IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 2, 2019, accepted January 25, 2019, date of publication February 6, 2019, date of current version April 5, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2897327

Part of Speech Tagging in Urdu: Comparison of Machine and Deep Learning Approaches

WAHAB KHAN[1], ALI DAUD[1], KHAIRULLAH KHAN[2], JAMAL ABDUL NASIR[1], MOHAMMED BASHERI[3], NAIF ALJOHANI[3], AND FAHD SALEH ALOTAIBI[3]

[1]Department of Computer Science and Software Engineering, International Islamic University Islamabad, Islamabad 44000, Pakistan
[2]Department of Computer Science, University of Science and Technology, Bannu 28100, Pakistan
[3]Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah 21589, Saudi Arabia

Corresponding author: Wahab Khan (wahab.phdcs72@iiu.edu.pk)

ABSTRACT In Urdu, part of speech (POS) tagging is a challenging task as it is both inflectionally and derivationally rich morphological language. Verbs are generally conceived a highly inflected object in Urdu comparatively to nouns. POS tagging is used as a preliminary linguistic text analysis in diverse natural language processing domains such as speech processing, information extraction, machine translation, and others. It is a task that first identifies appropriate syntactic categories for each word in running text and second assigns the predicted syntactic tag to all concerned words. The current work is the extension of our previous work. Previously, we presented conditional random field (CRF)-based POS tagger with both language dependent and independent feature set. However, in the current study, we offer: 1) the implementation of both machine and deep learning models for Urdu POS tagging task with well-balanced language-independent feature set and 2) to highlight diverse challenges which cause Urdu POS task a challenging one. In this research, we demonstrated the effectiveness of machine learning and deep learning models for Urdu POS task. Empirically, we have evaluated the performance of all models on two benchmark datasets. The core models evaluated in this study are CRF, support vector machine (SVM), two variants of the deep recurrent neural network (DRNN), and a variant of n-gram Markov model the bigram hidden Markov model (HMM). The two variants of DRRN models evaluated include forward long short-term memory (LSTM)-RNN and LSTM-RNN with CRF output.

INDEX TERMS Urdu, part of speech (POS), conditional random field (CRF), support vector machine (SVM), recurrent neural network (RNN), hidden Markov model (HMM).

I. INTRODUCTION

Part of speech (POS) tagging task is accomplished through the use of taggers, and taggers are composed of a numerous set of linguistic rules and the job of these taggers is to assign a corresponding syntactic tag to each and every word in a given text [2], [3]. Therefore, to step-up the accuracy of any natural language processing (NLP) system we are required to devolve on its ability that how efficiently and precisely it draws out associated information from a training data. The automated NLP systems which have the capability to produce more robust results in case of limited resources are considered as mature systems.

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

It is observed that the size of the training data and the quality of tagset used are the two major ingredients which greatly affect the performance of models which learns through automatic process such as machine learning and deep learning models. Therefore, we can say that POS tagging task not exclusively depends upon the dataset accustomed in training phase of the model, but in addition, the tagset used in the annotation is also equally important [3]–[5]. The two major components that are utmost necessary for the development a precise POS tagger are: a) a quantitative and a qualitative training data b) and an algorithm for appropriate POS tags prediction and its association with test data [3], [6]. Usually, POS tagging task is harder in languages having fewer linguistic resources compared to rich linguistic languages [3], [7].

NLP is one of the most authoritative domain of the selective information era. Interpreting knotty language vocalizations is

also a decisive portion of artificial intelligence. To analyze the mindset of a society, robust and accurate NLP systems are paramount necessary because people communicate most everything in language. There are a prominent diversity of subjacent tasks and machine learning models helping NLP applications.

Compared to ML approaches, Deep learning, in near past, over 5 years or so, has arousing intense research from a middling recession field of study made up of a conventual community of researchers to be so mainstream that its application can be found in the entire field of research where computational processing is involved. The fast emanation and superficial ascendance of deep learning approaches over conventional machine learning approaches on miscellany tasks have been astounding to see, and occasionally hard to explicate. Deep learning models showed record-setting performance in diverse areas such as NLP, image processing, language modeling, machine translation, computer vision, automatic speech recognition, audio recognition, Style imitation and bioinformatics [8]. The record-setting performance of DRNN in diverse areas motivated us that besides SVM we should measure our proposed work against DRRN.

In this current work, we accounted the exploitation of a machine and deep learning based system for Urdu POS. The major machine and deep learning models accounted in this study for Urdu POS includes: CRF, SVM , LSTM-RNN, LSTM-RNN with CRF output and HMM. Since CRFs in and of itself configured for problems implying sequence labeling because it mocks up the relationship between adjacent items in sequence. Therefore, its usage has been spotted in varied POS schemes and has demonstrated record-setting performance in diverse research areas including NLP.

Previously we have proposed CRF based Urdu POS tagging system which take advantage of both language dependent and language independent features however in this study, our objectives are concerned to (a) to analyze Urdu POS task and its corresponding challenges (b) design a smart and novel set of features solely based on context word window and (c) to explore an appropriate machine learning and deep learning based classifier that can contribute to improved results with a goal to promote other researchers to exercise it as a criterial testbed for experimentations in Urdu POS research. Speech recognition, information extraction,, text to speech etc. are some application areas of POS tagging task [9], [10].

The remaining paper is expended as: Urdu POS related work is keyed out in section 2, section 3 discovers Urdu part of speech challenges, section 4 briefly highlights CRF, SVM, RNN and HMM while section 5 is reserved to explain experimental evaluation, section 6 provides discussion and error analysis and finally, section 7 provides a conclusion.

II. RELATED WORK

The approaches adopted for Urdu POS task can be put into three categories namely rule-based, statistical and transformation-based learning.

In rule-based category the pioneer POS tagger for the Urdu language was formulated by Hardie [11]. He talked over numerous challenges of Urdu and formulated a tagset for Urdu employing the EAGLES guidelines for morpho-syntactic annotation of the dataset. The tagset proposed by Hardie [11], can be conceived as a rootage for the foundation of indispensable resource for Urdu POS tagging. The Hardie's proposed approach make use of the grammarian rules of Urdu in combination with the EAGLE rule of thumb for the task of Urdu POS annotation. The proposed POS tagger make use of about 270 rules and the reported accuracy was 90% [11].

One flaw of rule-based approaches is that they do not have the potency and manageability [12]. Also, to handle some new information in the same domain for which rules are constructed, rules need regular updates. Secondly, one is required to be fluent in the subject language and have expert skills in rules generation. Additionally, rules are of generic nature and cannot be applied to other language. Thirdly rules generation is practically more time-consuming.

In contrast of rule-based approaches, the present prevailing approaches used for handling POS issue in majority languages are established on supervised machine learning. The ongoing paramount method for covering POS task is supervised machine learning approach. The fundamental logic behind supervised models is that it automatically induces rules from prelabeled data, termed as training data. Adoption of supervised learning approaches for analysis of large dataset is initiating intense research interest [5]. Machine learning supported POS systems are desirable as such systems are flexible and updating of such system requires little time and little effort in case of accessibility of enough amount of training data [3].

For the first time, a statistical approach was proposed for Urdu POS task by Anwar *et al.* [10]. They achieved Urdu POS task with the use of tagger based on n-gram Markov model and consequently Backoff models, reported results were comparable with the existing ones. For experiment, they took EMILE corpus as training and testing. They performed experiments with two sorts of tagsets, the first tagset was consist of more than 250 tags while the second tagset contains 90 tags. The authors evaluated performance of their proposed tagger with both of the mentioned tagsets. After evaluation the small tagset and Backoff model recorded best accuracy figure of 95%.

In the research work of Sajjad and Schmid [13] presented four Urdu POS tagger and compared its performance with each other. The four taggers used are Tree Tagger, random forest (RF) tagger, TnT tagger and SVM tagger. They conducted experiments on a corpus of size about 110000 collected from cyberspace. The tagset used in their experiments contains total 42 syntactic categories. Each tagger acquired effective accuracy, however, SVM was superior among them with an accuracy of 95.66%.

Jawaid and Bojar [14] have exploited the use of SVM making use of voting scheme. Results of the experiments

TABLE 1. Summary of previous Urdu POS approaches.

Previous Approaches	Model	Dataset	Tagset	Results
Hardie [11]	Rule-Based 270 linguistic rules	EMILLE corpus	350 tags with 48 subcategories	90% accuracy
Anwar, et al. [10]	N-gram Markov model	(EMILE corpus) Training data of 1000 word	(a) 250 tags (b) 90 tags	95% accuracy
Sajjad and Schmid [13]	(a) Tree Tagger (b) RF tagger, (c) TnT tagger and (d) SVM model	Corpus of size about 110000 collected from cyberspace	42 tags	Best accuracy reported with SVM tagger is 95.66%
Jawaid and Ondřej [14]	SVM model	CRULP POS dataset	41 tags	87.98% accuracy
Jawaid, et al. [15]	SVM model	CRULP POS dataset	41 tags	88.74% accuracy
Naz, et al. [16]	Brill's Transformation-based learning (TBL) approach	Data of size 123755 words	36 tags	84% accuracy
Khan, et al. [1]	CRF, SVM	CLE and BJ Dataset	35 and 37 Tags	On CLE dataset 86.95% while on BJ dataset 93.56 %

proves that Urdu tagger performance gets to an adenoidal degree once the available linguistic resources for Urdu are aggregated. In their research work, the authors measured the performance of their voting scheme based SVM approach against morphological analyzer and with the existent parser of Urdu. The authors used the POS tagged data released by Research in Urdu Language Processing (CRULP) as training data for conducting experiments. Their proposed SVM tagger reaches the accuracy of 87.98%.

Jawaid *et al.* [15] extracted bulky sized Urdu digital text from various online sources and annotated it with the help of SVM.

After testing, the standalone POS tagger achieves 88.4% accuracy. Actually, the work of [15] is an extension of the work of Jawaid and Ondřej [14]. SVM Tool perform better than state-of-the-art taggers and Sajjad and Schmid confirmed this for Urdu.

Naz *et al.* [16] applied Brill's transformation-based learning (TBL) approach for Urdu POS task. When sufficient amount of train data available then Brill's TBL approach can automatically generate rules from the provided training data. The authors evaluated their proposed TBL based system on 123755 words training data which was annotated with tagset of size 36. Their TBL based system yielded accuracy of 84%. Table 1 presents summary of previous Urdu POS approaches.

III. THE URDU LANGUAGE

The key feature of languages scripted from right to left such as Urdu, also known as Arabic script-based languages [17], is the syntactically context-sensitiveness: the dependence of a letter's shape on its position in the text. This means that a letter can acquire a different shape depending on whether it begins or ends or appears in the middle of a word. Urdu is in addition, an inflectionally rich language [18]. Urdu morphology compounds numerous additional spoken languages along with its own [10]. Urdu Language Processing (ULP) requires extra attention from the research community, the main one being the scientifically interesting linguistic properties: rich morphology and free word-order (a concept can be represented by multiple word structures [3], [19]. Therefore, performing POS task in Urdu is highly challenging. In addition to the morphological richness property of Urdu, the main characteristics which make Urdu POS task more challenging and complex are [19]: lack of capitalization, free word order nature, loan vocabulary, affixes, and lack of standard linguistic resources.

The ever first POS tagset for the Urdu language is reported in [11] and is referred as Hardie tagset. Hardie provided 350 tags in his tagset. The tagset constructed by Hardie contains a larger number of tags and carrying various NLP tasks with this tagset poses a number of challenges.

Since developing efficient NLP resource with the help of Hardie tagset is very difficult due to its bulky size therefore,

TABLE 2. Sajjad Urdu POS tagset.

Main Tag	Sub Categories
Noun	Proper Noun (NP)
	Noun (NN)
Pronoun	Personal pronoun (PP)
	Reflexive(RP)
	Relative(REP)
	Adverbial(AP)
	KAF (KP)
	Adverbial KAF(AKP)
Demonstrative	KAF(KD)
	Adverbial(AD)
	Relative(RD)
	Personal(PD)
Auxiliaries	Aspectual auxiliaries(AA)
	Tense auxiliaries(TA)
Number	Cardinal(CC)
	Ordinal(OR)
	Fractional(FR)
	Multiplicative(MUL)
Conjunction	Coordinating(CC)
	Subordinating(SC)
Title	Pre-title(PRT)
	Post-title(POT)
Verb(VB)	---
SE (SE)	---
Adverb(ADV)	---
Genitive(G)	---
Genitive Reflexive(GR)	---
Phrase Marker(PM)	---
Adjectival Practical(A)	---
DATE (DATE)	---
Expression(EXP)	---
Quantifier(Q)	---
VALA (VALA)	---
Measuring Unit(MU)	---
Question Word(QW)	---
Negation(NEG)	---
Sentence Marker (SM)	---
Interjection(INT)	---
Intensifier(I)	---
KER(KER)	---

Sajjad and Schmid [13] designed a new Urdu POS tagset having 42 tags. The author created this new tagset in light of previous POS study and Urdu grammar rules. Though this new tagset provides very clear subclasses for pronouns and demonstrative however some part of speech such as verbs and tenses are inadequately defined. A detailed description of Sajjad POS tagset is provided in Table 2. In 2008 center for research on Urdu language processing proposed another tagset containing 46 POS tags. Compared to the previous tagset, in this tagset verbs and nouns are very fairly classified.

Muaz *et al.* [20] designed a new tagset containing 32 POS tags. Authors deeply analyzed previous POS tagset and compared each tagset with other to find differences and variation. After careful analysis of the variations found in previous tagsets, the authors designed this new tagset which is syntactically and computationally coherent tagset.

TABLE 3. Detailed description of CLE POS tagset.

Main Tags	Sub Categories
Noun	Proper Noun (NNP)
	Common Noun (NN)
Pronoun	Personal pronoun (PRP)
	Demonstrative(PDM)
	Possessive pronouns (PRS)
	Reflexive pronouns (PRF)
	Reflexive Apna (APNA)
	Relative Personal (PRR)
	Relative Demonstrative (PRD)
Verb	Main Verb Infinitive (VBI)
	Main Verb Finite (VBF)
Auxiliary	Aspectual auxiliaries (AUXA)
	Progressive auxiliaries (AUXP)
	Tense auxiliaries(AUXT)
	Modals auxiliaries (AUXM)
Residual	Foreign Fragment (FF)
Interjection	Interjection (INJ)
Adposition	Preposition (PRE)
	Postposition (PSP)
Symbols	Common (SYM)
	Punctuation (PU)
Adverb	Common (RB)
	Negation (NEG)
Particle	Common (PRT)
	Vala (VALA)
Conjunction	Coordinate Conjunction (CC)
	Subordinate Conjunction (SC)
	SC Kar (SCK)
	Pre-sentential (SCP)
Nominal Modifiers	Ordinal (OD)
	Fraction (FR)
	Multiplicative (QM)
	Adjective (JJ)
	Quantifier (Q)
	Cardinal (CD)

Recently, Tafseer *et al.* [21] released a CLE(Centre of Language Engineering) tagset for Urdu, containing 12 major categories with 35 subcategories. In CLE tagset Urdu morpho-syntactic categories of various part of speech are designed very carefully. The Penn Treebank and various Indian language tagset are the major resources which are used as guideline by the authors to create CLE tagset for Urdu. A detailed description of CLE POS tagset is provided in Table 3.

A. URDU PART OF SPEECH CHALLENGES

From earlier works it is discovered that several Urdu grammarians like Platts [22] Haq [23] sorts Urdu words into three basic POS categories. The three basic categories they defines

Sentence-I	ہے	ارادہ	کا	سونے	جلد	آج	میرا
	hey	irada	ka	soney	jald	aaj	mera
	VBF	NN	PSP	VBI	RB	NN	PRS
	Today I want to sleep early						
Sentence-II	ہے	ریٹ	کیا	تولہ	فی	کا	سونے	آج
	hey	rate	kia	tola	fi	ka	soney	aaj
	VBF	NN	RB	NN	PRE	PSP	NN	NN
	What is the rate per ton of gold?							

are (a) اسم (ism, noun) (b) فعل (fil, verb) (c) حرف (harf, particles) [13], [22]. On the other hand, Schmidt [24] is of the view that Urdu words acquires ten part of speech instead of three.

In morphological rich languages like English and Urdu, the task of POS tagging is challenging one. There can be number of reasons but the characteristic, that in morphologically rich languages, normally a single word can be used in more than one forms and can acquire multiple tags in different contexts is the prime one. Urdu, in addition, is both inflectionally and derivationally rich morphological language [25]. Verbs are generally conceived a highly inflected object in Urdu comparatively to nouns [25]. The major factors to which Urdu verb express inflection agreement are: case, number, gender and respect. In Urdu, it is usual that a single root verb might experience as many as twenty-five different inflected variations. So far, the main factors are concerned for which Urdu nouns express inflected agreement are: number, gender and case. Furthermore, the nouns express derivational alterations into adjectives [25].

Similarly, the most mature systems, such as for POS in English, rely heavily for accuracy on extrinsic linguistic resources such as annotated corpora, human-made dictionaries and gazetteers [26], [27]. The ULP community not only lacks these resources but also is dealing with a language without capitalization, a powerful clue for proper nouns identification, unlike NLP in a European language. Similarly, unlike English, Urdu does not support small and capital words and exhibits the free-word order behavior. POS in Urdu is, therefore, a difficult task, demanding a greater sophistication in linguistic analysis and the development of techniques for effective task performance.

The most predominant errors with machine-controlled POS systems is to precisely predict nouns, in noun-based phrases from other POS categories such as pronouns, proper nouns adverbs, and adjectives etc. In Urdu, proper nouns and common nouns are different from each other through a number of features, but it is still challenging to classify it from each other. Therefore, one needs to rely on contextual information of word in a phrase to perform NLP in Urdu. Moreover, in Arabic script based languages like Urdu, taking out useful information from context word is harder [16]. Below given sentences explains this situation.

In the above example, in both sentences, the word سونے (soney) appeared with unique meaning and tag. In sentence-I it is used in sense of sleeping e.g. سونے (soney, Sleep) and attain the VBI (main verb infinitive) tag while in sentence-II it is used in sense of gold e.g. سونے (soney, Gold) and attains the common noun (NN) tag. Below we provide some common Urdu POS challenges which we have already identified in our previous work.

TABLE 4. Pronouns used as adjective example.

Sentence No.1	دروازے پر کون ہے ؟
	Darvaze par kaun hai?
	Who is at the door?
Sentence No.2	دعوت میں کون کون آیا؟
	Davat mein kaun kaun aya?
	Which various people came to the party?

TABLE 5. Example of nouns used in place of adjectives.

Sentence No.1	ضرورتمند لوگوں کو خوراک دو۔
	Zaroratmand logun ku khurak do
	Give food to needy people
Sentence No.2	ضرورتمند کو خوراک دو۔
	Zaroratmand ku khurak do
	Give food to needy

1) PRONOUNS USED AS ADJECTIVES

The interrogative pronouns کون (kaun, who) and کیا (kya, what) are also used as adjectives. کون (kaun) means "Who?". It is occasionally also used as an adjective, qualifying a noun.

Kya means "what" referring to things. It is also used as an adjective qualifying a noun, especially before oblique case nouns where it means "which". Consider the example given in Table 4. In sentence 1 the word کون (kaun) is used as a pronoun while in sentence 2 the same word کون (kaun) is used as an adjective.

2) NOUNS USED AS ADJECTIVES

In Urdu grammar, it is usual that adjectives perform like a noun in noun based phrases. When nouns are took away, the antecedent adjectives act likewise a noun. Look into sentence 1 provided in Table 5 where adjective (ضرورت مند) and noun (لوگوں) occurs sequentially.

Similarly, In sentence 2 of the same table when the main noun (لوگوں) is moved out, now the adjective word (ضرورت مند) acquits likewise a noun rather than an adjective. The example given in Table 5 explains both cases.

TABLE 6. Example of usage of Adjective as an adverb.

Sentence No.1	وہ بڑا ذہین آدمی ہے۔
	Vo bara zahin admi hai
	He is very intelligent man
Sentence No.2	وہ بڑا آدمی ہے۔
	Vo bara admi hai
	He is big man

TABLE 7. Usage of Adjectivse in place of noun example.

Sentence No.1	احمد امیر ہے
	Ahmad ameer hay.
	Ahmad is rich
Sentence No.2	اسلام میں امیر کو غریب پر کوئی فوقیت حاصل نہیں ہے۔
	Islam main ameer ku gharib par koi foqiat hasal nehi hay
	In Islam, rich has no superiority over the poor

3) ADJECTIVES USED AS ADVERBS

A couple of adjectives may be practiced adverbially to change some other adjectives. The adjective بڑا (bara, big) can be used adverbially as an intensifier. Consider the example given in Table 6. In sentence 1 بڑا (bara, very) the word بڑا (bara, very) is employed as an adjectively while in second sentence the same word بڑا (bara, big) is employed as an adverb.

4) ADJECTIVES USED AS NOUNS

In Urdu it is also usual that galore number of adjectives likewise be employed in place of nouns. Consider the example given in Table 7. The word امیر (ameer, rich) of first sentence is employes as an adjective while in second sentence it is used as a noun. Also, some loan word such as نوجوان (young man) and غیر ملکی (foreigners) are classified as nouns and adjective.

IV. METHODLOGY

In the Incoming section we talk about the assorted models tested in this study. The models tested includes: CRF, SVM, LSTM-RNN, LSTM with CRF output and finally the N-gram language model.

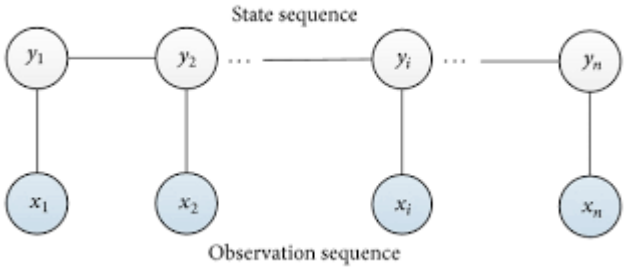

FIGURE 1. Graphical representation of linear chain CRF.

A. CONDITIONAL RANDOM FIELDS

CRFs [28] are common sequential models, widely used for segmentation and labelling tasks in NLP. In training phase, it can handle efficiently bulky size features independently [29].

CRFs are probabilistic models for tagging and segmenting sequential data. They support a conditional approach and structurally follow characteristics of an undirected graphical model, where the nodes comprise the tag sequence 'y' related to observing sequence 'x' [30]. For POS problem, token of a sentence represents observation sequence while POS tags corresponds to label sequence [31] (Figure 1).

In mathematical notation format, the conditional probability for each input sequence $X = (x_1, x_2, x_3 \ldots\ldots x_n)$ and tag sequence $Y = (y_1, y_2, xy_3 \ldots\ldots y_n)$ of a CRF model is given below:

$$P(y|x) = \frac{1}{Zx} exp\left(\sum_{e\epsilon E}\sum_{i}\lambda_i t_i\left(e, y|_e, x\right) + \sum_{v\epsilon V}\sum_{k}\mu_k s_k\left(v, y|_v, x\right)\right) \quad (1)$$

Z(x) denotes normalization agent and can denoted in below mathematical form:

$$Z(x) = \sum_{y}\left(exp\left(\sum_{e\epsilon E}\sum_{i}\lambda_i t_i\left(e, y|_e, x\right) + \sum_{v\epsilon V}\sum_{k}\mu_k s_k\left(v, y|_v, x\right)\right)\right) \quad (2)$$

where $y|_e$ denotes the edges of undirected graph while $y|_v$ represents vertices, t_i is the transition feature function of edges, s_k is a state feature function of nodes while λ_i and μ_k represent the weights assigned to the different feature or transfer characteristic functions of edges and nodes in the training phase.

The feature functions of edges define feature function between two back-to-back nodes in CRFs, therefore equation 1 can be rewritten as below:

$$P(y|x) = \frac{1}{Zx} exp\left(\sum_{i=1}^{n}\left(\sum_{j}\lambda_j t_j\left(e, y_{i-1}, x\right) + \sum_{k}\mu_k s_k\left(v, y_i, x\right)\right)\right) \quad (3)$$

TABLE 8. Feature function structure.

func1 = if (output class = PNN and feature e.g. The left context word of current word = "پڑھنے") return 1 else return 0
func2 = if (output class = NN and feature e.g. The left context word of current word = "پڑھنے") return 1 else return 0
....
....
....
funcN = if (output = QM and feature e.g. The left context word of current word = "پڑھنے") return 1 else return 0

where 'n' denotes the number of counts of each input sequence $X = (x_1, x_2, x_3 \ldots\ldots x_n)$ while y_i is the resultant output tag of the ith word of the sequence X. We can simplify the notations of state feature function by writing $s_k\,(y_i, x, i) = s_k\,(y_{i-1}, x, i)$ and it can be either transition feature function $t_j\,(e, y_{i-1}, x)$ or state feature function $s_k\,(v, y_i, x)$ and uniformly the two feature function can be represented as $f_k\,(y_{i-1}, y_i, x, i)$. Thus, the conditional probability of the corresponding observation sequence can be written as below:

$$p\,(y|x) = \frac{1}{Zx} exp\left(\sum_k \lambda_k f_k\,(y_{i-1}, y_i, x)\right) \tag{4}$$

The parameters (λk) in CRF can be estimated using iterative scaling algorithm, gradient based methods or L-BFGS methods [32], however in this study the libraries used for CRF are based on the L-BFGS method for parameter estimation.

1) CRF FEATURE FUNCTION

Feature functions are core supplements of the CRF training phase and are generated according to the mentioned features. Final features are synthesized by using feature function by going through the entire training and testing data. CRFs can use both real valued as well as binary valued features, however, in our experimentation all features are binary valued.

$$f_k\,(y_{i-1}, y_i, x, i) = \begin{cases} 1, & if\ y_{i-1} = ADJECTIVE\ and\ y_i = NOUN \\ 0, & Otherwise \end{cases}$$

The number of final features generated by feature function during encoding process ranges from thousands to several hundred thousand or even in millions, mainly depending on the (a) size of training data, (b) number of output classes and (c) the number of distinct strings expended from a given template. In this study, the number of context word window feature used are 7 while the number of output classes are 35 in case of CLE POS tagset. Thus, in case of unigram template, the total quantity of feature functions yielded after executing all the features mentioned in feature template for a single token will be ((1 × 7) x35 = 245). Similarly, for a record having total of 12 tokens, the number of features generated will be ((12 × 7) x35 = 2,940)). In our experiments, when a CRF evaluates any feature E.g. ''The left context word of current word'', for any token e.g. the token انگریزی (Angrazee, English) then the set of feature functions generated will be like those shown in Table 8:

B. SUPPORT VECTOR MACHINE (SVM)

Support vector machines (SVMs) [33], are supervised machine learning algorithm, designed to handle binary classification problems [34]. It utilizes a nonlinear mapping to convert the definitive preparing information into a higher size. Inside this new measurement, it hunts down the direct optimal dividing hyperplane. The hyperplane can be used to separate the data of two classes. In computational NLP, SVMs are put on to a bit of NLP tasks e.g. POS, NER, segmentation, content arrangement and so forth, and are accounted to have attained high exactness without falling under overfitting [35]. So far, the performance of SVM is concerned, it produces highly accurate results, but the training time is extremely slow.

Given the training data $\{(x_i, y_i)\}_{i=1}^n$ where $x_i \in R^D$ and represents the ith input vector and $y_i \in \{1, -1\}$ and represent the corresponding class tag related to x_i and n represents the number of inputs.

The primal concept obligatory for specifying a linear classifier is the scalar product between vector 'w' and vector 'x' symbolized as $\langle w, x\rangle = \sum_{j=1}^M w_j x_j$. A linear classifier is established on linear discriminant function of the below form:

$$f\,(x) = \langle w, x\rangle + b \tag{5}$$

where $f\,(x)$ is termed as discriminant function and its job is to (a) assign score to input x_i (b) to determine how to sort out it, w represents the weight vector while b is called the bias, a scalar quantity. In two dimensions, the classification rule for separating hyperplane can be written as:

$$f\,(x, w, b) = W.X + b = 0 \tag{6}$$

The hyperplane of SVM splits up the space into cardinal half spaces as per the sign of $f\,(x)$, that shows on which side of the hyperplane a point came in.

$$f\,(x, w, b) = sign(W.X + b) \tag{7}$$

So, any point that is located above the hyperplane meets the below presented stipulate.

$$W.X + b > 0 \tag{8}$$

Likewise above, any point that came under the hyperplane must satisfy the following condition:

$$W.X + b < 0 \tag{9}$$

In case of non-linear separable data, the non-linear SVM classifier define the discriminant or the decision

function for an input vector as below:

$$f(x) = sign(g(x)) \tag{10}$$

where $g(x)$ is given below:

$$g(x) = \sum_{i=1}^{m} w_i K(x, z_i) + b \tag{11}$$

where in case of $f(x) > 0$ indicates that x belongs to certain class while in case $f(x) < 0$ means x does not belong to any class. z_i represents support vectors while m is the number of support vectors and $K(x, z_i)$ represent the kernel responsible for mapping the input to higher dimensional space and can be expressed as dot product of two vectors such as $K(x, z_i) = K(x.z_i)$. We can exercise assorted kernels however the design and configuration of a proper kernel for a peculiar task is a trivial task. In below subsection we highlight some popular SVM kernel functions.

1) SVM KERNEL FUNCTION

The SVM can be conceived as a kernel machine, allowing users to commute its conduct by practicing an assorted kernel function. So far kernel function is related to, the kernel function is its distinctive component, responsible to map the input to higher dimensional space. The four common SVM kernel functions are listed below [36]:

a: THE LINEAR KERNEL

Almost all text classification problems are often linear separable, therefore for text classification tasks linear kernel is commonly recommended [37], [38]. However, in a few cases, the adoption of another kind of kernel could be more beneficial. Below are some fundamental advantages of linear kernel due to which researchers prefers its practice over other kernels in assorted tasks [39]: Since text holds many features and in case of the existence of galore features linear kernel performs well. Liner kernel provides faster training of SVM than other kinds of kernel. To train SVM with linear kernel it requires fewer parameters to optimize. Keeping in mind the advantages of the linear kernel, in this study the training of SVM is also achieved with help of linear kernel. Mathematically linear kernel can be expressed as below:

$$K(x, y) = x^T y \tag{12}$$

b: THE RADIAL-BASED FUNCTION (RBF) KERNEL

The radius-based function kernel also termed as Gaussian radial-based function kernel. In case of non-linear classification, it is a widely and most frequently adopted kernel function in SVM, mainly due to its fascinating features. The two parameters of RBF kernel are: C and Gamma (γ) where C is regularization parameter while γ represents the kernel coefficient. Both parameters are integral part to sustain high-pitched performance of the RBF SVM [40]. Mathematically RBF kernel can be expressed as below:

$$K(x, y) = \exp(-\gamma \parallel \mathrm{x} - \mathrm{y} \parallel^2) \tag{13}$$

c: THE POLYNOMIAL KERNEL

this kernel function essentially implies that the classifier considers not only the explicitly specified features, but also all available sets of size d of features. Mathematically polynomial kernel can be expressed as below:

$$K(x, y) = (\gamma x^T y + c)^d, \quad \gamma > 0 \tag{14}$$

where d denotes degree of polynomial and is kernel parameter while c is a constant term. All the three parameters e.g. γ, c and d are adjustable.

d: THE SIGMOID KERNEL

sigmoid kernel is evolved from artificial neural network family [41]. The SVM model employing a sigmoid kernel function is like to a two-layer neural net, this attracted researchers to practices it widely with SVM model for assorted tasks. Also, despite being exclusive conditionally positive definite, it has been ascertained to execute advantageously in practice. Mathematically it can be represented as below:

$$K(x, y) = \tanh\left(\gamma a x^T y + r\right), \quad \gamma > 0 \tag{15}$$

where γ and r represents two parameters of sigmoid kernel where γ can be viewed as scaling parameter of input data while r is constant term, and this parameter is responsible for controlling threshold values for mapping and is termed as shifting parameter [41].

2) SVM HYPERPARAMETERS

Picking out optimum hyperparameter measures for support vector machines is an authoritative and essential step in SVM designing phase for its successful application. The most common hyperparameters in the subject of SVM are: kernel, C and γ, where C symbolises regularization parameter which ascertains the trade-off between minimizing the training error and minimizing model complexity and Gamma γ represents the kernel coefficient in case of RBF, polynomial and sigmoid kernels. For all feature subset, the hyperparameters were adjusted to prevail the most estimable performance metrics practicing some search algorithm. Among others some optimization methods that commonly used for SVM hyperparameter optimization are random search, grid search, gradient descent algorithm and estimation of distribution algorithms [42]. The grid search algorithm goes through a 5-fold cross validation mechanism.

C. RECURRENT NEURAL NETWORK (RNN)

Recurrent neural networks (RNNs) [43] belong to the artificial neural networks family [44]. The naivest conceivable variant of the recurrent neural network is the Elman architecture [45] which is also named in literature as a conventional or simple recurrent neural network [46], [47]. Traditional neural networks(non-deep or General forward neural networks) [48] such as Multi-layer perceptron, feed-forward neural networks or autoencoders and convolutional neural networks (CNN) use up a readied sized input and yield

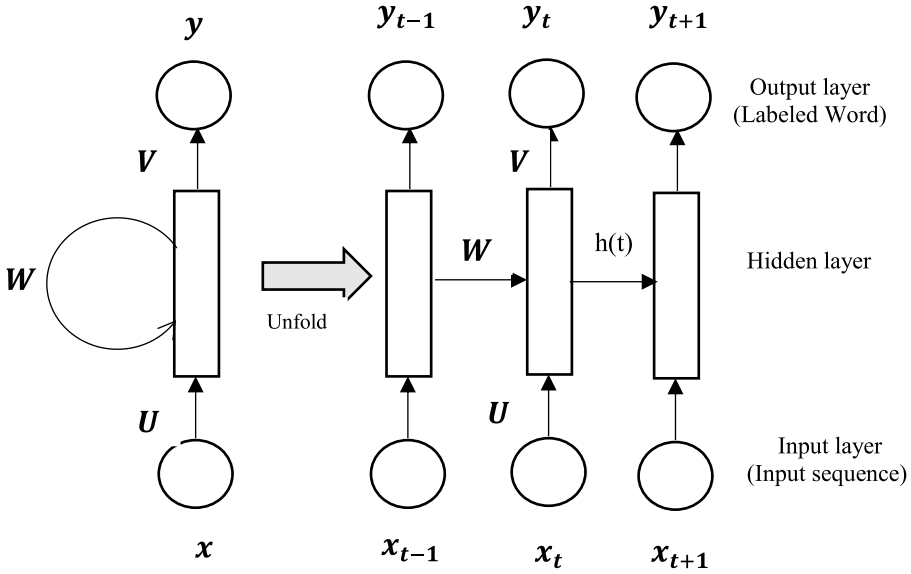

FIGURE 2. Folded and unfolded regular recurrent neural networks with single hidden recurrent layer.

fixed-size outputs. RNN, a type of forward neural network, but then, can manage input/output of arbitrary lengths. Traditional networks such as CNN are considered ideal for two-dimensional data processing such as images and videos [49] while RNNs are ideal for multi-dimensional or variable-length sequential data processing such as text and speech analysis [44], [50]. Similarly, in traditional neural networks, input and output are independent of each other while in RNN the outputs are largely reliant on the previous calculation. The internal structure (hidden layers) of an RNN contains recursive edges (feed-back connections) between units as compared to its counterparts, the traditional neural nets.

The hidden layers of RNNs which contain recursive edges are termed as memory units of the network as its calculation is based on the output of the previously hidden state, along with the input of the current input state. The hidden layers of RNNs are the core element, which captures information about input sequence. Therefore, with this information persistence capability of hidden layers neurons, as the time steps increase, the units get influenced by larger and larger neighborhood or in simple words the network acquires long-term dependencies, enabling RNNs to demonstrate propelling historical performance. Traditional neural networks and shallow structure classifiers such as CRFs, hidden Markov models (HMMs), maximum entropy (MaxEnt) and support vector machines (SVMs) do not support long-term dependencies, preventing them from developing information persistence [44], [51], [52].

Figure 2 demonstrates a conventional forward RNN being extended (or unfolded/unrolled) into a complete network. From unfolding/unrolling we plainly intend that we make out the network for the whole sequence. A conventional RNN has three layers: the input layer corresponds to the input to the network; the middle or hidden layer contains recursive edges that correspond to information persistence, and the output one represents the predicted output. In conventional RNNs, there are one-to-one relationships between the extended layers and input sequence. If the input sequence consists of three words, then the network will be extended to a three-layer neural network.

In figure above, U, W, and V correspond to the parameters for input, hidden and output layers, respectively, as calculated during the training phase, and are used to connect input, hidden and output layers, where x(t) is the input word with a dimension usually equal to the vocabulary size, and y(t) represents the output at output layer. The output layer has a dimension equivalent to the number of possible output classes [53]. These show the probability distribution over named entity classes in case of a NER task, while h(t) denotes the hidden layer and is used to keep historical information corresponding to the particular input vector.

The equations that regulate the calculation taking place in an RNN are given below:

$$h_t = \sigma\left(U.x(t) + W.h_{t-1}\right), \quad t = 1 \ldots .T \tag{16}$$

$$y\left(t\right) = g(Vh\left(t\right)) \tag{17}$$

x(t) represents any particular input at any time stamp (t) to the input layer, and corresponds to one vector.

h(t) is considered the memory unit of the network and corresponds to the hidden layer at any time stamp (t). Its calculation is based on the output of previous s s hidden state, along with the input of current input state;

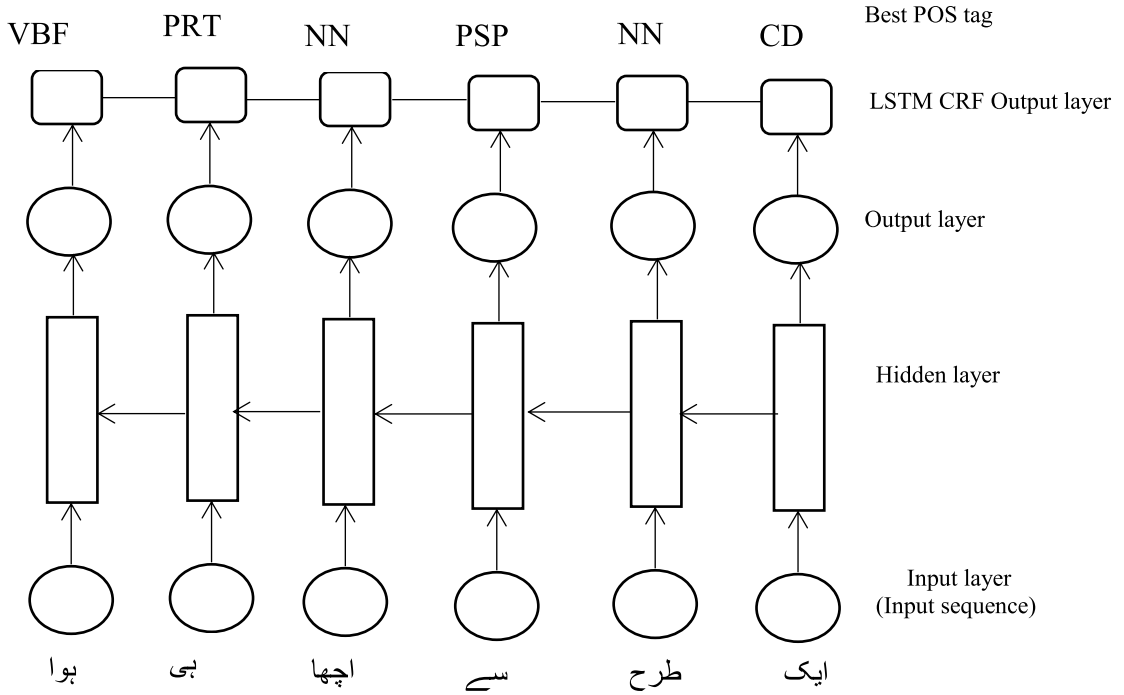

FIGURE 3. Illustration of Forward LSTM-RNN with CRF output.

g(Vh(t)) calculates y(t) by using the output of the hidden layers and weights between the hidden layer and output layer y. y(t) corresponds to output at time stamp (t);

$\sigma(Tanh)$ represents function, responsible for initializing the first hidden state.

Recurrent neural networks have become popular in NLP tasks such as language modeling and word-labeling. In word-labeling, RNNs — as do CRFs — assign tags to a sequence of words given as the input [54], [55]. RNNs have also recently been shown to be effective in a broad range of sequential labelling problems such as language modeling [56], [57], question answering [58], [59], speech recognition [60], translation [61], image captioning [62], NER [63] and text generation [57].

1) LSTM NETWORK

Traditional RNNs are inclined to vanishing gradient problem once trained on back-propagation through time (BPTT) initially proposed by [64], to accost the vanishing gradient problem a high-toned architecture of RNN termed as long short-term memory (LSTM) has been configured in 1997 by [65].

These days, LSTMs are a commonly used type of RNN and have proved to be more effective at arriving at long-run dependencies than traditional RNNs [66], [67]. Unlike to traditional RNNs, LSTM-RNNs employ a different function in hidden state calculations. The memory units in LSTMs are referred to as cells or black boxes, and their parameters comprise the outputs of the N − 1 layer and the current layer input. These cells or black boxes are responsible for deciding what to keep and what to erase. They then combine the previous layer, the current memory and the input [50], [66]–[68].

2) LSTM-CRF

A Simple or Forward LSTM network with CRF output is shown in Figure 3. In simple LSTM-CRF network the decision about the final tag is governed by two-layer e. g the LSTM layer and the CRF layer. The job of LSTM layer is to utilize past input feature while as the job of the CRF layer is to utilize the sentence level tag information. In below figure CRF layer is interpreted with help of undirected lines which link up back-to-back output layers. The input to the CRF layer is the state transition matrix and transition matrix represents past and future tags. The transition matrix enables CRF layer to predict the current tag by utilizing both the past and future tags [69]. We represent the transition matrix with $A_{i,j}$, holding the transition scores from i^{th} tag to j^{th} tag for a pair of successive time steps. Let $X = (x_1, x_2, \ldots, x_n)$ represents a sentence and we denote it with $[x]_1^T$ while $N([S]_1^T)_{i,t}$ represents transition matrix holding the transition scores yielded by the simple LSTM network for the given sentence $[x]_1^T$ along with sequence of tags. The score of a sentence along with a sequence of tags $[i]_1^T$ is then generated by the add up of transition scores and scores from LSTM network:

$$S\left([x]_1^T, [i]_1^T\right) = \sum_{t=1}^{T} ([A]_{[i]_{t-1},[i]_t} + N\left([S]_1^T\right)_{i,t} \quad (18)$$

These days dynamic programming techniques are practiced to the highest degree to compute the $A_{i,j}$ values and optimal

tag sequence for inference [28]. To generate probabilities of tag sequence $[i]_1^T$ softmax function is used and is given below:

$$p\left(y|\,[x]_1^T\right) = \frac{e^{s([x]_1^T,[i]_1^T)}}{\sum_{\tilde{c}\in I_s} e^{s([x]_1^T,\tilde{c})}} \tag{19}$$

where I_s corresponds to all potential tag sequence for a specified sentence $[x]_1^T$.

D. N-GRAM-BASED MODEL/HMM

The N-gram based models or Markov models are the class of probabilistic models that assume that using the history of previous N − 1 words in a sequence, we can predict the next word. The most common variants of N-gram models are the unigram, bigram, and trigram. For example, the probabilities of the bigram model are obtained using the previous (N − 1) word only while in the trigram model the probabilities of N − 2 are involved.

The general equations that regulate the calculation taking place in N-gram model to calculate the probability of an entire sequence using the bigram assumption is given below:

$$P\left(w_1^n\right) \approx \prod_{k=1}^{n} P\left(w_k|w_{k-1}\right) \tag{20}$$

In this study, we constructed a bigram model and used the bigram probabilities in building the hidden Markov model (HMM). Hidden Markov model is also termed a sequence classifier or sequence labeler. The basic function of a sequence classifier is: for given input sequence first to identify a class label for each token and then to assign the corresponding label to each token or word of the sequence. HMMs models have been applied to a wide variety of problems in text and speech processing, including topic segmentation, POS tagging, information extraction, and syntactic disambiguation [5].

The components of HMM [70] are (a) the initial probabilities distribution of hidden states represented with π, (b) a matrix of transition probabilities of states represented with A and (d) a matrix of emission probabilities or observation probabilities represented with B. The following equations formalize a general HMM model parameters.

$$M = (\pi, A, B) \tag{21}$$

where 'A' represents matrix of transition probabilities and the transition probability a_{ij} from a state a_i to a state a_j is the probability of the word with tag a_j following a word with a tag a_i.

$$A = \left(a_{ij}\right) \tag{22}$$

$$a_{ij} = \frac{|a_i a_j|}{|a_i|} \tag{23}$$

Similarly, 'B' represents a matrix of observation/emission probabilities which is the probability of a given word being generated given a particular tag and is formalized as below:

$$B = \left(b_i\left(o_t\right)\right) \tag{24}$$

$$b_i\left(o_t\right) = \frac{|o, a_i|}{|a_i|} \tag{25}$$

In this study, the transition probabilities are calculated from states contextual bigram model while the maximum likelihood estimation or the emission probabilities are calculated from the observation sequence bigram model. During the emission probabilities estimation of the observation sequence, the problem of probabilities estimation of unknown words in training data is handled with Laplace smoothing technique. After creation of A and B matrices, the Viterbi algorithm is used to find the optimal path in the search space.

E. FEATURES

In machine learning tasks, the performance of almost all models depends upon feature set. Therefore, in that respect, selection of most relevant feature set is an essential requirement of all machine learning models. The final results are greatly dependent on most relevant features [32]. Among others some commonly and widely adopted features reported in literature are: POS information, word affixes information, digit feature, surrounding words, length of word and named entity information etc. However, in this study we only considered the current word and its context word window as features.

As already stated, that previously we have proposed a CRF based POS system for Urdu which make use of both language dependent and language independent feature set, however in this study our one objective is to propose a faster machine learning based Urdu POS system with smart and novel context word window features instead of complex and large feature set that can improve upon the existing POS systems.

Context word engineering for appropriate tag selection is very critical. Below are the context word features which we defined for our CRF and DRNN models:

- Token: The current word itself
- Context words window: the various context words window features used in this study are listed below:
  - The word to the left of the current word
  - The word to the right of the current word
  - Joint use of Current word and the word to the left of the current word
  - Joint use of Current word and the word to the right of the current word
  - Joint use of Current word and N − 1, N − 2 left words of the current word
  - Joint use of Current word and N + 1, N + 2 right words of the current word

V. EXPERIMENT AND EVALUATION

We conducted 10-fold cross validation experiment for all models used in this study to symbolize the effectiveness of

TABLE 9. Consolidated statistics of CLE POS tagged dataset.

Tag	Occurrence Frequency	Tag	Occurrence Frequency	Tag	Occurrence Frequency	Tag	Occurrence Frequency
APNA	581	INJ	128	PRF	91	RB	1429
AUXA	3015	JJ	8090	PRP	4448	SC	1877
AUXM	499	NEG	966	PRR	711	SCK	674
AUXP	503	NN	29183	PRS	501	SCP	302
AUXT	3480	NNP	4735	PRT	1492	SYM	274
CC	3225	OD	502	PSP	17257	VALA	295
CD	2295	PDM	2122	PU	10369	VBF	11026
FF	234	PRD	139	Q	1381	VBI	1667
FR	69	PRE	13	QM	10	---	---

machine learning and deep learning approaches for Urdu POS task along with context word window features and word embedding as a feature for deep learning models.

A. DATASET

In the current study, the effectivity of both machine learning and deep learning models for POS in Urdu is showed on two gold standard dataset namely (a) the CLE dataset and (b)Bushra Jawaid dataset [15]. We refer Bushra Jawaid dataset in the remaining text as BJ dataset.

1) CLE DATASET

Center for Language Engineering (CLE) has released a very copious POS tagged dataset to promote research and computational activities in Urdu. The released dataset is termed as Urdu Digest POS Tagged, but herein we refer this dataset as CLE dataset. This dataset is available in two versions e.g. size of the first version is 100K words while the size of the second version is 1M words. In both datasets, all words are manually annotated with its corresponding POS tags. The contents of this dataset contains text from assorted news genre. Contents of this dataset are organized in two broad categories the information and imaginative.

Primarily tags in CLE Urdu POS tagset are put in twelve core categories with subdivisions, resulting in 35 unique POS tags [21]. The CLE tagset has been used to tag 100k words of the CLE Urdu Digest Corpus [21]. The annotation of CLE Urdu Digest dataset was achieved with the help of CLE POS tagset. The detailed description of CLE POS tagset can be seen in Table 3. Table 9 shows consolidated statistics of various POS tags extracted from the available version of CLE POS tagged dataset.

2) BJ DATASET

In 2014 Jawaid *et al.* [15] introduced the first bulky sized Urdu POS tagged dataset that is freely available.[1] The content of dataset is crawled from various Urdu website e.g. BBC Urdu, Urdu Planet and several other sites and automatically annotated with POS tags. Their released dataset contains about 95.4 million words distributed in about 5.4 million sentences. As Bushra Jawaid dataset is very big and for processing requires sophisticated hardware, therefore due to hardware limitations, for experiments we only considered a portion from this dataset comprising of 164466 words organized in hundred documents and 5000 sentences. Consolidated statistics of the portion of BJ dataset considered for experiments are provided in Table 10. The Bushra Jawaid dataset was annotated with Sajjad POS tagset and details of Sajjad tagset is already provided in Table 2. Table 11 shows consolidated statistics of various POS tags extracted from the available version of BJ dataset.

TABLE 10. Consolidated statistics of BJ dataset.

Total no. of words	164466
Total no of Documents	100
Total no. of Sentences	5000

B. IMPLEMENTATION DETAILS

Training and testing of the proposed machine and deep learning model were carried through CRFSharp libraries. For experimentation, we used C# based three open sources libraries for our proposed approaches e.g. CRFSharp libraries[2] for CRF approach, RNNSharp libraries[3] for DRNN approach, SVM_Text_Classifier[4] for support vector machines approach and Forward Viterbi - Hidden Markov Model[5] for HMM Viterbi decoding. The adoption of CRFSharp and RNNSharp packages have been reported in the literature for western languages to accomplish tasks such as semantic role labeling [71], term extraction [72], POS tagging [73], named entity recognition [74] and so on,

[1] https://lindat.mff.cuni.cz/repository/xmlui/handle/11858/00-097C-0000-0023-65A9-5

[2] https://github.com/zhongkaifu/CRFSharp
[3] https://github.com/zhongkaifu/RNNSharp
[4] https://github.com/alexandrekow/svmtutorial
[5] http://pcarvalho.com/forward_viterbi/

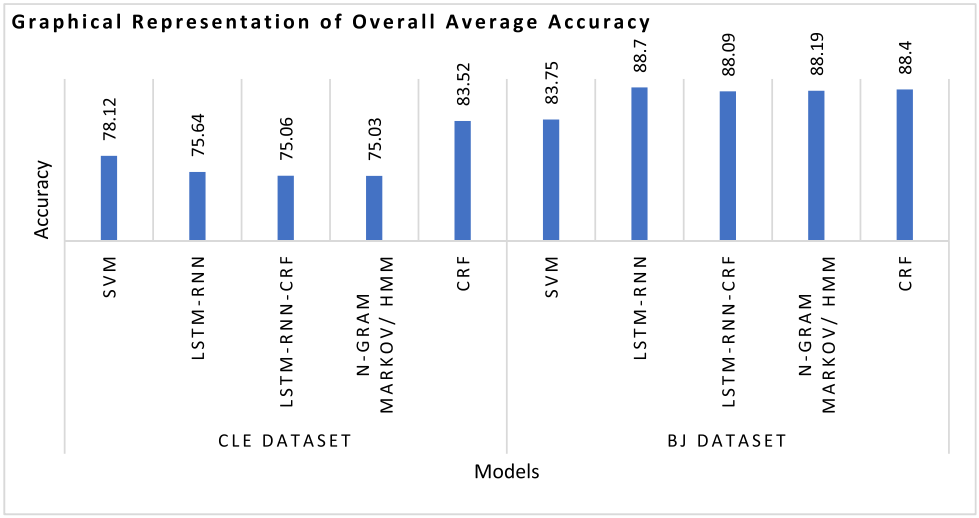

FIGURE 4. Graphical depiction of overall results.

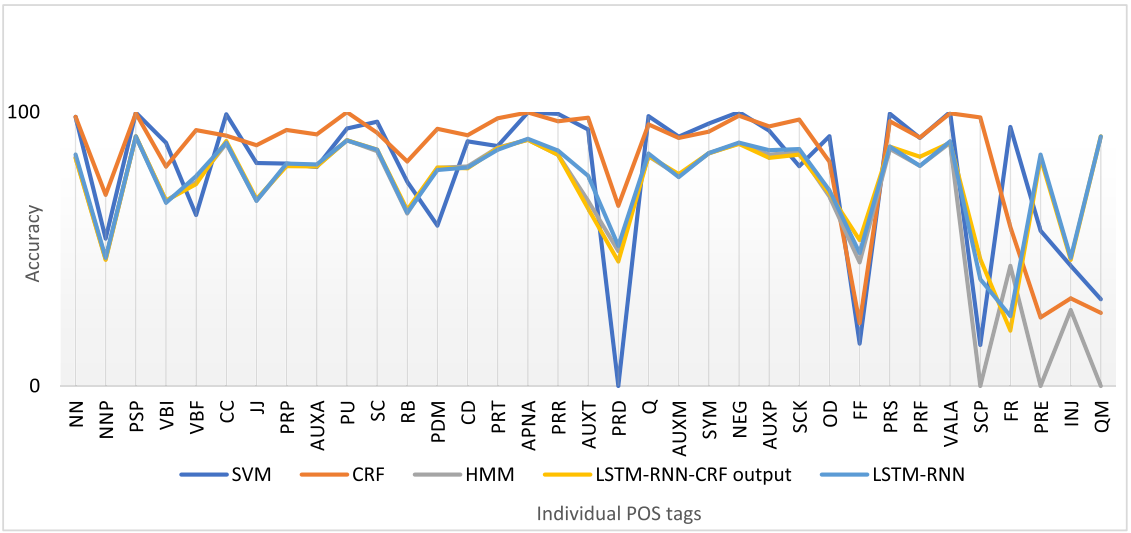

FIGURE 5. Graphical depiction of individaul pos tag results on CLE dataset.

while its adoption for the same task in South Asian languages including Urdu is still missing.

C. RESULTS

In this research work accuracy is used as an evaluation standard for performance measurement of all models.

$$Accuracy = \frac{Total\ number\ of\ corrected\ tags}{Total\ number\ of\ tags}$$

After performing 10-fold cross-validation experiments, the results figures of individual models show that on CLE dataset CRF model significantly performed better than other model while on BJ dataset LSTM-RNN model superseded all other model with significant margin.

Table 12 evince average accuracy of CRF, SVM, LSTM-RNN, LSTM-RNN with CRF output and HMM while Figure 4 graphically depicts Table 12 results.

When tested on CLE dataset, CRF recorded superior results by achieving an average accuracy value of 83.52% while SVM, LSTM-RNN, LSTM-RNN with CRF output and HMM models achieved averaged accuracy values of 78.12%, 75.64%, 75.06% and 75.03% which were not better than the results figures recorded by the CRF based approach. Similarly, when tested on BJ dataset LSTM-RNN recorded best average accuracy value of 88.7% whereas the SVM, RNN variants, CRF and HMM achieved average accuracy values of 83.75%, 88.09%, 88.4% and 88.19%. The reported results of SVM, LSTM-RNN-CRF, CRF and HMM on BJ dataset is not better than LSTM-RNN. All models on BJ dataset performed comparatively with little variation in results, except

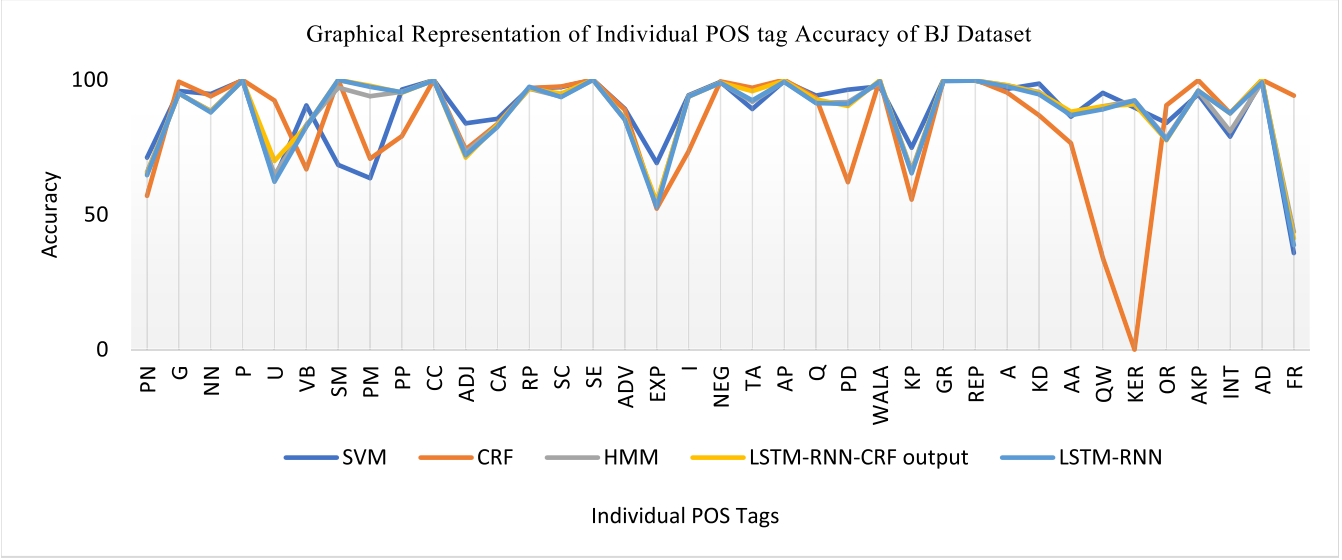

FIGURE 6. Graphical depiction of individual POS tag results on BJ dataset.

TABLE 11. Tag wise consolidated statistics of BJ POS tagged dataset.

Tag	Occurrence Frequency	Tag	Occurrence Frequency	Tag	Occurrence Frequency	Tag	Occurrence Frequency
A	375	EXP	351	NEG	2021	SC	4287
AA	5110	FR	32	NN	41135	SE	2930
AD	88	G	766	OR	275	SM	6897
ADJ	9915	GR	743	P	20549	TA	6398
ADV	2607	I	3469	PD	1807	U	78
AKP	459	INT	93	PM	3912	VB	20123
AP	1085	KD	420	PN	10908	VALA	513
CA	3461	KER	491	PP	5625	---	---
CC	3559	KP	196	Q	2156	---	---
REP	1094	RP	180	QW	358	---	---

TABLE 12. Average accuracy results of all models.

Datasets	Model	Accuracy
CLE dataset	SVM	78.12
	LSTM-RNN	75.64
	LSTM-RNN-CRF	75.06
	N-Gram Markov/ HMM	75.03
	CRF	**83.52**
BJ dataset	SVM	83.75
	LSTM-RNN	**88.7**
	LSTM-RNN-CRF	88.09
	N-Gram Markov/ HMM	88.19
	CRF	88.4

the SVM where the significance margin of other models is higher than 4.5% in accuracy. On CLE dataset the CRF based Urdu POS system superseded SVM approach by more than 5.4% on average accuracy while on BJ dataset the LSTM-RNN margin value is more than 4.5%. Similarly, when compared with LSTM-RNN and LSTM-RNN with CRF output approaches on CLE dataset, the average accuracy margin values of CRF is more than 7.88% while on BJ dataset the margin values of LSTM-RNN is 0.61% and 0.3%. Table 13 and Table 14 shows individual POS tag results on CLE and BJ datasets of CRF, SVM, LSTM, LSTM-RRNN with CRF output and HMM models while Figure 5 and FIGURE 6 graphically depict Individual POS results. The CRF based Urdu POS system on CLE dataset achieved a highest average accuracy of 99.95% for punctuation (PU) tag and the lowest average accuracy of 22.91% for Foreign Fragment (FF) tag while on the same dataset the SVM based Urdu POS, system recorded highest average accuracy value of 100% for both NEG and VALA tags and lowest zero average accuracy for PRD POS tag. Similarly, LSTM-RNN and LSTM-RNN with CRF output recorded highest average accuracy of 91.4%,91.8% for PSP, while lowest average accuracy of 25.49, 20.2% for FR POS tags while HMM based

TABLE 13. Average accuracy of individual POS tag on CLE dataset.

POS Tag	Accuracy				
	SVM	CRF	HMM	LSTM-RNN-CRF output	LSTM-RNN
NN	**98.22**	**98.2**	83.51	83.2	84.42
NNP	53.79	**69.77**	46.24	46.06	46.8
PSP	**99.76**	99.51	91.1	91.08	91.04
VBI	**88.77**	80.06	67.53	67.61	66.88
VBF	62.39	**93.38**	74.45	73.68	76.6
CC	**99.17**	91.35	88.49	89.38	88.77
JJ	81.34	**87.88**	67.64	68.22	67.6
PRP	81.14	**93.43**	80.61	80.24	81.14
AUXA	79.89	**91.81**	80.14	80.11	80.84
PU	93.93	**99.95**	89.67	89.69	89.57
SC	**96.47**	92.39	85.68	86.26	86.22
RB	74.56	81.91	62.95	64.26	63.28
PDM	58.53	**93.82**	79.6	79.72	78.79
CD	**89.29**	91.53	80.14	79.42	79.75
PRT	87.56	**97.69**	86.97	86.78	86.19
APNA	**99.85**	**99.85**	89.85	89.85	90.18
PRR	**99.29**	96.62	84.4	84.34	85.91
AUXT	93.56	**97.89**	67.52	64.71	76.67
PRD	0	65.74	49.24	45.43	51.13
Q	**98.49**	95.47	83.88	83.95	84.82
AUXM	**90.99**	90.53	76.73	77.31	76.19
SYM	**95.78**	92.78	84.85	84.96	85.03
NEG	**100**	98.62	88.55	88.32	88.82
AUXP	93.15	**94.71**	84.45	83.23	86.07
SCK	80.21	**97.24**	85.29	84.43	86.48
OD	**91.14**	81.83	69.15	70.26	71.37
FF	15.47	22.91	45.14	**53.24**	48.5
PRS	**99.37**	96.61	86.53	87.37	87.33
PRF	90.54	**90.69**	80.34	83.67	80.34
VALA	**100**	99.5	88.73	88.73	89.28
SCP	14.93	**98.01**	---	46.33	39
FR	**94.5**	58	43.83	20.2	25.49
PRE	56.67	25	---	83.2	**84.42**
INJ	43.89	31.94	27.7	46.06	**46.8**
QM	31.67	26.67	---	**91.08**	91.04

system recorded highest average accuracy of 91.1% for PSP and lowest of 27.7% for INJ POS tags.

The CRF based Urdu POS system on BJ dataset achieved a highest average accuracy of 100% for SM, CC, SE, AP, WALA, GR, REP and AD tags and the lowest zero average accuracy for KER tag while on the same dataset the SVM based Urdu POS, system recorded highest average accuracy value of 100% for SE,GR and AD tags and lowest average accuracy of 35.83% for FR POS tag. Similarly, LSTM-RNN recorded highest average accuracy of 100% for SE, while lowest average accuracy of 38.67% for FR POS tags and LSTM-RNN with CRF output recorded highest average accuracy of 100% for SE, GR and AD while lowest average accuracy of 41.17 % for FR tag. The HMM based system recorded highest average accuracy of 100% for SE tag and lowest of 43.67% for FR POS tags.

VI. DISCUSSION AND ERROR ANALYSIS

In this research work, we exploited comparative analysis of machine learning and deep learning approaches for POS task in Urdu offline digital text. On CLE dataset when the only language-independent feature was used, our CRF based POS system achieved better performance than SVM, LSTM-RNN, and LSTM-RNN with CRF output, indicating the potential of using CRF for Urdu POS task. We found that the performance improvement of CRF over SVM and RNNs was its power of employment of correlativity inside two tags.

The SVM classifiers during training stage make use of maximal margin conception and likewise experience the capacity to manage whole observations one at a time [75]. The core supplement of SVMs models are the kernel function and it is the kernel functions which are responsible for categorization of any particular data.

TABLE 14. Average accuracy of individual POS tag on BJ dataset.

POS Tag	Accuracy				
	SVM	CRF	HMM	LSTM-RNN-CRF output	LSTM-RNN
PN	71.19	57.04	**65.83**	65.24	64.7
G	95.86	**99.32**	94.95	94.96	95.03
NN	**94.76**	93.93	88.51	88.33	87.99
P	99.83	**99.98**	99.79	99.67	99.78
U	63.23	**92.36**	64.77	69.92	62.24
VB	**90.5**	66.85	83.27	82.84	82.69
SM	68.47	**100**	97.12	99.94	99.97
PM	63.54	70.78	93.93	**97.82**	97.44
PP	**96.4**	79.14	95.61	95.25	95.43
CC	99.89	**100**	99.82	99.87	99.79
ADJ	**83.86**	74.18	73.2	71.27	72.09
CA	85.52	83.67	83.11	83.31	82.53
RP	96.85	96.94	96.69	96.77	**97.44**
SC	97.28	**97.55**	93.81	94.9	93.62
SE	**100**	**100**	**100**	**100**	**100**
ADV	**89.24**	88.93	85.23	85.61	85.25
EXP	**69.2**	52.28	54.26	54.65	52.81
I	**94.31**	73.52	94.03	93.75	93.82
NEG	99.26	**99.42**	98.93	98.93	98.92
TA	89.19	97.04	91.84	**95.84**	92.39
AP	99.54	**100**	99.34	99.64	99.27
Q	**94.18**	93.42	91.52	92.72	91.37
PD	**96.36**	62.08	91.73	90.35	91.08
WALA	97.59	**100**	99.03	99.83	99.45
KP	74.84	55.64	**66.42**	65.78	65.38
GR	**100**	**100**	99.61	**100**	99.61
REP	99.71	**100**	99.73	99.66	99.73
A	96.67	95.28	97.96	**98.03**	97.53
KD	**98.6**	86.86	95.49	95.24	94.66
AA	86.34	76.51	87.06	**88.18**	86.93
QW	**95.18**	34.03	90.13	90.18	89.1
KER	89.74	0	**92.34**	90.91	92.31
OR	84.09	**90.55**	78.41	77.65	77.85
AKP	94.61	**99.85**	95.79	95.99	95.92
INT	79.01	**87.66**	80.96	87.62	87.63
AD	**100**	**100**	99.09	**100**	99.09
FR	35.83	**94.17**	43.67	41.17	38.67

During the training and testing phase of SVM, majority errors were occurred in case of when it mixed up proper nouns (PN) with common nouns (NN), VB and EXP part of speech.

Similarly to higher degree it also mixed up NN with PN, Adj, and VB part of speech. Table 15 presents summary of the SVM model mixed up tags when tested on BJ dataset.

CRFs are state-of-the-art sequential classifiers and draw together the soundest feature of generative plus classification models [76]. There are many types of features such as orthographic information of word, the affixes information, the POS information of the left and right words and many more, but in this study, we demonstrated the performance of CRF with only language-independent feature e.g. context words

TABLE 15. Confusion matrix of SVM on BJ dataset.

Current tag	PN	NN	ADJ	VB	EXP
PN	---	462	---	26	13
NN	165	---	45	31	5
ADJ	62	197	---	6	2
VB	20	174	9	---	---
CA	17	37	.5	6	2

window as it requires a minimal effort on feature engineering while producing better results. The simplicity of the proposed CRF based approach not only reduces computational time but also reduces human efforts of feature engineering. During decoding phase of test data CRF mixed up several POS tags with each other. Table 16 provided summary of CRF most mixed up tags on BJ dataset.

Since RNN-LSTM model showed superior performance comparatively to other models used in this study when tested it on BJ dataset. Therefore, this study proves that by using various RNN model architectures with sparse (i.e. template features) and dense features (i.e. word embeddings) we can obtain better POS tagging results for Urdu than a language-independent features based traditional CRF. Since RNN make use of both template as well as word embeddings feature. For DRNN training we considered the template feature/context word window already used in CRF training while the word embedding features are extracted from huge amount of unlabeled data. We used the Txt2Vec[6] project – a C# implementation – to generate our word embedding features. This project interprets specified words and phrases into their corresponding vectors, so that each dimension of the vector corresponds to a feature. DRNN assigns a class label to words in real time, for example assigning tags using the strategy of where and when a word occurs. In CRF, the input and output are directly connected without any middle layers, while in RNN inputs are connected with output through recurrent cells [77]. RNN holds each word in the training data as a high-dimensional real-valued vector and, in this vector representation, the chances that similar words will be close to each other increases, so the model takes advantage of the relationship between similar words. Word vector space representation therefore provides better performance for analogous words in an analogous linguistic context

In the above confusion matrixes, the statistics provided in each row represents the number of times a current tag of testing data was confused with irrelevant tag in testing phase. For example, in **Table 15** the current tag PN was confused 462 time with NN tag, 26 time with VB and 13 times with EXP tag. Similarly, the current tag ADJ was confused 62 times with PN tag, 197 times with NN, 6 time with VB and 2 times with EXP tag.

[6]https://github.com/zhongkaifu/Txt2Vec

TABLE 16. Confusion matrix of CRF on BJ dataset.

Current tag	PN	NN	SM	ADJ	VB
PN	---	178	40	29	19
NN	59	---	83	20	28
SM	51	138	---	13	9
ADJ	38	81	15	---	13
VB	19	80	23	14	---

VII. CONCLUSION

These days the state-of-the-art approaches that are widely adopted around the globe for the development of POS taggers, in almost all languages including Urdu, are based on machine and deep learning models [3], [5]. Building an effective feature is extremely necessary. Therefore, at heart, to do any learning at all, the system needs (a) annotated data, usually provided by annotators, (b) a good feature set, to generate a model from data and then use that model to classify new data.

In this work, therefore, in addition to comparison of machine and deep learning models we also proposed a very well-balanced context word features for both machine learning and deep learning models. The CLE and BJ gold standard dataset are used to frame and assess machine and deep learning-based Urdu POS tagger.

The experiments show that CRF based model performs better as compared to the SVM and RNNs and n-gram approaches on CLE dataset while DRRN models performs better on BJ dataset. In future work, we plan (a) to evaluate the performance of CRF on more sophisticated feature (b) Since in this study we have observed that LSTM-RNN performed comparatively with CRF therefore, in future we also plan to propose a more sophisticated deep learning-based Urdu POS system based on semi-supervised learning e.g. character-embedding learning from large unlabeled data (c) Moreover, in future we also plan to check that how much neural Urdu-English machine translation system can be benefited from this proposed Urdu POS system.

REFERENCES

[1] W. Khan *et al.*, "Lang resources & evaluation," 2018. doi: 10.1007/s10579-018-9439-6.
[2] T. Horváth, Z. Alexin, T. Gyimóthy, and S. Wrobel, "Application of different learning methods to Hungarian part-of-speech tagging," in *Proc. Int. Conf. Inductive Log. Program.*, 1999, pp. 128–139.
[3] A. Daud, W. Khan, and D. Che, "Urdu language processing: A survey," *Artif. Intell. Rev.*, vol. 47, no. 3, pp. 279–311, 2017.
[4] S. Mukund, "An NLP framework for non-topical text analysis in Urdu—A resource poor language," Ph.D. Dissertation, ProQuest LLC, State Univ. New York Buffalo, Buffalo, NY, USA, 2012.
[5] W. Khan, A. Daud, J. A. Nasir, and T. Amjad, "A survey on the state-of-the-art machine learning models in the context of NLP," *Kuwait J. Sci.*, vol. 43, no. 4, pp. 66–84, 2016.
[6] C. Biemann, "Unsupervised part-of-speech tagging employing efficient graph clustering," in *Proc. 21st Int. Conf. Comput. Linguistics 44th Annu. Meeting Assoc. Comput. Linguistics, Student Res. Workshop*, Jul. 2006, pp. 7–12.

[7] J. V. Graça, K. Ganchev, L. Coheur, F. Pereira, and B. Taskar, "Controlling complexity in part-of-speech induction," *J. Artif. Intell. Res.*, vol. 41, pp. 527–551, Aug. 2011.
[8] Y. Goldberg, "A primer on neural network models for natural language processing," *J. Artif. Intell. Res.*, vol. 57, pp. 345–420, Nov. 2016.
[9] D. Roth and D. Zelenko, "Part of speech tagging using a network of linear separators," in *Proc. 36th Annu. Meeting Assoc. Comput. Linguistics*, vol. 2, 1998, pp. 1136–1142.
[10] W. Anwar, X. Wang, L. Li, and X.-L. Wang, "A statistical based part of speech tagger for Urdu language," in *Proc. Int. Conf. Mach. Learn. Cybern.*, Aug. 2007, pp. 3418–3424.
[11] A. Hardie, "Developing a tagset for automated part-of-speech tagging in Urdu," Corpus Linguistics Conf., Dept. Linguistics, Lancaster Univ., UCREL Tech. Papers, 2003, vol. 16.
[12] R. Chiong and W. Wei, "Named entity recognition using hybrid machine learning approach," in *Proc. 5th IEEE Int. Conf. Cogn. Inform.*, Jul. 2006, pp. 578–583.
[13] H. Sajjad and H. Schmid, "Tagging Urdu text with parts of speech: A tagger comparison," in *Proc. 12th Conf. Eur. Chapter Assoc. Comput. Linguistics*, Mar. 2009, pp. 692–700.
[14] B. Jawaid and O. Bojar, "Tagger voting for Urdu," in *Proc. 24th Int. Conf. Comput. Linguistics*, 2012, pp. 135–144.
[15] B. Jawaid, A. Kamran, and O. Bojar, "A tagged corpus and a tagger for Urdu," in *Proc. LREC*, 2014, pp. 2938–2943.
[16] F. Naz, W. Anwar, U. I. Bajwa, and E. U. Munir, "Urdu part of speech tagging using transformation based error driven learning," *World Appl. Sci. J.*, vol. 16, no. 3, pp. 437–448, 2012.
[17] S. Hussain and M. Afzal, "Urdu computing standards: Urdu Zabta Takhti (UZT) 1.01," in *Proc. 21st Century IEEE Int. Multi Topic Conf. (INMIC)*, Dec. 2001, pp. 223–228.
[18] T. Ahmed and A. Hautli, "A first approach towards an Urdu WordNet," in *Proc. Linguistics Literature Rev.*, vol. 1, 2011, pp. 1–14.
[19] K. Riaz, "Rule-based named entity recognition in Urdu," in *Proc. Named Entities Workshop*, 2010, pp. 126–135.
[20] A. Muaz, A. Ali, and S. Hussain, "Analysis and development of Urdu POS tagged corpus," in *Proc. 7th Workshop Asian Lang. Resour.*, 2009, pp. 24–29.
[21] A. Tafseer *et al.*, "The CLE Urdu POS tagset," in *Proc. 9th Int. Conf. Lang. Resour. Eval. (LREC)*, 2015, pp. 2920–2925.
[22] J. T. Platts, *A Grammar of the Hindustani or Urdu Language*. London, U.K.: WH Allen, 1909.
[23] M. A. Haq, *Urdu Sarf-o-Nakhu*. New Delhi, India: Amjuman-e-Taraqqi, 1987.
[24] R. L. Schmidt, *Urdu, an Essential Grammar*. Hove, U.K.: Psychology Press, 1999.
[25] Q.-U.-A. Akram, A. Naseer, and S. Hussain, "Assas-Band, an affix-exception-list based Urdu stemmer," in *Proc. 7th Workshop Asian Lang. Resour.*, 2009, pp. 40–46.
[26] J. I. Kazama and K. Torisawa, "Exploiting Wikipedia as external knowledge for named entity recognition," in *Proc. Joint Conf. Empirical Methods Natural Lang. Process. Comput. Natural Lang. Learn.*, 2007, pp. 698–707.
[27] A. Daud, W. Khan, and D. Che, "Urdu language processing: A survey," *Artif. Intell. Rev.*, vol. 47, no. 3, pp. 279–311, 2016.
[28] J. Lafferty, A. McCallum, and F. C. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," presented at the 18th Int. Conf. Mach. Learn. (ICML), 2001.
[29] S. Žitnik, L. Šubelj, and M. Bajec, "SkipCor: Skip-mention coreference resolution using linear-chain conditional random fields," *PLoS ONE*, vol. 9, no. 6, 2014, Art. no. e100101.
[30] Y. Benajiba and P. Rosso, "Arabic named entity recognition using conditional random fields," in *Proc. Workshop HLT NLP Arabic World (LREC)*, 2008, pp. 143–153.
[31] A. Ekbal, R. Haque, and S. Bandyopadhyay, "Named entity recognition in Bengali: A conditional random field approach," in *Proc. 3rd Int. Joint Conf. Natural Lang. Process. (IJCNLP)*, 2008, pp. 589–594.
[32] S. Song, N. Zhang, and H. Huang, "Named entity recognition based on conditional random fields," *Cluster Comput.*, pp. 1–12, Sep. 2017. doi: 10.1007/s10586-017-1146-3.
[33] V. Vapnik, *Statistical Learning Theory*. New York, NY, USA: Springer-Verlag, 2013.
[34] L. C. Padierna, J. M. Carpio, A. Rojas, H. Puga, R. Baltazar, and H. Fraire, "Hyper-parameter tuning for support vector machines by estimation of distribution algorithms," in *Nature-Inspired Design of Hybrid Intelligent Systems* (Studies, in Computational Intelligence), vol. 667, P. Melin, O. Castillo, and J. Kacprzyk, Eds. Cham, Switzerland: Springer, 2017, pp. 787–800.
[35] P. J. Antony and K. P. Soman, "Kernel based part of speech tagger for Kannada," in *Proc. Int. Conf. Mach. Learn. Cybern. (ICMLC)*, Jul. 2010, pp. 2139–2144.
[36] N. E. Ayat, M. Cheriet, and C. Y. Suen, "Automatic model selection for the optimization of SVM kernels," *Pattern Recognit.*, vol. 38, no. 10, pp. 1733–1745, 2005.
[37] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," in *Proc. Eur. Conf. Mach. Learn.*, 1998, pp. 137–142.
[38] A. Ben-Hur, C. S. Ong, S. Sonnenburg, B. Schölkopf, and G. Rätsch, "Support vector machines and kernels for computational biology," *PLoS Comput. Biol.*, vol. 4, no. 10, 2008, Art. no. e1000173.
[39] C.-W. Hsu, C.-C. Chang, and C.-J. Lin, "A practical guide to support vector classification," Dept. Comput. Sci., Nat. Taiwan Univ., Taipei, Taiwan, Tech. Rep., 2003.
[40] Z. Xu, M. Dai, and D. Meng, "Fast and efficient strategies for model selection of Gaussian support vector machine," *IEEE Trans. Syst., Man, Cybern. B. Cybern.*, vol. 39, no. 5, pp. 1292–1307, Oct. 2009.
[41] H.-T. Lin and C.-J. Lin, "A study on sigmoid kernels for SVM and the training of non-PSD kernels by SMO-type methods," *Neural Comput.*, vol. 3, pp. 1–32, Mar. 2003.
[42] A. Rojas-Domínguez, L. C. Padierna, J. M. C. Valadez, H. J. Puga-Soberanes, and H. J. Fraire, "Optimal hyper-parameter tuning of SVM classifiers with application to medical diagnosis," *IEEE Access*, vol. 6, pp. 7164–7176, 2017.
[43] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by backpropagating errors," *Nature*, vol. 323, pp. 533–536, 1986.
[44] L. Deng, "A tutorial survey of architectures, algorithms, and applications for deep learning," *APSIPA Trans. Signal Inf. Process.*, vol. 3, pp. 1–29, Jan. 2014.
[45] J. L. Elman, "Finding structure in time," *Cognit. Sci.*, vol. 14, no. 2, pp. 179–211, Mar. 1990.
[46] T. Mikolov, M. Karafiát, L. Burget, J. Černocký, and S. Khudanpur, "Recurrent neural network based language model," in *Proc. INTERSPEECH*, 2010, p. 3.
[47] M. Bodén, "A guide to recurrent neural networks and backpropagation," A. Holst, Ed. DALLAS Project, NUTEK-Supported Project AIS-8: Application of Data Analysis With Learning Systems, 1999-2001, SICS, Tech. Rep. T2002:03, Kista, Sweden, 2001.
[48] T. Xie, H. Yu, and B. Wilamowski, "Comparison between traditional neural networks and radial basis function networks," in *Proc. IEEE Int. Symp. Ind. Electron. (ISIE)*, Jun. 2011, pp. 1194–1199.
[49] I. Arel, D. C. Rose, and T. P. Karnowski, "Deep machine learning—A new frontier in artificial intelligence research [research frontier]," *IEEE Comput. Intell. Mag.*, vol. 5, no. 4, pp. 13–18, Nov. 2010.
[50] A. Graves, A.-R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2013, pp. 6645–6649.
[51] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015.
[52] Z. Zhang, Z. Sun, J. Liu, J. Chen, Z. Huo, and X. Zhang. (2016). "Deep recurrent convolutional neural network: Improving performance for speech recognition." [Online]. Available: https://arxiv.org/abs/1611.07174
[53] K. Yao, B. Peng, G. Zweig, D. Yu, X. Li, and F. Gao, "Recurrent conditional random field for language understanding," presented at the IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP), May 2014.
[54] K. Yao, B. Peng, G. Zweig, D. Yu, X. Li, and F. Gao, "Recurrent conditional random fields," in *Proc. Neural Inf. Process. Syst., NIPS Deep Learn. Workshop*, 2013, pp. 1–9.
[55] C. Lyu, B. Chen, Y. Ren, and D. Ji, "Long short-term memory RNN for biomedical named entity recognition," *BMC Bioinf.*, vol. 18, no. 1, p. 462, 2017.
[56] K. Yao, G. Zweig, M.-Y. Hwang, Y. Shi, and D. Yu, "Recurrent neural networks for language understanding," in *Proc. INTERSPEECH*, 2013, pp. 2524–2528.
[57] I. Sutskever, J. Martens, and G. E. Hinton, "Generating text with recurrent neural networks," in *Proc. the 28th Int. Conf. Mach. Learn. (ICML)*, 2011, pp. 1017–1024.

[58] Z. Dai, L. Li, and W. Xu. (2016). "CFO: Conditional focused neural question answering with large-scale knowledge bases." [Online]. Available: https://arxiv.org/abs/1606.01994
[59] F. Ture and O. Jojic. (2016). "No need to pay attention: Simple recurrent neural networks work! (For answering 'Simple' questions)." [Online]. Available: https://arxiv.org/abs/1606.05029
[60] R. Serizel and D. Giuliani, "Deep-neural network approaches for speech recognition with heterogeneous groups of speakers including children," *Natural Lang. Eng.*, vol. 23, no. 3, pp. 325–350, 2016.
[61] M.-T. Luong, H. Pham, and C. D. Manning. (2015). "Effective approaches to attention-based neural machine translation." [Online]. Available: https://arxiv.org/abs/1508.04025
[62] A. Karpathy and L. Fei-Fei, "Deep visual-semantic alignments for generating image descriptions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3128–3137.
[63] D. Bonadiman, A. Severyn, and A. Moschitti, "Deep neural networks for named entity recognition in Italian," in *Proc. 2nd Italian Conf. Comput. Linguistics CLiC*, 2015, p. 51.
[64] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–538, 1986.
[65] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[66] H. Sak, A. Senior, and F. Beaufays. (2014). "Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition." [Online]. Available: https://arxiv.org/abs/1402.1128
[67] Z. Liu *et al.*, "Entity recognition from clinical texts via recurrent neural network," *BMC Med. Inform. Decis. Making*, vol. 17, no. 2, p. 67, 2017.
[68] V. M. Janakiraman. (2017). "Explaining aviation safety incidents using deep temporal multiple instance learning." [Online]. Available: https://arxiv.org/abs/1710.04749
[69] M. Gridach, "Character-aware neural networks for Arabic named entity recognition for social media," in *Proc. 6th Workshop South Southeast Asian natural Lang. Process. (WSSANLP)*, 2016, pp. 23–32.
[70] Z. Youzhi, "Research and implementation of part-of-speech tagging based on hidden Markov model," in *Proc. Asia–Pacific Conf. Comput. Intell. Ind. Appl. (PACIIA)*, Nov. 2009, pp. 26–29.
[71] M. Bilgin and M. F. Amasyali, "Semantic Role Labeling With Relative Clauses," *Int. J. Electron., Mech. Mechatron. Engineer*, vol. 6, no. 2, pp. 1165–1175, 2016.
[72] Y. Yin, F. Wei, L. Dong, K. Xu, M. Zhang, and M. Zhou. (2016). "Unsupervised word and dependency path embeddings for aspect term extraction." [Online]. Available: https://arxiv.org/abs/1605.07843
[73] M. Maimaiti, A. Wumaier, K. Abiderexiti, and T. Yibulayin, "Bidirectional long short-term memory network with a conditional random field layer for Uyghur part-of-speech tagging," *Information*, vol. 8, no. 4, p. 157, 2017.
[74] N. D. Phuong and V. T. N. Chau, "Automatic de-identification of medical records with a multilevel hybrid semi-supervised learning approach," in *Proc. IEEE RIVF Int. Conf. Comput. Commun. Technol., Res., Innov., Vis. Future (RIVF)*, Nov. 2016, pp. 43–48.
[75] G. Hoefel and C. Elkan, "Learning a two-stage SVM/CRF sequence classifier," in *Proc. 17th ACM Conf. Inf. Knowl. Manage.*, 2008, pp. 271–278.
[76] C. Raymond and G. Riccardi, "Generative and discriminative algorithms for spoken language understanding," in *Proc. INTERSPEECH*, 2007, pp. 1605–1608.
[77] Z. Huang, W. Xu, and K. Yu. (2015). "Bidirectional LSTM-CRF models for sequence tagging." [Online]. Available: https://arxiv.org/abs/1508.01991

WAHAB KHAN received the M.S. degree in computer science from the University of Science and Technology, Bannu, Pakistan, in 2009. He is currently pursuing the Ph.D. degree in computer science with International Islamic University Islamabad, Pakistan. His research interests include natural language processing, machine learning, and deep learning and data mining.

ALI DAUD received the Ph.D. degree from Tsinghua University, in 2010.

He is the Head of the Data Mining and Information Retrieval Group, International Islamic University Islamabad, Pakistan, where he is currently an Associate Professor with the Department of CS and SE. He has published about 70 papers in reputed international impact factor journals and conferences. He has completed supervising five Ph.D., 22 M.S., and 18 B.S. dissertation/theses. He has taken part in many research projects and was PI of two projects as well. His research interests include data mining, social network analysis and mining, probabilistic models, scientometrics, and natural language processing.

KHAIRULLAH KHAN received the Ph.D. degree in information technology from Universiti Teknologi PETRONAS, Malaysia, in 2012, where he worked on machine learning for the automatic detection of opinion targets from text. He is currently an Associate Professor with the Department of Computer Science, University of Science and Technology, Bannu, Pakistan.

JAMAL ABDUL NASIR received the Ph.D. degree in computer science from the Lahore University of Management University, Pakistan.

He is currently an Assistant Professor with the Department of CS and SE, International Islamic University Islamabad, Islamabad, Pakistan. His research interests include data science, natural language processing, text mining, text summarization, recommendation systems, distributed systems, and software systems quality.

MOHAMMED BASHERI received the bachelor's degree in computer education from King Abdulaziz University, Saudi Arabia, the master's degree in information technology from Griffith University, Australia, and the Ph.D. degree in computer science from the School of Engineering and Computer Science, Durham University, U.K. He is the Vice Dean of Development with the Faculty of Computing and IT, King Abdulaziz University. He has 15 years of experience as a Professional Academic.

NAIF ALJOHANI received the Ph.D. degree in computer science from the University of Southampton, U.K.

He is currently an Assistant Professor with the Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia. His research interests include linked open data, social networks, data mining.

FAHD SALEH ALOTAIBI received the Ph.D. degree in computer science from the University of Birmingham, U.K.

He is currently an Assistant Professor with the Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia. His research interests include natural language processing, social networks, and data mining.