Lecture 23: Spelling Correction

CSA3202 Human Language Technology

Mike Rosner, Dept ICS
December 2012

Contents

Acknowledgement

- Jurafsky & Martin: Speech and Language Processing (2000) [1], Chapter 5.
- See also `www.sultry.arts.su.edu.au/links/statnlp.html` for resources on statistical NLP

Speech Recognition, POS Tagging, Spelling Correction

- Despite apparent differences, these problems share many underlying similarities.
- All are concerned with the problem of accepting a string of input symbols and mapping them to possible sequences of output symbols.

Domain	**Input Sequence Type**	**Output Sequence Type**
Speech Recognition	Phones	Characters
POS-Tagging	Words	POS Tags
Spelling Correction	Characters	Characters

Spelling Correction

Kukich [3] breaks the field down into three increasingly broader problems:

- Detection of non-words (e.g. *graffe*).
- Isolated word error correction (e.g. *graffe* $\Rightarrow$ *giraffe*).
- Context dependent error detection and correction where the error may result in a valid word (e.g. *there* $\Rightarrow$ *three*) .

Spelling Error Patterns

According to Damereau (1964) 80% of all misspelled words are caused by **single-error misspellings** which fall into the following categories (for the word *the*)

- Insertion (*ther*).
- Deletion (*th*).
- Substitution (*thw*).
- Transposition (*hte*).

Because of this study, much subsequent research focused on the correction of single error misspellings.

Causes of Spelling Errors

Keyboard Based

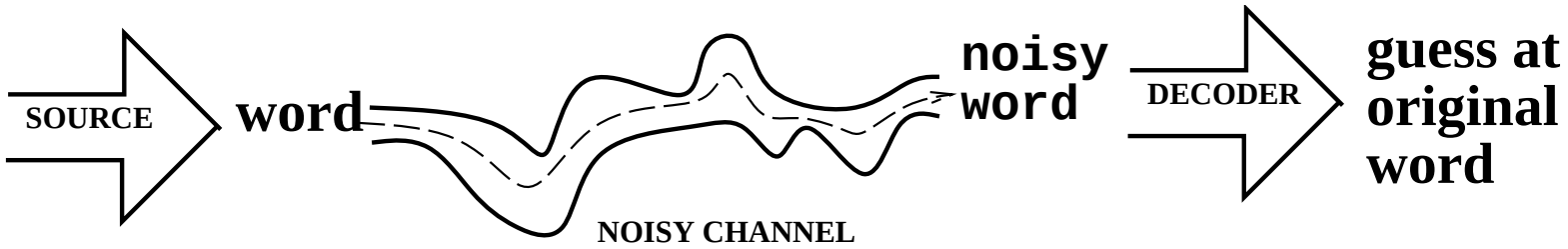
- Immediately adjacent keys in the same row of the keyboard (50% of the novice subsitutions (31% of all substitutions).
- Hitting corresponding key on opposite side of keyboard.
- 83% novice and 51% overall were keyboard errors

Cognitive

- Phonetic *seperate* - *separate*
- Homonym *there* - *their*.

OCR: mainly visual similarity (e.g. *m* - *rn*; substitutions; space deletions or insertions; failures.

Noisy Channel Model



- Language is generated and passed through a noisy channel.
- Resulting noisy data are received
- Goal is to recover the original data from the noisy data.
- Same model can be used in diverse areas of language processing e.g. spelling correction, morphological analysis, pronunciation modelling, machine translation.
- Metaphor comes from Jelinek's (1976) work on speech recognition, but algorithm is a special case of Bayesian inference (1763[1]).

[1] Bayes' friend Richard Price edited and presented this work in 1763, after Bayes' death, as *An Essay towards solving a Problem in the Doctrine of Chances*

Bayesian Classification Applied To Spelling Correction

- The noisy channel approach was first suggested by Kernighan, Church and Gale [2] and incorporated into the `correct` program which:
  - takes words rejected by the Unix *spell* progam. A word is rejected if it does not belong to an online wordlist (the dictionary);
  - generates a list of potentially correct words;
  - ranks them according to the Bayes equation
  - picks the one with the highest rank.
- We will follow the correction of the word *acress* which proceeds in two steps, proposing candidates, and ranking candidates.

Proposing Candidates

- Assume that correct word will differ from the misspelling by a single insertion, deletion, substitution or transposition.
- The candidate list is generated by applying any single transformation yielding a word occurring in a large online dictionary
- For instance, the typo *acress* yields: actress (d), cress (i), caress (t), access (s), across (s), acres (i), acres (i)

		Correct	Error	Position	
Error	Correction	Letter	Letter	(Letter #)	Type
acress	actress	t		2	deletion
acress	cress		a	0	insertion
acress	caress	ca	ac	0	transposition
acress	access	c	r	2	substitution
acress	across	o	e	3	substitution
acress	acres		s	5	insertion
acress	acres		s	4	insertion

Ranking Candidates

The second stage scores each correction. Let O be the typo and c range over a set C of candidate corrections. The most likely correction ĉ is then

$$\text{argmax}_{c \in C} P(c \mid O)$$

which by Bayes' rule is equivalent to

$$\hat{c} = \text{argmax}_{c \in C} P(O \mid c) P(c)$$

The prior probability P(c) can be estimated by

- Counting how often the word c appears in the corpus
- Normalising the count by dividing it by the number N of words in the corpus.

Smoothing

- Zero counts can cause problems,
- We add .5 to all counts (Laplace smoothing).
- Having done this, we must compensate by adding 0.5*V to the denominator for each word V in the vocabulary so that

$$P(c) = \frac{C(c) + 0.5}{N + 0.5V}$$

- The rationale is that by adding .5 to each word type we have effectively increased the size of the corpus by exactly .5V tokens.
- What is the effect on existing probabilities?

Prior Probability

Using the formula above, the prior probabilities come out as follows:

c	freq(c)	P(c)
actress	1343	.0000315
cress	0	.000000014
caress	4	.0000001
access	2280	.000058
across	8436	.00019
acres	2879	.000065

Computing the Likelihood Term $P(O \mid c)$

The likelihood term $P(O \mid c)$ is difficult if not impossible to *predict* in the general case (depends on arbitrary factors e.g. on who the typist is, the lighting conditions etc.)
However it can be *estimated* if we have a theory of how it was produced.
In the case of Kernighan et al, it is assumed that O arises from a single insertion, deletion, transposition or substitution, for which certain *a priori* probabilities are evident, e.g.

- the identity of the correct letter,
- how the letter was misspelled
- the surrounding context
- e.g. m and n are often confused (because they are pronounced similarly, they often crop up in the same contexts.

Computing $P(O \mid c)$

In fact Kernighan et al. ignored most of these factors and simply counted the occurrences of particular kinds of error occurring in a large corpus of errors.
Using this technique they constructed a series of **confusion matrices** for the different kinds of error.

- sub[x,y] - number of times x is substituted for y.
- ins[x,y] - number of times x was typed as xy
- del[x,y] - number of times xy was typed as x
- tran[x,y] - number of times xy was typed as yx

Using the Confusion Matrices

Using these matrices, they calculated $P(t \mid c)$ as follows, where c_p is the p^{th} character of word c. Recall that t represents the typo (i.e. the observation O)

$$P(t \mid c) = \frac{\text{sub}[t_p, c_p]}{\text{count}[c_p]}$$

$$P(t \mid c) = \frac{\text{tran}[c_p, c_{p+1}]}{\text{count}[c_p c_{p+1}]}$$

$$P(t \mid c) = \frac{\text{del}[c_{p-1}, c_p]}{\text{count}[c_{p-1}, c_p]}$$

$$P(t \mid c) = \frac{\text{ins}[c_{p-1}, t_p]}{\text{count}[c_{p-1}]}$$

Result

c	freq(c)	p(c)	p(O|c)	p(c)p(O|c)	%
actress	1343	.0000315	.000117	3.69×10^{-9}	37
cress	0	.000000014	.00000144	2.02×10^{-14}	0
caress	4	.0000001	.00000164	1.64×10^{-13}	0
access	2280	.000058	.000000209	1.21×10^{-11}	0
across	8436	.00019	.0000093	1.77×10^{-9}	18
acres	2879	.000065	.0000321	2.09×10^{-9}	21
acres	2879	.000065	.0000342	2.22×10^{-9}	23

- What do you conclude from this result?

Remarks and the Role of Context

- The algorithm predicts "acres" as the correct word.
- Yet the surrounding context
  ... was called a "stellar and versatile acress whose combination of sass and glamour has defined her..."
  makes it clear that the correct word is actually "actress".
- Clearly other methods are necessary to take account of context.

Producing Confusion Matrices

The algorithm described requires hand-annotated data to train the confusion matrices. This is expensive and slow to produce. Kernighan et al (1990) suggested the following iterative approach to the problem of constructing confusion matrices:

- Initialise matrices with equal values
- Spelling error correction algorithm is run on a set of spelling errors. This yields the errors paired with their corrections.
- Using this information in these pairs, the confusion matrices can now be recomputed.
- The program performed quite well, agreeing with 87% of the judgements of asking human judges concerning spelling errors that had two possible corrections.

References


Daniel Jurafsky and James H. Martin.
Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition (Prentice Hall Series in Artificial Intelligence).
Prentice Hall, 1st edition, 2000.


Mark D. Kernighan, Kenneth Church, and William A. Gale.
A spelling correction program based on a noisy channel model.
In *Proceedings of the Thirteenth International Conference on Computational Linguistics*, pages 205–210, 1990.


Karen Kukich.
Techniques for automatically correcting words in text.
ACM Comput. Surv., 24:377–439, December 1992.