Between Sound and Spelling: Combining Phonetics and Clustering Algorithms to Improve Target Word Recovery

Marcos Zampieri[1] and Renato Cordeiro de Amorim[2]

[1]Saarland University, Germany [2]Glyndŵr University and Birkbeck University of London, UK

Introduction

- Spell checking is an important part of several computer applications.
- Spell checkers need to accomplish two steps:
  - Identifying misspellings.
  - Providing accurate suggestions for the misspelled word.
- In this work we focused on target word recovery.
- Our method works under the assumption that some letters are more often misspelled for others (*d* and *t*; *p* and *b*; *m* and *n*).
  - Examples: *then, them* or *abrupt, aprupt, aprubt*.
- To model this we use:
  - A clustering algorithm applied to spell checking previously used in de Amorim and Zampieri (2013)
  - A simple phonetic algorithm: Soundex.
- Using Soundex, the strings 'Robert' and 'Rupert' are represented as 'R163'.

Related Work

- Verberne (2002)
- Mitton (2009)
- Lin and Wu (2013)
- de Amorim and Zampieri (2013)
- Pirinen (2014)

Clustering

- Our clustering algorithm, iPAM, is based on Partition Around Medoids (Kaufman and Rousseuw, 1990) and iK-Means (Mirkin, 2005).

  The medoid m_k of a given cluster S_k is equivalent to the entity $y_i \in S_k$ with the minimum sum of distances to all other $y_j \in S_k$. PAM minimises the criterion below.

$$W(S, M) = \sum_{k=1}^{K} \sum_{i \in S_k} \sum_{v \in V} (y_{iv} - m_{kv})^2, \quad (1)$$

  where M represents a set of medoids $m_1, m_2, ..., m_K$. The criterion represents the sum of distances between $m_k \in M$ and $y_i \in S_k$, for $k = 1, 2, ..., K$.
- PAM allows us to use a distance measure whose center is unknown. We can minimize the equation with three simple steps.
  - Randomly select K entities from Y as initial medoids.
  - Assign $y_i \in Y$ to the cluster formed by its closest medoid.
  - Update each medoid $m_k \in M$ to be equivalent to $y_i \in S_k$ with the minimum sum of distances to all other $y_j \in S_k$, for $k = 1, 2, ..., K$.
- We adapted iK-Means to work with medoids, rather than centroids, so it can initialize the PAM algorithm.

$$W(S_A, m_t) = \sum_{y_i \in S_A} d(y_i, m_t) + \sum_{y_i \in S_A} d(y_i, m_c), \quad (2)$$

  where S_A is an anomalous cluster in Y, m_c is the entity $y_i \in Y$ with the smallest sum of distances to all other entities $y_j \in Y$, and m_t is the entity $y_i \in S_A$ with the smallest sum of distances to all other entities $y_j \in S_A$, we can see m_t as a tentative medoid based on S_A, and m_c as the medoid of the data set Y.
- In order to complete the equations above we need to define d, which the function returning the distance between two strings.
- Our framework follows the idea that multiple distances can be combined. Distance between two strings x and z is given by:

$$d(x, z) = \sum_{t=1}^{T} w_t d_t(x, z) \quad (3)$$

  where w_t is the weight of distance d_t and T is the total number of distances used.

Soundex

In the experiments for this paper we have decided to use only two distances: Levenshtein and Soundex.

$$d(x, z) = w_1 * Lev(x, z) + w_2 * Lev(Sound(x), Sound(z)) \quad (4)$$

For every string, the Soundex algorithm works using the following 4 steps.

- Keep the first letter of the word and drop all other occurrences of a, e, i, o, u, y, h, w.
- Replace consonants with digits (after the first letter):
  - b, f, p, v → 1
  - c, g, j, k, q, s, x, z → 2
  - d, t → 3
  - l → 4
  - m, n → 5
  - r → 6
- If before step 1, two or more letters with the same number are adjacent in the word, only retain the first letter. Two letters with the same number separated by 'h' or 'w' are coded as a single number, whereas such letters separated by a vowel are coded twice.
- Iterate the previous step until obtaining one letter and three numbers. If word is too short so that it is not possible to assign three numbers, append it with zeros until there are three numbers.

Performance

We tested our method using: in the Birkbeck spelling error corpus containing

- A dictionary containing 57,046 entries.
- The Birkbeck spelling error corpus containing 36,133 misspellings of 6,136 target words.

Method	Cluster.	Cluster. + Phon.
Accuracy (%)	41.71%	47.85%
Accuracy (Nominal)	14,579 words	16,725 words
Total Number of Clusters	1,570 clusters	1,215 clusters
Cluster Size (Mean)	3.78 words	2.17 words
Cluster Size (Median)	2 words	1 word
Average Distance Calculations	3,251.4	3,175.3

- Our method uses a constant c and originally we experimented with $c = 1$.
- By increasing this constant one can expect to increase the accuracy of our method, at the cost of increasing the number of distance calculations as well. We experimented with other c values and here we report results obtained using $c = 2$ and $c = 3$.

Constant c Value	$c = 2$	$c = 3$
Accuracy (%)	50.44%	50.59%
Accuracy (Nominal)	17,631 words	17,270 words
Total Number of Clusters	1,570 clusters	1,215 clusters
Cluster Size (Mean)	2.23 words	2.24 words
Cluster Size (Median)	1 word	1 word
Average Distance Calculations	13,573	35,450

References

de Amorim, R., Zampieri, M.: Effective spell checking methods using clustering algorithms. In: Proceedings of Recent Advances in Natural Language Processing (RANLP2013), Hissar, Bulgaria (2013) 172-178

Kaufman, L., Rousseeuw, P.: Finding groups in data: an introduction to cluster analysis. Volume 39. Wiley Online Library (1990)

Lin, C., Chu, W.: Ntou chinese spelling check system in sighan bake-off 2013. In: Proceedings of the Seventh SIGHAN Workshop on Chinese Language Processing (SIGHAN-7), Nagoya, Japan (2013) 102-107

Mirkin, B.: Clustering for data mining: a data recovery approach. Volume 3. CRC Press (2005)

Mitton, R.: Ordering the suggestions of a spellchecker without using context. Natural Language Engineering 15 (2009) 173-192

Pirinen, T.: Weighted Finite-State Methods for Spell-Checking and Correction. PhD thesis, University of Helsinki (2014)

Verberne, S.: Context-sensitive spell checking based on word trigram probabilities. Master's thesis, University of Nijmegen (2002)

Marcos: marcos.zampieri@uni-saarland.de - Renato: r.amorim@glyndwr.ac.uk