Received October 18, 2021, accepted December 17, 2021, date of publication December 23, 2021, date of current version January 6, 2022.

Digital Object Identifier 10.1109/ACCESS.2021.3137905

Discovering Lexical Similarity Using Articulatory Feature-Based Phonetic Edit Distance

TAFSEER AHMED[1], MUHAMMAD SUFFIAN[2], MUHAMMAD YASEEN KHAN[1], AND ALESSANDRO BOGLIOLO[2]

[1]Center for Language Computing, Department of Computer Science, Mohammad Ali Jinnah University, Karachi 75400, Pakistan
[2]Department of Pure and Applied Sciences, Università degli Studi di Urbino Carlo Bo, 61029 Urbino, Italy

Corresponding author: Muhammad Yaseen Khan (yaseen.khan@jinnah.edu)

This work was supported by the Department of Pure and Applied Sciences, Università degli Studi di Urbino Carlo Bo, Italy.

ABSTRACT Lexical Similarity (LS) between two languages uncovers many interesting linguistic insights such as phylogenetic relationship, mutual intelligibility, common etymology, and loan words. There are various methods through which LS is evaluated. This paper presents a method of Phonetic Edit Distance (PED) that uses a soft comparison of letters using the articulatory features associated with their International Phonetic Alphabet (IPA) transcription. In particular, the comparison between the articulatory features of two letters taken from words belonging to different languages is used to compute the cost of replacement in the inner loop of edit distance computation. As an example, PED gives edit distance of 0.82 between German word 'vater' ([faːtər]) and Persian word 'پدر' ([pedær'), meaning 'father,' and, similarly, PED of 0.93 between Hebrew word 'שלום' ([ʃəɭɑm]) and Arabic word 'سلام' ([səɭaːm], meaning 'peace,' whereas classical edit distances would be 4 and 2, respectively. We report the results of systematic experiments conducted on six languages: Arabic, Hindi, Marathi, Persian, Sanskrit, and Urdu. Universal Dependencies (UD) corpora were used to restrict the comparison to lists of words belonging to the same part of speech. The LS based on the average PED between pair of words was then computed for each pair of languages, unveiling similarities otherwise masked by the adoption of different alphabets, grammars, and pronunciations rules.

INDEX TERMS Articulatory features, cognates, computational linguistics, edit distance, lexical similarity, natural language processing, phonetic matching.

I. INTRODUCTION

"Language is the road map of a culture. It tells you where its people come from and where they are going"—Rita Mae Brown.

Computational linguistics is a broad field of research that encompasses Natural Language Processing (NLP) and Natural Language Understanding (NLU). Applications include semantic and syntactic analyses, part of speech tagging, dependency parsing, text classification, keyword extraction, text summarization, text generation, automatic translation. Alongside the computational analysis of language for linguistic-specific purposes, NLP and NLU approaches are extensively applied to address problems in several application domains, including healthcare, finance, social sciences, and human-computer interaction.

Globalization, supported by unprecedented communication technologies, has created the needs and opportunities for cross-culture contamination, leading to multilingualism (i.e., people speaking more than one language) and to code-mixing/switching (i.e., transliteration) [1]–[3]. It is very difficult for a language to retain its vocabulary under the overwhelming influence of other languages. Thus, speakers have a chance of borrowing words—called *loanwords*—from other (donor) languages, using them directly, without translation. On the other hand, languages may have very similar words—called *cognates*—which come from the same etymological origin. While cognates are inherited from a common ancestor language, loanwords are due to a persistent contact between groups of people speaking different languages.

Cognates are not identical to each other. The form of the words changes over time due to the phenomenon of *sound change*. For example, consider the word for *father* in different Indo-Aryan languages, English: 'father' [faːðəɹ],

The associate editor coordinating the review of this manuscript and approving it for publication was Francesco Piccialli.

German: 'vater' [faːtər], Persian: 'پدر' [pedær], and Hindi: 'पिता' [pɪ.tɑː]. These words originated from the Proto-Indo-European (PIE) word 'peter,' are not identical, as their form has undergone sound changes that occurred in hundreds of years. Hence, traditional string matching would not work properly to find cognates and LS, which are of great interest not only to study phylogenetic relationships, linguistic distance, and cross-contamination among languages (and populations), but also to evaluate their mutual intelligibility.

In this paper, we present a modified string matching algorithm that is based on the articulatory features of the words under comparison. We introduce Phonetic Edit Distance (PED) and we use it to evaluate the pairwise lexical similarity of six different languages: Arabic, Hindi, Marathi, Persian, Sanskrit, and Urdu. These six languages represent a significant benchmark for the proposed approach because they have different scripts, namely, Devanagari for Sanskrit, Hindi, and Marathi and Perso-Arabic for Arabic, Persian, and Urdu, and two of them, namely, Urdu and Hindi, are known to be mutually intelligible languages [4]. The main contributions of this paper are:

1) The definition of PED is based on articulatory features.
2) The development of a script-independent algorithm to evaluate the lexical similarity between pairs of languages.

A scoring system for pronunciation of words using the weighted phonetic edit distance to measure the error rate [5] could be considered similar to the first contribution of the proposed work. Another similar work was done related to our second contribution, barely computing normal edit distance on the orthographic transcription of words for three languages of Perso-Arabic script [6]. In comparison to the proposed contributions, the first work [5] lacks the notion of articulatory features for the edit distance and the latter work [6] lacks the notion of phonetic edit distance.

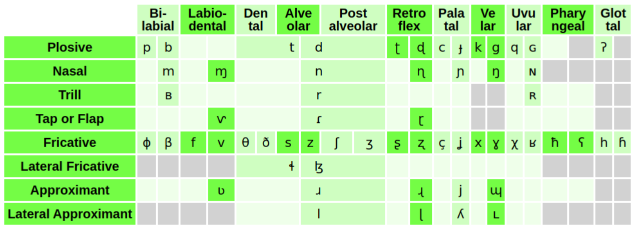
The organization of the paper is as follows: literature review for string matching, phonetic (string) matching, and LS of the languages is presented in section II; detailed discussion on the proposed PED is provided in section III; methodology of the paper is described in section IV; the evaluation and explanation of results are provided in section V; conclusion and future works of proposed work are highlighted in section VI, and in the last bibliographical references are provided.

II. LITERATURE REVIEW

This section covers the literature review of the string and phonetic matching algorithms, as well as approaches for LS calculation.

A. STRING AND PHONETIC MATCHING

Research work contributed by Ukkonen [7] stands as the premier and the most famous amongst all edit-distance (ED) based string matching algorithms. It gives the number of operations (insertion, deletion, or replacement) required to transform a string into another string. For example, the strings *fax* and *axe* have an edit distance of 2 as we perform two operations, i.e., deleting `f` of the first string and then inserting `e` at its end for transforming it into the second string. There are variations in the ED algorithm to make it adapt for the different tasks in Natural Language Processing (NLP) and Computational Biology (CB) e.g., spell correction [8] and string alignment [9]. However, its main shortcoming is that it considers the letters used in the string as discrete-distinct units. In the comparison, either two letters are similar (in this case no/zero operation is needed to transform `p` into another `p`), or they are different (one operation is required to transform `p` into `b`).



	Bi-labial		Labio-dental		Dental		Alveolar		Post alveolar		Retroflex		Palatal		Velar		Uvular		Pharyngeal		Glottal	
Plosive	p	b					t	d			ʈ	ɖ	c	ɟ	k	g	q	ɢ			ʔ	
Nasal		m		ɱ				n				ɳ		ɲ		ŋ		ɴ				
Trill		ʙ						r										ʀ				
Tap or Flap				ⱱ				ɾ				ɽ										
Fricative	ɸ	β	f	v	θ	ð	s	z	ʃ	ʒ	ʂ	ʐ	ç	ʝ	x	ɣ	χ	ʁ	ħ	ʕ	h	ɦ
Lateral Fricative							ɬ	ɮ														
Approximant				ʋ				ɹ				ɻ		j		ɰ						
Lateral Approximant								l				ɭ		ʎ		ʟ						

FIGURE 1. IPA chart for pulmonic consonants. (Courtesy International Phonetic Association).

In contrast to the letter matching algorithms, Zobel and Dart [10] presented a phonetic matching algorithm that gives the similarity of strings based on sounds of corresponding letters. Similarly, the Soundex algorithm has provided six equivalence classes of letters. The algorithm discards the vowels and transforms the consonants in the string into their mapped phonetic class (except the first letter), respectively [11]. Following this scheme, both 'robert' and 'rupert' are transformed to the same string `r163` as both `b` and `p` belong to the same sound class `b`, `f`, `p`, and `v` having encoding value 1. Philips [12] introduces a double Metaphone, which is widely employed for spell-checking applications.

Daitch–Mokotoff (DM Soundex) [13] and Beider-Morse Phonetic Matching (BMPM) [14] propose rule-based algorithms that also utilize the encoding scheme for matching the similar-sounding/homophonic names written in different languages. For example, the same strings spelled differently in different languages; *Schwarz* in German, *Szwarc* in Polish, *Şvarţ* in Romanian, *Chvarts* in French, and Шварц in Russian. Hence, these algorithms, through the ED, can be used to find the cognates according to the phonetic similarity. However, extending these algorithms to other languages is difficult as there are many languages and scripts. Thus, we specifically need an algorithm that applies phonetic matching without providing the equivalence classes of letters of similar sound toward defining acoustic-phonetic equivalence for vowels, as suggested in work by Broad [15], of homophonic letters and complicated rules of spelling.

The chart related to sound articulations is presented in figure 1, where it could be seen that IPA's are arranged according to the place and manner of articulation. The sounds

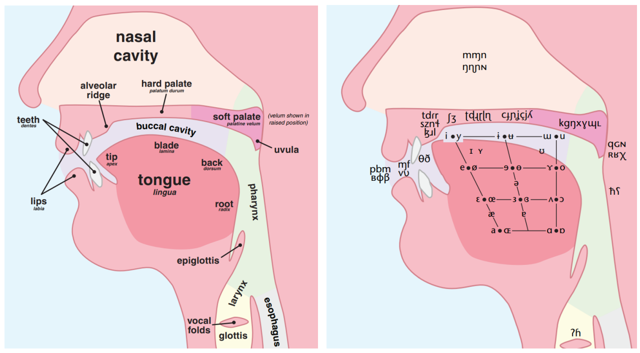


FIGURE 2. IPA articulation points (left) human vocal tract (right) IPA (vowels, consonants) articulation points, figure courtesy [16].

p and b are similar because they have the same place of articulation i.e. Bilabial [17], and similarity of articulation, i.e. Plosive [18]. Further, the difference between these sounds is another articulatory feature, i.e. voicing [19], for example, b is a voiced consonant and p is an unvoiced consonant. There are other features, e.g. aspirated and Pharyngeal etc. This could be seen more clearly in figure 2, where all the IPA-based vowels and consonants are shown with their articulation points in the human vocal tract. For which the IPA is represented by diacritical marks. Hence, we cannot consider IPA symbols as atomic. An IPA symbol (or a sound) can be represented by a set of features.

B. LEXICAL SIMILARITIES

Since the languages inherit words from a common ancestor language, it is quite evident that languages of the same family have many homophonic words for the same concept. The difference in the sound of these words is widely studied and it is commonly recognized that the change of the sound is systematic.

Grimm's Law dealt with sound change in Germanic languages (for example English, German, Dutch, and Swedish, etc.) [20]. It presented the observation that some of the voiced stops of the older language changed to voiceless stops, and similarly, voiceless stops changed into voiceless fricatives. There are other similar studies, e.g. Dahl's Law [21] and Verner's Law [22] confirming the results.

Consider the example of the word *father* originating from [pəter] in Proto–Indo–European (PIE). It is different in different Indo-European languages due to the change in sounds. It became [fader] in Proto-Germanic, and [faðer] in English.

Coleman [23] showed the dialect distances between two texts by using the Levenshtein distance, Cosine similarity, Hamming distance, and the ASCII-based hashing methods. Kondrak [24] presented a method to identify the cognates in the languages with inter-related vocabulary sets. His study endorsed the notion of language similarity that it should be measured with phonetic multi-valued features, instead of orthographic measures like the longest common subsequence ratio.

Do *et al.* [25] presented an approach, namely, WNSim, for the similarity analysis of words with a concern of their synsets in WordNet for a specific part of speech (PoS). They also presented the lexical level matching (LLM) technique by combining word-level similarity to compute phrase level and sentence level similarity. Similarly, a string level similarity computation for identification of source code was reported in [26], using the Rabin-Karp rolling hashing algorithm that outperforms various other algorithms. Gomaa *et al.* [27] published a survey on three types of text similarity, i.e., string-based, corpus-based, and knowledge-based. In string-based text similarity, they stressed the need for a technique that should use phonetic features for the same task.

Dijkstra *et al.* [28] conducted an experiment about the cross-language similarity based on English and Dutch cognate sets, for example, English: 'lamp' [læmp], Dutch: 'lamp' [lɑmp] and English: 'flood' [flʌd], Dutch: 'vloed' [vlut]. These cognate sets are useful for bilingual translations. Similarly, Schepens [29] showed the cross-language distribution of cognates based on phonetics and high frequencies. Their results show that cognate frequency was reduced in less-closely related language pairs as compared to more closely related language pairs. Garcia and Souza [30] compute the LS for the English and Portuguese languages on 500 high-frequency words. The genetic difference (the genetic difference is small when one language is descended from the other or if both languages are descended from a common ancestor language) between both languages is reported as less than 30%. But, the calculations of orthographic and phonetic similarity by using the Levenshtein distance for English and Portuguese shows that the readers of both languages got benefit from the orthographic similarity in reading Portuguese and English words.

Some researcher [28], [29], [31] reveal that the phonological and orthographic similarity matters in finding the cross-lingual cognates. Similarly, a different study used six datasets for linking records across the languages like English and French, it evaluated their record linkages from DBpedia knowledge-base [32], [33]. Khan [34] analyzed the historical background related to the origin of French and Urdu words and concluded the existence of genetic affinity between Urdu and French language, Although it showed the similarity of words based on semantic, phonetic, and etymology, it lacks the computational model. Siew and Vitevitch [35] did similar work to the proposed work in which the phonological and orthographic similarity structures of English words are characterized in a network of language, also the links are made between the words orthographically and phonologically to check their similarity. Another study [6] showed some work on the lexical similarity of three languages written in Perso-Arabic script. It employed the orthographic transcription technique for the word lists (set of lemmas) that share the same part of speech. The word lists mapped into IPA for the language, followed by computing their edit distance through Levenshtein distance.

TABLE 1. IPA-based vowel weights per articulation positions; the articulation positions are shown in italics and the weights are enclosed in parentheses.

Weights		*front* (0)	*near-front* (0.25)	*central* (0.5)	*near-back* (0.75)	*back* (1)
close	(0)	/i/,/y/		/ɨ/,/ ʉ/		/ɯ/,/u/
near-close	(0.17)		/ɪ/,/ʏ/		/ʊ/	
close-mid	(0.33)	/e/,/ø/		/ɘ/,/ɵ/		/ɤ/,/o/
mid	(0.5)			/ə/		
open-mid	(0.67)		/ɛ/,/œ/		/ɜ/,/ɞ/	/ʌ/,/ɔ/
near-open	(0.83)		/æ/		/ɐ/	
open	(1)			/a/,/ɶ/		/ɑ/,/ɒ/

III. PHONETIC EDIT DISTANCE

The Phonetic Edit Distance (PED) method takes two IPA strings (words) as input and returns the phonetic distance between them. Likewise, in the standard ED, if the strings are the same then the distance is zero, thus, in our case if the sounds are the same then the phonetic distance between them is zero. Otherwise, if there is a mismatch, then the resulted distance depends not only on the operations of insertion and deletion but also depends on the phonetic similarity of the sounds that are replaced.

In the standard ED, the distance of IPA strings [pɛn] and [bɛnd] is $\Phi\left([\text{pɛn}], [\text{bɛnd}]\right) = 2$, where Φ denotes function for standard ED, as the second string is formed by replacing /p/ with /b/ (bearing the operational cost 1) and inserting /d/ in the end (again, bearing the operational cost 1), hence the sum of all operational costs is 2. However, with the proposed method, the PED for the same pair of IPA strings is $1 < \Psi < 2$, where Ψ denotes the PED, as the operational cost of replacement is not fixed (as 1), and/since it varies between [0, 1] depending on the phonetic similarity of the replaced sounds. Hence, as /p/ and /b/ in the given IPA strings are more similar in articulation (as discussed in section II-A), their replacement cost would be less (for example, 0.2 w.r.t the proposed system) than the cost of replacing /p/ by /b/ (i.e., 1 w.r.t the result of the standard edit distance). The major building blocks of this method are further discussed in section III-A–III-C.

A. ARTICULATORY FEATURES

The proposed system (framework) will work on the IPA encoded strings (words), the words of the language will first be converted into IPA. Since we need to find the distance of sounds of the IPA string, so, we have created the features with continuous values. For vowels, we used two continuous features, back and open, (see the labels at the top, and left in the chart presented in figure 3, and table 1) and one binary feature rounded.

The rounded vowels are placed at right in the pairs of vowels given in figure 3. Further, the feature back represents all labels (i.e., *front*, *near-front*, *central*, *near-back*, and *back*); and the values corresponding to these labels are set in a range [0, 1]; the label *front* gets value = 0 while the farthest label *back* gets value = 1—and the rest of labels that

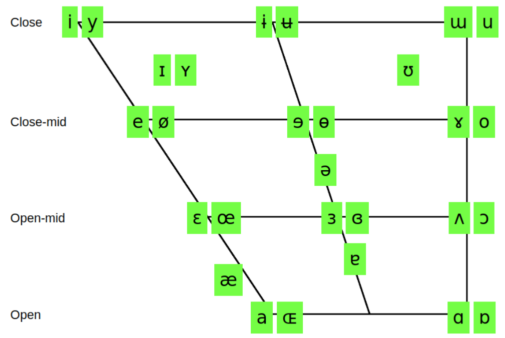


FIGURE 3. IPA chart with the articulation of vowels. (Courtesy: International Phonetic Association).

come between these two, get the value at equal intervals, i.e., *near-front* = 0.25, *central* = 0.50, and *near-back* = 0.75. Similarly, the feature open represents all labels (i.e., *close*, *near-close*, *close-mid*, *mid*, *open-mid*, *near-open*, and *open*); and, likewise the feature back, the values corresponding to open are set in a range [0, 1]; *viz.* label *close* gets value = 0 and the farthest label *open* gets value = 1; and the values for the rest of labels are set at the equal intervals, i.e., *near-close* = 0.17, *close-mid* = 0.33, *mid* = 0.50, *open-mid* = 0.67, and *near-open* = 0.83.

Lastly, the value for the feature *rounded*, which shows the binary characteristic, is either 0 or 1 corresponding to the roundness or non-roundness of a vowel. For clarifying the assignment of weights and their distance, table 1 is provided, in which the vowels are placed on a similar order of features as presented in the standard chart in figure 3. For example, consider the vowel /i/, looking its position in vowel chart presented in figure 3 and taking the corresponding value from table 1, we can set *open* = 0, *back* = 0 and *rounded* = 0. Similarly, the vowel /u/ has *open* = 0, *back* = 1 and *rounded* = 1. The vowel /a/ has *back* = 0.25, *open* = 1 (as it is a *near-front* and *open* vowel) and *rounded* = 1.

For consonants, we proposed the following features and data-type of their respective values: *place* (continuous), *manner* (discrete), *voiced* (binary), *airflow* (discrete), *aspirated* (binary), and *pharyngeal* (binary) shown in table 2.

The feature *place* represents the place of articulation. As these places are present on the continuum inside the human mouth (see figure 2), therefore, we assign relative positions as the value of these features. Hence, the bilabial

TABLE 2. Articulatory features and the weight assignments for consonants.

№	Articulatory Features	Range of values for articulatory feature
1	Label	IPA symbol from the list of vowels and consonants
2	Type	*vowel* or *consonant*
3	Method	*NA* or 0
4	Place	*NA* or in $[0, 0.05, 0.10, \cdots, 0.95]$
5	Manner	*NA* or in {st, fr, ns, af, fl, ap}
6	Voiced	*NA* or 1, 0, -1
7	Aspirated	*NA* or 0, 1
8	Open	*NA* or in $\{0, 0.17, 0.33, \cdots, 0.83, 1\}$
9	Front	*NA* or in $\{0, 0.25, 0.50, 0.75, 1\}$
10	Rounded	*NA* or in 0, 1

position (lips) has the value of 0.05, the dental position (teeth) has the value of 0.15, and the glottal position (throat) has the value of 0.95. The other features have discrete or binary values, this could be seen in figure 1, in the first column. As the bilabial position is at the start it has a value of 0.05 and the glottal position is in the last and has a value of 0.95. There are two meta-features, *label* and *type*, along with these articulatory features. The feature label has the IPA of the sound and the feature type encodes whether the sound is a consonant or a vowel.

B. FINDING THE DISTANCE OF SOUND

The articulatory features presented in section III-A are crafted in such a way that similar sounds have similar feature values. Also, we have different approaches for vowel–vowel and consonant–consonant comparison. Consonants are not compared with vowels and vice versa.

Algorithm 1 Phonetic Difference for Vowel (PDV)

```
1: procedure PDV($w, x$)    ▷ $w$ and $x$ be the IPA character for vowels.
2:     $\delta_{ob} \leftarrow \Theta\left(\langle w_{open}, w_{back}\rangle, \langle x_{open}, x_{back}\rangle\right)$    ▷ Calculating Manhattan distance $\Theta(\cdot)$ between $w$ and $x$ using open and back features.
3:     if $\delta_{ob} > 0.5$ then
4:         $\delta_r \leftarrow \Theta(\langle w_{rounded}, x_{rounded}\rangle)$
5:         return $\frac{1}{3} \cdot (\delta_{ob} + \delta_r)$
6:     else
7:         return $\frac{1}{3} \cdot (\delta_{ob} + 1)$
8:     end if
9: end procedure
```

To compare two vowels, we employed Manhattan Distance [36] and assigned equal weight to all of the features under the same type, such as all non-binary features are weighted $\frac{2}{3}$ of the distance, and binary features are weighted $\frac{1}{3}$ of the distance. Consider algorithm 1 for the vowel comparison, where w and x are the features for the vowels, provided $w \neq x$, Manhattan Distance is denoted by $\Theta(\cdot)$; δ_{ob} and δ_r are the variables representing the distances of continuous and binary features.

Algorithm 2 Phonetic Difference for Consonant (PDC)

```
1: procedure PDC($w, x$)    ▷ $w$ and $x$ be the IPA character for consonants.
2:     $\delta_m \leftarrow \xi\left[\langle w_{manner}, x_{manner}\rangle\right]$
3:     $\delta_p \leftarrow \Theta\left(\langle w_{place}, x_{place}\rangle\right)$
4:     $\delta_{m+p} \leftarrow \delta_m + \delta_p$
5:     if $\delta_{m+p} > \alpha$ then
6:         return $\delta_{m+p}$
7:     else
8:         $\delta_{m+p} \leftarrow \delta_{m+p} \cdot \frac{2}{3}$
9:         $\delta_y \leftarrow \Theta(\langle w_{voiced}, x_{voiced}\rangle) \cdot \frac{1}{5}$
10:        $\delta_{rf} \leftarrow \Theta(\langle w_{\tilde{*}}, x_{\tilde{*}}\rangle) \cdot \beta$
11:        return $\delta_{m+p} + \delta_y + \delta_{rf}$
12:    end if
13: end procedure
```

Similarly, for consonants comparison, as shown in algorithm 2, we gave weight $\frac{2}{3}$ to the place and manner features. The remaining weight $\frac{1}{3}$ is allocated to all other features. The feature voiced has weight $\frac{1}{5}$, the remaining features (i.e., airflow and aspirated) have weight, $\beta = \left(1 - \frac{2}{3} - \frac{1}{5}\right)$. The list of other features could be extended without reducing the weight of three main features (*place*, *manner*, *voiced*). Moreover, place and manner features are more significant than any other feature having binary (or tertiary) values. Hence, we include the distance of other features only when the distance added by place and manner (δ_{m+p}) is less than or equal to the threshold $\alpha = \frac{1}{2} \cdot \left(\delta_{m+p}\right)$. If the combined distance is larger than α, then we return that distance (scaling it as out of 1) without adding the distance of other features. We have a rule-based distance for the feature manner, hence, the lookup in the dictionary ξ is made, which results in the distance when the key $\langle w_{manner}, x_{manner}\rangle$ is given. Lastly, $\tilde{*}$, as mentioned in line 10, shows the Manhattan Distance for all of the remaining features.

C. MODIFICATION IN EDIT DISTANCE

We modified Edit Distance to account for the phonetic similarity of sounds. The cost of insertion and deletion remains 1 (as in the standard ED). However, the replacement cost is calculated by calling the function PDV or PDC, as described in Algorithms 1 and 2 respectively. These functions return a real number between [0, 1] i.e., 0 for the same sound and 1 for an entirely different sound. The pseudo-code for the modified form of Phonetic Edit Distance (PED) is given in algorithm 3. Let us take two examples of calculating the PED for the words in different languages written in different scripts, Hebrew and Arabic are both Semitic languages, and hence many cognates. One of the cognate pairs is the greeting words 'שלומ' ([ʃəɭɑm]; meaning 'peace') and 'سلام' ([səɭaːm]; meaning 'peace'). The list of articulatory features corresponding to these words which are used to measure the distance is presented in tables 3 and 4, and to see the weights of each letter like vowels could be seen in table 1 and consonants in the website,[1] respectively. In the IPA-based strings, there are two mismatches, i.e., the consonant /ʃ/ is to be replaced by the consonant /s/, and the vowel /ɒ/ is to be replaced by /aː/. In standard ED (Φ; the notation for standard Edit Distance function), this would cost one operation, but with the application of PED on the feature lists of these consonants it will cost 0.267 and 0.667. As in the case of letters /ʃ/ and /s/, both are consonants, so the look-up will be made from the chart of consonants (placed in online repository[2]) and the letters /ɐ/ and /aː/ are vowels so the look-up will be made from the table 1, but, for this example corresponding values of these letters are reported in table 3 and 4.

Algorithm 3 Phonetic Edit Distance (PED)

```
procedure PED($S, T$)    ▷ $S$ and $T$ be the string of word; provided $S \neq T$.
    $s \leftarrow$ be the length of $S$ IPA string.
    $t \leftarrow$ be the length of $T$ IPA string.
    if $\min(s, t) = 0$ then
        return $\max(s, t)$
    else if $S[s-1] = T[t-1]$ then
        pass
    else
        ins_cost $\leftarrow$ PED($S[0 : s-1], T$) + 1
        del_cost $\leftarrow$ PED($S, T[0 : t-1]$) + 1
        $\varphi \leftarrow 0$    ▷ Initial value for phonetic-difference.
        if $\varrho(S[s-1]) = \varrho(T[t-1]) =$ consonant then    ▷ $\varrho(\cdot)$ be the function that inspects the type of IPA character.
            $\varphi \leftarrow$ PDC($S[s-1], T[t-1]$)
        else if $\varrho(S[s-1]) = \varrho(T[t-1]) =$ vowel then
            $\varphi \leftarrow$ PDV($S[s-1], T[t-1]$)
        end if
        rep_cost $\leftarrow$ PED($S[0:s-1], T[0:t-1]$) + $\varphi$
    end if
    return min(ins_cost, del_cost, rep_cost)
end procedure
```

The following steps are involved in the computation of PED based on articulatory features.

- Consonant letter distance:
    1) The *method* is the same for both consonants making a distance of 0.

[1]https://github.com/msnizami/LexSim/ (A repository holding reference material regarding the article such as chart for consonants and chart for vowels, owned by the author).

[2]Ibid; see footnote 1.

TABLE 3. Articulatory features corresponding to the Hebrew word 'שלומ' ([ʃəɭɑm]; meaning 'peace').

	Meta Features	IPA Letters				
	Label →	/ʃ/	/ə/	/ɭ/	/ɒ/	/m/
	Type →	Cons	Vow	Cons	Vow	Cons
Phonetic Articulatory Features	Method	0	–	0	–	0
	Place	0.45	–	0.25	–	0.05
	Manner	fr	–	ap	–	ns
	Voice	0	–	-1	–	-1
	Aspirated	0	–	0	–	0
	Open	–	0.5	–	1	–
	Back	–	0.5	–	1	–
	Rounded	–	0	–	1	–

TABLE 4. Articulatory features corresponding to the Arabic word 'سلام' ([səɭaːm]; meaning 'peace').

	Meta Features	IPA Letters				
	Label →	/s/	/ə/	/ɭ/	/aː/	/m/
	Type →	Cons	Vow	Cons	Vow	Cons
Phonetic Articulatory Features	Method	0	–	0	–	0
	Place	0.45	–	0.25	–	0.[illegible]
	Manner	fr	–	ap	–	ns
	Voice	0	–	-1	–	-1
	Aspirated	0	–	0	–	0
	Open	–	0.5	–	1	–
	Back	–	0.5	–	0.33	–
	Rounded	–	0	–	1	–

    2) The weights of *place* are different making a distance of 0.20.
    3) The *manner* of both is fricative making a distance of 0.
    4) The letter /ʃ/ is *un-voiced* and /s/ is *voiced*, but the feature *voiced* is less important as compared to *place* and *manner* so it is assigned a minute weight of 0.067, making a distance of 0.067 ($\frac{1}{5}$ of the less important features).
    5) For other features as *aspirated*, *open*, *back*, and *rounded* both consonants are the same making a distance of 0. Hence, the difference of *place* and *voice* resulting in its PED (Ψ; be the notation for Phonetic Edit Distance function) to 0.267.
- Vowel letter distance:
    1) For *open*, the letter /aː/ and /ɒ/, both are 1, making a distance of 0.
    2) For *back*, /aː/ is 0.33 (although placed in *near-front* but it is *near-front* and near to *central*) and /ɒ/ is 1, making distance of 0.667.
    3) For *rounded*, both /aː/ and /ɒ/ are rounded, making a distance of 0. Hence, making the overall difference of both of these vowels to 0.667.

Hence, the Ψ (of articulatory features) of these two strings, that are Arabic and Hebrew, is 0.267 + 0.667 = 0.934 (despite having two replacements). Similarly, the PED of

German word 'vater' ([faːtər]) and Persian word 'پدر' (pidar, [pedær]) having four replacements, i.e., /f/ → /p/ = 0.1, /aː/ → /e/ = 0.223, /t/ → /d/ = 0.217, and /ə/ → /æ/ = 0.277, is eventually calculated as 0.817.

IV. METHODOLOGY

In this section, we discuss and motivate the requirements deriving the design of the experiment. We show the systematic approach of calculating PoS-wise (Part of Speech) lexical similarity between the two languages. Lexical similarity (LS) between two languages is calculated based on the count/ratio of similar words present in two languages.

There are three major issues/queries in the calculation of lexical similarity.

1) How do we decide whether the two compared words are similar or not?
2) How many words will be compared?
3) Do we compare any of the words or do we choose the root of the word on some criterion?

The proposed PED method gives a reply to the first question. The method does not require to decide manually whether two words are similar or not. The PED gives edit distance as a measure of similarity, and word lists can be compared using this measure. A method of alignment and lexical similarity calculation is presented in section IV-C.

The second and third questions are about choosing the appropriate word lists. Nizami *et al.*, [6] proposed the calculation of lexical similarity based on different PoS. As annotated, UD corpora are available for more than 60 languages, and all of those corpora use the same PoS tag set. Thus, it is a good choice to run an experiment with UD corpora. As it has been mentioned earlier that the languages have similar words due to various reasons, some word classes e.g. pronouns and numbers are similar due to genetic affinity, so that, we can say that languages belonging to the same family or sub-family have cognates due to genetic affinity (i.e., inheritance from the parent or ancestor language). The other word classes, e.g. nouns and proper nouns, may have a significant influence due to borrowing from a genetically unrelated or distant language. Hence, PoS-wise similarity will portray different aspects of the LS and this gives the reply to the second question. As we choose to extract PoS-wise word lists from the UD corpora, the count of words to be compared, the count depends on the words present in the corpora. As manually selecting the words from long lists (e.g. of nouns and verbs) is not feasible, we use all the words extracted from the lists.

A. MATERIAL AND METHOD

The main setup with its parts is described in sections IV-B and IV-C, where corpora, chosen languages, chosen PoS tags, and the computing parts like word-list creation, word-to-IPA, IPA-to-Articulatory features, and then the computation of similarities are described. The modified edit distance calls two algorithms for the phonetic difference of vowels and consonants. These algorithms are already defined in sections III-B and III-C for finding the distance of sound and the modified edit distance respectively. The complete system diagram to show the whole process of finding lexical similarity is shown in figure 4.

In the system diagram (figure 4), the first block represent the dataset taken from the universal dependencies platform for all languages. The next block is about the extraction of word -lists on the basis of part of speech for all the languages subjected for lexical similarity in this study. Next step converts the words into respective IPA and assigns articulatory features. Then, the next block PED is the main method of lexical computation. i.e. Phonetic Edit Distance, which calls the procedure's PDV (Phonetic difference for Vowels) and PDC (Phonetic difference for Consonants) to compute the lexical similarity of two strings and then recursively for two languages.

B. LANGUAGES, SCRIPTS, CORPORA, AND PoS TAGS

For the LS experiment, we chose six languages that are written in two entirely different scripts, i.e., Perso-Arabic and Devanagari. The reasons behind choosing these languages are: first, the Universal Dependency (UD) corpora in considerable size of these six languages are available; and secondly, the conversion system of text-to-IPA for these languages/scripts is created. The six languages involved in the experiments are Arabic, Persian, and Urdu (all written in Arabic script [13], [37]), as well as Hindi, Marathi, and Sanskrit (all written in Devanagari script [37]). The PoS tags involved in this experiment are: adposition; auxiliary; coordinating and subordinating conjunctions; determiner; particles; pronouns; nouns; proper nouns; and verbs.

C. CALCULATING LEXICAL SIMILARITY

Important components of the LS experiment are systematically described in the following sub-sections

1) CREATING WordLists

The UD corpora are available on the website of Universal Dependencies. The corpora have the tagged information of dependency structures modeled in CoNLL-U format [38]. We extracted PoS-wise word lists by processing these structures. The dependency structures have lemma corresponding to the word, e.g. the lemma like corresponds to the words like, likes, liked, and liking. Thus, we chose the lemma instead of the word, as it reduces the size of the word list and the need for comparison of words in different lists. Hence, the word-lists and words mentioned in the following text are lemma-lists and lemmas respectively. However, we retain the use of the term 'word' in the following discussion.

2) WORD-TO-IPA CONVERSION

This component of the system converts the word into the corresponding IPA string. It is considered that the word-to-IPA conversion is available for many languages, for example, Brierley *et al.* [38] developed a verified scheme of word to IPA mapping for Arabic script aligning with pronunciations

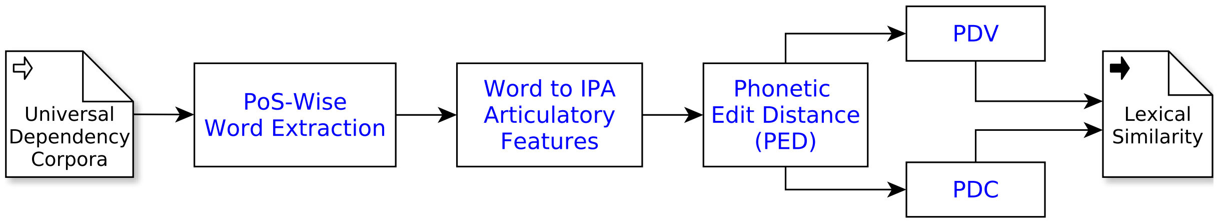

Universal Dependency Corpora
PoS-Wise Word Extraction
Word to IPA Articulatory Features
Phonetic Edit Distance (PED)
PDV
PDC
Lexical Similarity

FIGURE 4. **Overall scheme of the proposed work for finding the lexical similarity.**

carefully, as this scheme was developed for 'Quranic Arabic' including short vowels, therefore, it is not suitable for our proposed languages as the Devanagari script languages (Hindi, Marathi, Sanskrit) do not follow such short vowels. Also, each language has its limitations and extra rules of pronunciation which cause issues to map a corresponding IPA letter from the standard IPA chart, so, to include all the complex parts of phonetics of each language is a very complex task and out of the scope of this work. As most of the paper/online dictionaries have IPA entries corresponding to the word, for Devanagari script languages IPA charts corresponding to script characters are presented in these works [39]–[44] and Wiktionary.[3]

Moreover, many languages have a simple one-to-one letter to IPA mapping. Hence, we infer that IPA strings are easier to obtain and entertain in this system. Also, IPA is a standard and it is easy to make a comparison across the languages by representing in the same encoding scheme of IPA. We implemented a small module for the orthographic word-to-IPA conversion for Arabic and Devanagari script along with phonetic features. We handcrafted the rules and developed the phonetic features (discussed in section III) by using the Wiktionary[4] and [39]–[44] works, for the Arabic script we used the IPA chart developed by CLE[5] and Brierley *et al.* [38]. Devanagari script does not have short vowels, and we omitted these too in the Arabic script conversion, as it is to be compared with the Arabic script. This module is an *ad hoc* arrangement because we can have multiple methods for mapping of word to IPA in different languages, and even for the same language. Currently, we implement a mapper for script-to-IPA.

However, a one-to-one mapping of letters to IPA creates some questions, like what if the pronunciation of two same letters is different in different languages based on previous and next character? Therefore, we used and cross-verified our approach of converting word to IPA with a well-recognized approach namely `Epitran`[6] [45] presented on GitHub as an open-source. As Epitran does not provide IPA for short vowels in Arabic script languages (Arabic, Persian, Urdu) and replaces the same letter in the outcome. For the sake of Arabic script languages, we resolved this problem by making a look-up in our character to IPA mapper[7] for the short vowels and missing characters by `Epitran`. In addition to it, the IPA characters returned by Epitran were holding different Unicode's due to the different language script (for example, any same character for Urdu and Arabic could have different Uni-code) were normalized by mapping on a single character by using the `UrduHack`'s[8] normalization module.

3) IPA-TO-ARTICULATORY FEATURES

The list of articulatory features for phonetic matching is presented (in table 1 and 2) in section III. The IPA string is converted into a list of articulatory features. Currently, we have designed an articulatory feature vector for six languages, three Arabic script (Arabic, Persian, Urdu) and three Devanagari script (Hindi, Marathi, Sanskrit). The mapper or file containing the articulatory features corresponding to characters is present at the address in footnote-1. However, an enhancement could be made by extending the list of features and features corresponding to the remaining IPA symbols.

4) COMPUTING LEXICAL SIMILARITIES

The components described in the above sections give us a (PoS-wise) list of words of different languages. For computing the LS of two languages, we need all the words in language one to be converted into a feature vector including articulatory feature weights for each character. For example, if a word consists of five characters, then the word will be tokenized and a list of five characters will be developed including articulatory feature weights for each character representing the word as a unit of a feature vector. A detailed discussion on the conversion of language words into IPA-based articulatory features is given in section IV. This way, all the words (feature vectors) will be compared from language one to all the words (feature vectors) of language two. Similarly, this process was repeated for all the words of language two on language one and the average similarity results are quoted in the paper.

Now, for the lexical similarity, consider the two word-lists, L_1 and L_2; $L_1 \neq L_2$, entertaining the same PoS. Algorithm 4 shows the comparison of words in L_1 and L_2 to find the LS in two languages. The words w and x, as shown in steps 3 and 4

[3]https://en.wiktionary.org/wiki/Appendix:Hindi_pronunciation
[4]https://en.wiktionary.org/wiki/Appendix:Arabic_script
[5]https://www.cle.org.pk/Downloads/langproc/Urdu_IPA_to_Sampa.pdf
[6]https://github.com/dmort27/epitran
[7]see footnote 1 above.
[8]https://github.com/urduhack/

of algorithm 4, encompass articulatory features. In step 5, the distance of w and x is computed by calling the function PED from Algorithm 3.

The minimum value of edit distance as a result of the comparison of a word w of L_1 with all the words x of L_2 one by one is added to the Ψ all that have the overall value of edit distance. The word which has the minimum distance in this step represented as $\Psi[\gamma]$ is aggregated to the total distance represented as Ψ_{all}. Dividing the Ψ_{all} by the length of L_1 (as shown in step 8) gives the average per letter PED ($\mu\Psi$) of the two lists, also this algorithm repeated for the words of language two over the words of language one. Then, the average similarity of both languages is computed by procedure AvgLexSim. If this $\beta\Psi = 0$ then the lists are identical, similarly, if it is 1 then the lists are entirely different. A smaller value (closer to 0) shows that most of the words are the same or similar in sound. The larger values (closer to 1) show that most of the words in the two lists are different.

Algorithm 4 Lexical Similarity Between Two Lists

```
1: procedure LexSim($L_1, L_2$)    ▷ $L_1$ and $L_2$ be the list of words in two different languages.
2:     $\Psi_{all} \leftarrow 0$
3:     for every word $w$ in $L_1$ do
4:         for every word $x$ in $L_2$ do
5:             $\Psi[x] \leftarrow \mathrm{PED}(w, x)$
6:         end for
7:         $\gamma \leftarrow$ be the key in the $\Psi$ where value is the least in all key-value pairs.
8:         $\Psi_{all} \leftarrow \Psi_{all} + \Psi[\gamma]$
9:     end for
10:    $\mu\Psi \leftarrow \Psi_{all}/|L_1|$
11:    return $\mu\Psi$
12: end procedure
1: procedure AvgLexSim($L_1, L_2$)    ▷ $L_1$ and $L_2$ be the list of words in two different languages.
2:     $\mu\Psi_1 \leftarrow$ LexSim($L_1, L_2$)
3:     $\mu\Psi_2 \leftarrow$ LexSim($L_2, L_1$)
4:     $\beta\Psi \leftarrow \frac{\mu\Psi_1 + \mu\Psi_2}{2}$
5:     return $\beta\Psi$
6: end procedure
```

V. RESULTS AND DISCUSSION

We compared PoS wise word lists of six languages using the algorithms described above, and the results are presented in figure 5. These languages have genetic as well as social affinities with each other. Arabic belongs to the Semitic family of languages. Persian belongs to the Iranian branch of the Indo-European$\rightarrow$Indo-Iranian languages.

Urdu, Hindi, Marathi, and Sanskrit belong to Indo–Aryan branch of the Indo–European$\rightarrow$Indo–Iranian languages. Sanskrit is an ancient language; however, the other three are modern languages. Urdu has much social interaction with Arabic and Persian, so it has borrowed vocabulary and phonetics from these languages; otherwise, Hindi and Urdu are different variants of the same language. Arabic, Persian, and Urdu are written in Arabic script, while Hindi, Marathi, and Sanskrit are written in the Devanagari script.

Figure 5 depicts the PoS wise lexical similarity using heat maps. The lighter color shows a higher value of PED per letter and hence, lower similarity. The darker color shows a lower value of PED per letter and hence, higher similarity. We did not calculate the similarity of some pairs when there are less than 5 words in one of the lists. The tiny or empty lists are not suitable for inferring some results.

We find that for five out of ten PoS, the top two similar languages (i.e., having the lowest PED) are Indo Aryan languages. These languages w.r.t to PoS are: for adpositions Persian–Urdu and Persian–Hindi; for auxiliaries Arabic–Urdu and Arabic–Hindi; for determiners Arabic–Urdu and Hindi–Marathi; for particles Urdu–Arabic, and Hindi–Marathi; and for pronouns Hindi–Sanskrit and Hindi–Marathi.

All of these PoS are the closed-class, i.e. new words are usually not added to these lists. For the remaining PoS, we have Hindi–Marathi and Urdu–Arabic for coordinating conjunctions; Urdu–Persian and Persian–Hindi for Subordinating conjunctions. These, too, belong to the closed-class of words. Persian and Hindi belong to the same sub-family of Indo-Iranian languages; however, we expect a closer affinity of Marathi with Hindi or Urdu.

Further, we find Arabic–Persian and Urdu–Arabic for nouns, and Urdu–Arabic and Urdu–Hindi, as the most similar (i.e., having lowest PED) languages for proper nouns. These PoS are open-class, and we expect to borrow them from Arabic for the words of these PoS.

Amongst all, the verb shows the most matching results. We find substantial similarity of Arabic verbs with Urdu, which is correct. Urdu borrows many verbs from Arabic in daily usage language. The other good matching of verbs is found in Hindi–Sanskrit and Hindi-Marathi languages. Hindi and Marathi borrow many verbs from Sanskrit (the older language from both of these languages).

Although the results of the proposed methodology have a limitation of false positives. Thus, we assume that the average of the false positives will be the same in all of the pairs. This assumption holds for the majority of the PoS wise language pairs. However, it may need some modification for two small words or too small word lists. When we browsed through the content of different PoS wise word lists, we found that the parallelism of similar design principles does not hold in some cases. The count of pronouns in Urdu, Hindi, and Marathi are 110, 63, and 24, respectively. Similarly, the count of Urdu, Hindi, and Marathi auxiliaries are 78, 49, and 19, respectively.

There should not be such a difference in closed-class word lists of closely related languages. The reason is excluding a subclass of words as they give PoS or putting the inflected form as the lemma in the CoNLL-U structures.

The lexical similarity of two or more languages needs the most relevant and frequently used content to compare. For

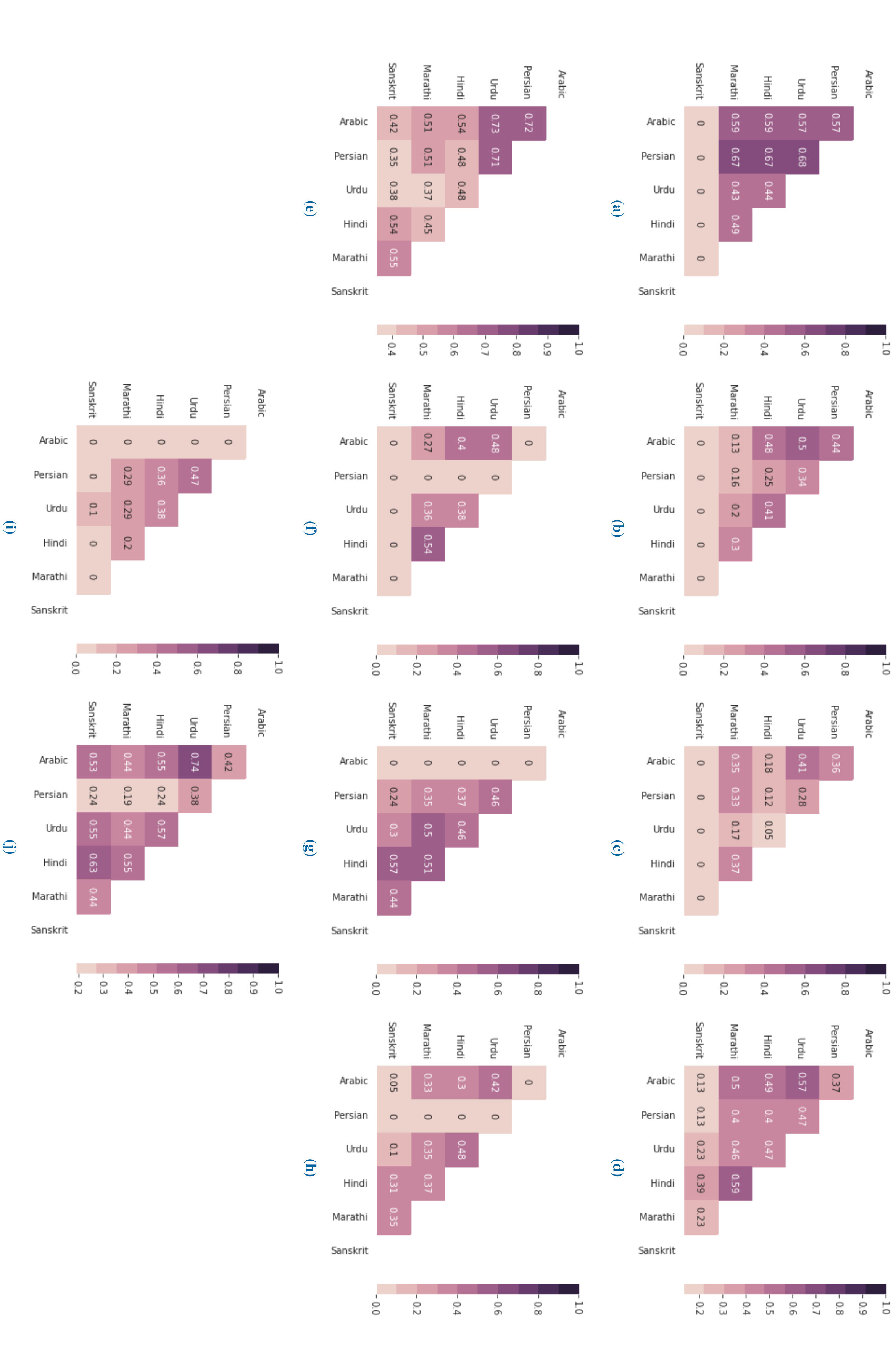


FIGURE 5. Heat maps for the PoS-wise lexical similarity. The sub-figures reveals (a) adposition, (b) auxiliary, (c) cconj, (d) det, (e) nouns, (f) particle, (g) pronouns, (h) proper noun, (i) sconj, and (j) verbs.

this task, we picked the PoS of each language from UD. Many words have much difference in the written script but are pronounced or voiced the same.

To underlying this, we used phonetic distance using articulatory features which is very near to real comparison of the languages for identifying the ratio of similarity and borrowing of words from other languages. In our experiment, it is analyzed that the verbs and nouns of Arabic, Persian, and Urdu are very similar, although Persian is older than the Urdu language which shows that Urdu borrowed many words from Persian.

Similarly, Urdu and Hindi are similar as both languages have the same region and sharing of writers and speakers. Arabic is similar to Urdu and Persian in nouns and proper nouns as all three languages share a religious perspective. Also, it is analyzed that the data for other languages like Marathi and Sanskrit was not enough on UD for the equal comparison with the rest of the languages taken for the experiment. The overall experiment and its results are satisfying.

VI. CONCLUSION AND FUTURE WORK

We presented an algorithm for articulatory feature-based phonetic edit distance (PED). This algorithm helps to find lexical similarity and to identify the loan words (cognates) that have different spellings for which IPA mapping is different among languages. We used the PED to find the lexical similarity of PoS wise lists of different pairs of languages. Most of the calculated similarities are in agreement with the genetic affinity of the compared languages. Despite the languages have different scripts, but we have aligned all the languages with the modified method of phonetic edit distance based on articulatory features. This method has given us good results which are comprehensible in comparison to the simple string matching which shows more difference and less similarity.

Hence, the method can be used on a more extensive set of languages after removing the following limitations. The current work used a mapping of the letter(s) to IPA (although we used and verified it with `Epitran` it also lacks the short vowels handling). A better approach is the usage of digitally available lexicographic resources, e.g. dictionaries, etc. It will resolve the many matters of ambiguity, e.g. letter(s) to IPA, silent letters, pronunciation, and unwritten diacritic marks. One can work on a better set of corpora or better cleaning, as we found some problems of (non-)parallel design, implementation errors, and cleaning with UD corpora. A better corpus could be a manually or automatically tagged parallel corpus, e.g. Europarl parallel corpus or Wikipedia dumps.

REFERENCES

[1] F. Genesee, "Dual language in the global village," in *Pathways to Multilingualism: Evolving Perspectives on Immersion Education*, T. W. Fortune and D. J. Tedick, Eds. Bristol, U.K.: Multilingual Matters, 2008, ch. 2, pp. 22–46, doi: 10.21832/9781847690371-005.

[2] J. Errington, "Colonial linguistics," *Annu. Rev. Anthropol.*, vol. 30, no. 1, pp. 19–39, 2001.

[3] H. Cixous, L. Burke, T. Crowley, and A. Girvin, Eds., *The Routledge Language and Cultural Theory Reader*, 1st ed. New York, NY, USA: Routledge, 2003.

[4] B. Comrie, *The Major Languages of South Asia, the Middle East and Africa*, 1st ed. London, U.K.: Routledge, 2003, p. 336, doi: 10.4324/9780203412336.

[5] R. Karhila, A.-R. Smolander, S. Ylinen, and M. Kurimo, "Transparent pronunciation scoring using articulatorily weighted phoneme edit distance," 2019, *arXiv:1905.02639*.

[6] M. S. Nizami, M. Y. Khan, and T. Ahmed, "Towards a generic approach for pos-tagwise lexical similarity of languages," in *Proc. Int. Conf. Intell. Technol. Appl.* Springer, 2019, pp. 493–501.

[7] E. Ukkonen, "Algorithms for approximate string matching," *Inf. Control*, vol. 64, nos. 1–3, pp. 100–118, Jan. 1985.

[8] E. S. Ristad and P. N. Yianilos, "Learning string-edit distance," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 5, pp. 522–532, May 1998.

[9] T. Jiang, G. Lin, B. Ma, and K. Zhang, "A general edit distance between RNA structures," *J. Comput. Biol.*, vol. 9, no. 2, pp. 371–388, Apr. 2002.

[10] J. Zobel and P. Dart, "Phonetic string matching: Lessons from information retrieval," in *Proc. 19th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, 1996, pp. 166–172.

[11] E. J. Yannakoudakis and D. Fawthrop, "The rules of spelling errors," *Inf. Process. Manage.*, vol. 19, no. 2, pp. 87–99, Jan. 1983.

[12] L. Philips, "The double metaphone search algorithm," *C/C++ Users J.*, vol. 18, no. 6, pp. 38–43, 2000.

[13] R. S. Rangila, M. S. Thirumalai, and B. Mallikarjun, "Bringing order to linguistic diversity: Language planning in the British Raj," Lang. India, Bloomington, MN, USA, 2001, vol. 1, no. 6.

[14] A. Beider, "Beider-morse phonetic matching: An alternative to soundex with fewer false hits," *Avotaynu: Int. Rev. Jewish Geneal.*, vol. 24, no. 2, p. 12, 2008.

[15] D. J. Broad, "Toward defining acoustic phonetic equivalence for vowels," *Phonetica*, vol. 33, no. 6, pp. 401–424, Nov. 1976.

[16] A. Atkielski, "Phonetic transcription can be a useful tool for teaching or correcting pronunciation in the ESL/EFL classroom," 2005.

[17] C. M. Scott and R. L. Ringel, "Articulation without oral sensory control," *J. Speech Hearing Res.*, vol. 14, no. 4, pp. 804–818, Dec. 1971.

[18] P. J. Roach, "Laryngeal-oral coarticulation in glottalized English plosives," *J. Int. Phonetic Assoc.*, vol. 9, no. 1, pp. 2–6, Jun. 1979.

[19] P. K. Kuhl and J. D. Miller, "Speech perception by the chinchilla: Voiced-voiceless distinction in alveolar plosive consonants," *Science*, vol. 190, no. 4209, pp. 69–72, Oct. 1975.

[20] F. Kortlandt, "Proto-germanic obstruents and the comparative method," *North-Western Eur. Lang. Evol.*, vol. 52, no. 1, pp. 3–7, 2007. [Online]. Available: https://www.jbe-platform.com/content/journals/10.1075/nowele.52.01kor, doi: 10.1075/nowele.52.01kor.

[21] L. Lombardi, "Dahl's law and privative [voice]," *Linguistic Inquiry*, vol. 26, no. 2, pp. 365–372, 1995.

[22] G. K. Iverson and J. C. Salmons, "Laryngeal enhancement in early germanic," *Phonology*, vol. 20, no. 01, pp. 43–74, May 2003.

[23] J. Coleman and J. Pierrehumbert, Eds., "Computational phonology," in *Proc. 3rd Meeting ACL Special Interest Group Comput. Phonology*, Assoc. Comput. Linguistics, 1997. [Online]. Available: https://aclanthology.org/W97-1100

[24] G. Kondrak, "Identifying cognates by phonetic and semantic similarity," in *Proc. 2nd Meeting North Amer. Chapter Assoc. Comput. Linguistics Lang. Technol. (NAACL)*. Pittsburgh, PA, USA: Assoc. Comput. Linguistics, 2001, pp. 1–8, doi: 10.3115/1073336.1073350.

[25] Q. Do, D. Roth, M. Sammons, Y. Tu, and V. Vydiswaran, "Robust, lightweight approaches to compute lexical similarity," Comput. Sci. Res. Tech. Rep., Univ. Illinois, Champaign, IL, USA, Tech. Rep., 2009, vol. 9.

[26] H. Kaur and R. Maini, "Assessing lexical similarity between short sentences of source code based on granularity," *Int. J. Inf. Technol.*, vol. 11, no. 3, pp. 599–614, Sep. 2019.

[27] W. H. Gomaa and A. A. Fahmy, "A survey of text similarity approaches," *Int. J. Comput. Appl.*, vol. 68, no. 13, pp. 13–18, Apr. 2013.

[28] T. Dijkstra, K. Miwa, B. Brummelhuis, M. Sappelli, and H. Baayen, "How cross-language similarity and task demands affect cognate recognition," *J. Memory Lang.*, vol. 62, no. 3, pp. 284–301, Apr. 2010.

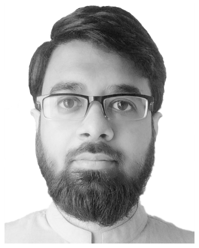
[29] J. Schepens, T. Dijkstra, F. Grootjen, and W. J. B. van Heuven, "Cross-language distributions of high frequency and phonetically similar cognates," *PLoS ONE*, vol. 8, no. 5, May 2013, Art. no. e63006.

[30] M. I. M. García and A. M. B. de Souza, "Lexical similarity level between English and Portuguese," *ELIA*, vol. 14, pp. 145–163, Nov. 2014, doi: 10.12795/elia.2014.i14.06.

[31] H. Carrasco-Ortiz, M. Amengual, and S. T. Gries, "Cross-language effects of phonological and orthographic similarity in cognate word recognition: The role of language dominance," *Linguistic Approaches Bilingualism*, vol. 11, no. 3, pp. 389–417, Sep. 2021.
[32] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer, and C. Bizer, "DBpedia—A large-scale, multilingual knowledge base extracted from Wikipedia," *Semantic Web*, vol. 6, no. 2, pp. 167–195, 2015.
[33] Ö. Ö. Çakal, M. Mahdavi, and Z. Abedjan, "CLRL: Feature engineering for cross-language record linkage," in *Proc. EDBT*, 2019, pp. 678–681.
[34] A. A. Khan, "Lexical affinities between Urdu and French," *Eurasian J. Hum.*, vol. 1, no. 1, pp. 1–19, 2015. [Online]. Available: https://www.eurasianjournal.org/my_article/volume-1/
[35] C. S. Siew and M. S. Vitevitch, "The phonographic language network: Using network science to investigate the phonological and orthographic similarity structure of language," *J. Experim. Psychol., Gen.*, vol. 148, no. 3, p. 475, 2019.
[36] C. Sammut and G. I. Webb, *Encyclopedia of Machine Learning and Data Mining*, 2nd ed. New York, NY, USA: Springer, 2017.
[37] B. Comrie, *The Major Languages of South Asia, the Middle East and Africa*. Evanston, IL, USA: Routledge, 2003.
[38] C. Brierley, M. Sawalha, B. Heselwood, and E. Atwell, "A verified arabic-IPA mapping for Arabic transcription technology, informed by quranic recitation, traditional Arabic linguistics, and modern phonetics," *J. Semitic Stud.*, vol. 61, no. 1, pp. 157–186, Mar. 2016.
[39] K. Samudravijaya, P. Rao, and S. Agrawal, "Hindi speech database," in *Proc. INTERSPEECH*, 2000, pp. 456–459.
[40] K. Arora, S. Arora, S. R. Singla, and S. S. Agrawal, "Sampa for Hindi and Punjabi based on their acoustic and phonetic characteristics [C]," in *Proc. Int. Oriental COCOSDA Conf*, Hanoi, Vietnam, 2007, pp. 17–22.
[41] R. K. Joshi, T. N. Dharmadhikari, and V. V. Bedekar, "The phonemic approach for Sanskrit text," in *Sanskrit Computational Linguistics*. Berlin, Germany: Springer, 2007, pp. 417–424.
[42] K. Samudravijaya, "Indian language speech label (ILSL): A de facto national standard," in *Advances in Speech and Music Technology*. Singapore: Springer, 2021, pp. 449–460.
[43] G. Kalpesh, "The use of sanskrit, an ancient language, as a tool to evaluate cleft palate speech problems," *Indian J. Plastic Surg.*, vol. 40, no. 2, pp. 91–93, Jul. 2007.
[44] T. A. Hall, "The historical development of retroflex consonants in indo-aryan," *Lingua*, vol. 102, no. 4, pp. 203–221, Aug. 1997.
[45] D. R. Mortensen, S. Dalmia, and P. Littell, "Epitran: Precision G2P for many languages," in *Proc. 11th Int. Conf. Lang. Resour. Eval. (LREC)*, 2018, pp. 1–5.

TAFSEER AHMED received the Ph.D. degree from the Universität Konstanz, Germany, in 2009. He has teaching and research experience of more than 20 years. He has worked on lexical functional grammar, universal dependencies, PoS tagging, author attribution, transliteration, named entity recognition, and other NLP applications for Pakistani languages specially Urdu. He has worked on Urdu Propbank for the University of Colorado Boulder. He was a co-PI of the DAAD, Germany, funded project "Urdu Text to Speech: Understanding Intonation."

MUHAMMAD SUFFIAN received the B.S. degree in computer science from Sukkur IBA University, in 2016, and the M.S. degree in computer science from Mohammad Ali Jinnah University, Karachi, Pakistan, in 2018. He is currently pursuing the Ph.D. degree with the Università degli Studi di Urbino Carlo Bo, Italy. From 2016 to 2020, he was with the Department of Computer Science, Mohammad Ali Jinnah University, as a Lecturer during the same period while he was the part of the Center for Language Computing, Mohammad Ali Jinnah University. His research interests include human-centered artificial intelligence, machine learning, deep learning, natural language processing, computational linguistics, and explainable artificial intelligence.



MUHAMMAD YASEEN KHAN received the B.S. degree in software engineering from the University of Karachi, in 2010, and the M.S. degree in software engineering from the Karachi Institute of Economics and Technology (KEIT), in 2015. He is currently pursuing the Ph.D. degree in computer sciences with Mohammad Ali Jinnah University (MAJU), Pakistan. His experience in the software development industry includes working as a (Vendor) Software Development Engineer for Microsoft at Mazik Global/Tyler Technologies Inc.; a Data Scientist at Cubix Labs; and working with Love For Data as the Principal Research and Development Engineer at NLP. His research interest includes the combination of applied artificial intelligence, psychology, sociology, and NLP.

ALESSANDRO BOGLIOLO received the Laurea degree in electrical engineering and the Ph.D. degree in electrical engineering and computer science from the University of Bologna, Italy, in 1992 and 1998, respectively. From 1992 to 1999, he was with the Department of Electronics and Computer Science (DEIS), University of Bologna. In 1995 and 1996, he was with the Computer Systems Laboratory (CSL), Stanford University, CA, USA. From 1999 to 2002, he was with the Department of Engineering (DI), University of Ferrara, Italy. He joined the Università degli Studi di Urbino Carlo Bo, Italy, in 2002, where he is a Full Professor of computer systems. In 2019, he co-founded Digit srl, benefit corporation for digital social innovation. His research interests include mobile crowdsensing, sensor networks, computational linguistic, and digital platforms for sustainability and participatory social innovation.