Global Journal of Enterprise Information System

DOI: 10.18311/gjeis/2018/19822

MetaSoundex Phonetic Matching for English and Spanish

K. Koneru* and C. Varol

Department of Computer Science, Sam Houston State University, TX 77341, USA; keerthi.amigos@gmail.com, cvarol@shsu.edu

Abstract

Researchers confront major problems while searching for various kinds of data in large imprecise databases, as they are not spelled correctly or in the way they were expected to be spelled. As a result, they cannot find the word they sought. Over the years of struggle, pronunciation of words was considered as one of the practices to solve the problem effectively. The technique used to acquire words based on sounds is known as "Phonetic Matching". Soundex was the first algorithm developed and other algorithms such as Metaphone, Caverphone, DMetaphone, Phonex etc., are also used for information retrieval in different environments. The main contribution of this paper is to analyze and implement the newly proposed MetaSoundex algorithm for fixing ill-defined data in English and Spanish languages. The newly developed MetaSoundex algorithm addresses the limitations of well-known phonetic matching techniques, Metaphone and Soundex. Specifically, the new algorithm provided results that are more accurate compared to both Soundex and Metaphone algorithms and has higher precision compared to Soundex, thus reducing the noise in the considered arena.

Keywords: Information Retrieval, Metaphone, Metasoundex, Misspelled Words, Phonetic Matching, Soundex

Paper Code (DOI): 19822; **Originality Test Ratio:** 19%; **Submission Online:** 23-Jan-2018; **Manuscript Accepted:** 30-Jan-2018; **Originality Check:** 03-Feb-2018; **Peer Reviewers Comment:** 13-Feb-2018; **Double Blind Reviewers Comment:** 20-Mar-2018; **Author Revert:** 24-Mar-2018; **Camera-Ready-Copy:** 27-Mar-2018; **Editorial Board Excerpt:** 30-Mar-2018.

Editorial Board Excerpt: *Initially at the Time of Submission (ToS) submitted paper had a 19% plagiarism which is an accepted percentage for publication. The editorial board is of an observation that paper had a subsequent surveillance by the blind reviewer's which at a later stages had been rectified and amended by an authors (Koneru and Varol) in various phases as and when required to do so. The reviewer's had in an initial stages comment with minor revision with a following remark which at a short span restructured by an authors. The comments related to this manuscript is extremely noticeable both subject-wise and research wise by the reviewers during evaluation and further at blind review process too. All the comments had been shared at a variety of dates by the authors' in due course of time and same had been integrated by the author in addition. By and large all the editorial and reviewer's comments had been incorporated in this very paper at the end and further the paper had been earmarked and decided under* ***"Empirical Research Paper"*** *category as its highlights and emphasize the pure and first hand information in relation to MetaSoundex Phonetic Matching for English and Spanish.*

1. Introduction

Information deterioration is an intensive problem for organizations in the present era. With the increase in the amount of information saved day by day, there is a desperate need for locating the mistyped data. Organizations are facing great challenge to maintain the quality of data in information systems with various sources of data damage. Whenever the data is assimilated from multiple sources, it is a challenge to recognize the duplicate information due to the existence of misspelled data for the same record. As a result, the information of organization always ends up at risk. To address these challenges, techniques such as string matching, phonetic matching, and data linkage have been used. Apart from other techniques that depend on variations in letters, phonetic matching is mainly contingent on variations in sound to identify the misspelled data. As a result, the misspelled data from multilingual sources can also be identified using phonetic matching.

Soundex was the naive algorithm proposed and other algorithms like Metaphone, Caverphone, DMetaphone, Phonex etc., are also used for retrieving nearest matches for misspelled data. As per the research[15], it was clearly observed that there is no concrete technique for retrieving nearest matches. Soundex has high accuracy than other algorithms but has huge overhead due to its high false positives. Metaphone has high efficiency in spite of its low accuracy, due to its low overhead. Hence, this paper mainly involves the proposal, implementation, and analysis of a hybrid algorithm, *MetaSoundex*. It is observed that MetaSoundex has an accuracy of 84.5% for a real-life dataset, which is improved over Soundex (80%) and Metaphone (58%).

The rest of this paper is structured as follows. Differences between string matching and phonetic matching are discussed in next section. Section three describes background of phonetic matching algorithms. Section four explains in detail about Soundex, Metaphone, and MetaSoundex, which is the initial contribution of this paper. Section five defines the evaluation

*Author for correspondence

Table 1. String matching *vs* phonetic matching[26,29]

	String Matching	Phonetic Matching
Matching	Matches data based on patterns of substrings	Matches data based on the similar pronunciations
Involves	Addition, Deletion or Substitution of Letters	Conversion of data to phonetic patterns
Applications	Applied in Search Engines, Bio-Informatics, spell checkers, digital forensics etc.	Used in name retrieval in enquiry lines, record linkage and fraud detection. Gaining its importance in spell checkers and; search engines.
Prominence	Mainly used for matching names and nouns from English Language	Can be used in multi-lingual environment, where diversities in pronunciation or writing styles may be present.

The study in this paper particularly focuses on phonetic matching because:

a. It is not explored as much as string matching and still relies on old techniques.

b. Of increase in the voice-to-text translation applications, where phonetic matching plays a crucial role.

metrics and describes the experimental setup used in this study. The main contribution of this paper is presented in section six in which the results of newly proposed MetaSoundex are analyzed and compared with the existing algorithms. Finally, this paper is concluded and future work is pointed out in section seven.

2. String Matching *vs* Phonetic Matching

Phonetic matching is one of the important techniques that plays a major role in variety of fields such as digital investigation involving voice memos and voice mails, transcriptions, and voice apps such as 'Siri', 'Google' voice etc.[3] to provide suggestions for misspelled words. Phonetic comparison meticulously identifies the words that are most likely to sound similar. It obtains the quantitative analysis of pronunciations[32] between speech forms and spellings of words, whereas, string matching mainly involves insertion, deletion, and substitution of letters to find the near matches[26,29]. Table 1 describes the most common differences between string matching and phonetic matching.

3. Phonetic Matching Algorithms

Information retrieval is one of the major viewpoints of data mining application areas[28]. However, the information may not be consistent over the considered arena due to various causes. The different sources of variations can be spelling variations (typographical errors, substituted letters or by addition or omission of letters), phonetic variations (discrepancies in phonetic structure of words), double names or double first names (names having more than one word), change of name[27] (individual undergoes change of name).

Of the different criteria mentioned above, the research in phonetic variations led to the development of phonetic matching algorithms, which obtains worthwhile approximate matches to the misspelled words.

3.1 Evolution of Phonetic Matching Algorithms

The evolution of phonetic matching has come into frame when there is a hardship in the retrieval of information[5]. The main goal of phonetic matching algorithms is to encode homophones to the same representation so that they can be matched despite of minor differences in spelling[1,10]. The technique of obtaining words using sounds was used in the US census since the late 1890's, but a concrete solution to this was first proposed and patented by Robert C. Russell in 1912 as Soundex algorithm[27]. The background of various phonetic matching algorithms is discussed.

3.1.1 Soundex

The earliest algorithm in the literature is Soundex developed by Odell and Robert C. Russell in 1912, which generates a four-digit code retaining its first letter. The authors patented the algorithm in 1918[22]. The process mainly encodes consonants while a vowel is not encoded unless it is the first letter. Arguably, Soundex is one of the most widely known of all phonetic algorithms. It is used as a standard feature in applications like mySQL, oracle, etc. Because of the few disadvantages like dependency on the first letter, failure of detection of silent consonants, limit to the four characters of encoding, and high overhead in the retrieved matches, Soundex can only be used in applications where high false positives and false negatives can be tolerated[27].

3.1.2 Beider-Morse Phonetic Matching (BMPM)

Beider and Morse implemented an improvement to Soundex to reduce the number of false positives and false negatives, known as Beider-Morse Phonetic Matching (BMPM). Beider, *et al*, has also mentioned that the algorithm is extended to languages other than English, with the application of some generic rules to obtain the phonetic codes[5]. Varol, *et al*, discussed BMPM as a hybrid technique with a 6-letter encoded code in which the percentage of

irrelevant matches can be abated by 70%[33]. A set of tables representing the pronunciation rules for specific languages are designed for BMPM, where the language of the word can be recognized from its spelling. The design includes nearly 200 rules to specify the language in this technique. If the language cannot be determined, special kind of generic rules are used to encode the word.

3.1.3 NYSIIS

NYSIIS algorithm was developed in 1970 as a part of New York State Identification and Intelligence System project headed by Robert L. Taft, which produces a canonical code similar to Soundex[13]. Unlike Soundex, NYSIIS retains the information regarding position of vowels in the encoded word by transforming them all to 'A'. It generates only alphabetic code and is extensively used in record linkage system[4,12,30].

3.1.4 Daitch Mokotoff Soundex

Daitch Mokotoff Soundex System was developed by Randy Daitch and Gary Mokotoff of the Jewish Genealogical Society (New York) in 1985. The algorithm is mainly used for determining the near matches with Eastern European surnames, which include Russian and Jewish names. Similar to Soundex, the algorithm also encodes into digits by extending it to a complete 6-digit code. The conversion rules of Daitch Mokotoff Soundex are much complicated than Soundex as they involve groups of characters for encoding (2016)[31].

3.1.5 Phonex and Phonix

Phonex is a technique of encoding words after pre-processing. In order to overcome defects of Phonex, Phonix has been introduced with a number of transformations in the beginning, ending, and in the middle of the word[33]. Phonix is considered to be the variant of Soundex, where a prior mapping involves nearly 160 letter-group conversions to normalize the string. For example, X is converted to 'ECS', PSv is converted to Sv (where 'v' is any vowel) if it occurs at the start of string. Phonix also produces a four letter code like Soundex, which is highly useful when an exact index search is required but, due to the truncation of code, it is not beneficial when the complete string matching should be assessed[34].

3.1.6 Metaphone

In 1990, a new technique considering diphthongs (combination of two or more letters) of words was developed by Lawrence Philips, known as Metaphone[18]. It indexes the original word based on the pronunciation rules in English. It retains more information than other variants of Soundex as the letters are not defined into groups[21]. The final code of Metaphone includes 16 consonant letters but retains the vowels, if present at the beginning of the word. Bhattacharjee, *et al* had stated that the technique is mainly used for data cleaning in the text files to remove erroneous data[6]. Pande, *et al* detailed that Metaphone has its extended usage in stemming, which improves performance in Information Retrieval (IR)[23]. David Hood cited that though the algorithm is sensitive to combination of letters like 'TH', it is not subtle enough with the vowels especially at the postvocalic L and R[13].

3.1.7 Double Metaphone

Double Metaphone, popularly known as DMetaphone, is an enhancement to Metaphone algorithm by Lawrence Phillips in 2000. It is distinctive from other algorithms as it generates two code values – one representing the basic version and other representing the alternate version[24]. Unlike Soundex, DMetaphone encodes groups of letters called diphthongs according to a set of rules[33]. The encoding process involves rules, which consider the words from different origins such as Eastern European, Italian, Chinese and other languages.

3.1.8 Caverphone

In pace, the specified algorithms are not suitable for a particular database, named Caversham, which is mainly used for data source linkage. The algorithm, known as Caverphone, which is analogous to Metaphone with some rules subsequently applied, was enforced by David Hood in 2002 to encode the data of Caversham database[23]. The algorithm was later improvised in 2004 to Caverphone 2.0, to increase its accuracy and efficiency by applying more set of rules. David Hood[13] also stated that the algorithm is efficient by giving precise matches when compared to Soundex and Metaphone algorithms for linking data sources[7].

3.1.9 Spanish Soundex

In 2012, Am´on *et al* had proposed an improvement to Soundex algorithm by including Spanish letters making it feasible to obtain phonetic codes for Spanish words[2]. The encoding also removes the dependency on the first letter by converting all the letters into digits. As a result, Spanish Soundex is more accurate than the original Soundex in finding near matches for Spanish words. In 2014, Angeles, *et al* had improvised the algorithm to make the encryption code resizable[10].

3.1.10 Spanish Metaphone

Alejandro Mosquera[19] had developed Metaphone algorithm for Spanish language by adapting the techniques from the algorithm used for English Language[19]. Unlike Spanish Soundex, Spanish Metaphone retains the information related to vowels. The encoded word results in groups of characters.

In spite of many phonetic matching algorithms, there is still a need to develop a proper algorithm to achieve higher data quality as every algorithm has its own disadvantages[27]. Soundex is one of the prominent algorithms having high accuracy but it has very low precision due to the large overhead. Metaphone is a well-known phonetic matching algorithm comprising of rules involving vowels and sounds of diphthongs but has less accuracy. The major contribution of this work is to overcome such shortcomings and propose a new algorithm, MetaSoundex, in Engliash and Spanish, where the encoding process includes both the vowel and diphthong sounds. As these sounds are reflected, the number of false positives is reduced.

4. Methodology

Complication in the recovery of data is the result of type errors, misspelled words, inconsistent expression habit, and different formats. With typographical errors, often there would be interchanging of letters or misspelling of words. Such problems can be addressed by phonetic matching algorithms such as Soundex, Metaphone, and Caverphone, etc. In this section, we are going to discuss in detail about the existing Soundex and Metaphone algorithms of English and Spanish languages and describe the functionality of newly proposed MetaSoundex algorithm.

Soundex

To obtain Soundex code following steps should be followed. Let the input word be **w**. Convert all letters into upper case. Retain the first letter in the word **w.**

Algorithm: Soundex
set A = {*A, E, H, I, O, U, W, Y* }, **B** = { *B, F, P, V* }, **C** = { *C, G, J, K, Q, S, X, Z* }, **D** = { *D, T* }, **E** = { *L* }, **F** = { *M, N* }, **G** = {*R*}
set i = 0
while i >= 1 ***and*** i < **w**.*length*
  if w.*charAt*(i) ∈ **A**
    w.*charAt*(i) = **0**
  end
  else if w.*charAt*(i) ∈ **B**
    w.*charAt*(i) = **1**
  end
  else if w.*charAt*(**i**) ∈ **C**
    w.*charAt*(**i**) = **2**
  end
  else if w.*charAt*(**i**) ∈ **D**
    w.*charAt*(**i**) = **3**
  end
  else if w.*charAt*(**i**) ∈ **E**
    w.*charAt*(**i**) = **4**
  end
  else if w.*charAt*(**i**) ∈ **F**
    w.*charAt*(**i**) = **5**
  end
  else if w.*charAt*(**i**) ∈ **G**
    w.*charAt*(**i**) = **6**
  end
end

From word **w**, all pairs of same digits and zeroes are removed. The first four characters of word **w** are considered to be Soundex code[7,22].

Spanish Soundex

The Spanish Soundex algorithm is similar to Soundex in generating the encoded word. The following steps should be followed for obtaining the code. Let the input word be **w**. Convert all letters into upper case.

Algorithm: Spanish Soundex
set A = {*A, E, H, I, O, U, W* }, **B** = { *B, V* }, **C** = { *F, H* }, **D** = { *D, T* }, **E** = { *S, G, Z, X* }, **F** = { *Y, LL, L* }, **G** = { *N, Ñ, M* }, **H** = { *Q, K* }, **I** = { *G, J* }, **J** = { *R, RR* }
set i = 0
while i >= 0 ***and*** i < **w**.*length*
  if w.*charAt*(i) ∈ **A**
    remove w.*charAt*(i)
  end
  else if w.*charAt*(i) = ***P***
    w.*charAt*(i) = **0**
  end
  else if w.*charAt*(i) ∈ **B**
    w.*charAt*(i) = **1**
  end
  else if w.*charAt*(i) ∈ **C**
    w.*charAt*(i) = **2**
  end
  else if w.*charAt*(i) ∈ **D**
    w.*charAt*(i) = **3**
  end
  else if w.*charAt*(i) ∈ **E**
    w.*charAt*(i) = **4**
  end
  else if w.*charAt*(i) ∈ **F**
    w.*charAt*(i) = **5**
  end
  else if w.*charAt*(i) ∈ **G**
    w.*charAt*(i) = **6**
  end

```
        else if w.charAt(i) ∈ H
                w.charAt(i) = 7
        end
        else if w.charAt(i) ∈ I
                w.charAt(i) = 8
        end
        else if w.charAt(i) ∈ J
                w.charAt(i) = 9
        end
    end
```

From word **w**, all pairs of same digits are removed. Unlike Soundex, the resultant code is independent of first letter of the word[2].

Metaphone

To obtain Metaphone code following steps should be followed. Let the input word be **w**. Convert all letters into upper case. Drop all the duplicate letters from **w** except *C*.

Algorithm: Metaphone

set A = {*K, G, P*}, **B** = {*CIA, CH* }, **C** = {*SCH, C* }, **D** = {*CI, CE, CY* }, **E** = { *DGE, DGI, DGY* }, **F** = { *GH, GN, GNED*}, **G** = {*GI, GE, GY, ^GG* }, **H** = {*A, E, I, O, U* }, **I** = { *CK, Q* }, **J** = { *PH, V*}, **K** = { *SH, SIO, SIA* }, **L** = {*TIAO, TH, TCH*}

```
set i = 0
while i >= 0 and i < w.length
    if w.charAt(i) ∈ A and w.charAt(i + 1) == N
      w.charAt(i) = N
    end
    if w.charAt(i) == A and w.charAt(i + 1) == E
      w.charAt(i) = E
    end
    if w.charAt(i) == W and w.charAt(i + 1) == R
      w.charAt(i) = R
    end
    if w.charAt(w.length-2) == M and w.charAt
    (w.length-1) = B
      w.charAt(i) = B
    end
end
    if w contains B
      replace with X
    end
    if w contains C
      replace with K
    end
    if w contains D
      replace with S
    end
    if w contains E
      replace with J
    end
    set i = 0
    while i >= 0 and i < w.length
    if w.charAt(i) == G
    w.charAt(i) = K
    end
    if w.charAt(i) == D
      w.charAt(i) = T
    end
end
    if w contains G
      replace with J
    end
while i >= 0 and i < w.length
    if w.charAt(i) == H and (w.charAt(i-1) ∈ H or
    w.charAt(i+1) ∈ H)
      remove H
    end
    end
    if w contains I
      replace with K
    end
    if w contains J
      replace with F
    end
    if w contains K
      replace with X
    end
    if w contains L
      remove T
    end
    if w.charAt(0) == W and w.charAt(1) == H
      remove H
    end
set i = 0
while i >= 0 and i < w.length
    if (w.charAt(i) == W or w.charAt(i) == Y) and
    w.charAt(i+1) ∉ H
      remove w.charAt(i)
    end
    if w.charAt(i) == Z
      w.charAt(i) = S
    end
end
```

From word **w**, all vowels are removed and the obtained output is considered as Metaphone code[18].

Spanish Metaphone

Let the input word be **w**. Convert all letters into lower case.

Algorithm: Spanish Metaphone

set A = {*A, E, I, O, U* }, i = 0

```
while i >= 0 and i < w.length
    if w.charAt(i) == á
      w.charAt(i) = A
    end
    if w.charAt(i) == c and w.charAt(i+1) == h
      w.charAt(i) = X
       remove w.charAt(i+1)
    end
    if w.charAt(i) == Ç
      w.charAt(i) = S
    end
    if w.charAt(i) == é
      w.charAt(i) = E
    end
    if w.charAt(i) == í
      w.charAt(i) = I
    end
    if w.charAt(i) == ó
      w.charAt(i) = O
    end
    if w.charAt(i) == ú or w.charAt(i) == ü
      w.charAt(i) = U
    end
    if w.charAt(i) == ñ
      w.charAt(i) = N
      w = w.substring(0,i) +"Y"+w.substring(i+1, w.length)
    end
    if w.charAt(i) == g and w.charAt(i+1) == ü
      w.charAt(i) = W
      remove w.charAt(i+1)
    end
    if w.charAt(i) == b
      w.charAt(i) = V
    end
    if w.charAt(i) == l and w.charAt(i+1) == l
      w.charAt(i) = Y
      remove w.charAt(i+1)
    end
end
w.toUpperCase() //convert all letters to uppercase and remove duplicate letters except C
set i = 0
while i >= 0 and i < w.length
    if w.charAt(i) == C and w.charAt(i+1) == C
      w.charAt(i) = X
    remove C
    end
    if w.charAt(i) == C and (w.charAt(i+1) == E or w.charAt(i+1) == I)
      w.charAt(i) = Z
      remove w.charAt(i+1)
    end
    if w.charAt(i) == G and (w.charAt(i+1) == E or w.charAt(i+1) == I)
      w.charAt(i) = J
    remove w.charAt(i+1)
    end
    if w.charAt(i) == H and w.charAt(i+1) □ A
      remove H
    end
    if w.charAt(i) == Q and w.charAt(i+1) ≠ U
      remove K
    else
      remove w.charAt(i) and w.charAt(i+1)
    end
    if w.charAt(i) == W
      w.charAt(i) = U
    end
    if (w.charAt(i) == S or w.charAt(i) == X) and w.charAt(i+1) ∈ A
      w = "E" + w;
    end
    end
```

The obtained **w** is the encoded Spanish Metaphone code[19].

4.1 MetaSoundex Algorithm

Though Soundex and Metaphone are naïve algorithms being used in different applications as embedded tools, each of them have their own disadvantages. Soundex mainly depends on the first letter of the word. It has a high overhead in retrieving the near matches and it does not consider the phonetic sounds of vowels. In spite of the fact of addressing the above problems in Metaphone algorithm, it only has less accuracy in obtaining the proper matches to the misspelled word. To overcome the limitations in both algorithms, a new algorithm is developed, namely, MetaSoundex. The schematic design of MetaSoundex algorithm is shown in Figure 1.

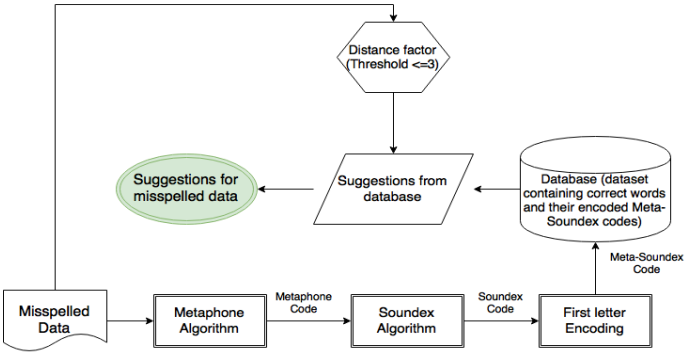

Figure 1. Schematic design of suggestions retrieval for MetaSoundex algorithm.

Table 2. Phonetic codes – English and Spanish.

Language	Word	Soundex Code	Metaphone Code	MetaSoundex Code
English	CAPABLE	C114	KPBL	5140
Spanish	CINEMÁTICA	46634	KNMTK	76637

The schematic design shows the retrieval of suggestions for the misspelled data using MetaSoundex algorithm. As shown in the Figure 1, the misspelled data is given as input to the Metaphone algorithm to obtain the Metaphone code. As a result, the phonetic sounds of vowels and diphthongs are retained. This Metaphone code is given as input to Soundex algorithm, which converts the existing groups of characters to numbers. The generalization of characters to numbers improves the accuracy for retrieving suggestions. But, the obtained code retains the dependency on first letter due to Soundex encoding. To remove this dependency, the first letter is encoded using transformations in Daitch-Mokotoff Soundex algorithm. The obtained MetaSoundex code is sent to database to retrieve the suggestions of the misspelled data. To further reduce the unnecessary overhead, a distance factor using Levenshtein Edit Distance (LED) is applied on retrieved suggestions, which is further detailed in section 4.2. The database shown in the Figure 1 comprises of correct words and their corresponding MetaSoundex codes for both English and Spanish languages.

MetaSoundex

MetaSoundex algorithm is a hybrid algorithm of Soundex and Metaphone as discussed earlier. The step by step encoding of MetaSoundex algorithm is detailed below. Let the input word be **w.**

1. Convert all the letters of the word **w** to upper case.
2. Encode the word using Metaphone to retain the vowel sounds and diphthong combinations.
3. Encode the obtained string using Soundex algorithm.
4. If the language is English, then the first letter is encoded using transformations in Daitch Mokotoff Soundex algorithm to remove the dependency on first letter.

The pseudo code of the MetaSoundex algorithm is discussed below:

```
Algorithm: MetaSoundex
set A = {A, E, I, O, U}, B = { J, Y }, C = { D, T }, D = { S, Z, C },
        E = { X, G, H, K, Q }, F = { N, M }, G = { B, F,V, P, W },
        H = { L }, I = { R }
set i = 0, language        //either English or Spanish
w = Metaphone(w)
w = Soundex(w)
if language is English
    while i >= 0 and i < w.length
    if w.charAt(i) ∈ A
      w.charAt(i) = 0
    end
    else if w.charAt(i) ∈ B
        w.charAt(i) = 1
    end
    else if w.charAt(i) ∈ C
        w.charAt(i) = 3
    end
    else if w.charAt(i) ∈ D
        w.charAt(i) = 4
    end
    else if w.charAt(i) ∈ E
        w.charAt(i) = 5
    end
    else if w.charAt(i) ∈ F
        w.charAt(i) = 6
    end
    else if w.charAt(i) ∈ G
        w.charAt(i) = 7
    end
    else if w.charAt(i) ∈ H
        w.charAt(i) = 8
    end
    else if w.charAt(i) ∈ I
        w.charAt(i) = 9
        end
    end
end
```

Table 2 shows the exemplary phonetic codes generated using Soundex, Metaphone, and MetaSoundex for both English and Spanish words.

4.2 Distance Factor for Filtering Retrieved Approximate Matches - MetaSoundex

The generated MetaSoundex code can be used to obtain the approximate matches for the given misspelled data. After the approximate matches are retrieved, the distance factor between the misspelled word and the retrieved matches is calculated using LED method[9] to reduce the unnecessary overhead. The threshold of the distance is set to 3, as the maximum number of errors in the synthetic data is less than 3, whereas for real-world data the distance factor is mostly observed to be 3. If LED is less than or equal to 3, then the word is filtered to be nearest match for the misspelled word. For example, the MetaSoundex code of the misspelled word "PROBLMS" is 7614. The retrieved suggestions from the database for the given MetaSoundex code

are "PROBLEMS", "PROVOLONES", and "PROPYLS" each having the distance factor of 1,5,3 respectively, from the given misspelled word. After applying the distance factor, the final suggestions are "PROBLEMS", "PROPYLS", which reduced the unnecessary overhead.

5. Testing

In this work, the accuracy of the MetaSoundex is compared with the existing algorithms in both English and Spanish languages. In English, the proposed algorithm is compared with five algorithms – Soundex, Metaphone, Caverphone, DMetaphone, NYSIIS, whereas in Spanish, the proposed algorithm is compared with Spanish Soundex and Spanish Metaphone. The evaluation metrics, experimental setup, and preparation of pre-processed datasets used to compare the accuracy and efficiency of the algorithms are discussed below[14].

5.1 Evaluation Metrics

The performance of phonetic matching algorithms used for information retrieval is evaluated by calculating precision, recall and F – measure.

Precision gives the total number of true positives obtained over the total number of suggestions for the obtained true positives.

$$p = \frac{\sum p}{\sum Number\ of\ suggested\ words\ for\ each\ corrected\ word} \quad (1)$$

where, $P = \begin{cases} 1, \textit{ if the word is corrected} \\ 0, \textit{ if the word is not corrected} \end{cases}$

$p = cumulative\ precision\ of\ an\ algorithm$

Recall provides the total number of relevant words over the total number of suggestions. It can also be referred as accuracy.

$$R = \frac{Number\ of\ corrected\ words}{Total\ number\ of\ misspelled\ words} \quad (2)$$

where, R = *recall or accuracy of an algorithm.*

The F – measure is calculated based on precision and recall and is defined as the harmonic mean of precision and recall. It is given by,

$$F = \frac{2 \times P \times R}{P + R} \quad (3)$$

Where, $F = F - measure\ of\ the\ algorithm$.

5.2 Experimental Setup

The design of the experimental analysis supports two languages, English and Spanish. Two input files - one with correct data indicated as "reference data file" and other with ill-defined data represented as "incorrect data file" are used to retrieve approximate matches from Spanish dictionary and English dictionary for both Spanish and English languages, respectively.

The simulator generates phonetic codes by executing respective phonetic matching algorithms of the corresponding language, for the errant data. These codes are compared to the phonetic codes present in database and the matched word lists are retrieved as the approximate suggestions. These matched words are evaluated by comparing with the reference file to calculate precision and recall, which would symbolize the better algorithm.

5.3 Dataset Preparation

Data pre-processing is considered to be an important phase in data mining because the data that is collected from various sources lacks consistency, which makes it unsuitable to directly apply data processing algorithms[35]. The raw data can also be incomplete with missing values of some attributes. In some cases, we can encounter noisy data with some unwanted values to a given attribute. As a result, we preprocess the data into a suitable format to apply different algorithms.

5.3.1 Reference Dataset Preparation

Until now, various experiments were conducted on finding phonetic matches for misspelled words of personal names[27]. But there is only little exploration in finding the phonetic matches for dictionary words using these algorithms. Hence, in this project we mainly concentrated on obtaining the phonetic matches for misspelled words of English and Spanish diction, which are considered as reference datasets.

The reference datasets for the experiment are prepared as follows. For the English dictionary dataset, all the words are extracted from the reference[17] and a list is formed. This list comprises of 267,750 correct, non-duplicate words. Phonetic codes are generated for each of these words, by applying the algorithms discussed in section 3. A dataset is created with these English words and their corresponding phonetic codes. This dataset is used as a reference dataset for obtaining the suggestions for misspelled English words.

Similarly, Spanish wordlist is extracted from the reference[8]. The list consists of 95,487 correct words. Phonetic codes are generated using Spanish phonetic matching algorithms. Another dataset, having these Spanish words and their corresponding phonetic codes are created to use as reference for retrieving suggestions to misspelled words.

5.3.2 Synthetic Dataset Preparation

According to Kukich[16], nearly 80% of problems of misspelled words can be addressed either by addition of a single letter, or replacement of single letter or swapping of letters. Therefore, synthetic datasets are generated by executing addition, deletion, swapping, and replacement of letters.

From the above mentioned correct word list of English language, different pairs of synthetic ill-defined datasets are generated by randomly selecting the words. Each pair consists of correct words as reference data and their corresponding manipulated words as misspelled data. The generation of synthetic datasets is shown in Figure 2.

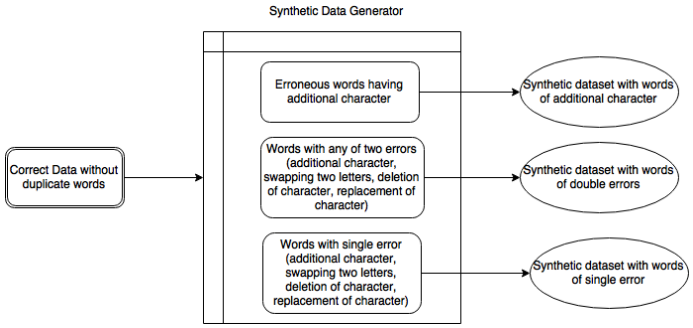

Figure 2. Synthetic datasets generation for analysis of various algorithms.

While creating the manipulated data, words with three types of errors are generated - words with additional character, words having single error (replacement or substitution of character or swapping of two characters), and words having double errors (two single errors). The generated words are accumulated into datasets of size 800. Four datasets are generated for each type of error. Hence, a total of twelve pairs of correct and manipulated datasets are generated. By the same token, twelve pairs of correct and manipulated datasets are generated with data size of 800 for the Spanish language.

5.3.3 Real-World Misspelled Data

Apart from the synthetic data, the performance of the algorithms is also analyzed on real-world data. For English, the misspelled data is referred from[11] having nearly 4,200 misspelled words along with their corresponding correct words. In the same way, the Spanish data is retrieved from[25]. As there is only little research in the field of misspelled words in Spanish language, the data size of misspelled words is only about 100.

6. Results and Discussion

The work in this paper illustrates the performance of different algorithms on datasets of particular size having various types of errors - single error, double error, and additional character.

6.1 Analysis on Synthetic Data – English and Spanish

The values of recall and precision for different algorithms tested on synthetic data are shown in Table 3 and 4 for Spanish and English languages, respectively.

From the above experimental data, it can be clearly observed that the state-of-the-art MetaSoundex algorithm has highest accuracy, whereas, Metaphone has the lowest accuracy of all the algorithms. It can also be observed that the value of recall is highly dependent on the type of error. The recall value is low for the wordlist having two errors in each word irrespective of language. From the results, it can be stated that the precision

Table 3. Precision and recall values of different algorithms for synthetic data - Spanish

Algorithm	Additional Character		Double Error		Single Error	
	Precision	Recall	Precision	Recall	Precision	Recall
Soundex	0.029	0.1	0.019	0.02	0.026	0.127
Metaphone	0.18	0.05	0.3	0.01	0.17	0.07
NYSIIS	0.055	0.15	0.0305	0.025	0.013	0.187

Table 4. Precision and recall values of different algorithms for synthetic data - English.

Algorithm	Additional Character		Double Error		Single Error	
	Precision	Recall	Precision	Recall	Precision	Recall
Soundex	0.003	0.45	0.0034	0.27	0.0038	0.37
Metaphone	0.05	0.21	0.095	0.09	0.17	0.162
MetaSoundex	0.016	0.53	0.021	0.328	0.023	0.408
DMetaphone	0.0033	0.40	0.004	0.223	0.005	0.331
Caverphone	0.02	0.33	0.04	0.128	0.057	0.204
NYSIIS	0.01	0.337	0.0157	0.05	0.013	0.23

of Soundex is least for any type of error in both the languages due to the retrieval of high false positives while the precision of Metaphone is high in all the cases.

For English, Soundex shows its high recall value in the second place, followed by DMetaphone, NYSIIS and Caverphone in succession. Analogous to English, in Spanish Soundex shows its high recall value in the second place, followed by Metaphone.

The performance evaluation for different algorithms on the synthetic dataset of English and Spanish words is shown in Figure 3.

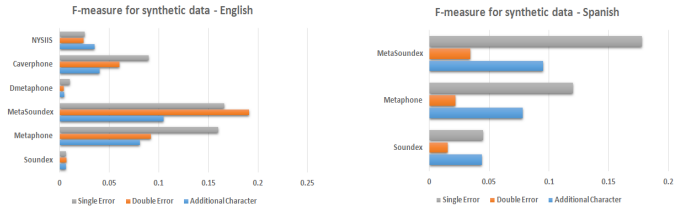

Figure 3. F-measure of different algorithms on synthetic data for English and Spanish languages.

Figure 3 indicate a substantial increase in the efficiency of MetaSoundex algorithm over Soundex and Metaphone. Though Metaphone has high precision, it is less efficient than MetaSoundex due its low accuracy. From the experimental analysis, it can be clearly stated that MetaSoundex has better accuracy than all other algorithms for any data size and type of error, reducing the number of false positives and noise in the retrieved suggestions.

In English language, the highest value of F-measure of MetaSoundex is followed by Metaphone and Caverphone. Soundex and DMetaphone show the highest unnecessary overhead in all the considered arenas. Though DMetaphone has noticeable recall values, it has low precision similar to Soundex due to retrieval of suggestions for both the primary and secondary codes.

MetaSoundex has reduced unnecessary overhead along with the high recall value ensuring that the algorithm reduces noise and can be used in various applications where count of false positives plays a major role. For the synthetic data, based on the type of error, MetaSoundex shows high efficiency for the erroneous list having two errors, while it reflects low value for the words having additional character. Figure 3, it can also be inferred that all other algorithms show average F-measure for double errors for English words.

In the same way, in Spanish language, it can be observed that the results are dependent on type of errors. All the three algorithms show least performance for the words with double errors while highest performance for the words with single error.

6.2 Analysis on Real-World Data – English and Spanish

In addition to the analysis on synthetic dataset, the experimental analysis is also conducted on the real-world ill-defined data to check the efficiency of the algorithms. The data size of the real-world English dataset is 4200 but for the Spanish language the size is nearly 100. The recall and precision values of different algorithms for English and Spanish languages are shown in Table 5.

From the above, it can be stated that the MetaSoundex has the exceptional recall value showing its high accuracy on the real-world data followed by Soundex while Metaphone has the lowest accuracy rate in correcting the misspelled words. The F-measure for different algorithms on the real-world dataset of English and Spanish words is shown in Figure 4.

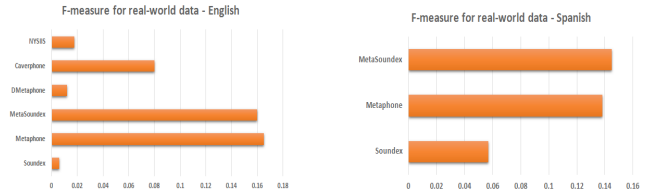

Figure 4. F-measure of different algorithms on real-world misspelled data for English and Spanish languages.

As shown above, for English language, Metaphone shows highest efficiency with a miniature difference to the MetaSoundex algorithm. In spite of low recall Metaphone shows better efficiency

Table 5. Precision and recall values of different algorithms for real-world data – English and Spanish.

Algorithm	English		Spanish	
	Precision	Recall	Precision	Recall
Soundex	0.003	0.8	0.033	0.2
Metaphone	0.096	0.575	0.136	0.14
MetaSoundex	0.012	0.845	0.103	0.24
DMetaphone	0.006	0.75	–	–
Caverphone	0.046	0.62	–	–
NYSIIS	0.009	0.69	–	–

due its high precision, reducing unnecessary overhead. The efficiency of MetaSoundex has an exceptional increase over Soundex, showing that the state-of-the-art MetaSoundex has achieved high precision over Soundex. Though Caverphone has low recall, it shows a better F-measure due to its decent value of precision, which is followed by NYSIIS, DMetaphone, and Soundex.

By the same token, for the real-world data of Spanish language, Meta-Soundex has the highest F-measure compared to other algorithms reducing the unnecessary suggestions. In spite of its high precision, Metaphone has the lowest accuracy of all the three compared algorithms. Soundex has the least efficiency as the precision is very less compared to other algorithms.

From the above analysis on synthetic data and real-world data, it can be clearly stated that MetaSoundex has better values of recall and precision. The accuracy of MetaSoundex is observed to be improved over Soundex as the dependency on the first letter is removed in the MetaSoundex algorithm. Also, the high precision of MetaSoundex is due to the reduced false positives as the algorithm retains the sounds of vowels and diphthongs by the application of rules in Metaphone. As a result, the improved accuracy over existing algorithms and the improved precision over Soundex (which is considered as one of the more accurate algorithms) made MetaSoundex more balanced and efficient than other algorithms.

7. Conclusions

In this paper, we presented an overview of various phonetic matching algorithms in English and Spanish languages. We explained how newly developed MetaSoundex algorithm is different from the existing phonetic matching algorithms. The functionality of different phonetic matching algorithms for both English and Spanish language is illustrated. Then, we justified the need to implement the state-of-the-art MetaSoundex algorithm. The main purpose of the proposed approach is to improve the recall and precision over the existing algorithms, thus increasing accuracy and reduce the noise in retrieved suggestions for misspelled words from various sources.

To improve the recall and precision, a new hybrid algorithm, MetaSoundex, is proposed, whose implementation is mentioned in detail. The efficiency of this algorithm is evaluated and compared with the existing algorithms such as Soundex, Metaphone, DMetaphone, Caverphone, and NYSIIS. The analysis is performed on different datasets having three types of errors, namely, additional character, single error (substituted letter, missing of a letter), and words with double errors (more than one single error) along with the real-world misspelled data. From the experiments, it can be clearly affirmed that MetaSoundex has improved recall and precision over existing algorithms. Also, the implementation of distance factor in MetaSoundex algorithm facilitates to improve the precision over other phonetic matching algorithms.

In this paper, the analysis is performed on English and Spanish languages as both of them are most widely spoken languages across the globe (35)[20]. The development of phonetic matching algorithms and the application of MetaSoundex can also be extended to other languages based on the requirement, which can be considered as future work as it would require more observance and experimental analysis.

8. Acknowledgments

This work was supported by Sam Houston State University. The authors would like to express their gratitude to the anonymous reviewers, whose helpful comments and suggestions were advantageous to improve the quality of this paper.

9. References

1. Angeles PM, Gamez AE, Moncada GJ. Comparison of a Modified Spanish Phonetic, Soundex, and Phonex Coding Functions During Data Matching Process. International Conference on Informatics, Electronics and Vision (ICIEV). 2015. https://doi.org/10.1109/ICIEV.2015.7334028.
2. Amón I, Moreno F, Echeverri J. Algoritmo Fonético Para Detección De Cadenas De Texto Duplicadas En El Idioma Espa-ol, Revista Ingenierías Universidad de Medellín. 2012; 11(20):127–38.
3. Arkfeld MR. Audio: Solving the Riddle and Avoiding Sanctions for the Forgotten "Electronically Stored Information" (ESI), Law Technology News. 2013. http://www.nexidia.com/media/1768/white-paper-audio-the-forgotten-esi-arkfeld.pdf.
4. Balabantaray RC, Sahoo B, Lenka SK, Sahoo DK, Swain M. An Automatic Approximate Matching Technique Based on Phonetic Encoding for Odia Query, IJCSI International Journal of Computer Science Issues. 2012; 9(3).
5. Beider, Morse SP. Phonetic Matching: A Better Soundex. 2010. http://stevemorse.org/phonetics/bmpm2.htm.
6. Bhattacharjee AK, Mallick A, Dey A, Bandypoadhay S. Enhanced Technique for Data cleaning in text files, International Journal of Computer Science Issues. 2013; 10(5).
7. Carstensen A. An Introduction to Double Metaphone and the Principles behind Soundex. 2005. http://www.b-eye-network.com/view/1596.
8. Diccionario. http://www.deperu.com/diccionario/.
9. Hassan D, Aickelin U, Wagner C. Comparison of Distance metrics for hierarchical data in medical databases, International Joint Conference on Neural Networks (IJCNN). 2014. https://doi.org/10.1109/IJCNN.2014.6889554, https://doi.org/10.2139/ssrn.2828084.
10. Haunts S. Phonetic String Matching: Soundex. 2014. https://stephenhaunts.com/2014/01/17/phonetic-string-matching-soundex/.

11. Hempel B. Fuzzy tools. 2014. https://github.com/brianhempel/fuzzy_tools/blob/master/accuracy/test_data/sources/misspellings/misspellings.txt.
12. Hobbs S. New York State Identification and Intelligence System (NYSIIS) Phonetic Encoder. 1990. http://www.dropby.com/NYSIIS.html.
13. Hood D. Caversham Project Occasional Technical Paper. 2004. http://caversham.otago.ac.nz/files/working/ctp060902.pdf.
14. Kelkar BA, Manwade KB. Identifying Nearly Duplicate Records in Relational Database, IRACST - International Journal of Computer Science and Information Technology and Security (IJCSITS). 2012; 2(3).
15. Koneru K, Pulla VSV, Varol C. Performance Evaluation of Phonetic Matching Algorithms on English Words and Street Names: Comparison and Correlation. 5th International Conference on Data Management Technologies and Applications, 2016. p. 57-64. https://doi.org/10.5220/0005926300570064.
16. Kukich K. Techniques for automatically correcting words in text, ACM Computing Surveys. 1992; 24(4). https://doi.org/10.1145/146370.146380.
17. Lawler J. An English Words List. 1999. http://www-personal.umich.edu/.
18. Lawrence P. Hanging on the Metaphone, Computer Language. 1990; 7(12).
19. Mosquera A. Phonetic Indexing with the Spanish Metaphone Algorithm. 2012. http://www.amsqr.com/2012/02/phonetic-indexing-with-spanish.html.
20. Most Widely Spoken Languages in the World. 2014. http://www.infoplease.com/ipa/A0775272.html.
21. Nikita. Phonetic Algorithms. 2011. http:// ntz-develop.blogspot.com/2011/03/phonetic-algorithms.html.
22. Odell MK, Russell RC. Patent nos. 1,261,167 and 1,435,683. 1918 and 1922.
23. Pande BP, Dhami HS. Application of Natural Language Processing Tools in Stemming, International Journal of Computer Applications (0975 – 8887). 2011; 27(6).
24. Philips L. The Double Metaphone Search Algorithm. 2000. http://www.drdobbs.com/the-double-metaphone-search-algorithm.
25. Planeta C. Las 20 palabras peor pronunciadas en espa-ol. 2008. http://www.planetacurioso.com/2008/10/30/las-20-palabras-pero-pronunciadas-en-espanol/.
26. SaiKrishna V, Rasool A, Khare N. String Matching and its Applications in Diversified Fields, International Journal of Computer Science Issues. 2012; 9(1).
27. Shah R, Singh DK. Analysis and Comparative Study on Phonetic Matching Techniques, International Journal of Computer Applications. 2014; 87(9). https://doi.org/10.5120/15236-3771.
28. Singh V, Saini B. An Effective Pre-Processing Algorithm for Information Retrieval Systems, International Journal of Database Management Systems (IJDMS). 2014; 6(6). https://doi.org/10.5121/ijdms.2014.6602.
29. Singla N, Garg D. String Matching Algorithms and their Applicability in various Applications, International Journal of Soft Computing and Engineering (IJSCE). 2012; 1(6).
30. Snae C. A Comparison and Analysis of Name Matching Algorithms, International Journal of Computer, Electrical, Automation, Control and Information Engineering. 2007; 1(1).
31. Soundex Coding. 2016. http://www.jewishgen.org/InfoFiles/soundex.html.
32. Sundeep C, Srikantha R. Analysis of Phonetic Matching Approaches for Indic languages, In International Journal of Advanced Research in Computer and Communication Engineering. 2012; 1(2).
33. Varol C, Talburt JR. Pattern and Phonetic Based Street Name Misspelling Correction. Eighth International Conference on Information Technology: New Generations; 2011. https://doi.org/10.1109/ITNG.2011.101.
34. Zobel J, Dart P. Phonetic String Matching: Lessons from Information Retrieval. Nineteenth Annual International ACM SIGIR conference on Research and development in Information Retrieval; 1996. https://doi.org/10.1145/243199.243258.
35. Zhang S, Zhang C, Yang Q. Towards databases mining: Pre-processing collected data, Applied Artificial Intelligence. 2003; 17(5–6):545–61. DOI: 10.1080/713827180. https://doi.org/10.1080/713827180.

Annexure-I

MetaSoundex Phonetic Matching for English and Spanish

ORIGINALITY REPORT

19%

SIMILARITY INDEX

PRIMARY SOURCES

1	www.scitepress.org Internet	900 words — 13%
2	shsu-ir.tdl.org Internet	129 words — 2%
3	Edy Portmann, Tam Nguyen, Jose Sepulveda, Adrian David Cheok. "chapter 7 Fuzzy Online Reputation Analysis Framework", IGI Global, 2012 Crossref	38 words — 1%
4	downloads.openqm.com Internet	35 words — 1%
5	web.stanford.edu Internet	28 words — < 1%
6	www.jewishgen.org Internet	21 words — < 1%
7	Lei Chen, Jiahuang Ji, Zihong Zhang. "Wireless Network Security", Springer Nature, 2013 Crossref	20 words — < 1%
8	"Speech and Computer", Springer Nature, 2017 Crossref	15 words — < 1%
9	@?@?@?@?@?, @?.. "@?@?@?@?@?@?@? @?@? @?@?@?@?@?@?@?@?@?@? @?@? @?@?@?@?@?@?@? @?@? @?@?@?@? @? @? @?@?@?@?@?@?@?@? @? @?@?@?@?@?@?@?@?@? @?@?@?@?: @?@?@?@?@?@?@?@? @? @?@?@?	12 words — < 1%

@?@?@?@?@?@?@?@? @?@?@?@?@?@?@?@?@?@?@? @?@?@?@?@?@?@?@?@?@?@? (@?@?@?@?@?@?@?@? @? @?@?@?@?@?@?@?@?@? @?@?@?@?@?@?@?@? @? @?@?@?@?@? @?@?@?@?)", Russian Literature, 20080215
Crossref

10 Manoj Kumar Sharma, Debasis Samanta. "Word Prediction System for Text Entry in Hindi", ACM Transactions on Asian Language Information Processing, 2014 — 12 words — < 1%
Crossref

11 thinkmind.org — 12 words — < 1%
Internet

12 iisit.org — 11 words — < 1%
Internet

13 Zahra Pezeshki, Syed Ali Soleimani Ivari. "Applications of BIM: A Brief Review and Future Outline", Archives of Computational Methods in Engineering, 2016 — 11 words — < 1%
Crossref

14 ntz-develop.blogspot.com — 9 words — < 1%
Internet

15 "Pattern Recognition and Image Analysis", Springer Nature, 2013 — 8 words — < 1%
Crossref

16 "Smart Computing and Informatics", Springer Nature, 2018 — 8 words — < 1%
Crossref

EXCLUDE QUOTES ON
EXCLUDE BIBLIOGRAPHY ON
EXCLUDE MATCHES OFF

Source: http://www.ithenticate.com/

Prevent Plagiarism in Publication

The Editorial Board had used the ithenticate plagiarism [http://www.ithenticate.com] tool to check the originality and further affixed the similarity index which is 19% in this case *(See Annexure-I)*. Thus the reviewers and editors are of view to find it suitable to publish in this *Volume-10, Issue-1, January-March, 2018*.

Citation:
K. Koneru and C. Varol
"MetaSoundex Phonetic Matching for English and Spanish",
Global Journal of Enterprise Information System. Volume-10, Issue-1, January-March, 2018. (http://informaticsjournals.com/index.php/gjeis)
DOI: 10.18311/gjeis/2018/19822
Conflict of Interest:
Author of a Paper had no conflict neither financially nor academically.