Sindh Univ. Res. Jour. (Sci. Ser.) Vol.43 (2) 119-126 (2011)

SINDH UNIVERSITY RESEARCH JOURNAL (SCIENCE SERIES)

Sindhi Diacritics Restoration by Letter Level Learning Approach

J. A. MAHAR, G. Q. MEMON, AND H. SHAIKH*

Faculty of Engineering, Science and Technology, Hamdard University, Karachi
E-mails:, gqmemon@yahoo.com, hidayat.shaikh@salu.edu.pk
Ph: +92-334-2727937; Fax: +92-243-9280439

Corresponding author Javed A. Mahar, email: mahar.javed@gmail.com,

Received 24rth June 2011and Revised 12th August 2011)

Abstract: Sindhi is one of those languages that require diacritics for exact reading and comprehension, but in routine compositions diacritics are almost ignored. Hence it brings about many syntactical, morphological and phonological ambiguities for computational processing. The diacritics can be restored at letter and word levels, in this paper, letter level learning method is used for the task of Sindhi diacritics restoration in which surrounding letters of the specific letter are calculated and stored into a feature vector in order to compare them with the new examples which are input from the non-diacritized text. These letters are computed with different window sizes, the N=5 is observed most efficient one. The *k*-nearest neighbor classifier is implemented for the classification of instances and at last, the nearest instance is taken for the replacement of non-diacritized letter. The evaluation of results is represented in terms of Diacritic Error Rate (DER), which is 1.9%. The proposed approach is tested on Sindhi but can be used for other Arabic script based languages because the character set of Sindhi is the superset of Arabic character set.

Keywords: Diacritics Restoration; Sindhi Language; K-NN; Letter Level Learning;

1. INTRODUCTION

Diacritization helps the language for being comprehensible, vivid, audible, and easily readable. Arabic script based languages has two types of vowels: Long vowels are written as normal letters and short vowels are written as punctuation marks (Safadi, *et al.,* 2006), however short vowels are placed only where there is inadequate context. People mostly don't use diacritics while writing or typing text because it's a time taking way, in such situation, the reader can comprehend only on the bases of his own knowledge and experience about the concerned language however, context of the words can also be a helping sign. The absence of diacritics creates large number of possible vowel combinations for the same set of characters which composes the word. Furthermore, the written form of a homographic word having several pronunciations and with each one carrying different meaning creates the semantic and syntactic ambiguities for readers as well as for computational processing, without disambiguation, it is too difficult to produce the correct pronunciation of words (Elshafei, 1991) Schlippe, *et al.,* 2008)

Guessing diacritics is indeed a very essential component for developing various natural language and speech synthesis systems using Arabic script based writing systems (Kirchho, *et al.,* 2002 Zitouni, *et al.,* 2006, Shaalan, *et al.,* 2009) For the widespread usage of computers in linguistics applications; Sindhi texts need to be supplied with diacritics in order to be correctly vocalized before being processed. Thus the application of automatic diacritics restoration for Sindhi computational processing is as important as the life for a language.

The diacritics restoration is a lexical disambiguation task in natural language processing Nguyen, and Ock, (2010) This problem has been solved in some languages by various researchers using word level Gal (2002) Elshafei, *et al.,* (2006) Harby, *et al.,* (2008) Ali, (2009) Alghamdi, *et al.,* (2010) letter level Mihalcea, (2002) Mihalcea, and Nastase, (2002) Zitouni and Sarikaya, (2008) combination of word and letter level Ananthakrishnan, *et al.,* (2005) Nelken and Shieber (2005) Tufis, and Ceausu, (2007) Tufis, and Ceausu, (2008) Schlippe, *et al.,* (2008) and grapheme level Kanis, and Muller, (2005) Wagacha, *et al.,* (2006) Pauw, *et al.,* (2007) Roth, *et al.,* (2008) approaches; each approach has qualities and deficiencies with respect to different natures of languages.

*Department of Computer Science, Shah Abdul Latif University, Khairpur

Most of the researchers used word level approach for diacritics restoration whereas few selected letter level approach. For instance, Maximum Entropy models are used by (Zitouni and Sarikaya, 2008) in which three features i.e., lexical, segment-based and parts of speech tags are integrated, the combination of these features yield high level accuracy and achieved the diacritic error rate of 5.1%. The instance based learning mechanism for diacritic restoration of Romanian language is presented by (Mihalcea, 2002) that acts at letter level. Simple features and their rules are used in learning algorithm. Four ambiguous pairs of letters are considered for learning, for each one, text is scanned and generated all possible examples encountered in the corpus. The attributes are formed by N letters to the left and right of ambiguous letter and target attribute is the ambiguous letter itself. The proposed algorithm is particularly useful for handling unknown words and accuracy of over 99% is achieved. Same learning algorithm is implemented on four languages namely Czech, Romanian, Polish and Hungarian by (Mihalcea, and Nastase, 2002) and the average accuracy of 98% is achieved.

2. MATERIALS AND METHODS

Material

The corpus of language is necessary for computational exploitation. Therefore, various lexicons for Arabic and other languages are available on the web but there are no large electronic dictionaries available, nor extensive corpora of Sindhi language, only small-sized dictionaries are available which are not reliable to perform any method for diacritics restorations. Any rich language like Sindhi has ability to continuously adopt words of other languages to meet the modern standards of communications. Since five decades, the big difference could be seen in written and spoken Sindhi. The corpora of Sindhi language are classified as modern Sindhi text and old Sindhi text, therefore, two kinds of corpuses: Corpus of Modern Sindhi Language (CMSL) and Corpus of Shah Jo Risalo (CSJR) are developed which have fully diacritized text.

Corpus of Modern Sindhi Language (CMSL)

The corpus of recently published literature about arts, sports, politics, environment and music is collected from different genres like newspapers, magazines and books from Internet. The pages are downloaded in HTML and PDF formats, after downloading; pages are converted into their plain text equivalents. The CMSL contains approximately 3 million words. The detailed statistical information of this corpus is shown in **(Table 1)**.

Table 1: Statistical Information of CMSL

Corpus Type	Sentences	Word Tokens	Word Types
Arts	31157	685504	48416
Sports	22184	421496	38317
Politics	40333	847071	65164
Environment	33900	640104	49546
Music	18544	330792	37088
Total	**146,118**	**2,924,967**	**238,531**

Corpus of Shah Jo Risalo (CSJR)

The corpus of old Sindhi is collected from شاه جو رسالو (Shah Jo Risalo) because in this book original Sindhi language is used by the great poet Shah Abdul Latif Bhitai. The CSJR contains approximately 27360 words. Shah Jo Risalo managed by Aadwani Aadvani, K. (2009) is used which is divided into 30 سُرَ (Chapters), each chapter is based on poems and lays. The total number of poems is 1579 and approximate length of a poem is from 2 to 11 lines. There are 43 lays in this corpus and a lay consists of approximately from 4 to 16 lines. The statistical information of the words in CSJR is shown in **(Table 2)**.

Table 2: Words Information of CSJR

Total No. of Words	27360
Total No. of Word Types	10894
Total No. of Non-Critical Words	9085
Total No. of Critical Words	1809

Method

The diacritization task at both, word and letter level can be achieved through various statistical methods such as hidden markov model, maximum entropy and others. In order to accomplish maximum possible accuracy, these methods are used with the combination of linguistic tools, i.e. parts of speech tagger and morphological analyzer. In this paper, diacritics restoration is performed at letter level with the use of instance based learning, also known as memory based learning. It is a distinct feature of this method that no supporting tools are required for its process. The major aim to use this mechanism is to produce a unique methodology for those languages which don't have a large number of lexical and semantic resources, particularly with which word based diacritization, can bring in a great many difficulties that embed hurdles to perform the task of diacritics restoration. Another advantage of this mechanism is that it can deal also with those words which are out of vocabulary; hence such method may be used generally with many languages for diacritics restoration regardless of linguistic and grammatical rules.

Learning Algorithms

Among all, machine learning research is especially focused for automatically learning the recognition of complex patterns so that the decision basing on data should be made efficiently. There lies a complication because the data set of total possible conditions provided maximum possible inputs is too extensive to be covered with the processed training data set. Therefore learner should generalize from stored training data, so as to enable for producing a better output in upcoming cases.

The purpose of using the instance based learning algorithm is to compare new problem instances with the instances that are stored in memory. Instance based learning has the advantage over other machine learning methods is its ability of adoption for its model to previously unprocessed data. On the contrary, other methods usually need total set of instances for re-analyzing when only one set is changed.

K-nearest neighbor algorithm is the simplest example of an instance based learning one Hullermeier, *et al.,* (2001) whereas the K-NN method classifies the objects based on closest training examples in the feature space. The basic model is given below:

$$f(x_q) = \frac{\sum_{i=1}^{k} f(x_i)}{k} \qquad (1)$$

Every input instance is compared with all the nearest neighbors through the process of classifier KNN, finally the method opts for the most often represented ones. A multidimensional array stores training examples consisting vectors of features. A category label indicates each example with regard to its class. A majority of votes along with its neighbors classifies the given object in this method. If the value of k is equal to 1 then object is assigned to the class of its nearest neighbor. Cross-validation technique is used for the determination of the value of k; the given data creates the basis for the determining process.

There are two components in memory based system: a learning component that is memory based and other is performance component; the latter is similarity based one. **(Fig.1)** depicts the architecture of memory based system. The learning component directly appends training instances to the memory, hence called memory based. An instance is actually a fixed-length vector of n feature-value pairs; it consists on information field which contains the classification of that particular feature-value vector, where as the

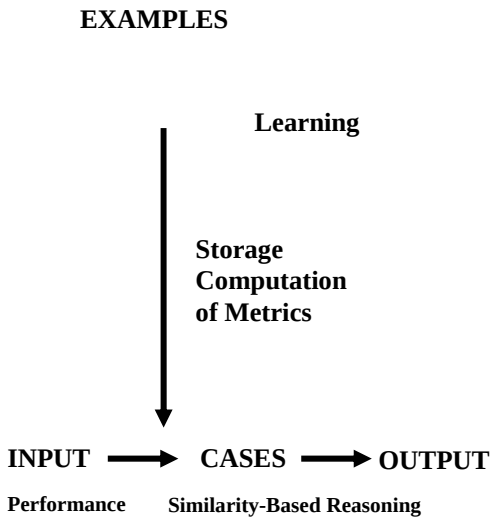

Fig.1. Architecture of Memory Based Learning

performance component is basically a product of the learning component that is used as a base to map input to output; this eventually turns into the form of performing classification. During the process of classification, a new test instance is given to the system, making use of distance metric Δ(X, Y) computes the similarity between the new instances X and entire examples Y in memory, this process is performed with the use of Overlap metric where distance between instances is shown by n features, and δ shows the distance per feature.

$$\Delta(X, Y) = \sum_{i=1}^{n} \delta(x_i, y_i) \qquad (2)$$

Where:

$$\delta(x_i, y_i) = \begin{cases} abs\left(\frac{x_i - y_i}{\max_i - \min_i}\right) & \text{if numeric, else} \\ 0 & \text{if } x_i = y_i \\ 1 & \text{if } x_i \neq y_i \end{cases}$$

The described method of this metric counts the total number of feature-values in both patterns whether matching or mismatching so that domain knowledge bias must be added to weight. The statistical information can be computed for the weight of features by examining which are better predictors of the class labels. Each and every feature is examined separately with Information Gain (IG) weighting (Lee, C., Lee, G. G. 2006) and measured for the quantity of information that it produces to stored knowledge for the right class label. Between the situations with and

without knowledge of the value of the feature I, IG of this feature is measured by computing the difference in no certain situation.

$$w_i = H(C) - \sum_{v \in V_i} P(v) \mathrm{x} H(C \mid v) \qquad (3)$$

$$\Delta(X, Y) = \sum_{i=1}^{n} w_i \ \delta(x_i, y_i) \qquad (4)$$

The process of extrapolation is performed by assigning the most frequent class within the stored set of most similar examples as the category of new test examples. When some situation brings about a tie among categories, a tie breaking resolution method is made in use through that the value of the *k* parameter increases by 1, and the additional nearest neighbors at the new *K*th distance are joint to the current nearest neighbor set (*k* must be rearranged to its user defined value), if the tie still persists then with the highest overall occurrence of training data set the class label is chosen, if the occurrence is also matched then method takes the first class that was initially encountered while processing the training instance file.

Features

The features that are taken into use in any algorithm have great impact on the eventual accuracy. In this regard, surrounding letters are watched, with a particular notation assigned to white spaces, commas and dots, similar to (Mihalcea 2002). Such set of features contributes surprisingly well for achieving the highest accuracy. A particular text is scanned for each ambiguous pair of letters then all possible examples which encountered while processing the given corpus are generated. The N letters at both sides left and right of the ambiguous letter form all characteristics in an example, where as the target attribute is also the ambiguous letter itself.

3. IMPLEMENTATIONS AND RESULTS

The ambiguous letters appoint the target attribute for being learned; hence the work is done at low level of letters. Three vital diacritics i.e. Zabar, Zair and Pesho used frequently in Sindhi are considered in experiments. They are also called short vowels. There are other diacritic symbols in Sindhi but their use is infrequent in routine applications. Where as, the previously said three symbols keep great importance in daily used compositions. Its interesting that no matter how large or short is the number of diacritics in a language, it does not influence the accuracy achieved in the process of diacritization.

Feature Vectors

For the purpose of experiments, the corpuses of CSJR and CMSL are selected, 1,338,867 examples are extracted from CMSL and 325,607 examples are extracted from CSJR with a particular notation assigned to white spaces (SP), commas (CO) and dots (DO). These examples are stored into a multidimensional array that is based on the features of vectors. The sample of corpus is depicted in **(Fig.2)** and the sample of feature vectors extracted from this part of corpus is shown in **(Table 3)**.

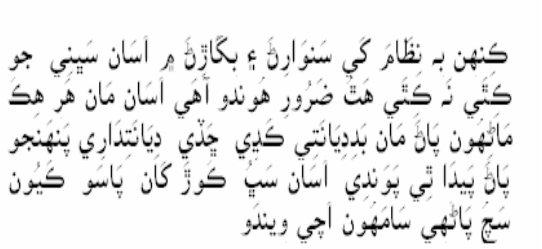
كنهن بہ نظام کي سنوارڻ ۽ بگاڙڻ ۾ آسان سينئي جو
کٿي نہ کٿي هٿ ضرور هوندو آهي آسان مان هر هڪ
ماڻهون پاڻ مان بدديانتي کديي جڏي ديانتداري پنهنجو
پاڻ پيدا ٿي پوندي آسان سڀ کوڙ کان پاسو کيون
سڀ پاڻهي سامهون آچي ويندو

Fig.2. Sample Part of Text from Proposed Corpus CMSL

Table 3: Sample Feature Vectors from Corpus CMSL

Letters	Feature Vectors
ـَـ	: ا ,ن ,ت ,ي ,SP ,ٺ ,SP ,چ ,ڊ : پ ,ا ,س ,و ,SP ,SP ,CO ,ن ,و ,ي
ـِـ	: ن ,هه ,ن ,SP ,ب ,SP ,SP ,SP ,SP ,SP : ي ,SP ,ج ,و ,SP ,ٿ ,SP ,ک ,ٿ ,ي : ٿ ,ي ,ن ,هه ,ٿ ,SP ,هه ,ي ,ٿ
ـُـ	: SP ,هه ,ر ,SP ,هه ,و ,ٹ ,ا ,م ,SP : ن ,SP ,س ,پ ,SP ,ا ,ک ,SP ,ڙ ,و
هَ	: SP ,ک ,ٿ ,ي ,SP ,و ,ر ,ض ,SP ,ٿ : ن ,د ,و ,SP ,ا ,ا ,س ,ا ,DO ,ي : SP ,م ,ا ,ن ,SP ,SP ,ک ,هه ,SP ,ر : ر ,ي ,SP ,پ ,ن ,پ ,SP ,و ,ج ,ن
هِ	: ن ,SP ,ن ,هه ,ن ,ن ,ک ,SP ,SP ,SP : ن ,SP ,هه ,ر ,SP ,هه ,ٹ ,م ,SP ,ک : چ ,SP ,پ ,ا ,ٹ ,م ,ا ,س ,SP ,ي
هُ	: ض ,ر ,و ,ر ,SP ,SP ,و ,د ,ن ,و : ک ,SP ,م ,ا ,ٹ ,ٹ ,ا ,پ ,SP ,و : ي ,SP ,س ,ا ,م ,چ ,ا ,SP ,ن ,و

Results

Among the entire set of examples extracted from both corpuses for every ambiguous letter set, 100 examples are taken for testing from each set. The total

number of letter sets is 52 so that 100 x 52=5,200 examples are set aside for testing and the rest is used for training the learner. From the input text, system selects each letter one by one along with its first neighbors from each side and compares with the stored examples through the process of KNN classifier. The system calculates the value of each feature of vector and then stored them into created metric. All values are weighted and assigned the labels regardless of matching or mismatching, examples are classified according to the assigned class labels, and finally the method opts for the most often represented ones. The results with CMSL and CSJR are shown in Table 4 and Table 5 respectively, the tables encompasses the sets of ambiguous letters, the number of examples extracted from the corpus for each ambiguous set, and the precision obtained with the instance based learner used for application at letter level.

Table 4: Ambiguous Set of Letters and Examples with CMSL

Ambiguous Set	Total Examples	Precision Achieved	Ambiguous Set	Total Examples	Precision Achieved
اَ اُ اِ	126738	95.63%	زُ ز زَ	2449	96.84%
بَ بُ بِ	30307	99.59%	سُ س سَ	33284	96.94%
پَ پُ پِ	16941	99.11%	شُ ش شَ	10744	99.55%
پُ پِ پَ	19759	98.88%	صُ ص صَ	3952	95.76%
تَ تُ تِ	29663	99.36%	ضُ ض ضَ	1231	96.94%
ٹُ ٹِ ٹَ	20488	99.55%	طُ طِ طَ	1473	98.89%
ٹُ ٹِ ٹَ	18721	99.01%	ظُ ظِ ظَ	499	98.58%
ٹُ ٹِ ٹَ	11693	99.87%	عُ عِ عَ	15491	94.91%
ثَ ثُ ثِ	4652	97.96%	غُ غِ غَ	2890	95.92%
پَ پُ پِ	20083	99.19%	فُ فِ فَ	12840	99.41%
جَ جُ جِ	40621	93.84%	قُ قِ قَ	1752	99.70%
جِهِ جِهَ جِهُ	7381	97.88%	قُ ق قَ	1157	99.51%
چُ چِ چَ	1392	99.50%	کُ کِ کَ	45032	99.03%
چُ چِ چَ	713	99.62%	کُ کِ کَ	33495	99.19%
چُ چِ چَ	21852	98.71%	گُ گِ گَ	17720	99.66%
چُ چِ چَ	18139	99.55%	گِهِ گَهَ گَه	8502	95.55%
حُ حِ حَ	23799	99.98%	گُ گِ گَ	885	99.11%
خُ خِ خَ	11831	99.77%	گُ گِ گَ	382	97.59%
دُ دِ دَ	36426	99.99%	لُ ل لَ	40894	99.96%
ڈُ ڈِ ڈَ	7493	94.77%	مُ م مَ	55275	99.77%
دُ دِ دَ	474	96.71%	نُ ن نَ	121690	93.61%
ڈُ ڈِ ڈَ	31695	99.33%	ڻُ ڻ ڻَ	881	97.99%
دُ دِ دَ	5195	99.82%	وَ وُ و	81661	99.38%
دُ ذِ ذَ	1019	97.99%	هِ هَ هه	130685	96.88%
رُ ر رَ	43832	99.01%	ءَ ءُ ء	16044	95.72%
ڑُ ڑ ڑَ	8493	99.22%	يُ ي يَ	138559	93.62%

Table 5: Ambiguous Set of Letters and Examples with CSJR

Ambiguous Set	Total Examples	Precision Achieved	Ambiguous Set	Total Examples	Precision Achieved
اَ اُ اِ	22684	95.21%	زُ ز زَ	4954	97.48%
بَ بُ بِ	4951	98.66%	سُ س سَ	11834	98.77%
پَ پُ پِ	1644	98.11%	شُ ش شَ	1721	98.91%
پُ پِ پَ	2593	98.19%	ص صَ صُ	574	97.22%
تَ تُ تِ	6451	98.26%	ضُ ض ضَ	323	98.23%
ٹُ ٹِ ٹَ	4641	98.82%	طُ طِ طَ	278	98.04%
ٹُ ٹِ ٹَ	1432	97.06%	ظُ ظِ ظَ	128	99.01%
ٹُ ٹِ ٹَ	986	98.87%	عُ عِ عَ	2141	96.89%
ثَ ثُ ثِ	717	96.69%	غَ غُ غِ	313	94.77%

ٻَ ٻُ ٻِ	4873	98.29%	فَ فُ فِ	1045	98.95%
جَ جُ جِ	14166	94.35%	قَ قُ قِ	212	95.88%
جهِ جهَ جهُ	3994	93.55%	قَ قُ قِ	383	98.33%
چَ چُ چِ	496	96.13%	كَ كُ كِ	17521	98.52%
ڄَ ڄُ ڄِ	281	97.41%	كَ كُ كِ	10571	98.89%
ڇَ ڇُ ڇِ	5832	98.61%	گَ گُ گِ	5447	98.22%
ڇهَ ڇهُ ڇهِ	3198	96.04%	گهَ گهُ گهِ	1193	95.82%
ڃَ ڃُ ڃِ	1959	98.39%	ڳَ ڳُ ڳِ	225	98.94%
ڃهَ ڃهُ ڃهِ	8736	98.07%	ڳَ ڳُ ڳِ	132	98.05%
دَ دُ دِ	8264	99.05%	لَ لُ لِ	14013	99.44%
ڈَ ڈُ ڈِ	493	96.96%	مَ مُ مِ	19925	98.36%
ڊَ ڊُ ڊِ	219	97.99%	نَ نُ نِ	27918	94.93%
ڈَ ڈُ ڈِ	15605	98.55%	ٽَ ٽُ ٽِ	581	94.91%
ڊَ ڊُ ڊِ	715	98.19%	وَ وُ وِ	18662	98.66%
ڈَ ڈُ ڈِ	313	94.63%	هَ هُ هِ ه	16568	97.33%
رَ رُ رِ	16382	98.53%	ءَ ءُ ء	4794	90.64%
ڑَ ڑُ ڑِ	1147	97.06%	يَ يُ يِ	31379	93.98%

The calculated accuracy with CMSL is 98.95% and 97.32% accuracy is achieved with CSJR. Experimental results show that the accuracy rate is directly related with the size of corpus. The DER of 1.04% is achieved with CMSL and DER of 2.68% is achieved with CSJR. The cumulative DER of 1.9% is calculated through this mechanism. **(Fig.3)** shows the diacritic error rates using both corpuses. The corpus of Sindhi language containing 2952327 words is selected, and when the size of training data increases, proportionally accuracy rate of learner results into a greater number.

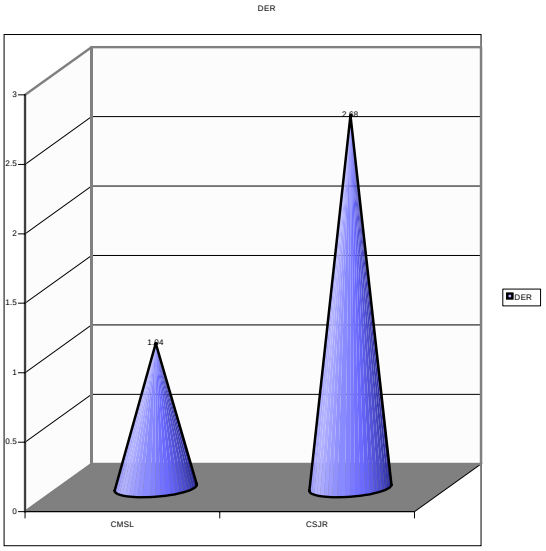

Fig.3. Calculated DER Using Corpuses CSJR and CMSL

In terms of window size, the greatest and most efficient accuracy was observed for ten accompanying letters that are nearest ones (i.e. N=5). Three different window sizes are examined for determining the size of context through which our problem would best be modeled. Among window sizes of two, six, and ten letters (i.e. N= 1, 3, 5), particularly ten first letters are taken into consideration for experiments. N=5 resulted the best outcomes among all others. Comparison of results with CMSL is presented in Table 6 and Table 7 shows the comparative results with CSJR.

Table 6: Obtained Results for Window Size of Two Six and Ten Letters with CMSL

Ambiguous Set	N=1	N=3	N=5
ا ِ ا	92.31%	94.86%	95.63%
بِ بَ بُ	95.22%	99.21%	99.59%
بَ بُ بِ	95.72%	98.31%	99.11%
بُ بِ بَ	96.69%	96.98%	98.88%
تِ تَ تُ	94.53%	96.61%	99.36%
تُ تِ تَ	91.28%	97.77%	99.55%
تُ تِ تَ	96.08%	98.63%	99.01%
تُ تِ تَ	97.39%	98.16%	99.87%
ٹِ ٹَ ٹُ	95.55%	96.83%	97.96%
پِ پَ پُ	95.36%	98.44%	99.19%

Table 7: Obtained Results for Window Size of Two Six and Ten Letters with CMSL

Ambiguous Set	N=1	N=3	N=5
ا ِ ا	93.55%	94.17%	95.21%
بُ بِ بَ	95.98%	96.33%	98.66%
بِ بِ بَ	93.83%	96.63%	98.11%
پُ پِ پَ	95.05%	97.19%	98.19%
تُ تِ تَ	95.73%	96.23%	98.26%
تُ تِ تَ	95.83%	97.06%	98.82%
تُ تِ تَ	94.08%	94.29%	97.06%
تُ تِ تَ	94.71%	96.49%	98.87%
ٹُ ٹِ ٹَ	91.11%	93.22%	96.69%
پُ پِ پَ	96.61%	97.95%	98.29%

4. CONCLUSION

The letter level learning method is instance based one in which *K*-NN algorithm is applied. A feature vector is the calculation and storage point of surrounding letters. Different window sizes are experimented for the computation of letters, i.e. N=2, N=3, N=5, N=6, among which N=5 is discovered the most efficient one. Proposed mechanism of diacritics restoration is trained on the corpus of diacritized text of CSJR and CMSL and then tested with non-diacritized text of the same corpus. Two diacritics restoration systems have been developed for Sindhi language that work at word level; one uses WordNet approach Mahar, and Memon, (2011) and the other does N-grams approach Mahar, and Memon, (2011). The DER of 9.3% is achieved with WordNet and N-grams approach shows the DER of 5.91%. This paper presents Letter Level Learning approach for Sindhi diacritics restoration and achieved DER is 1.9% which is the lowest of the previously developed systems.

REFERENCES:

Ali, A. R. (2009) "Automatic Urdu Diacritization", MS(CS) Thesis, Department of Computer Science, National University of Computer and Emerging Sciences and Center for Research in Urdu Language Processing, Lahore, Pakistan.

Alghamdi, M., Z. Muzaffar and H. Alhakami (2010) "Automatic Restoration of Arabic Diacritics: A Simple, Purely Statistical Approach", Arabian Journal for Science and Engineering, Vol. (**35**):137-155.

Ananthakrishnan, S., S. Bangalore, and S. Narayanan, (2005) "Automatic Diacritization of Arabic Transcripts for Automatic Speech Recognition", In Proceedings of the International Conference on Natural Language Processing, Kanpur, India.

Aadvani, K. (2009) "Shah Jo Risalo", 2nd Edition, Sindhica Academy, Karachi, Pakistan.

Elshafei, M. (1991) "Towards an Arabic Text-To-Speech System", The Arabian Journal for Science and Engineering, Vol. **(16)**: Number 4, 565-583.

Elshafei, M., H. Al-Muhtaseb, and M. Alghamdi, (2006) "Statistical Methods for Automatic Diacritization of Arabic Text", In the Proceedings of the 18th National Computer Conference, Riyadh. Vol. (**18**): 301-306. Conference.

Gal Y. (2002) "An HMM Approach to Vowel Restoration in Arabic and Hebrew", ACL-02 Workshop on Computational Approaches to Semitic Languages, Association for Computational Linguistic, Philadelphia, Pennsylvania, 1-7.

Harby, A. A., M. A. Shehawey, and R. S. Barogy, (2008) "A Statistical Approach for Quran Vowel Restoration", ICGST International Journal on Artificial Intelligence and Machine Learning, Vol. (**8**):No 3, 9-16.

Hullermeier, E., D. Dubois, and H. Prade, (2001) "Instance-Based Prediction in the Framework of Possibility Theory", (Available from: citeseerx.ist. psu.edu/viewdoc/ summary?doi =10.1.1.16. 4629Pp.

Kirchho, K., J. Bilmes, J. Henderson, R. Schwartz, M. Noamany, P. Schone, G. Ji, S. M. Das, F. Eganhe, D. Vergyri, D. Liu, and N. Duta, (2003) "Novel Approaches to Arabic Speech Recognition", Annual Report from 2002 the Johns hopkins, Summer Workshop. In the proceddings of the IEEE international conference on Acoustics, speech and signal procedding Hong Kong.

Kanis, J., and L. Muller, (2005) "Using Lemmatization Technique for Automatic Diacritic Restoration", in Proceedings of the Speech Communication, Moscow, 255-258.

Lee, C., and G. G. Lee, (2006) "Information Gain and Divergence-Based Feature Selection for Machine Learning-Based Text Categorization, An International Journal of Information Processing and Management-Special Issue: Formal Methods for Information Retrieval, Vol. (**42**): Issue 1, 155-165.

Mihalcea, R. F. (2002) "Diacritic Restoration: Learning from Letters Versus Learning from Words", Lecture Notes in Computer Science, Vol. (**2276**): 96-113.

Mihalcea, R. and C. Nastase, (2002) "Letter Level Learning for Language Independent Diacritics Restoration", Proceedings of 6th Workshop on Computational Language Learning, Vol. (**20**):1-7. Proceedings.

Mahar, J. A., and G. Q. Memon, (2011) "Lexicon Based Diacritics Restoration using WordNet for Sindhi, Internationa Journal of Academic Research, Vol. (**3**): Number 2, Part 1, 37-43.

Mahar, J. A., and G. Q. Memon, (2011) "Automatic Diacritics Restoration System for Sindhi", Sindh University Research Journal (Science Series), Vol. (**43**): Number 1, 43-50.

Nguyen, K. H., and C. Y. Ock, (2010) "Diacritic Restoration in Vietnamese: Letter Based vs. Syllable Based Model", Springer- Verlag Berlin Heidelberg, 631-636.

Nelken R, and S. M. Shieber (2005) "Arabic Diacritization using Weighted Finite-State Transducers", ACL Workshop on Computational Approaches to Semitic Languages, Association for Computational Linguistic, Ann Arbor, Michigan, 79-86.

Pauw, G. D., and P.W. Wagacha, and G.D. Schryver, (2007) "Automatic Diacritic Restoration for Resource-Scarce Languages", In Proceeding of TSD, Springer-Verlag Berlin Heidelberg, 70-179.

Roth, R., O. Rambow, and N. Habash, (2008) "Arabic Morphological Tagging, Diacritization and Lemmatization using Lexeme Models and Feature Ranking", In the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technology, Columbus, 117-120.

Shaalan, K., H. Abo Bakr and I. Ziedan, (2009) "A Hybrid Approach for Building Arabic Diacritizer", In Proceedings of the EACL Workshop on Computational Approaches Semitic Languages, Athens, Greece, 27-35. Proceedings.

Safadi, H., O. Dakkak, and N. Ghneim, (2006) "Computational Methods to Vocalize Arabic Texts", In Proceedings of the 2nd Workshop on Internationalizing Speech Synthesis Markup Language. Proceedings.

Schlippe, T., T. Nguyen, and S. Vogel, (2008) "Diacritization as a Machine Translation Problem and as a sequence Labeling Problem", The 8th Conference of the Association for Machine Translation in the Americas, Waikiki, Hawaii, 270-278.

Tufis, D. and A. Ceausu, (2007) "Diacritic Restoration in Romanian Texts", Recent Advances in Natural Language Processing Workshop: A Common Natural Language Processing Paradigm for Balkan Languages, Borovets, Bulgaria. Processing.

Tufis, D. and A. Ceausu, (2008) "DIAC+: A Professional Diacritic Recovering System", In Proceedings of Language Resources and Evaluation Conference, Marakkech, Morocco,167-174 Processing.

Wagacha, P., G. De Pauw, and P.Githinji, (2006) "A Grapheme-Based Approach for Accent Restoration in Gıkuyu", In Proceedings of the 5th International Conference on Language Resources and Evaluation, Genoa, Italy, ELRA, 1937–1940. Processing.

Zitouni, I., S. Jeffrey and R. Sarikaya, (2006) "Maximum Entropy Based Restoration of Arabic Diacritics", In Proceedings of 21st International Conference on Computational Linguistic and 44th Annual Meeting of the ACL, Sydney, Australia, 577-584. Conference.

Zitouni I., and R. Sarikaya, (2008) "Arabic Diacritic Restoration Based on Maximum Entropy Models", Computer Speech and Language, Vol. 23, 257-276.