




# Spellchecker

---

Tahira Naseem



---

*To err is human  
To correct is spellchecker*

# Spelling Errors

---

Spelling errors can generally be divided into two types

- **Typographic** errors occur when writer knows the correct spelling of the word but mistypes the word by mistake, these errors are mostly related to the keyboard.
- **Cognitive** errors (also called orthographic errors) occur when writer does not know or has forgotten the correct spelling of a word



# Why we make spelling mistakes

---

- Keyboard Adjacencies
- Shift-key Characters
- Phonetic Similarity
- Visual Similarity



# How to correct them

---

*Using Spellchecker*



# Components of a Spellchecker

---


- Lexicon
- A bunch of algorithms for
  - Lexicon lookup i.e. error detection
  - Error correction
  - Ranking of corrections.
  - Morphological Preprocessing



# Lexicon Storage Techniques

---

- Code Based Techniques
- Trie Based Techniques
- Minimal Finite Automata
- N-Gram Inverted List
- Cross Indexing
- Bloom's Filter (Hashing)

- 
- 
- For achieving efficient detection and high performance of correction, different data structures of lexicon can be used for error detection and error correction.



# Morphological Preprocessing

---

- In highly inflected languages, rather than saving all forms of words, only root forms are saved and input words are first trimmed to get the root word and then these root words are checked in the dictionary. This is done to save space.



# Categories of Spelling Correction

---

Spelling Correction can be of two types

- Automatic

Error is automatically replaced by correction without user intervention. Automatic error correction is the requirement for those speech processing and NLP (Natural Language Processing) related systems where human intervention is not possible.

Spellchecker should be able to provide one best correction suggestion.

- Interactive

User can interactively select one of the suggested corrections for replacement.

spellchecker can give multiple correction suggestion.



# Types of Correction Techniques

---

- Isolated Word Error Correction
- Context based Error Correction

# Isolated Word Error Correction Techniques

---

Mainly four types of correction techniques

are used

- Soundex and its variants
- Edit Distance techniques
- N-Gram based techniques
- Probabilistic techniques



# SOUNDEX

---

- Soundex algorithm tries to assign common codes to similar sounding words and names.

# The Algorithm

---

1. Retain the first letter of the name, and drop all occurrences of a,e,h,i,o,u,w,y in other positions.
2. Assign the following numbers to the remaining letters after the first:  
b, f, p, v → 1      d, t → 3      m, n → 5  
c, g, j, k, q, s, x, z → 2      l → 4      r → 6
3. If two or more letters with the same code were adjacent in the original name (before step 1), omit all but the first.
4. Convert to the form "`letter, digit, digit, digit" by adding trailing zeros (if there are less than three digits), or by dropping rightmost digits (if there are more than three).



# Problems with Soundex

---

- Low Hit Rate  
one fourth of the relevant words go undetected
- Low Precision  
Large size words sets are detected  
only one third of detected words are actually relevant

# Reasons of Shortcomings

---

- Large size of group 2.  
decreasing its size will reduce the size of detected result set of words.
- Retaining initial letter as it is.
- Ignoring the possibility of sounds produced by more than one letters.
- Ignoring the omission and change in sounds of letters in consonants clusters. dg→j, tch→ch
- Ignoring the ending characters of long words

# Possible enhancements

---

- Use smaller groups of characters
- Assign code to leading letter
- N- Grams Substitution
- Multiple length codes
- Multiple codes for one word
- Overlapping groups of letters
- Position Information

Phonix is a Soundex variant that makes some of these enhancements.

# Phonix

---

- Phonix is a Soundex variant
- Codes assigned to letters are different
  - 1→b, p      2→c, g, j, k, q      3→d,
  - 4→l          5→m, n                      6→r
  - 7→f, v      8→s, x, z
- prior to code generation, string is standardized by applying some letter groups' substitutions also called N-gram substitution. For example the sequence *tch* is mapped to *ch*, *ph* is mapped to *f*.



# Edit Distance Techniques

---

- Single Error Technique
- Levenshtien Edit Distance
- Editex



# Single Error Technique

---

[Damerau 1964] showed that 80% of spelling errors belong to one of the four classes of single errors.

These classes are:

1. Single letter insertion
2. Single letter deletion
3. Single letter substitution
4. Transposition of two adjacent letters



# The Algorithm

---

- Generate all those strings from which the erroneous word can be formed by applying any of the single error operations.
- Check the strings in the dictionary
- Filter out those strings which are not valid words in the language.
- Present remaining words as suggestions

# Performance of Single Error technique

---

- Damerau reports 96.4% accuracy for single errors, including multiple errors it gives overall 84% accuracy.
- Correctly Identified                    812
- Incorrectly Identified                    30
- not identified                            122
- **TOTAL**                                    **964**



# Problems

---

- 20% of the errors (multiple errors) are being completely ignored.
- No weight is given to phonetic similarity. For example according to the algorithm, for the misspelling 'regect', both 'reject' and 'regent' are equally likely corrections.



# Levenshtien Edit Distance

---

- Levenshtein distance between two strings is the number of editing operations (insertion, deletion, substitution and transposition) required to convert one string into the other.
- The Levenshtein distance technique, like Damerau's single spelling error technique, measures the distance between two character strings in terms of insertion, deletion, substitution and transposition, but it caters multiple errors as well.
- Dynamic programming techniques are used to compute distance.



# The Algorithm

---

- Compute edit distance between erroneous word and all dictionary words.
- Select those dictionary words whose edit distance is within a pre specified threshold value.
- Present these words as suggestions



# Problem

---

- Again, phonetic similarities are not being considered.



# Editex

---

- A mixture of Soundex and edit distance
- Editex forms overlapping groups of letters.
- If a letter from a Editex group is substituted by another letter from the same group then the contribution of this substitution operation to the edit distance is half of the any other single error operation.

# Editex Codes

---

1 - b, p

2 - c, k, q

3 - d, t

4 - l, r

5 - m, n

6 - g, j

7 - f, p, v

8 - s, x, z

9 - c, s, z



# Problem

---

- Though phonetic similarity is being taken into consideration but other factor causing spelling mistakes, like keyboard adjacencies, visual similarity etc. are still being ignored.



# N-Gram Based Techniques

---

- An N-gram is a sequence of N adjacent letters in a word where N can be 1, 2, 3....
- They are used for both; error detection and error correction.
- The more N-grams two strings share the more similar they are.

# The Algorithm

---

- In a method given by Pfeifer [1995] a similarity coefficient  $\delta$  can be calculated by dividing the number of common N-grams of the two strings by the total number of N-grams in two strings.  
let A and B be two strings  
$$\delta = A \cap B / A \cup B$$
- Those dictionary words are considered possible correction whose similarity score is higher than a pre specified threshold.

# Problems

---

- N-gram techniques don't show good performance on short words. For example when using trigrams, the words of length 3 will share no trigram with themselves just after introduction of single error.
- N-gram similarity measure works best for insertion and deletion errors, well for substitution errors, but very poor for transposition errors.



# Probabilistic Techniques

---

- Single Edit Distance with Probabilistic Ranking
- Noisy Channel Model



# Probabilistic Ranking

---

- Generate candidate correction using Single Error Technique.
- Rank Corrections using confusion matrices for all types of single errors.



# Noisy Channel Model

---

- Consider the phenomenon of making spelling mistakes as the process of sending text through a noisy communication channel, which introduces errors in the text. Our task is to find the most probable transmitted word (correct dictionary word) for a received erroneous string (misspelling).



# Generic Algorithm

---

- The model assigns a probability to each correct dictionary word for being a possible correction of the misspelling. The word with highest probability is considered the closest match (or the actual intended word).

# Formal Description

---

- Formally the problem can be stated as follows,


Let  $D$  be a dictionary and  $w_i$  be any word in  $D$ , for a misspelled string  $s$  not present in  $D$  our task is to find such  $w \in D$  for which

$P(w|s)$  is maximum

Hence  $w = \underset{w_i}{\operatorname{argmax}} P(w_i|s)$

Applying Bayes' rule we can rewrite this probability expression as

$$P(w|s) = (P(s|w)P(w)) / P(s)$$



---

to maximize  $P(w|s)$  we have to maximize only  $P(s|w)P(w)$ , since  $P(s)$  remains constant.

The first of these terms  $P(s|w)$  is the probability of typing string  $s$  when word  $w$  was intended, it is called the *a posteriori* probability or channel model.

$P(w)$  is the probability that a writer will type  $w$  from among all the dictionary words. It is called source-model or language-model.

Many different techniques have been devised to model the source and channel behavior.

# An implementation by Brill & Moore (2002)

---

To compute  $P(s/w)$ , both  $s$  and  $w$  are divided into partitions  $r_1, r_2, r_3 \dots$

$$P(s/w) = P(r_{1s}/r_{1w}) * P(r_{2s}/r_{2w}) * P(r_{3s}/r_{3w}) \dots$$

To compute this we must first know all  $P(\alpha \rightarrow \beta)$  or  $P(\alpha/\beta)$  where  $\alpha$  and  $\beta$  are letters or groups of letters from alphabets. They are computed through training.

This is letter based model, phone base model can be applied by first making letter to phone conversions using Fisher's N-Grams

# Fisher's N-grams

---

- Fisher's N-grams assign probabilities to letter to phone translation based on context. The form of these substitution rules are as follows:

[Lm.T.Rn→ph1p1.ph2p2]

for example:

[b.b.o]→\_.9.b.1

b in the context of a and o becomes silent  
90% of the times



# Performance

---

- 95.58% for one best
- 99.06% for 4 best

Probabilistic techniques are generic and domain adaptive, but they require training. Training requires hand coded errors or corpus.

# Comparison

---

| <b>TECHNIQUE</b>                     | <b>PERFORMANCE</b>          |
|--------------------------------------|-----------------------------|
| Soundex                              | Precision=33%<br>Recall=75% |
| Phonix                               | 33%                         |
| Single Error                         | 84%                         |
| Levenshtein                          | <b>60%</b>                  |
| Probabilistic Weighted Edit Distance | 87% <b>78%</b>              |
| N-Gram Vectors                       | <b>52%-74%</b>              |
| Probabilistic N-Grams                | 75%-78% <b>75%</b>          |
| Noisy Channel                        | 99%                         |



# Real Word Errors

---

- Real word errors are those typographic errors that result in a valid dictionary word not intended by the typist. For example typing 'form' when 'from' was intended.
- It is difficult to detect real word errors, context sensitive techniques are required to detect and correct such errors.
- Near 40% of the errors are real word errors.



# Context Based Error Correction

---

Context Based Error Correction  
techniques

are required for

- Ranking of correction suggestions
- Detection of real word errors



# How to Use Context

---

- Real-word spelling errors can be viewed as violation of NLP constraints, there are mainly five types of constraints in NLP,
  - Lexical
  - Syntactic
  - Semantic
  - Discourse
  - Pragmatic
- Context based correction techniques generally make use of syntactic and semantic knowledge to identify real word errors.



# Statistical Approach

---

- Statistical context based correction techniques make use of word digram and trigram probabilities in order to capture collocation trends.
- POS N-Gram Probabilities are also used.



## Which Techniques should be used


---

- Combination of different techniques generally give better results compared to the individual techniques.
- Probabilistic techniques are most generic but they require training.

# Suggested Readings

---

- Kukich, K. Techniques for automatically correcting words in text, *ACM Computing Survey*, Vol. 14, No. 4, pp 377-439, December 1992.
- Brill, E. and Moore, R. C. An Improved Error Model for Noisy Channel Spelling Correction. *In proceedings of 38th Annual meeting of Association for Computational Linguistics*, pp. 286-293, 2000.
- Toutanova, K. and Moore, R. C. Pronunciation Modeling for Improved Spelling Correction, *In proceedings of 40th Annual meeting of Association for Computational Linguistics*, July 2002, pp. 144-151.

- 
- 
- Speech and Language Processing.  
By Daniel Jurafsky and James H. Martin
  - Zobel, J. and Dart, P. *Phonetic String Matching: Lessons from Information Retrieval.*