Enhancing Sindhi Word Segmentation Using Subword Representation Learning and Position-Aware Self-Attention

WAZIR ALI¹ 2, JAY KUMAR³, SAIFULLAH TUMANI⁴ 5 6, REDHWAN NOUR⁷, ADEEB NOOR⁸, AND ZENGLIN XU⁹ (Senior Member, IEEE)

¹College of Computer Science and Information Systems, Institute of Business Management, Karachi 75190, Pakistan
²SoloGen AI Pvt. Ltd., Karachi 75190, Pakistan
³Faculty of Computer Science, Dalhousie University, Halifax, NS B3H 4R2, Canada
⁴Hessianflux OÜ, 10115 Tallinn, Estonia
⁵BioQuant, IPMB, Heidelberg University, 69120 Heidelberg, Germany
⁶Computer Science Department, Bahria University, Karachi, Sindh 75290, Pakistan
⁷Department of Computer Science, College of Computer Science and Engineering, Taibah University, Medina 42353, Saudi Arabia
⁸School of Computer Science and Technology, Hebing Institute of Technology, Shenzhen 510085, China

Corresponding author: Wazir Ali (aliwazirjam@gmail.com)

This work was supported by the National Key Research and Development Program of China under Grant 2018YFB1005100 and Grant 2018YFB1005104.

ABSTRACT Sindhi word segmentation is a challenging task due to space omission and insertion issues. The Sindhi language itself adds to this complexity. It’s cursive and consists of characters with inherent joining and non-joining properties, independent of word boundaries. Existing Sindhi word segmentation methods rely on designing and combining hand-crafted features. However, these methods have limitations, such as difficulty handling out-of-vocabulary words, limited robustness for other languages, and inefficiency with large amounts of noisy or raw text. Neural network-based models, in contrast, can automatically capture word boundary information without requiring prior knowledge. In this paper, we propose a Subword-Guided Neural Word Segmenter (SGNWS) that addresses word segmentation as a sequence labeling task. The SGNWS model incorporates subword representation learning through a bidirectional long short-term memory encoder, position-aware self-attention, and a conditional random field. Our empirical results demonstrate that the SGNWS model achieves state-of-the-art performance in Sindhi word segmentation on six datasets.

INDEX TERMS Attention mechanism, neural network, long short-term memory, representation learning, word segmentation.

I. INTRODUCTION

Word segmentation is a fundamental and challenging task in Sindhi text classification and other Natural Language Processing (NLP) tasks [1]. A word segmenter determines the word boundaries in the shape of beginning and ending [2]. It has been largely investigated in many space-delimited languages including English [3], [4], Arabic [5], Urdu [6] and non-space delimited languages including Chinese [7], Japanese [8], and Burmese [9]. However, the word segmentation in low-resource Sindhi language has not been studied well [10], mainly due to the lack of language resources and its morphological complexities.

Sindhi word segmentation is a challenging task because it exhibits space omission and space insertion problems [11]. This is partly because the Arabic script, which is cursive in nature, consists of characters that have inherent joining and non-joining attributes regardless of a word boundary [1]. Apart from the discussed problems, there is no gold-standard benchmark corpus for Sindhi to evaluate the segmentation task [10]. In summary, the SWS task is difficult, important, and not studied as a sequence modeling problem. The...
previous approaches mainly rely on rule-based methods [1], [11], [12], [13], which have certain limitations such as the inability to deal with Out-Of-Vocabulary (OOV) words, less robustness for other languages, and the algorithms’ inefficiency to deal with a large amount of noisy text. Thus, the existing approaches lack the robustness and applicability towards open-source implementation.

Neural networks have gained great attention in the NLP community [14], [15], [16]. On the one hand, state-of-the-art sequence tagging systems rely heavily on large amounts of task-specific knowledge [17]. This knowledge comes in the form of hand-engineered features and data pre-processing. While neural network models have been extensively investigated for separate word segmentation in many space-delimited and non-space-delimited languages, Sindhi word segmentation has not yet been addressed as a sequence tagging problem using neural network models.

Recently, RNN variants, including LSTMs and BiLSTMs, have been successfully exploited for sequence tagging problems [14], [18], [19]. However, LSTMs are limited in that they can only encode sequences in a unidirectional way. This limitation is addressed by BiLSTMs, which stack two LSTMs to create a bidirectional encoder [20]. The BiLSTMs employ a simultaneous training strategy in both positive and negative directions. This is an ideal solution because the bidirectional network can access context from both the right and left directions in sequence labeling, leading to a better understanding of text sequences.

We propose a language-independent SGNWS model that incorporates neural model with a powerful BiLSTM encoder, Subword Representation Learning (SRL), and self-attention mechanism. We convert word segmentation into a sequence tagging problem. To the best of our knowledge, this is the first attempt to tackle SWS as a sequence labeling task. Our novel contributions are listed as follows:

• The proposed SGNWS model eliminates the need for highly complex, hand-engineered or rule-based word segmentation algorithms by adopting BiLSTM encoder, SRL, and self-attention mechanism.

II. RELATED WORK

The word segmentation has been well studied in English [3], [4], Arabic [5], Chinese [21], and many other languages. Little work has been carried out to address the word segmentation problem for Sindhi. Presently, Sindhi language is being written in two famous writing system of Persian-Arabic and Devanagari [22]. However, Persian-Arabic is the standard script of Sindhi [11], which contains rich morphology due to the frequent usage of prefixes and suffixes to express inflections and derivations, which makes it complex morphological language.

The existing SWS models rely on the rule-based approaches [10]. Initially, [12] coined the SWS problem by introducing the first word segmentation model using several rule-based algorithms. The proposed model was evaluated on a small dataset 16,601 lexicon with segmentation error rate (SER) of 9.54%. Later, [11] proposed another rule-based model with 91.76% segmentation accuracy. They performed the word segmentation in three steps: the first step consists of input and segmentation with white space, second step is used for the segmentation of simple and compound words, while the third step deals with the segmentation of complex words. Moreover, [1] proposed a SWS model by evaluating the dataset of 157K tokens obtained from news corpus and dictionary lexicon. Their proposed model achieves good performance on the dictionary lexicon, but poorly performed in dealing with news and books corpus. [13] proposed two algorithms for Sindhi lexicon lemmatization and stemming. The rule-based methods have certain limitations such as inability to deal with out of vocabulary words, less robustness for other datasets or languages, and the algorithms’ inefficiency to deal with a large amount of noisy or raw text.

Recently, neural sequence to sequence models including RNNs have largely gained popularity [23], [24] by greatly simplifying the learning and decoding process in a number of NLP applications including word segmentation [5], [25]. The BiLSTM [20] is an ideal solution to learn the sequences in a language because it has access to both the contexts of right and left directions. The BiLSTM network has been largely employed for word segmentation in Chinese [26], [27], Japanese [8] and Arabic [5], and other languages [28], [29], [30] by achieving excellent performance without relying on any external feature engineering strategies. On the one hand, state-of-the-art sequence tagging systems rely on large amounts of task-specific knowledge [17] in the form of hand-engineered features and data pre-processing. On the other hand, the performance of neural models rely on the incorporating unsupervised neural representation learning including classical [31], character-level [32], contextualized [33] and task-oriented [34]. More recently, the integration of self-attention mechanism [35] in neural
models has also yielded new state-of-the-art results word segmentation [21]. Moreover, the last layer of neural models has a significant impact on the performance. The CRF [36] is broadly used in the sequence tagging tasks [8], [17], [37] for decoding in neural models. Taking advantage of language-independent neural models for SWS, we propose a model that efficiently captures the character-level information with SRL by converting the segmentation into a sequence tagging problem.

III. SINDHI MORPHOLOGY
The Persian-Arabic is a standard writing script for Sindhi, which is cursive and written from right to left direction [38]. It contains rich morphology [11] due to the frequent usage of prefixes and suffixes to express inflections and derivations, which makes it a complex morphological language. The alphabet of Sindhi Persian-Arabic consists of 52 basic letters, 29 derived from Arabic language, 03 from the Persian language, and 20 modified letters [10]. It also uses 03 secondary letters, 07 honorific symbols and diacritic marks [38], [39]. Interestingly, the shape of some letters in Sindhi change the form according to their position in a word [1], such letters are referred as joiners. Thus, a joiner have at most four shapes; i) initial ii) middle iii) final and iv) isolated, as Table 2 depicts an example of some letters. Whereas the position-independent letters having final or isolated form are referred as non-joiners. Specifically, white spaces are used to detect word boundaries in Sindhi. However, writers omit a hard space between two words. Therefore, a phrase or a sentence that ends with non-joiner letters becomes one token. In the first case, the words are joined with their preceding and succeeding words in the absence of white space, which leads to misspellings. In the second case, the shape of characters remains identical even in the absence of white space. Due to position-independent and space-independent letters, the SWS exhibits both challenges of space insertion and space omission [12], [39].

| Ending | Middle | Initial | Isolated | Roman |
|--------|--------|---------|----------|-------|
| ب | م | ل | ن | Bz |
| ت | د | آ | ن | Td |
| س | ا | ن | | Sn |
| ژ | ن | ه | | Zn |
| گ | ا | ن | | Gn |
| ں | ن | ن | | Nn |

A. SPACE OMISSION
The space omission is a common phenomenon in Sindhi words that end with the non-joiner letters. However, the absence of white space exhibit the correct shape of words such as Table 4 shows an example of a Sindhi sentence with and without the use of white space. But computationally, that sentence consists of one token without the use of white spaces between words. Whereas the sentence consists of eight tokens with the use of white space between words. Therefore, the omission of white space between words ending with non-joiner letters raises a computational issue.

B. SPACE INSERTION
Another challenge in SWS arises when combining two or more root words (morphemes) form a new standalone single word (see Table 5). In such cases, writers omit white space if the first morpheme ends with a joiner letter. However, white space prevents it’s joining with the next morpheme so that the word retains a valid visual form. The missing space insertion leads to the formation of compound words and often misspelling. Hence, white space is essential in this case for the ease of readability and correct spelling of Sindhi words.

IV. THE PROPOSED MODEL
In this section, we describe the proposed SGNWS model in detail. Firstly, we introduce the tagging scheme and employ language-specific setting based on the morphological complexities, space-omission and space-insertion issues. We illustrate the neural structures of our SGNWS model, which includes three core parts: the encoder, self-attention, and decoder. Then, the training details of the proposed model are presented.

A. LABEL SET
We modeled the word segmentation as character-level sequence tagging [26]. Theoretically, word boundary can be predicted with binary classification in word segmentation, but in practice, fine-grained tag sets [40] produce high segmentation accuracy. Following the work [27], we employ four tags [B, I, E, S] to indicate the position of letters at the beginning [B], inside [I], ending [E] of a word, or a single-character/symbol [S], respectively. Additionally, [X] is used to represent the white space to delimit word boundaries. A sentence, as an example of the proposed tagging scheme is depicted in Table 6 by assigning the proposed tags to a sentence.

B. LANGUAGE-SPECIFIC PREPROCESSING
Based on the topological factors and baseline models, we employ the following language specific preprocessing steps.

1) Firstly we separate punctuation marks, then use space-delimited syllables for word boundary detection.
2) In order to tackle space omission problem, we use SRL based on concatenated trigram representations.
3) We split the large sentences with white-space if the length exceeds more than 300 tokens and do not consider sentences having tokens less than 5.
4) The regular hard-space is tagged as [X] in the dataset and punctuation marks are tagged as [S]. However,
TABLE 3. Complete list of Sindhi joiner and non-joiner letters, (i) denote joiner letters (ii) non-joiners, and (iii) non-joiner secondary letters.

| Joiner | Non-Joiner | Non-Joiner Secondary |
|--------|------------|----------------------|
| .      | ,          | (                   |
| )      | )          | )                   |
| !      | !          | !                   |
| ?      | ?          | ?                   |
| '      | '          | '                   |
| ”      | ”          | ”                   |

TABLE 4. An example of a Sindhi sentence, all words end with the non-jointer letters. (i) denote the words with white space (the tokens are separated with "," symbol), (ii) without white space (iii) Roman transliteration of Sindhi sentence (iv) is the english translation of a Sindhi sentence.

Ali ae Ahmed gadd Bazar ayya huwa. (i) 
Ali and Ahmed came to Bazar together. (ii)

C. ENCODER

The model takes character-level input \( x_t = c_1, c_2, c_3, \ldots, c_i \) of character unigrams \( c_i \), character bigrams \( c_i, c_{i+1} \), character trigrams \( c_i, c_{i+1}, c_{i+2} \), and 4-grams of each word \( w_n \) for SRL, as depicted in Table 7.

1) SUBWORD REPRESENTATION LAYER

We make the Subword representations from three types of discrete sources: character unigram \( c_i \), bigram \( c_i, c_{i+1} \), and trigram \( c_i, c_{i+1}, c_{i+2} \), \( (i \in [1, n]) \) in this form. Formally, we exploit three looking-up tables \( E^c, E^{bc}, E^{bhc} \) for character unigrams, bigrams, and trigrams. Give one character unigram \( c_i \), its representation \( E^c_i \) is obtained by indexing from \( E^c \), one character bigram \( c_i c_{i+1} \), its representation \( E^{bc} \), and one trigram \( c_i, c_{i+1}, c_{i+2} \) its representation \( E^{bhc} \) is obtained by indexing from \( E^{bhc} \) accordingly. The three looking-up tables are model parameters which would be learned during training.

2) LSTM INPUTS

In this layer, the BiLSTM encoder takes input of different neural features to the exploration of character-level unigram, bigram, and trigram based features as inputs. The input representations of both right-to-left and left-to-right at \( t \)th position always use the features ending at \( c_i \), where the last input trigrams right-to-left and the left-to-right LSTMs are \( c_i c_{i+1}, c_{i+1} \), and \( c_{i-1} c_{i-2}, c_i \), respectively. We employ a non-linear feed-forward neural layer to combine character-level unigrams, bigrams, and bigrams, formalized as follows:

\[
\bar{x}_t = \tanh \left( W_c \left[ E^c_i, E^{bc}_{i-1}, E^{bhc}_{i-2} \right] + b_c \right)
\]

\[
\bar{x}_t = \tanh \left( W_c \left[ E^c_i, E^{bc}_{i-1}, E^{bhc}_{i-2} \right] + b_c \right), \quad (1)
\]

3) BILSTM NETWORK

LSTM network [41] encodes the information in one direction \( \overrightarrow{h} \). However, the BiLSTM network [20] has the ability to encode the sequences from both right \( \overrightarrow{h} \) as well as left \( \overleftarrow{h} \) contexts. The BiLSTM network computes the forward \( \overrightarrow{h} \) and then backward \( \overleftarrow{h} \) hidden states. Finally, the context vector is created by concatenating both hidden states, \( \overrightarrow{h} \) and \( \overleftarrow{h} \), for the final output \( h_t = [ \overrightarrow{h_t}; \overleftarrow{h_t} ] \) of the network. The processing flow in the BiLSTM network can be expressed as

multi-word tokens without hard-space such as numerical expressions 689.0967, date 25-06-2020, money 4736$, etc., are assigned continuous tags.

FIGURE 1. The architecture of the proposed SGNWS model. The input sentences \( x_1, x_2, \ldots, x_n \) are converted into a sequence of character-level subword representations as input. The BiLSTM network learns how to obtain subword features for identifying word boundaries. The output of the BiLSTM encoder layer is fed into the self-attention layer before decoding. The notation \( Q = K = V = H \) signifies that the query, key, and value vectors used in the attention mechanism are all derived from the same hidden layer \( (H) \), obtained from the output of a BiLSTM encoder layer. Finally, we employ a CRF to obtain the predicted label sequence.
TABLE 5. Sindhi word types with an example of space insertion, along with English translation. (i) represent the words with white space (‘−’ symbol represents space), and (ii) without space. The Roman transliteration is given for ease of reading.

| Word Type | i. | ii. | Roman | English Translation |
|-----------|----|-----|-------|---------------------|
| Affix     | بحاس  | بحاس | Be-hisaa  | Uncountable |
| Reduplicate | هداي هداي  | هداي هداي | haidai hodai | Here and there |
| Compound  | صاحب قدرت  | saahib-o-Quadrat | Powerful |
| Borrowed  | موبائل فون  | Mobile phone | Mobile Phone |
| Abbreviation | اين سايچ  | اين سايچ | Ain Cee Aich Dee | NCHD |

TABLE 6. An example of employed character-level sequence tagging scheme for SWS task. The [X] label represents the white spaces. The given Sindhi sentence can be read from right to left, and the Roman transliteration of each Sindhi token can be read from left to right.

| Roman Transliteration | ahai | boli | qadeemi | Sindhi |
|-----------------------|------|------|---------|--------|
| Sindhi Persian-Arabic  | س ن قی ک د وی ب و لی آ ط ی | نمی | قیمتی | بولی آطی |
| Character Sequence     | S E I B X E I I B X E I I B |

TABLE 7. An example of Sindhi subword decomposition for subword representation learning.

| Representation | Decomposition |
|----------------|---------------|
| Word           | سنیتندی |
| Characters     | سنیتندی |
| Bigrams        | سنیتندی |
| Trigrams       | سنیتندی |

follows:

\[
\begin{bmatrix}
\tilde{c}_t \\
\tilde{f}_t \\
\tilde{o}_t \\
\end{bmatrix} = \begin{bmatrix}
\sigma \\
\sigma \\
\tanh
\end{bmatrix} \begin{bmatrix}
W^T x_t \\
h_{t-1}
\end{bmatrix} + b \tag{2}
\]

\[
c_t = i_t \odot \tilde{c}_t + f_t \odot c_{t-1} \tag{3}
\]

\[
h_t = o_t \odot \tanh(c_t) \tag{4}
\]

where \(i_t, f_t, \text{ and } o_t\) indicate the input, forget, and output gates; \(\sigma()\) represents the sigmoid function; \(W^T\) and \(b\) are the trainable parameters; \(x_t\) is an input vector of the current time step; \(\odot\) represents the dot product function.

D. POSITION-AWARE SELF-ATTENTION

The limitation of an encoder-decoder model is their fixed-length connection [42] of the intermediate semantic vector. Hence, the encoder compresses the whole sequence of information into a fixed-length vector. Recently, self-attention [35] has been widely used in word segmentation [21], [43] to tackle the limitations of encoder-decoder structure to extract as much feature information as possible from the input sequences. We introduce a token-level position-aware multi-head attention to the output \(H = [h_1, h_2, h_3, \ldots, h_n]\) of concatenated BiLSTM states to learn the character-level structure of words and capture token-level dependency. The self-attention layer [35] before the CRF layer [44] automatically focuses on the specific Sindhi tokens that play a decisive role in word segmentation. We use the position-aware scaled dot product to calculate the similarity between query \(Q\) and key \(K\) matrices to obtain weights of each word. Then, normalize the obtained score and calculate the weights using softmax. Afterwards, obtain the final attention by a weighted sum of value matrix \(V\) and the weights. The intuition behind position-aware attention mechanism is described as follows:

\[
\text{attn}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V \tag{5}
\]

where \(Q \in \mathbb{R}^{n \times 2d_h}, K \in \mathbb{R}^{n \times 2d_h}, V \in \mathbb{R}^{n \times 2d_h}\), are the query, key and value matrices. \(d_h\) is the dimension of both \(\overrightarrow{h}\) and \(\overleftarrow{h}\) hidden units of BiLSTM, which equals to \(2d_h\). Firstly, the attention mechanism linearly maps \(Q, K\) and \(V\) in \(h\) times with different weights matrices. Then \(\overrightarrow{h}\) projections parallelly perform scaled dot-product attention. Finally, the results of an attention layer are concatenated and once again mapped to get the new representations. In this paper, we use self-attention as a set \(Q = K = V = H\), aims to capture the long dependencies between any two tokens in the input sentence, where \(H\) represents the output of the BiLSTM. Formally, the function can be expressed as under:

\[
h_i = \text{attn}\left(Q\omega_i^Q, K\omega_i^K, V\omega_i^V\right) \tag{6}
\]

\[
H' = (h_i \parallel \ldots \parallel \text{head}_b)w_o \tag{7}
\]
where \( w \tilde{w} \in \mathbb{R}^{2d_k \times d_k} \), \( w \tilde{w} \in \mathbb{R}^{2d_h \times d_k} \), \( w' \tilde{w}' \in \mathbb{R}^{2d_k \times d_k} \) are the trainable projection parameters and \( d_k = 2d_h / h \). \( w_a \in \mathbb{R}^{2d_h \times 2d_h} \) are the trainable parameters.

**E. OUTPUT LAYER**

The decoder finds a next-action step conditioned [45] on historical actions. The overall decoding framework of our SGNS model is shown by the upper part of Figure 1. Following sequence tagging models such as [45] and [46]. The generated output from the self-attention layer fed to CRF for prediction. Once all \( a_t \) are generated from the network for each \( x_t \), a decoder takes them to predict a sequence of segmentation labels \( \hat{y} = \hat{y}_1 \hat{y}_2 \cdots \hat{y}_t \) for \( x \) as,

\[
\hat{y} = \text{Decoder}(A)
\]

here \( A = a_1a_2 \cdots a_t \) is output sequence. The decoder can be implemented by using softmax or CRF. The softmax can be illustrated as:

\[
\hat{y}_i = \arg \max_{y_i \in T} \frac{\exp (a_i)}{\sum_{j=1}^{T} \exp (a_j)}
\]

where \( a_i \) is the value at dimension \( t \) in \( a_t \).

The CRF decoder can be illustrated as:

\[
\hat{y}_i = \arg \max_{y_i \in T} \frac{\exp (W_c \cdot a_i + b_c)}{\sum_{y_{i-1} \in \mathcal{Y}} \exp (W_c \cdot a_i) + b_c}
\]

where \( W_c \in \mathbb{R}^{1 \times |T \times T|} \) and \( b_c \in \mathbb{R}^{1 \times T} \) are trainable parameters to model the transition for \( y_{i-1} \) to \( y_i \).

**F. TRANSDUCTION**

The non-segmental multiword tokens identified by the main network are transduced into corresponding components in an additional step. We transduce the identified non-segmental multiword tokens in a context-free fashion. For multiword tokens with two or more valid transductions, we only adopt the most frequent one. We use a generalizing approach to process non-segmental multiword tokens. If there are more than 200 unique multiword tokens in the training set, we train a self-attention-based neural encoder-decoder model equipped with shared BiLSTM. Overall, we utilize context to identify non-segmental multiword tokens and then apply a combination of dictionary and sequence-to-sequence encoder-decoder to transude them.

**V. EXPERIMENTAL SETUP**

This section provides details about the experimental setting of baselines as well as proposed SGNS models. We use TensorFlow [47] deep learning framework for the implementation of all neural models on GTX 1080-TITAN GPU to run all the experiments.

**A. DATASETS**

We utilize the recently proposed unlabeled Sindhi text corpus [48] in the experimental setting. We convert segmentation into a sequence tagging problem using B, I, E, S, X tagging scheme and split the dataset into 80% for training and 20% for development and test sets. Table 8 shows the complete statistics of the datasets.

**TABLE 8. Statistics of the proposed training datasets used in the experiments are presented in the table below. Sent., Tok., UW, and AWL denote the number of sentences, tokens, unique words, and average word length in each dataset, respectively. We concatenate all the domains and represent the combined dataset as SDSEG.**

| Datasets | Sent | Tok | UW | AWL |
|----------|------|-----|----|-----|
| KN       | 24,212 | 601,910 | 10,721 | 3.687 |
| AA       | 19,736 | 521,257 | 14,699 | 3.660 |
| WK       | 14,557 | 669,623 | 11,830 | 3.738 |
| Tw       | 10,752 | 159,130 | 17,379 | 3.820 |
| Bk       | 22,496 | 430,923 | 16,127 | 3.684 |
| SDSEG    | 91,753 | 2,382,843 | 70,737 | 3.717 |

**B. BASELINE MODELS**

We conduct several baseline experiments by training the LSTM, BiLSTM using softmax and CRF decoders. We train character-level [27] subword representations in baseline experiments to analyze and compare the performance of proposed model. The brief description of each baseline model is described as follows:

1) **BiLSTM**: Our first baseline utilizes the BiLSTM encoder, which is leveraged with character-level representations, bigrams, and SRL. We employ a softmax classifier in the last layer of the network for decoding tag sequences. The BiLSTM benefits from encoding both forward and backward sequences, efficiently capturing word information at the morphemic level.

2) **BiLSTM-CRF**: The second baseline model is built upon a BiLSTM-CRF network with a hyperparameter setting similar to that of the BiLSTM encoder. We utilize character-level representations, bigrams, and SRL. CRF inference is employed in the last layer of the network for decoding purposes.

**C. EVALUATION METRICS**

We report word boundary precision, recall, and F1-scores [49]. Precision measures the proportion of correctly predicted word boundaries among all predicted boundaries. Recall measures the proportion of correctly predicted boundaries compared to the actual number of boundaries in the text. F1-score, the harmonic mean of precision and recall, provides a balanced view of these two metrics.

\[
\text{Precision} = \frac{\text{#(correctly predicted tags)}}{\text{#(predicted tags)}} \times 100\% \quad (11)
\]

\[
\text{Recall} = \frac{\text{#(correctly predicted tags)}}{\text{#(true tags)}} \times 100\% \quad (12)
\]

\[
F_1\text{-score} = \frac{2 \times (\text{Precision} \times \text{Recall})}{\text{Precision + Recall}} \times 100\% \quad (13)
\]

**D. PARAMETER SETTING AND TRAINING**

The training procedure involves updating the network’s parameters based on the training data. We train the baselines
and proposed model using the log-likelihood function. The log-likelihood give strong performances [50] compared to global learning [51] to maximize F1-score. We split the dataset into training, development, and test sets. Variational dropout [52] of 0.25% is employed to recurrent units [53]. The softmax is used for label classification in baseline models, CRF is added in the last layer of the BiLSTM-CRF and SGNWS models. The Gradient normalization is used to improve the performance [54], which re-scales the gradient when the norm goes over a threshold. We use BiLSTM state size of 200 layers, and character embedding size of 64. We train the models for 40 epochs, and use the Nadam optimizer [55] with initial learning rate of 0.01. We use beam decoding with beam size of 5. For each configuration, we trained and evaluated different settings of a manually tuned hyperparameters grid, varying initial learning rate, and input and recurrent dropout rates. All the hyper-parameters for baseline models and SGNWS are kept similar for performance difference and fair comparison. Our SGNWS is implemented by grouping sentences of similar lengths into the same bucket and padding them to the same length. We construct sub-computational graphs for each bucket so that sentences of different lengths are processed more efficiently [49].

### TABLE 9. Optimal hyper-parameters for SRL, baselines, and proposed SGNWS model.

| Hyperparameter                        | Range  |
|---------------------------------------|--------|
| SRL                                   |        |
| E<sup>0</sup> representation dimension| 64     |
| E<sup>1</sup> representation dimension| 64     |
| E<sup>2</sup> representation dimension| 64     |
| E<sup>3</sup> representation dimension| 64     |
| E<sup>4</sup> representation dimension| 64     |
| Window size                           | 5      |
| Subword representation dimension      | 100    |
| Epochs                                | 100    |
| Neural models                         |        |
| Optimizer                             | Nadam  |
| Initial learning rate                 | 0.01   |
| Gradient normalization                | 5.0    |
| BiLSTM state size                    | 200    |
| Dropout                               | 0.25   |
| Decay rate                            | 0.05   |
| Batch size                            | 32     |
| Epochs                                | 40     |

### VI. RESULTS AND ANALYSIS

In this section, we firstly report the results of baselines with different configurations on SGSSEG benchmark dataset. Then, we describe the validity of the proposed on different datasets as well as SDSEG on the same configurations.

### A. BASELINE RESULTS

We investigate the capability of baseline neural models to handle words of different lengths. The table presents the results, demonstrating that the best performances are achieved when the n-gram length is three under all settings. We utilize the BiLSTM encoder with softmax and CRF decoder. The empirical results on the SDSEG dataset are showcased, reporting overall precisions, recalls, and F1-scores. Several observations can be drawn from the results. Table 10 illustrates the performance comparison of all models on the SDSEG dataset. Firstly, the LSTM achieves a stable baseline F-Score of 95.29% on the development and 94.32% on the test set. The BiLSTM model outperforms both baselines in both the development and test sets due to its bidirectional learning capabilities. However, the BiLSTM-CRF outperforms both BiLSTM baselines, indicating the dominance of CRF over the softmax classifier. Moreover, adding character-level features to the BiLSTM-CRF model surpasses all three baselines. However, BiLSTM-CRF with bigram and trigram-based subword representations yield results close to those of the BiLSTM-CRF+Char model, highlighting the superiority of the character-level approach and the performance gain it provides. We have observed the effects of employing various subword representations.

### TABLE 10. Neural baselines for Sindhi word segmentation on the SDSEG test set. P%, R%, F1% denote the Precision, Recall, and F1-score.

| Model                     | P%     | R%     | F1%    |
|---------------------------|--------|--------|--------|
| BiLSTM-Char               | 96.52  | 94.28  | 95.74  |
| BiLSTM-Char+bigram        | 96.42  | 94.28  | 95.87  |
| BiLSTM-SRL                | 96.81  | 97.74  | 96.53  |
| BiLSTM-CRF-Char           | 96.11  | 95.87  | 96.23  |
| BiLSTM-CRF-Char+bigram    | 96.82  | 97.26  | 96.78  |
| BiLSTM-CRF-SRL            | 97.17  | 97.23  | 97.39  |

### B. FINAL RESULTS

The proposed SGNWS model has yielded superior results compared to the baselines, as shown in Table 10 for test data. According to the results, SRL proves to be beneficial for the word segmentation task in Sindhi language. The proposed SGNWS model outperforms all the baselines on the SDSEG dataset as well as on five different datasets: Kawish newspaper (KW), Awami-Awaz newspaper (AA), Wikipedia dumps (WK), Twitter (Tw), and books (Bk). Moreover, Figure 2 illustrates the training accuracy and loss of the SGNWS model, alongside other models, on the SDSEG dataset. Our proposed SGNWS model surpasses the baselines with a high F1-score of 98.51% on the test set using the SDSEG dataset. This observation indicates that SRL is beneficial in capturing more word boundaries and semantic information for the word segmentation of Sindhi text. The SGNWS model achieves state-of-the-art performance on KW and SDSEG datasets.

Table 11 presents the final results on the test datasets of Kawish, Awami-Awaz, Wikipedia dumps, Twitter, Books, and SDSEG. We present the performances of the SGNWS neural model by utilizing subword representation learning and self-attention. According to the results, it is evident that the model achieved the best performance on the SGSSEG dataset, followed by the Kawish and Books datasets. We compare the SGNWS model with the discrete baseline system under a purely supervised setting. The BiLSTM-CRF-CRL...
model achieves similar performances on the SDSEG dataset compared to the SGNWS model.

### TABLE 11. Evaluation of the SGNWS model on various datasets. Bold font denote the best results.

| Domain         | P%  | R%  | F1%  |
|----------------|-----|-----|------|
| Kawish News (KN)| 98.56 | 98.34 | 98.12 |
| Awami Awaz (AA)| 98.17 | 97.67 | 97.90 |
| Wiki-dumps (WK)| 97.25 | 96.84 | 96.37 |
| Twitter (Tw)   | 97.41 | 97.29 | 97.24 |
| Books (BK)     | 97.57 | 98.28 | 97.83 |
| SDSEG          | 98.48 | 98.64 | 98.51 |

### VII. CONCLUSION

Word segmentation is an essential and non-trivial task in the Sindhi language. While white spaces between words serve as indicators for predicting word boundaries, the presence of space omission and insertion introduces ambiguity into the segmentation process. In this paper, we introduced the SGNWS model, specifically designed to address the challenges associated with Sindhi word segmentation. Our proposed model possesses the capability to automatically learn and extract subword features with minimal language-specific settings. The SGNWS model achieves superior performance in SWS due to its high efficiency and robustness in sequential modeling tasks, enabling it to effectively capture word information at the morphemic level for predicting word boundaries. Empirical results across various unlabeled datasets demonstrate the effectiveness of SGNWS, showcasing state-of-the-art performance across all datasets. Our proposed model serves as an effective and elegant neural solution for Sindhi word segmentation, with potential applicability to other sequence tagging problems.

### REFERENCES

[1] Z. Bhatti, I. A. Ismaili, W. J. Soomro, and D. N. Hakro, “Word segmentation model for Sindhi text,” *Amer. J. Comput. Res. Repository*, vol. 2, no. 1, pp. 1–7, 2014.

[2] C. Ding, Y. K. Thu, M. Utiyama, and E. Sumita, “Word segmentation for Burmese (Myanmar),” *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, vol. 15, no. 4, pp. 1–10, Jun. 2016.

[3] L. Polka and M. Sundara, “Word segmentation in monolingual infants acquiring Canadian English and Canadian French: Native language, cross-dialect, and cross-language comparisons,” *Infancy*, vol. 17, no. 2, pp. 198–232, Mar. 2012.

[4] L. Wang, Z. Cao, Y. Xia, and G. De Melo, “Morphological segmentation with window LSTM neural networks,” in *Proc. AAAI Conf. Artif. Intell.*, 2016, vol. 30, no. 1, pp. 2842–2848.

[5] A. Almuhareb, W. Alsanie, and A. Al-Thubaity, “Arabic word segmentation with long short-term memory neural networks and word embedding,” *IEEE Access*, vol. 7, pp. 12879–12887, 2019.

[6] H. B. Zia, A. A. Raza, and A. Athar, “Urdu word segmentation using conditional random fields (CRFs),” in *Proc. 27th Int. Conf. Comput. Linguistics*, 2018, pp. 2562–2569.

[7] L. Li, Y. Dai, D. Tang, X. Qiu, Z. Xu, and S. Shi, “MarkBERT: Marking word boundaries improves Chinese BERT,” in *Proc. CCF Int. Conf. Natural Lang. Process. Chin. Comput.*, China: Springer, 2023, pp. 325–336.

[8] Y. Kitagawa and M. Komachi, “Long short-term memory for Japanese word segmentation,” in *Proc. 32nd Pacific Asia Conf. Lang., Inf. Comput.*, 2018, pp. 279–288.

[9] S. Zhang, C. Mao, Z. Yu, H. Wang, Z. Li, and J. Zhang, “Word segmentation for Burmese based on dual-layer CRFs,” *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, vol. 18, no. 1, pp. 1–11, Mar. 2019.

[10] W. A. Janro, “Sindhi language processing: A survey,” in *Proc. Int. Conf. Innov. Electr. Eng. Comput. Technol.* (ICEEECT), Apr. 2017, pp. 1–8.

[11] J. Mahar, H. Shaikh, and G. Memon, “A model for Sindhi text segmentation into word tokens,” *Sindh Univ. Res. J., Sci. Ser.*, vol. 44, no. 1, pp. 43–48, 2012.

[12] J. A. Mahar, G. Q. Memon, and S. H. Danwar, “Algorithms for Sindhi word segmentation using lexicon-driven approach,” *Int. J. Academic Res.*, vol. 3, no. 3, 2011.

[13] M. A. Dootio and A. I. Wagan, “Automatic stemming and lemmatization process for Sindhi text,” *J. Social Sci. Interdiscipl. Res.*, vol. 6, no. 2, pp. 19–28, 2017.

[14] F. Zaman, O. Maqbool, and J. Kanwal, “Leveraging bidirectional LSTM with CRFs for pashito tagging,” *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, vol. 23, no. 4, pp. 1–17, Apr. 2024.
B. S. Al-Smadi, “DeBERTa-BiLSTM: A multi-label classification model of Arabic medical questions using pre-trained models and deep learning,” Comput. Biol. Med., vol. 170, Mar. 2024, Art. no. 107921.

A. Murat and S. Ali, “Low-resource POS tagging with deep affix representation and multi-head attention,” IEEE Access, vol. 12, pp. 66495–66504, 2024.

X. Ma and E. Hovy, “End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF,” in Proc. 54th Annu. Meet. Assoc. Comput. Linguistics, 2016, pp. 1064–1074.

P. Rai, S. Chatterji, and B.-G. Kim, “Deep learning-based sequence labeling tools for nepali,” ACM Trans. Asian Low-Resou. Lang. Inf. Process., vol. 22, no. 8, pp. 1–23, Aug. 2023.

H. Kiliç Ünlü and A. Çetin, “Keyword extraction as sequence labeling with classification algorithms,” Neural Comput. Appl., vol. 35, no. 4, pp. 3413–3422, Feb. 2023.

M. Schuster and K. K. Paliwal, “Bidirectional recurrent neural networks,” IEEE Trans. Signal Process., vol. 45, no. 11, pp. 2673–2681, Nov. 1997.

S. Duan and H. Zhao, “Attention is all you need for Chinese word segmentation,” in Proc. Conf. Empirical Methods Natlang. Process. (EMNLP), 2020, pp. 3862–3872.

R. Motlani, “Developing language technology tools and resources for a resource-poor language: Sindhi,” in Proc. NAACL Student Res. Workshop, 2016, pp. 51–58.

I. Sebová and K. Grošlager, “Deep learning for load forecasting: Sequence to sequence recurrent neural networks with attention,” IEEE Access, vol. 8, pp. 36411–36426, 2020.

J. C.-W. Lin, Y. Shao, Y. Djenouri, and U. Yun, “ASRN: A recurrent neural network with an attention model for sequence labeling,” Knowl.-Based Syst., vol. 212, Jan. 2021, Art. no. 106548.

L. Gan and Y. Zhang, “Investigating self-attention for Chinese word segmentation,” IEEE/ACM Trans. Audio, Speech, Language Process., vol. 28, pp. 2933–2941, 2020.

X. Chen, X. Qiu, C. Zhu, P. Liu, and X. Huang, “Long short-term memory neural networks for Chinese word segmentation,” in Proc. Conf. Empirical Methods Natlang. Process., 2015, pp. 1197–1206.

Y. Shao, C. Hardmeier, J. Tiedemann, and J. Nivre, “Character-based joint segmentation and POS tagging for Chinese using bidirectional RNN-CRF,” in Proc. 8th Int. Joint Conf. Natlang. Process., 2017, pp. 173–183.

J. Yang, Y. Zhang, and F. Dong, “Neural word segmentation with rich pretraining,” in Proc. 55th Annu. Meet. Assoc. Comput. Linguistics, 2017, pp. 839–849.

Y. Yao and Z. Huang, “Bi-directional LSTM recurrent neural network for Chinese word segmentation,” in Proc. Int. Conf. Natlang. Inf. Process., 2016, pp. 345–353.

J. Ma, K. Ganchev, and D. Weiss, “State-of-the-art Chinese word segmentation with bi-LSTMs,” in Proc. Conf. Empirical Methods Natlang. Process., 2018, pp. 4902–4908.

T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” in Proc. Adv. Natlang Inf. Process. Syst., 2013, pp. 3111–3119.

P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, “Enriching word vectors with subword information,” Trans. Assoc. Comput. Linguistics, vol. 5, pp. 135–146, Dec. 2017.

A. Ahbik, D. Blythe, and R. Vollgraf, “Contextual string embeddings for sequence labeling,” in Proc. 27th Int. Conf. Comput. Linguistics, 2018, pp. 1638–1649.

Q. Liu, H.-Y. Huang, Y. Gao, X. Wei, Y. Tian, and L. Liu, “Task-oriented word embedding for text classification,” in Proc. 27th Int. Conf. Comput. Linguistics, 2018, pp. 2023–2032.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” in Proc. Adv. Natlang Inf. Process. Syst., 2017, pp. 5998–6008.

J. D. Lafferty, A. McCallum, and F. C. Pereira, “Conditional random fields: Probabilistic models for segmenting and labeling sequence data,” in Proc. 18th Int. Conf. Mach. Learn., 2001, pp. 282–289.

V. S. R. Bhuniredypalli, S. R. Koppula, and N. Koppula, “Enhanced conditional random field: Short-term memory for name entity recognition in English texts,” Concurrency Comput., Pract. Exper., vol. 35, no. 9, 2023, Art. no. e7640.

W. A. Narejo and J. A. Mahar, “Morphology: Sindhi morphological analysis for natural language processing applications,” in Proc. Int. Conf. Comput., Electron. Electr. Eng. (ICE Cube), Apr. 2016, pp. 27–31.

M. U. Rahman, “Towards Sindhi corpus construction,” in Proc. Conf. Lang. Technol., 2010, pp. 39–48.

H. Zhao, C. Huang, M. Li, and B.-L. Lu, “Effective tag set selection in Chinese word segmentation via conditional random field modeling,” in Proc. 20th Pacific Asia Conf. Lang., Inf. Comput., 2006, pp. 87–94.

S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural Comput., vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

M. Liao, J. Zhang, Z. Wan, F. Xie, J. Liang, P. Lu, C. Yao, and X. Bai, “Scene text recognition from two-dimensional perspective,” in Proc. AAAI Conf. Artif. Intell., 2019, vol. 33, no. 1, pp. 8714–8721.

P. Cao, Y. Chen, K. Liu, J. Zhao, and S. Liu, “Adversarial transfer learning for Chinese named entity recognition with self-attention mechanism,” in Proc. Conf. Empirical Methods Natlang. Process., 2018, pp. 182–192.

Y. Jia and X. Ma, “Attention in character-based BiLSTM-CRF for Chinese named entity recognition,” in Proc. 4th Int. Conf. Math. Artif. Intell., Apr. 2019, pp. 1–4.

M. Zhang, N. Yu, and G. Fu, “A simple and effective neural model for joint word segmentation and POS tagging,” IEEE/ACM Trans. Audio, Speech, Language Process., vol. 26, no. 9, pp. 1528–1538, Sep. 2018.

Z. Huang, W. Xu, and K. Yu, “Bidirectional LSTM-CRF models for sequence tagging,” 2015, arXiv:1508.01991.

M. Abadi et al., “TensorFlow: A system for large-scale machine learning,” in Proc. 12th USENIX Symp. Operating Syst. Design Implement. (OSDI), 2016, pp. 265–283.

W. Ali, J. Kumar, J. Lu, and Z. Xu, “Word embedding based new corpus for low-resource language: Sindhi,” 2019, arXiv:1911.12579.

Y. Shao, “Cross-lingual word segmentation and morpheme segmentation as sequence labelling,” in Proc. 1st Workshop Multi-Lang. Process. Globalising World, 2017, pp. 75–80.

M. J. van der Laan and S. Rose, “Why machine learning cannot ignore maximum likelihood estimation,” in Handbook of Matching and Weighting Adjustments for Causal Inference. Boca Raton, FL, USA: CRC Press, 2017, pp. 483–500.

D. Andor, C. Alberti, D. Weiss, A. Severin, A. Presta, K. Ganchev, S. Petrov, and M. Collins, “ Globally normalized transition-based neural networks,” in Proc. 54th Annu. Meet. Assoc. Comput. Linguistics, 2016, pp. 2442–2452.

Y. Yang, J. Huang, and D. Hu, “Sparsely dynamically expandable network via variational dropout,” Appl. Soft Comput., vol. 160, Jul. 2024, Art. no. 111705.

I. Salehin and D.-K. Kang, “A review on dropout regularization approaches for deep neural networks within the scholarly domain,” Electronics, vol. 12, no. 14, p. 3106, Jul. 2023.

L. Huang, J. Qin, Y. Zhou, F. Zhu, L. Liu, and L. Shao, “Normalization techniques in training DNNs: Methodology, analysis and application,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 45, no. 8, pp. 10173–10196, Aug. 2023.

D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in Proc. 3rd Int. Conf. Learn. Represent., 2015.

WAZIR ALI received the Ph.D. degree in computer science and engineering from the University of Electronic Science and Technology of China (UESTC). During the course of the Ph.D. degree, he has outstanding research experience. Since 2022, he has been an Assistant Professor with the Computer Science Department, Institute of Business Management, Karachi, Pakistan. His research interests include language models, information extraction, text classification, representation learning, and opinion mining. His research papers have been published in scientific journals and top natural language processing (NLP) conferences, including the Association of Computational Linguistics (ACL), the Language Resources and Evaluation Conference (LREC), and the Workshop on Computational Approaches To Subjectivity, Sentiment and Social Media Analysis (WASSA). One of his papers was nominated for the Best Paper Award in the Student Research Workshop Associated with Recent Advances in Natural Language Processing (RANLP-2021). He has been serving as part of the Scientific Committee at WASSA and Insights Conferences, since 2022. He is also serving as a reviewer for various scientific journals in the area of natural language processing.

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply. 185141
JAY KUMAR received the M.Phil. degree in computer science from Quaid-i-Azam University, Islamabad, Pakistan, in 2018, and the Ph.D. degree in computer science and technology from the School of Computer Science and Engineering, University of Electronics Science and Technology of China, Chengdu, China, in 2021. He is currently a Postdoctoral Fellow with Dalhousie University, Halifax, NS, Canada. His research interests include data stream mining, natural language processing, and spatiotemporal data mining.

SAIFULLAH TUMANI received the Ph.D. degree in computer science from the University of Electronic Science and Technology of China, with a focus on vehicle re-identification for advanced monitoring systems. He is currently a Postdoctoral Researcher of bioinformatics with Heidelberg University, Germany. His research interests include the intersection of machine learning, computer vision, and bioinformatics, with a focus on developing innovative solutions for complex problems. With a strong background in computer science and engineering, he continues to explore the applications of artificial intelligence and machine learning in diverse fields, aiming to bridge the gap between technology and real-world challenges.

REDHWAN NOUR received the Ph.D. degree in computer science from the University of Colorado Boulder. He is currently an Associate Professor with the College of Computer Science and Engineering, Taibah University. His research interests include cognitive web access, human-AI interaction, inclusive design, and machine learning.

ADEEB NOOR received the B.S. degree in information technology from Taibah University, Saudi Arabia, in 2005, and the M.S. degree in computer science and the Ph.D. degree in bioinformatics from the University of Colorado Boulder, in 2010 and 2015, respectively. He is currently a Full Professor with the Faculty of Computing and Information Technology, King Abdulaziz University, where he has also served as a Consultant to the Vice President of Development. He is also a Visiting Professor with the Computational Bioscience Research Center (CBRC), KAUST. Prior to his current role, he was the IT Director of the King Abdulaziz University Hospital, where he received several awards, including the prestigious Jeddah Award for Governmental Excellence. He has also founded two startups. His research interests include healthcare digital transformation, drug-drug interactions, and personalized medicine.

ZENGLIN XU (Senior Member, IEEE) received the Ph.D. degree in computer science and engineering from The Chinese University of Hong Kong, Hong Kong, in 2009. He is currently a Full Professor with Harbin Institute of Technology, Shenzhen, China. He has been with Michigan State University, East Lansing, MI, USA; Cluster of Excellence, Saarland University and Max Planck Institute for Informatics, Saarbrücken, Germany; Purdue University, West Lafayette, IN, USA; and the University of Electronic Science and Technology of China, Chengdu, China. His research interests include machine learning and its applications in computer vision, health informatics, and natural language processing. He was a recipient of the Outstanding Student Paper Honorable Mention of AAAI 2015, the Best Student Paper Runner-Up of ACML 2016, and the 2016 Young Researcher Award from Asia Pacific Neural Network Society (APNNS).