from corpora/2

A corpus $S$ (vocabulary $W$) may be easily turned into an SFSA.

The skeleton is a trie:

States are the prefixes of words: $Q = \text{Pr}(W)$
The initial state: $q_I = \epsilon$
Next-state function:
Stochastic dictionaries from corpora/2

A corpus $S$ (vocabulary $W$) may be easily turned into an SFSA.

The skeleton is a trie:

States are the prefixes of words: $Q = \Pr(W)$

The initial state: $q_I = \epsilon$

Next-state function:

$$\delta(q, a) = qa \quad \forall q, qa \in Q$$
Stochastic dictionaries from corpora/3

Then, the trie is made probabilistic.
Stochastic dictionaries from corpora/3

Then, the trie is made probabilistic.

Prefix frequencies from (smoothed) relative frequencies $f_S(w)$ in corpus $S$: 
Then, the trie is made probabilistic.

Prefix frequencies from (smoothed) relative frequencies $f_S(w)$ in corpus $S$:

$$c_S(q) = \sum_{z \in \Sigma^*: qz \in W} f_S(qz)$$
Then, the trie is made probabilistic.

Prefix frequencies from (smoothed) relative frequencies $f_S(w)$ in corpus $S$:

$$c_S(q) = \sum_{z \in \Sigma^*: qz \in W} f_S(qz)$$

used to estimate probabilities:
Then, the trie is made probabilistic.

Prefix frequencies from (smoothed) relative frequencies $f_S(w)$ in corpus $S$:

$$c_S(q) = \sum_{z \in \Sigma^*: qz \in W} f_S(qz)$$

used to estimate probabilities:

$$\pi(q) = \frac{f_S(q)}{c_S(q)} \quad p(q, a) = \frac{c_S(qa)}{c_S(q)}$$

The resulting trie can be compacted (merging subtrees having similar probability structure). Not in this work.
Stochastic dictionaries from corpora/4

One can define most probable continuation at each state:
One can define most probable continuation at each state:

\[ C(q) = \arg\max_{z \in \Sigma^*} f_S(qz). \]
Stochastic dictionaries from corpora/4

One can define most probable continuation at each state:

\[ C(q) = \arg\max_{z \in \Sigma^*} f_S(qz). \]

If memory is available, it may be precomputed and stored.
Using stochastic dictionaries for predictive-text entry/1

Keypad $K = \{1, 2, \ldots, 9\}$.

Letters in key $k$: $\Sigma_k \subset \Sigma$.
Using stochastic dictionaries for predictive-text entry/2

[User interface]

Blind keying (looking at the keyboard):

If we want barcelona:


we have been lucky!
Using stochastic dictionaries for predictive-text entry/3

[User interface]

Blind keying (looking at the keyboard):

But, if we want cara,

[2] [2] [7] [2] =barc-

a prefix of barcelona happens to be more likely. More options?
Using stochastic dictionaries for predictive-text entry/4

[User interface]

Now, looking at the screen:
Using stochastic dictionaries for predictive-text entry/4

[User interface]

Now, looking at the screen:

[2]
Using stochastic dictionaries for predictive-text entry/4

[User interface]

Now, looking at the screen:

[2]          a
Using stochastic dictionaries for predictive-text entry/4

[User interface]

Now, looking at the screen:

[2] a
[2] [2]
Using stochastic dictionaries for predictive-text entry/4

[User interface]

Now, looking at the screen:

[2]
[2] [2]  a
c a
Using stochastic dictionaries for predictive-text entry/4

[User interface]

Now, looking at the screen:

[2]  a
[2] [2] [7]
Using stochastic dictionaries for predictive-text entry/4

[User interface]

Now, looking at the screen:

[2] a
Using stochastic dictionaries for predictive-text entry/4

[User interface]

Now, looking at the screen:

```
[2]   a
```
Using stochastic dictionaries for predictive-text entry/4

[User interface]

Now, looking at the screen:

```
[2]       a
[2] [2] [7] [2] barc-
```
Using stochastic dictionaries for predictive-text entry/4

[User interface]

Now, looking at the screen:

[2]  a
Using stochastic dictionaries for predictive-text entry/4

[User interface]

Now, looking at the screen:

```
[2]  a
[2] [2] [7] [2]  barc-
```
Using stochastic dictionaries for predictive-text entry/4

[User interface]

Now, looking at the screen:

```
[2]
[2] [2]
[2] [2] [7]
```

```
a
ca
bar
barc-
NO!
casa
```
Using stochastic dictionaries for predictive-text entry/4

[User interface]

Now, looking at the screen:

[2]  a
Using stochastic dictionaries for predictive-text entry

[User interface]

Now, looking at the screen:

```
[2]        a
[2] [2] [7] [2] barc-
```

NO! NO!
Using stochastic dictionaries for predictive-text entry/4

[User interface]

Now, looking at the screen:

[2]  a
[2] [2] [7] [2] [*] [*]  barç-
Using stochastic dictionaries for predictive-text entry/4

[User interface]

Now, looking at the screen:

```
[2] a
```
Using stochastic dictionaries for predictive-text entry/4

[User interface]

Now, looking at the screen:

```
[2] a
```
Using stochastic dictionaries for predictive-text entry/4

[User interface]

Now, looking at the screen:

\[
\begin{align*}
&[2] & a \\
&[2] [2] & ca \\
&[2] [2] [7] & bar \\
&[2] [2] [7] [2] & barc- \quad \text{NO!} \\
&[2] [2] [7] [2] [+] & casa \quad \text{NO!} \\
&[2] [2] [7] [2] [+] [+] & barç- \quad \text{NO!} \\
&[2] [2] [7] [2] [+] [+] [+] & cara
\end{align*}
\]
Using stochastic dictionaries for predictive-text entry/4

[User interface]

Now, looking at the screen:

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>[2]</td>
<td>[2]</td>
<td>a</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Using stochastic dictionaries for predictive-text entry/4

[User interface]

Now, looking at the screen:

```
[2]  a
[2] [2] [7] [2]  barc-
[2] [2] [7] [2] [+]  casa  NO!
```
Using stochastic dictionaries for predictive-text entry/4

[User interface]

Now, looking at the screen:

```
[2]    a
```
Using stochastic dictionaries for predictive-text entry/4

[User interface]

Now, looking at the screen:

```
[2]  a
```
Using stochastic dictionaries for predictive-text entry/5

[User interface]

Suggested completions (à la eZiTexT) decrease the number of keystrokes. Suppose we want cerdanyola
Using stochastic dictionaries for predictive-text entry/5

[User interface]

Suggested completions (à la eZiTexT) decrease the number of keystrokes. Suppose we want cerdanyola

[2]
Using stochastic dictionaries for predictive-text entry/5

[User interface]

Suggested completions (à la eZiTexT) decrease the number of keystrokes. Suppose we want cerdanyola

[2] a
Using stochastic dictionaries for predictive-text entry/5

[User interface]

Suggested completions (à la eZiTexT) decrease the number of keystrokes. Suppose we want cerdanyola

[2]  a  a?
Using stochastic dictionaries for predictive-text entry/5

[User interface]

Suggested completions (à la eZiTexT) decrease the number of keystrokes. Suppose we want cerdanyola

[2] a a?
[2] [3]
Using stochastic dictionaries for predictive-text entry/5

[User interface]

Suggested completions (à la eZiTeXT) decrease the number of keystrokes. Suppose we want cerdanyola

[2] a a?
[2] [3] ce
Using stochastic dictionaries for predictive-text entry/5

[User interface]

Suggested completions (à la eZiTexT) decrease the number of keystrokes. Suppose we want cerdanyola

| [2] | a | a? |
| [2] [3] | ce | cent? |
Using stochastic dictionaries for predictive-text entry/5

[User interface]

Suggested completions (à la eZiTexT) decrease the number of keystrokes. Suppose we want cerdanyola

[2] a a?
[2] [3] ce cent?
Using stochastic dictionaries for predictive-text entry/5

[User interface]

Suggested completions (à la eZiTExt) decrease the number of keystrokes. Suppose we want cerdanyola

```
[2]                a       a?
[2] [3]            ce      cent?
[2] [3] [7]        cer-
```
Using stochastic dictionaries for predictive-text entry/5

[User interface]

Suggested completions (à la eZiTexT) decrease the number of keystrokes. Suppose we want cerdanyola

[2] a a?
[2] [3] ce cent?
Using stochastic dictionaries for predictive-text entry/5

[User interface]

Suggested completions (à la eZiTexT) decrease the number of keystrokes. Suppose we want cerdanyola

\[
\begin{array}{ccc}
\end{array}
\]
Using stochastic dictionaries for predictive-text entry/5

[User interface]

Suggested completions (à la eZiTexT) decrease the number of keystrokes. Suppose we want cerdanyola

<table>
<thead>
<tr>
<th>2</th>
<th>3</th>
<th>7</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>a?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ce</td>
<td>cent?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cer-</td>
<td>cert?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ade-</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Using stochastic dictionaries for predictive-text entry/5

[User interface]

Suggested completions (à la eZiTeXT) decrease the number of keystrokes. Suppose we want cerdanyola

\[
\begin{array}{ll}
[2] & a \\
[2] [3] & ce \\
[2] [3] [7] [3] & adre-
\end{array}
\]

a?  cent?
ce  cert?
cer-  certa?
Using stochastic dictionaries for predictive-text entry/5

[User interface]

Suggested completions (à la eZiTeXT) decrease the number of keystrokes. Suppose we want cerdanyola

```
```

```
a               a?
ce             cent?
cer-           cert?
adre-         adreça?
```
Using stochastic dictionaries for predictive-text entry/5

[User interface]

Suggested completions (à la eZiTexT) decrease the number of keystrokes. Suppose we want cerdanyola

```
[2]  a
[2] [3]  ce
[2] [3] [7]  cer-
[2] [3] [7] [3]  adre-
[2] [3] [7] [3] [2]  adreça-
```
Using stochastic dictionaries for predictive-text entry/5

[User interface]

Suggested completions (à la eZiTexT) decrease the number of keystrokes. Suppose we want cerdanyola

```
[2]     a           a?
[2] [3] ce          cent?
```
Using stochastic dictionaries for predictive-text entry/5

[User interface]

Suggested completions (à la eZiTexT) decrease the number of keystrokes. Suppose we want cerdanyola

[2]  
[2] [3]  
[2] [3] [7]  
[2] [3] [7] [3]  

a  
ce  
cer-  
adre-  
adreç-  
a?  
cent?  
cert?  
adreça?  
adreça?
Using stochastic dictionaries for predictive-text entry/5

[User interface]

Suggested completions (à la eZiTexT) decrease the number of keystrokes. Suppose we want cerdanyola

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>a?</td>
<td>a?</td>
<td>ce</td>
<td>cent?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cer-</td>
<td>cer?</td>
<td>adre-</td>
<td>adreça?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>adreça?</td>
<td>adreça?</td>
<td>cerdan-</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Using stochastic dictionaries for predictive-text entry/5

[User interface]

Suggested completions (à la eZiTeXT) decrease the number of keystrokes. Suppose we want cerdanyola

\[
\begin{array}{ll}
[2] & a \\
[2] [3] & ce \\
& adreça? \\
& cerdanyola? \\
\end{array}
\]

YES!
Using stochastic dictionaries for predictive-text entry/5

[User interface]

Suggested completions (à la eZiTeXT) decrease the number of keystrokes. Suppose we want cerdanyola

[2] a a?
[2] [3] ce cent?

Using stochastic dictionaries for predictive-text entry

[User interface]

Suggested completions (à la eZiTexT) decrease the number of keystrokes. Suppose we want cerdanyola

[2]
[2] [3]
[2] [3] [7]
[2] [3] [7] [3]

a
ce
cer-
adre-
adreç-
cerdan-

a?
cent?
cert?
adreça?
adreça?
cerdanyola?

YES!

Using stochastic dictionaries for predictive-text entry/5

[User interface]

Suggested completions (à la eZiTexT) decrease the number of keystrokes. Suppose we want cerdanyola

\begin{itemize}
  \item [2] a
  \item [2] [3] ce
  \item [2] [3] [7] cer-
  \item [2] [3] [7] [3] cert?
  \item [2] [3] [7] [3] [2] adre-
  \item [2] [3] [7] [3] [2] [6] adreça?
  \item [2] [3] [7] [3] [2] [6] cerdan-
  \item [2] [3] [7] [3] [2] [6] [<complete>] cerdanyola (validated)
\end{itemize}
Using stochastic dictionaries for predictive-text entry/6

[User interface]

Exceptions:

- No prefix for key sequence? BEEP! and fallback to traditional multitap.

- Can't find word with [*]? Delete and use traditional.
Using stochastic dictionaries for predictive-text entry/7

[Computation] For a keystroke sequence \( v \), compute the set of candidate prefixes:

\[
\text{cand}(\epsilon) = \{\epsilon\}
\]

\[
\text{cand}(vk) = \{x\sigma \in Q : x \in \text{cand}(v), \sigma \in \Sigma_k\}
\]

and use probabilities to sort:

\[
P(\epsilon) = 1; \quad P(q\sigma) = P(q)p(q, \sigma).
\]

If \( \text{cand}(vk) = \emptyset \), store \( \text{cand}(v) \) for selection and traditional continuation.

Show incomplete prefixes with dangling hyphen: \textbf{barc}-
Experimental evaluation/1

900 000 words from Catalan newspaper *Avui* (November 2000).

Remove hapax legomena: 25000 words, 68000 prefixes.

Corpus frequencies for words versus number of different words for the corresponding keystroke sequence:

<table>
<thead>
<tr>
<th>Ambiguity</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>51.5%</td>
</tr>
<tr>
<td>2</td>
<td>13.6%</td>
</tr>
<tr>
<td>3</td>
<td>24.5%</td>
</tr>
<tr>
<td>4</td>
<td>6.1%</td>
</tr>
<tr>
<td>&gt; 4</td>
<td>5.0%</td>
</tr>
</tbody>
</table>
Experimental evaluation/2

Fraction of words correctly reproduced by selecting the most frequent word associated to the corresponding keypad sequence: 86.0%.

(A random selection would have yielded 68.7% correct).
Experimental evaluation/3

Very simplified: ergonomic and attentional issues not considered.

Additional simplification of user’s behavior:

• Continuations are used as soon as possible

• The “next option” key is used only after keying the whole word.

Number of keypresses per letter averaged over a corpus.
Experimental evaluation/4

Results:

<table>
<thead>
<tr>
<th>Text</th>
<th>keystrokes per letter</th>
<th>new, no compl.</th>
<th>new, with compl.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>traditional</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Training text</td>
<td>2.019</td>
<td>1.083 (53.6%)</td>
<td>0.898 (44.5%)</td>
</tr>
<tr>
<td>More news</td>
<td>2.018</td>
<td>1.116 (55.3%)</td>
<td>0.940 (46.6%)</td>
</tr>
<tr>
<td>Synthetic messages</td>
<td>1.938</td>
<td>1.347 (69.5%)</td>
<td>1.256 (64.8%)</td>
</tr>
</tbody>
</table>

Note: corpus bias (newspaper)!

But: Message corpora hard to build (privacy, language identification, spelling, personal abbreviations).
Concluding remarks/1

Main conclusions:
Concluding remarks/1

Main conclusions:

- Predictive text entry system easily built from text corpora.
Concluding remarks/1

Main conclusions:

- Predictive text entry system easily built from text corpora.

- Keystroke reduction distinct even for out-of-corpus text.
Concluding remarks/2

Future work:
Concluding remarks/2

Future work:

- Use existing dictionaries in addition to corpus.
Concluding remarks/2

Future work:

- Use existing dictionaries in addition to corpus.

- Improve treatment of forms containing hyphens and apostrophes.
Concluding remarks/2

Future work:

- Use existing dictionaries in addition to corpus.
- Improve treatment of forms containing hyphens and apostrophes.
- Improve study using a more realistic interface and setting.