information

Article

Uyghur–Kazakh–Kirghiz Text Keyword Extraction Based on Morpheme Segmentation

Sardar Parhat [1], Mutallip Sattar [1], Askar Hamdulla [2] and Abdurahman Kadir [1,*]

[1] College of Information Management, Xinjiang University of Finance and Economics, Urumqi 830012, China
[2] College of Information Science and Engineering, Xinjiang University, Urumqi 830046, China
* Correspondence: ar@xjufe.edu.cn

Abstract: In this study, based on a morpheme segmentation framework, we researched a text keyword extraction method for Uyghur, Kazakh and Kirghiz languages, which have similar grammatical and lexical structures. In these languages, affixes and a stem are joined together to form a word. A stem is a word particle with a notional meaning, while the affixes perform grammatical functions. Because of these derivative properties, the vocabularies used for these languages are huge. Therefore, preprocessing is a necessary step in NLP tasks for Uyghur, Kazakh and Kirghiz. Morpheme segmentation enabled us to remove the suffixes as the auxiliary unit while retaining the meaningful stem and it reduced the dimension of the feature space present in the keyword extraction task for Uyghur, Kazakh and Kirghiz texts. We transformed the morpheme segmentation task into the problem of labeling the morpheme sequences, and we used the Bi-LSTM network to bidirectionally obtain the position feature information of character sequences. We applied CRF to effectively learn the information of the preceding and following label sequences to build a highly accurate Bi-LSTM_CRF morpheme segmentation model, and we prepared morpheme-based experimental text sets by using this model. Subsequently, we used the stem vectors' similarity to modify the TextRank algorithm, subsequent to the training of the stem embedding vector using the Doc2vec algorithm, and then we performed a text keyword extraction experiment. In this experiment, the highest F1 scores of 43.8%, 44% and 43.9% were obtained for three datasets. The experimental results show that the morpheme-based approach provides much better results than the word-based approach, which shows the stem vector similarity weighting is an efficient method for the text keyword extraction task, thus proving the efficiency of morpheme sequence for morphologically derivative languages.

Keywords: Uyghur–Kazakh–Kirghiz; keyword extraction; morpheme segmentation; stem extraction; stem vector; TextRank

Citation: Parhat, S.; Sattar, M.; Hamdulla, A.; Kadir, A. Uyghur–Kazakh–Kirghiz Text Keyword Extraction Based on Morpheme Segmentation. *Information* **2023**, *14*, 283. https://doi.org/10.3390/info14050283

Received: 13 February 2023
Revised: 28 April 2023
Accepted: 29 April 2023
Published: 10 May 2023

1. Introduction

Keywords are important phrases with a semantic meaning in a text. By extracting the keywords, the most representative lexical units in the text can automatically be identified. Therefore, the text keyword extraction method has important applications in text mining [1], information retrieval [2] and natural language processing [3]. In text searching tasks, keywords are extensively used to classify the search results and help users to rapidly obtain specific data. However, automatic keyword extraction is a challenging task because of the complexity of natural language, the heterogeneity of input text type and the different types of keywords that need to be extracted. An effective representation and retention of the semantic context information are important steps in the text keyword extraction process.

Uyghur, Kazakh and Kirghiz are morphologically rich, agglutinative languages sharing similar grammar and lexical structures. In the written form, officially, Arabic script is used for these three languages; at the same time, people also extensively use the Latin alphabet for mobile messages or on social networks such as WeChat. Sentences in these languages are composed of naturally separated words, which are formed by affixes attached to

the stem. A stem is a unit with independent semantics and is part of an open set, and affixes are the functional units and a closed set. The function of affixes is very important. They can change the meaning of a stem and derive new stems (vocabularies). As a result of these derivative properties, these languages have many combinations of morphemes, and greatly expand the available vocabulary. Therefore, through pre-processing operations, such as morpheme segmentation and word stem extraction, the meaningful and effective features of Uyghur, Kazakh and Kirghiz texts can be retained, thus, different forms of words can be merged into the same feature space, which can effectively reduce the repetition rate and dimension of features. This notion is presented in the example below:

Arabic script form.

دوختۇرلاردا دوختۇرغا خاس دوختۇرلۇق ئەخلاقى بولىشى كېرەك. (Uyghur)

دارىگەرلەر دارىگەرگە تيىپسلى دارىگەرلىك ٴتٸتكاگى بولسى كەرەك. (Kazakh)

دارىگەرلەر دارىگەرگە كەرەكتىك دارىگەرلىك ئەخلاكسن ساكتاسى كەرەك. (Kirghiz)

English meaning: Doctors must have the medical ethics.

Latin alphabet form.

(Uyghur) doHturlarda doHturGa Has doHturluq vAHlaqi bolixi kerAk.

(Kazakh) darigerler darigerge tiesili darigerlik vetikagi bolsi kerek.

(Kirghiz) darigArlAr darigArgA kArAklik darigArlik AHlakin saktasi kArAk.

After morpheme segmentation.

(Uyghur) **doHtur** + lar + da **doHtur** + Ga Has **doHtur** + luq vAHlaq + i bolix + i kerAk.

(Kazakh) **dariger** + ler **dariger** + ge tiesili **dariger** + lik vetikagi bolsi kerek.

(Kirghiz) **darigAr** + lAr **darigAr** + gA kArAk + lik **darigAr** + lik AHlak + in saktas + i kArAk.

The three sentences above contain seven words, and each sentence has three words which share the same stem (shown in), which are /doHtur/, /dariger/ and darigAr/ (English meaning: doctor). Subsequent to morpheme segmentation and stem extraction operations on these sentences, these three words grouped to only one stem; therefore, the dimensionality of the text feature space is considerably reduced, as shown in Table 1.

Table 1. Uyghur–Kazakh–Kirghiz word variants.

Stem	Variants	Affix
(doctor) Uyghur:doHtur Kazakh:dariger Kirghiz:darigAr	(on the doctors)	
	Uyghur:doHturlarda = doHtur + lar + dA	lar + dA
	(doctors)	
	Kazakh:darigerler = dariger + ler	ler
	Kirghiz: darigArlAr = darigAr + lAr	lAr
	(to the doctor)	
	Uyghur:doHturGa = doHtur + Ga	Ga
	Kazakh:darigerge = dariger + ge	Ge
	Kirghiz:darigArgA = darigAr + gA	gA
	(medicine)	
	Uyghur:doHturluq = doHtur + luq	luq
	Kazakh:darigerlik = dariger + lik	lik
	Kirghiz:darigArlik = darigAr + lik	lik

Various changes in the morphological structures of these languages easily cause surface form explosion and lead to the problem of language resource scarcity. Furthermore, the data collected from the Internet contain various dialects, dubious spelling and encoding that challenge the reliability of the textual information processing task [4].

This study proposes a morphological segmentation method for Uyghur, Kazakh and Kirghiz based on the Bi-LSTM and CRF model, and a method of text keyword extraction for these languages based on the Doc2vec_TextRank model. First, we built the Bi-LSTM_CRF morpheme segmentation model, then we used this model to perform morpheme segmentation and stem extraction separately on Uyghur, Kazakh and Kirghiz text sets which were collected from the Internet; then, we generated sets of morpheme sequences. Subsequently,

the Doc2vec embedding vector technique is used for vectorization and the stem cosine similarity calculation in order to adjust the probability transition matrix in the TextRank algorithm. Based on this new probability transition matrix, the TextRank weights of the candidate stems were calculated and the keyword extraction task was conducted. The contributions of our proposed method can be summarized as follows:

- Starting from the derivative morphology of Uyghur, Kazakh and Kirghiz languages, we propose a multilingual morpheme segmentation method based on Bi-LSTM_CRF model. We introduce a character embedding vector to efficiently retain the contextual information, and we solve the problem of data sparsity.
- Stem embedding vector similarity is applied to weight the TextRank algorithm and conducted text keyword extraction task. By categorizing the semantic information and reducing redundancy, the keyword extraction efficiency is improved.
- Based on the comparative experimental results on different units such as word and stem, the effect of morphological processing is verified for some derivative languages.

The rest of this paper is organized as follows. The next section presents the related works. Section 3 discusses our proposed method. Section 4 explains the experimental results. In the last section, we provide the conclusion and ideas for future work.

2. Related Works

2.1. Morpheme Segmentation and Stem Extraction

Scholars have conducted extensive research on stem extraction in major languages such as English and there are several commonly used stem extraction tools, such as Porter Stemmer, KStem Token Filter, Hunspell Stemmer, and so on. Research on the stem extraction of low-resource languages, such as Uyghur, Kazakh and Kirghiz, was started relatively late and is still in progress. Some works on Uyghur, Kazakh and Kirghiz morpheme segmentation and stem extraction have been reported in [5–10]. Rana et al. [5] proposed a method for Uyghur morpheme segmentation based on a combination of rules and dictionaries, and an Uyghur word stemmer is developed using left-to-right analysis of the lexical connection rules and Lovin algorithm. Saidiyaguli et al. [6] proposed an Uyghur stem extraction method based on an N-gram model and, according to the constraints of Uyghur word formation, using the parts of speech and contextual stem information features to extract the Uyghur word stem. The experimental results show that hybrid features can improve the efficiency of Uyghur stem extraction to a certain extent. Ulan et al. [7] combined the Kazakh word formation rules with the statistical characteristics of the stem–affix connection point and used a N-gram language model to extract the Kazakh word stems. Gulinazi et al. [8] used a combination of the lexical analysis and bidirectional full segmentation to segment the Kazakh morphemes, matched the segmentation result with the pre-prepared stem table to extract the Kazakh word stems and conducted Kazakh text classification based on the stem unit. Kaibierhan [9] used a hybrid strategy of combining the rules and stem–affix dictionaries to extract Kirghiz word stems. Previous studies on Uyghur, Kazakh and Kirghiz morpheme segmentation and stem extraction methods [5–9] mentioned above were mainly based on manually collected rules or suffix-based stemming methods; therefore, the extraction results obtained from using these methods were ambiguous. In addition, numerous words appear only once in the training corpus; therefore, numerous morpheme units that were collected after manually segmenting these words for training the model also appeared in the corpus no more than once. The inclusion of such morphemes in the corpus creates a data sparsity problem in the estimation of probability when training the segmentation model. Abudukelimu et al. [10] introduced deep learning into the Uyghur morpheme segmentation task and established an Uyghur morphological segmentation model using a bidirectional gated recurrent unit (Bi-GRU) neural network. This model automatically learnt feature representations from the experimental data and alleviated the problem of difficulty in ensuring the coverage in manual feature design, and it improved the efficiency of Uyghur morpheme segmentation.

2.2. Text Keyword Extraction

At present, the mainstream text keyword extraction methods used in the field of natural language processing include the word frequency statistics method, the topic model method, the vocabulary graph model method and the machine learning method. TFIFD [11,12] is a typical statistics-based keyword extraction method, which is efficient and simple and outperforms some of the more complex keyword ranking methods; however, this method performs information mining tasks based only on term and inverse document frequencies; therefore, it cannot reflect the complex semantic information of the text. The LDA model [13] is a representative topic model in the field of text keyword extraction. This model performs well with large training corpora and when dealing with longer textual information; however, the keywords extracted by the LDA model are often too broad to reflect the topic of the article and the model performs poorly on short texts. The methods based on machine learning mainly include the SVM [14], Naive Bayes classifiers [15]. This method transforms the keyword extraction task into a binary classification problem to determine whether or not a candidate word is a keyword and requires the preparation of the manually labeled experimental data and the training of the classification model. The lexical graph model mainly includes the TextRank model [16–19], which maintains that the importance of a word is determined by the votes of other related words, and the weight of a given word determines the importance of the voting process. The graph model has the advantages of linguistic knowledge and domain independence.

Several studies conducted on Uyghur and Kazakh text keyword extraction tasks are been reported in [20–23]. In [20], the frequency of occurrence of words in the text was obtained by weighting the feature items, and the positional information of the given word present in the text was also considered. The TFIDF weighting factor was set for the feature item to calculate the final weight of the candidate word, and the keywords were then extracted from Uyghur texts. In [21], based on the TextRank algorithm, information on the words' positions and word frequency features were used as weighting items, and the keyword extraction task was conducted on Uyghur texts, and this result was used to conduct Uyghur text classification. The work in [21] presented the text classification results, and the detailed results of the keyword extraction were not mentioned. The authors of [22] made improvements to the TFIDF algorithm and used the position features and frequency of words to extract keywords from Kazakh text. The authors of [23] used the TextRank, SAD (Sparse Discriminant Analysis) and SVM algorithms to extract the keywords from the Uyghur text and then conduct an Uyghur text sentiment classification based on the keyword extraction results. This paper presented the final sentiment classification results with improved accuracy. To date, there is a lack of research conducted on the keyword extractions performed on the Kirghiz text, and most recently, research on the keyword extraction in Uyghur and Kazakh texts has been intermittent, and the latest relevant research has not been found in publicly available academic resources. Although the keyword extraction methods used for Uyghur and Kazakh texts used the information in the text to extract the keywords, a traditional text representation method was used to represent the content of the text, and the main features of the words formed parts of the speech and word frequency. In this way, the important semantic information of the text's context was overlooked in the calculation process; therefore, these features could not provide sufficient semantic information for clustering or classification processes, resulting in the low accuracy of the keyword extraction methods.

3. Proposed Method

In this study, an approach for extracting the keywords from the Uyghur, Kazakh and Kirghiz texts is proposed. This study consists of five main steps, namely, text data collection; text pre-processing which includes spellchecking, morpheme segmentation and stem extraction; Doc2vec-based stem embedding vector representation and weight *adjustment* on the TextRank algorithm; TaxtRank value calculation and selection; and a final step of evaluation. For evaluation of the proposed method, we used the accuracy, precision,

recall rates and F1 measure. We also compared our method with some of the previous approaches from the related works. The steps of the proposed method are presented in Figure 1.

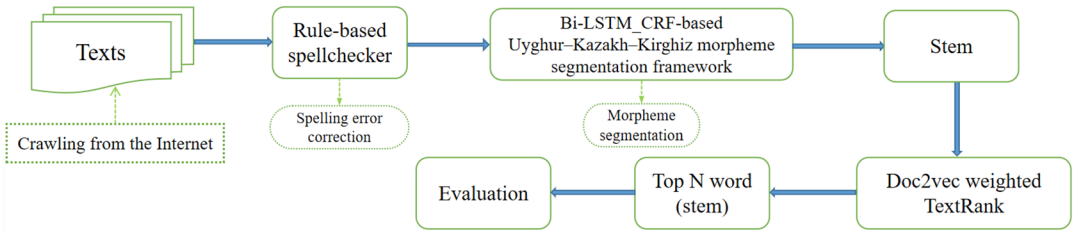

Figure 1. Steps of the proposed method.

3.1. Morpheme Segmentation Method Based on the Bi-LSTM_CRF

Recently, LSTM and Bi-LSTM networks have been widely used in natural language processing tasks, such as parts-of-speech tagging [24,25], Chinese word segmentation [26,27] and named entity recognition [28], and they have achieved outstanding results. In order to improve the efficiency of morpheme segmentation and stem extraction processes for Uyghur–Kazakh–Kirghiz, this study drew on the idea of Chinese word segmentation and transformed the task of morpheme segmentation into the label classification problem of morpheme sequences, and built a Uyghur–Kazakh–Kirghiz morpheme segmentation model based on the Bi-LSTM and CRF networks. A character embedding vector was introduced to continuously represent the word formation sequences so that the data sparsity problem caused by the discrete representation of morphemes when training the segmentation model could be alleviated.

3.1.1. The Bi-LSTM Model

The Bi-LSTM network is a bidirectional RNN network structure that was further improved and generalized by Alex Graves et al. [29] on the basis of the LSTM network. The Bi-LSTM network has bidirectional hidden layers, including forward and backward layers. These layers have the ability to capture the information of sentences in two different directions; therefore, the final output of the network depends on the forward calculation of the forward hidden layer and the backward calculation of the backward hidden layer, and then the states of the two hidden layers are ultimately connected. The Bi-LSTM network's structure is presented in Figure 2.

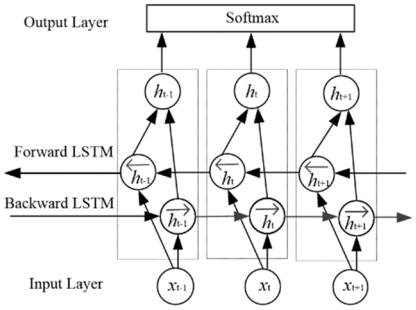

Figure 2. Structure of the Bi-LSTM network.

Let us suppose that $(\overrightarrow{h_1}, \overrightarrow{h_2}, \overrightarrow{h_3}, \cdots, \overrightarrow{h_n})$ and $(\overleftarrow{h_1}, \overleftarrow{h_2}, \overleftarrow{h_3}, \cdots \overleftarrow{h_n})$ are the hidden state sequences of the forward and backward LSTM networks, respectively. Then, the two directions of the LSTM networks can be calculated to obtain the forward and backward hidden states $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$ at time t, as is presented in Equations (1) and (2):

$$\overrightarrow{h_t} = \overrightarrow{\sigma_h}(\overrightarrow{U} \cdot x_t + \overrightarrow{W} \cdot \overrightarrow{h_{t-1}} + \overrightarrow{b}) \quad (1)$$

$$\overleftarrow{h_t} = \overleftarrow{\sigma_h}(\overleftarrow{U} \cdot x_t + \overleftarrow{W} \cdot \overleftarrow{h_{t-1}} + \overleftarrow{b}) \tag{2}$$

The forward and backward hidden states are combined to obtain the final output of the hidden state h_t at time t, as is presented in Equation (3):

$$h_t = \left[\overrightarrow{h_t^T}; \overleftarrow{h_t^T}\right] \tag{3}$$

We added a linear SoftMax layer after the LSTM network and classified the sequence of the attained labels.

3.1.2. The CRF Model

CRF [30] is an undirected graph model that has the advantages of its simple labeling and high efficiency. In relation to an observed sequence that must be labeled, the CRF model calculates the joint probability distribution of all sequences to infer the corresponding state sequences. For the sequence labeling task, the CRF model jointly decodes the input sentence by considering the correlation evident between the adjacent labels to obtain the best label chain for a given input sentence and normalize the global features to obtain a global optimal solution. The chain structure of the CRF model is presented in Figure 3.

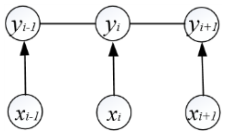

Figure 3. Chain structure of the CRF model.

If we assume that $X = (x_1, x_2, \cdots, x_n)$ is the given observable sequence and $y = (y_1, y_2, \cdots, y_n)$ is the corresponding label sequence, the conditional probability of label sequence y that appears can be calculated by Equation (4):

$$p(y|X) = \exp(\sum_j \lambda_j t_j(y_{i-1}, y_i, X, i) + \sum_k u_k s_k(y_i, X, i) \tag{4}$$

where $t_j(y_{i-1}, y_i, X, i)$ is the probability transition function representing the probability of a transition from the $(i-1)$-th token to the i-th token in the label sequence y for the observation sequence X, $s_k(y_i, X, i)$ is a state function representing the probability of label y_i at the i-th position for the observation sequence X, and λ_j and u_k are the weight parameters.

3.1.3. The Bi-LSTM_CRF-Based Uyghur–Kazakh–Kirghiz Morpheme Segmentation Model

The Bi-LSTM_CRF model is a neural network structure that combines the Bi-LSTM and CRF models. In this model, the SoftMax classification layer after the hidden layer of the Bi-LSTM network is replaced by the CRF layer, and the vector output from the Bi-LSTM network is used as the input of the CRF layer to obtain the final probability output of the model. When the characters in Uyghur, Kazakh or Kirghiz words are labeled according to the word formation characteristics of morphemes and used as inputs for the Bi-LSTM_CRF model, the Bi-LSTM model can realize the effective retention of the context information and the information of the preceding and following label sequences can also be effectively used by the CRF model; thus, the globally optimal label sequence with the highest probability can be obtained. The network structure of the Bi-LSTM_CRF model is presented in Figure 4.

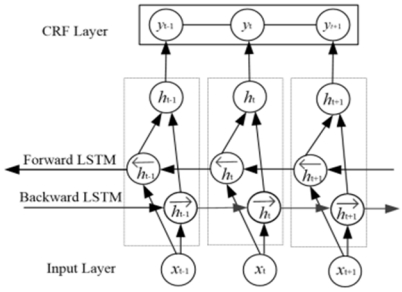

Figure 4. Structure of the Bi-LSTM_CRF model.

When the CRF layer is added after the output layer of the Bi-LSTM network, the CRF layer needs to adopt the state transition matrix as a parameter. Let A be the state transition matrix and let P be the output matrix of the bidirectional LSTM network. The output result predicted by the Bi-LSTM_CRF model for the state sequence $y = (y_1, y_2, \cdots, y_n)$ corresponding to the observed sequence $X = (x_1, x_2, \cdots, x_n)$ can then be represented by the scoring function presented in Equation (5):

$$p(X, y) = \sum_{i=0}^{n} A_{y_i, y_{i+1}} + \sum_{i=1}^{n} P_{i, y_i} \tag{5}$$

where A_{y_i, y_j} represents the probability of transition from states y_i to y_j and P_{i, y_i} represents the probability that the i-th character in the input observation sequence is the y_i-th label.

We used the datasets obtained from [4], which included 10,000 Uyghur, 5000 Kazakh and 3000 Kirghiz sentences, to prepare a lexical corpus, including 17,230 Uyghur, 14,516 Kazakh and 9460 Kirghiz words, and labeled these words using the BMES labeling method. That is, the characters of all words in the lexical corpus were labeled with {B, M, E, S} to represent the position of the morphemes that constitute a word, where B represents the starting character of the morpheme in the word, M represents the middle character of the morpheme in the word, E represents the end character of the morpheme in the word and S represents a morpheme with a single character. For example, the Kazakh word "qezmEtneN" (English meaning: of work) would be labeled as:

"q/B e/M z/M m/M E/M t/E n/B e/M N/E". The labeled data set was used as the input in the Bi-LSTM_CRF model, where the Bi-LSTM model extracted the inherent features of the Uyghur, Kazakh and Kirghiz languages, and the CRF model predicted the label of the sequences. The Bi-LSTM_CRF-based Uyghur–Kazakh–Kirghiz morpheme segmentation framework is presented in Figure 5.

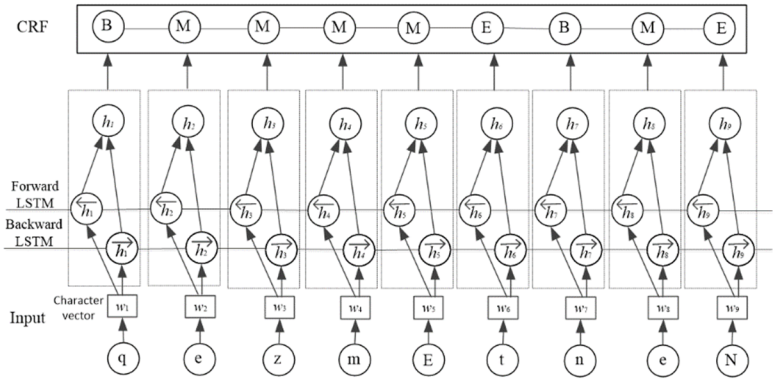

Figure 5. Structure of the Bi-LSTM_CRF-based Uyghur–Kazakh–Kirghiz morpheme segmentation model.

After the Kazakh word "qezmEtneN" was segmented by the Bi-LSTM_CRF model, it was divided into two parts: "qezmEt" (work) + neN (suffix).

In this study, the labeled Uyghur, Kazakh and Kirghiz word datasets were divided into training, validation and the test sets, according to a split ratio of 0.75:0.10:0.15. When training the Bi-LSTM_CRF model, the epochs of model training and the number of hidden layer neurons were both set to 100, the learning rate was set to 0.001 and the dropout ratio was set to 0.5. The mini-batch method was used to train the model, in which the batch size was set to 128, and the model was optimized using the Adam optimization function. The Word2vec tool was used for training the character vectors, and the size of the training window was set to 5, the number of iterations was set to 10 and the batch word was set to 10,000. The character vector dimension is an important hyperparameter, which directly affects the segmentation effect of the model. In order to determine the optimal dimensions of the character vector, we selected a range from 50 to 300 dimensions, with steps of 50 dimensions, and used the validation set to test the effect of different character vector dimensions on the segmentation performance, as shown in Figure 6. The accuracy was determined as the number of correctly segmented words out of all the segmented words.

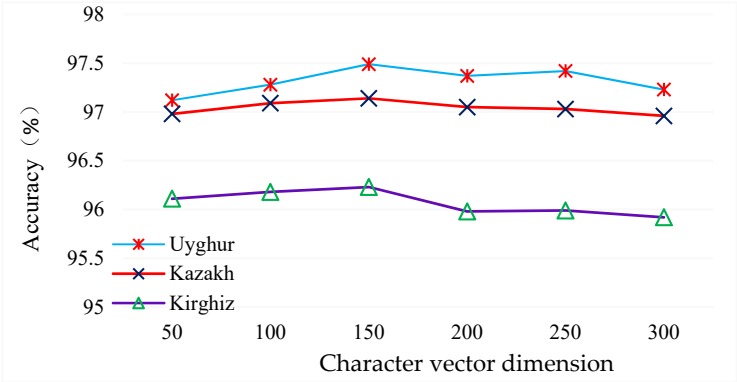

Figure 6. Influence of the character vector dimensions on the accuracy of the segmentation model.

As can be observed from Figure 6, when the character vector presented approximately 150 dimensions, the model's segmentation performance was the best. Therefore, 150 dimensions were selected as the optimal number for the character vectors in the subsequent experiments. Figure 7 presents the effect of changes in the number of iterations of the model on the accuracy of the morpheme segmentation.

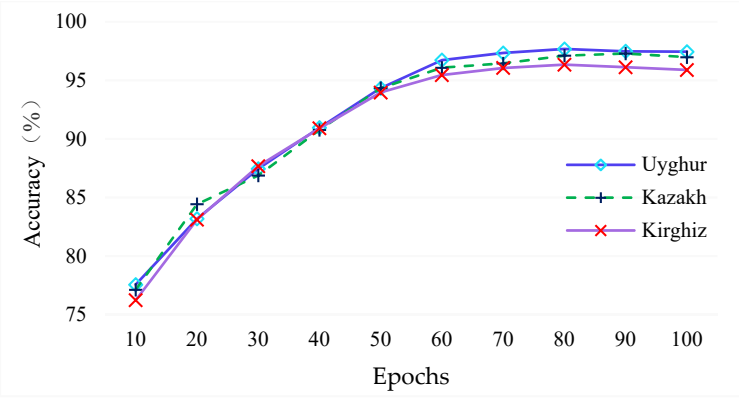

Figure 7. Influence of iterations on the accuracy of segmentation.

As can be observed from Figure 7, with an increase in the number of iterations, the model began to learn the positional features of the preceding and following character sequences more completely; therefore, the model's accuracy of segmentation also improved. When the number of iterations reached approximately 80, the segmentation accuracy of the model on the three datasets attained the highest values of 97.68%, 97.29% and 96.34%, after which the model started to converge.

3.2. Text Representation Method Based on the Doc2vec Model

Following the morpheme segmentation and stem extraction procedures, the subsequent step was to effectively represent the text features. The Doc2vec model [31] is an improved text representation model based on Word2vec. The word embedding vector represented by Doc2vec not only considers the semantic information between words, but also compresses the dimensions of the vectors; at the same time, the word order information is also considered, which offsets the shortcomings of Word2vec in ignoring the influence of word order in the text. The Doc2vec model predicts words by training two neural networks: a distributed memory (DM) model and a distributed bag of words (DBOW) model. The DM model predicts the occurrence probability of a word, given the context and document vector. The DM model [31] is presented in Figure 8.

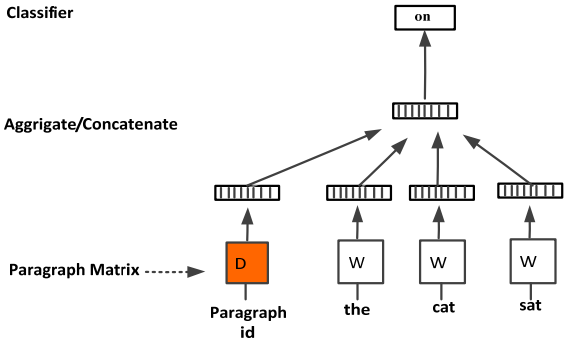

Figure 8. The DM model.

During the process of training the DM model, each document ID and all the words present in the corpus were first initialized as N-dimensional vectors. The document vector and context vocabulary were then fed into the model, and the hidden layers accumulated these vectors to obtain an intermediate vector, which was used as the input to the SoftMax output layer. During the document training process, the document ID remained the same and shared the same document vector, which was equivalent to using the semantics of the entire sentence when predicting the probability of a word.

The DBOW model predicts the probability of a random set of words appearing in a document based on a given document vector. In the process of training a single document, the document vector is shared, that is to say, the probability of a word's occurrence is predicted by using the semantic information of the whole document. The DBOW model [31] is presented in Figure 9.

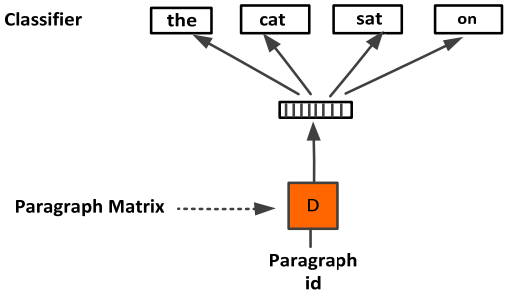

Figure 9. The DBOW model.

The DBOW model is a three-layer neural network. This model ignores the context of the input sentence and predicts random words that appear in the paragraph. That is, in each iteration, a window is sampled from the text, then a random word from that window is used as the prediction target, and it is predicted using the model, where the input to the model is the paragraph vector. We used the DBOW model to generate document vectors.

After we generated the word (stem) vector based on the Doc2vec model, the Euclidean distance was used to calculate the cosine similarity between two words to quantify their similarity. If the cosine value is higher, the two words are more similar and the degree of association is stronger, whereas a lower cosine value means that two words are less similar and the degree of association is weaker. Let us suppose that W_i and W_j are two words represented by word vectors. The similarity between them can be calculated by using Equation (6):

$$Sim(W_i, W_j) = \omega_{ij} = \frac{\vec{W_i} \cdot \vec{W_j}}{\|\vec{W_i}\| * \|\vec{W_j}\|} \tag{6}$$

3.3. Text Keyword Extraction Method Based on the TextRank Algorithm

The TextRank algorithm is a graph-based ranking algorithm that uses a text graph to represent the relationships between words in a given text [32]. Subsequent to building the text graph, each word in the document was regarded as a node, and the PageRank algorithm [33] was used to calculate the TextRank score of each node. Initially, all the words contained in the text graph were considered as candidate keywords, and the top n words with the highest TextRank scores were selected as the final keywords following the calculation of the PageRank. The TextRank score for a given node is calculated by using Equation (7):

$$TR(w_i) = (1-d) + d \times \sum_{w_j \in in(w_i)} \frac{ew_{ji}}{\sum_{v_k \in out(w_j)} ew_{jk}} TR(w_j) \tag{7}$$

where $TR(w_i)$ is the TextRank score of the node w_i and d is the damping coefficient, which represents the probability of any node randomly jumping to other nodes in the word graph; d is a constant between 0 and 1, and usually takes 0.85. Moreover, $in(w_i)$ represents the set of nodes pointing to the node w_i, $out(w_j)$ represents the set of nodes pointed to by the node w_j, and ew_{ji} and ew_{jk} are the edge weights between two nodes, that is, ew_{ji} and ew_{jk} represent the random weight probability matrix of a transition from nodes j to i and from nodes j to k, respectively, as shown in Equation (8):

$$WT = \begin{bmatrix} ew_{11} & ew_{12} & \cdots & ew_{1n} \\ ew_{21} & ew_{22} & \cdots & ew_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ ew_{n1} & ew_{n2} & \cdots & ew_{nn} \end{bmatrix} \tag{8}$$

A text graph is defined as an undirected graph G = (V, E), where V represents the set of words and E represents the set of edges between the words. Text graphs are created using two steps: first, the candidate keywords are selected from the text and considered as nodes in V; second, edges are built between words within a given window size W. The edges of the text graph are constructed by using a co-occurrence relationship [34], which is controlled by setting a sliding window for a given text. Specifically, if two words w_i and w_j appear in a sentence in the text with a window size of W, then we treat w_i and w_j as a node and add it to V, then build an edge between w_i and w_j, and the weight of the edge is 1; if not, there is no connection evident between w_i and w_j, and the weight of the edge is 0.

In fact, language is a natural phenomenon, and the semantic relationship between language units is present at different degrees; therefore, links of different strengths are present between the nodes. Using the traditional TextRank algorithm to calculate edge weights leads to the sparsity problem of probability transfer matrices. The word vector similarity generated by the Doc2vec model overcomes the sparsity problem in traditional text representation and includes the semantic information between the words. Therefore, we can use the Doc2vec algorithm to represent Uyghur, Kazakh and Kirghiz texts with word (stem) embedding vectors, and use the semantic information contained in the embedding

vector representation to weigh the probability of a transition occurring between the word graph nodes in TextRank, that is, using the similarity of the word vectors generated by the Doc2vec model to represent the relationships between the nodes of the traditional TextRank algorithm can ensure that some useful information is retained in the process of calculating the TextRank weight, thus improving the performance of extracting keywords from Uyghur, Kazakh and Kirghiz texts, as shown in Equation (9). The transition probability matrix after the edge weights have been replaced by the Doc2vec similarity is presented in Equation (10):

$$TR(w_i) = (1-d) + d \times \sum_{w_j \in in(w_i)} \frac{\omega_{ji}}{\sum_{v_k \in out(w_j)} \omega_{jk}} TR(w_j) \tag{9}$$

$$WT^* = \begin{bmatrix} \omega_{11} & \omega_{12} & \cdots & \omega_{1n} \\ \omega_{21} & \omega_{22} & \cdots & \omega_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \omega_{n1} & \omega_{n2} & \cdots & \omega_{nn} \end{bmatrix} \tag{10}$$

Then, the TextRank values of all candidate words (stems) were calculated on the basis of the new transition probability matrix by iteratively using Equation (9). After obtaining the TextRank values of all the nodes in the Uyghur–Kazakh–Kirghiz text graph, we selected the top n important stems with the highest TextRank values as keywords.

4. Experimental Results and Analysis

4.1. Experimental Corpus and Pre-Processing

At present, the research conducted on textual information processing methods in Uyghur–Kazakh–Kirghiz is still in the preliminary stage. There are no standard Uyghur, Kazakh and Kirghiz keyword extraction text datasets publicly available. In this study, we used the datasets obtained from [35], which consisted of over 8000 Uyghur texts in 9 categories (100 newly added texts for each category downloaded from the same data source) and over 7000 Kazakh texts in 8 categories (100 newly added texts for each category downloaded from the same data source) to prepare the Uyghur and Kazakh experimental texts. We built the Kirghiz text dataset by using a web crawling technique to download Kirghiz texts in six categories from the Xinjiang Radio and Television Station's website. In this experiment, 1000 texts were randomly selected from each dataset.

Due to the influence of other languages on Uyghur, Kazakh and Kirghiz and individuation, the texts downloaded from the Internet are prone to spelling errors. Therefore, we developed a spellchecker for the Uyghur, Kazakh and Kirghiz texts. The program analyzed the syllable rules of Uyghur, Kazakh and Kirghiz to detect the majority of misspelt words so that the spelling errors in a given word could be corrected. The flowchart of the spellchecker is presented in Figure 10. We used this program to correct spelling errors evident in the texts.

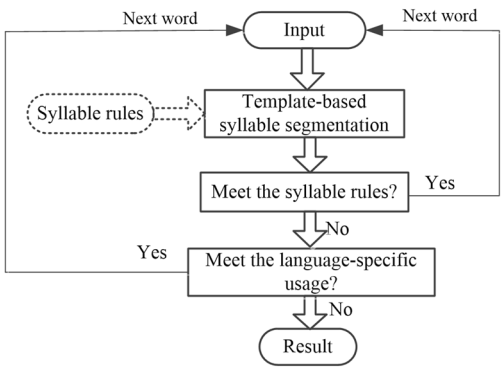

Figure 10. Flowchart of the Uyghur–Kazakh–Kirghiz spellchecker.

We then used the Uyghur–Kazakh–Kirghiz morpheme segmentation model to perform the morpheme segmentation and stem extraction operations on three text datasets, and then used the self-built stop word (stem) dictionary to implement stop word filtering. The Bi-LSTM_CRF model can lead to a significant reduction in the feature space dimensions of the texts, as shown in Table 2, which produces the stem extraction results obtained for all the collected texts.

Table 2. Reduction in vocabulary number by stem extraction.

# of Category	Language	No. of Word Vocabulary	No. of Stem Vocabulary	Stem-Word Vocabulary Ratio (%)
9	Uyghur	89,819	26,810	29.85
8	Kazakh	84,205	25,354	30.11
6	Kirghiz	35,643	11,024	30.93

It can be observed in Table 2 that, following the morpheme segmentation, the stem vocabulary decreased to approximately 30% of the word vocabulary. After the stems had been extracted from the texts, the stem vector was trained using the DBOW model. During the training stage, the vector dimension was set to the default value of 100, and the training window and learning rate were set to 10 and 0.025, respectively.

4.2. Experimental Results and Analysis

The performance of the keyword extraction method proposed in this study was evaluated by precision rate P, recall rate R and the F1 value. The formulae for calculating these are as follows:

$$P = T/A$$
$$R = T/M$$
$$F1 = 2 \times P \times R/(P + R)$$

where T represents the total number of correctly extracted keywords, A represents the number of all extracted keywords, and M represents the number of keywords selected manually.

In this study, keyword extraction methods, such as TF, TFIDF [20,22], TextRank [21,23] and Word2vec_TFIDF [36] (our previous work), were used to perform a comparison with the proposed method, and keyword extraction experiments were conducted on the datasets prepared in this study. As, besides our previous work, other given experimental results from the related works used different data sets, and the number of the experimental data was too small, we just applied the methods from the related works and experimented on our data sets for fair comparison. In the experiment, we first manually selected three keywords (stems) for each experimental text, and then we used TF, TFIDF, TextRank, Word2vec_TFIDF and the proposed method to extract three, four, five and seven keywords. The comparative experimental results are presented in Tables 3–6.

Table 3. Comparison results of extracting three keywords.

Method	Language	Result (%)		
		P	R	F1
TF	Uyghur	26.7	32.7	29.4
	Kazakh	26.5	32.4	29.2
	Kirghiz	25.9	31.8	28.5
TFIDF	Uyghur	27.7	32.9	30.1
	Kazakh	27.5	33.4	30.2
	Kirghiz	27.5	33.2	30.1
TextRank	Uyghur	33.4	43.4	37.7
	Kazakh	32.9	42.7	37.2
	Kirghiz	32.7	42.3	36.9

Table 3. *Cont.*

Method	Language	Result (%)		
		P	R	F1
Word2vec_TFIDF	Uyghur	34.5	38.8	36.5
	Kazakh	34.3	38.6	36.3
	Kirghiz	34.3	38.3	36.2
Doc2vec_TextRank	Uyghur	42.8	44.1	43.4
	Kazakh	43.2	44.7	43.9
	Kirghiz	42.9	44.3	43.6

Table 4. Comparison results of extracting four keywords.

Method	Language	Result (%)		
		P	R	F1
TF	Uyghur	24.9	30.8	27.5
	Kazakh	24.7	30.5	27.3
	Kirghiz	24.1	29.9	26.7
TFIDF	Uyghur	30.9	34.1	32.4
	Kazakh	31.1	34.3	32.6
	Kirghiz	31	34.1	32.5
TextRank	Uyghur	33.1	44.2	37.9
	Kazakh	32.8	44.1	37.6
	Kirghiz	32.4	44.3	37.4
Word2vec_TFIDF	Uyghur	36.3	43.2	39.5
	Kazakh	36.4	43	39.4
	Kirghiz	35.8	43.1	39.1
Doc2vec_TextRank	Uyghur	42.1	45.3	43.6
	Kazakh	42.5	45.6	44
	Kirghiz	42.4	45.5	43.9

Table 5. Comparison results of extracting five keywords.

Method	Language	Result (%)		
		P	R	F1
TF	Uyghur	23.2	29.7	26.1
	Kazakh	23.5	29.6	26.2
	Kirghiz	22.9	29.1	25.6
TFIDF	Uyghur	32.3	39.5	35.5
	Kazakh	32.8	39.9	36
	Kirghiz	32.4	39.8	35.7
TextRank	Uyghur	33.7	42.6	37.6
	Kazakh	33.5	42.7	37.5
	Kirghiz	33.1	42.3	37.1
Word2vec_TFIDF	Uyghur	40.6	45.7	43
	Kazakh	40.9	45.5	43.1
	Kirghiz	40.1	44.9	42.4
Doc2vec_TextRank	Uyghur	41.4	46.5	43.8
	Kazakh	41.3	46.9	43.9
	Kirghiz	41.5	46.4	43.8

Table 6. Comparison results of extracting seven keywords.

Method	Language	Result (%)		
		P	R	F1
TF	Uyghur	22.8	30.4	26.1
	Kazakh	22.5	30.8	26
	Kirghiz	22.1	30.5	25.6
TFIDF	Uyghur	31.8	40.2	35.5
	Kazakh	31.4	40.5	35.4
	Kirghiz	31.7	39.9	35.3
TextRank	Uyghur	32.6	43.9	37.4
	Kazakh	32.9	43.6	37.5
	Kirghiz	32.6	43.5	37.3
Word2vec_TFIDF	Uyghur	39.7	45.9	42.6
	Kazakh	39.5	45.9	42.5
	Kirghiz	39	45.2	41.9
Doc2vec_TextRank	Uyghur	39.8	47.2	43.2
	Kazakh	39.2	47.3	42.9
	Kirghiz	39.7	47.2	43.1

It can be observed in Tables 3–6 that when the number of extracted keywords was four, the F1 values of the Kazakh and Kirghiz texts reached the highest values of 44% and 43.9%, respectively. With five keywords, the F1 value of the Uyghur text attained the highest score of 43.8%. When the number of extracted keywords increased, the P, R and F1 values of the TF algorithm that was run on the three data sets began to decrease, while the values of P, R and F1 of the TFIDF algorithm gradually increased. However, the overall effect was significantly lower than the method proposed in this study. Among them, with different numbers of keywords, there was a gap of approximately 6–13% in the F1 values between TFIDF and the method proposed in this study. When the number of keywords increased, the traditional TextRank algorithm produced relatively stable results, but the effect lagged behind the proposed method; among them, the F1 value was lower than approximately 5.5%. When the number of keywords extracted was low, the P, R and F1 values of the Word2vec_TFIDF method were significantly lower than that of the proposed method; as the number of keywords increased, the values of P, R and F1 began to increase (when the number of keywords was seven, each value slightly decreased), and the gap with the method presented in this study began to narrow. The highest F1 value for the three datasets was approximately 0.8% less than for the method presented in this study. For different numbers of keywords, the values of P, R and F1 of the proposed method all maintained a relatively stable level.

In order to verify the impact of morpheme segmentation and stem extraction processes on the performance of the keyword extraction task in derivative languages, Doc2vec embedding vectors are trained on word sequence prior to conducting morpheme segmentation on the datasets and keyword extraction is conducted based on the words. The results of the keyword extraction task based on words and stem units were compared, as shown in Table 7.

As can be observed in Table 7, as the number of keywords increased, the value P, based on the word and stem units, began to decrease, while the values of R and F1 began to increase (when the number of keywords was seven, the F1 value decreased). Stem units provided better extraction results at all keyword levels, where the P, R and F1 values obtained with stem units were approximately 2% higher than those with word units. In particular, when the number of extracted keywords was four, the F1 value for the stem units on the three datasets was approximately 2.8% higher than that of the word units. It can be observed from the results that stem units can provide better results in textual

information processing tasks than word units and prove that morpheme segmentation can improve the effect of the text keyword extraction.

Table 7. Comparisons of extraction results based on word and stem units.

Method	No. of Keywords	Language	Vocabulary Unit	Result (%)		
				P	R	F1
Doc2vec_TextRank	3	Uyghur	Word	40.5	41.3	40.9
			Stem	42.8	44.1	43.4
		Kazakh	Word	40.7	41.5	41.1
			Stem	43.2	44.7	43.9
		Kirghiz	Word	41.1	41.7	41.4
			Stem	42.9	44.3	43.6
	4	Uyghur	Word	39.7	42.8	41.2
			Stem	42.1	45.3	43.6
		Kazakh	Word	39.3	42.4	40.8
			Stem	42.5	45.6	44.0
		Kirghiz	Word	39.8	42.7	41.2
			Stem	42.4	45.5	43.9
	5	Uyghur	Word	39	44.2	41.4
			Stem	41.4	46.5	43.8
		Kazakh	Word	38.8	44.5	41.5
			Stem	41.3	46.9	43.9
		Kirghiz	Word	39.2	44.3	41.6
			Stem	41.5	46.4	43.8
	7	Uyghur	Word	37.1	45.6	40.9
			Stem	39.8	47.2	43.2
		Kazakh	Word	37.5	45.3	41.0
			Stem	39.2	47.3	42.9
		Kirghiz	Word	37.6	45.5	41.2
			Stem	39.7	47.2	43.1

5. Conclusions and Future Work

Text keyword extraction is basic work in text information processing tasks. Automatic keyword extraction becomes challenging because of the complexity of natural languages and the heterogeneity of document types. Uyghur, Kazakh and Kirghiz are agglutinative languages with similar grammatical and lexical structures, where words are formed by adding affixes to stems. Some affixes change the meaning of the stem and derive new stems. As a result of their derivative characteristics, these languages have multiple combinations of morphemes and their vocabulary tends to greatly increase in size. Therefore, morpheme segmentation and stem extraction processes are some of the important methods used in the research to improve the performance of keyword extraction. This study discussed a morpheme segmentation method based on the Bi-LSTM_CRF deep neural network and a keyword extraction method based on Doc2vec and TextRank models. The task of extracting keywords from Uyghur, Kazakh and Kirghiz texts was implemented with different keyword extraction methods and language units based on various numbers of keywords. The experimental results show that the proposed method, which is based on morpheme segmentation, Doc2vec and TextRank methods, obtained F1 values of 43.8%, 44% and 43.9% on the three datasets. Compared with the extraction results based on other language unit and keyword extraction methods, the keyword extraction performance of proposed method was significantly improved. It can be observed that effective pre-processing techniques, such as morpheme segmentation and stem extraction, can improve the efficiency of natural language processing tasks for derivative languages such as Uyghur, Kazakh and Kirghiz.

As they are low-resource languages, Uyghur, Kazakh and Kirghiz experimental text data are relatively small. In future works, we will further expand the corpus data size by

using methods such as data augmentation and cross-lingual processing. We will also extend the morpheme segmentation method to incorporate other derivative languages, such as Uzbek and Mongolian, and conduct keyword extraction research on these languages. Additionally, we can apply keyword extraction methods to study sentiment analysis.

Author Contributions: Conceptualization, S.P. and A.K.; methodology, S.P. and A.H.; software, S.P. and M.S.; validation, A.K., A.H. and S.P.; formal analysis, S.P. and M.S.; investigation, S.P., M.S., A.H. and A.K.; resources, A.H. and A.K.; data curation, M.S.; writing—original draft preparation, S.P.; writing—review and editing, M.S., A.H. and A.K.; visualization, S.P. and M.S.; supervision, A.H. and A.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by National Natural Science Foundation of China, grant number 61662073, 2020 Xinjiang Uyghur Autonomeus Ragion Tianchi Doctor Plan Project, Xinjiang University of Finance and Economics School Level Scientific Research Foundation Project, grant number 2022XGC022.

Data Availability Statement: The experimental corpora in this study were built by crawling the texts from the publicly available resources, such as http://uyghur.people.com.cn/ (accessed on January 2018–April 2018), http://uy.ts.cn/ (accessed on November 2017–March 2018), http://kazakh.people.com.cn/ (accessed on January 2018–April 2018), http://kazakh.ts.cn/ (accessed on November 2017–March 2018), http://kirgiz.acradio.cn (accessed on January 2018–May 2018).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Salton, G.; Buckley, C. Term-weighting Approaches in Automatic Text Retrieval. *Inf. Process. Manag.* **1988**, *24*, 513–523. [CrossRef]
2. Firoozeh, N.; Nazarenko, A.; Alizon, F.; Daille, B. Keyword extraction: Issues and methods. *Nat. Lang. Eng.* **2019**, *11*, 1–33. [CrossRef]
3. Ruhul, A.; Madhusodan, C. Algorithm for Bengali Keyword Extraction. In Proceedings of the International Conference on Bangla Speech and Language Processing, Sylhet, Bangladesh, 21–22 September 2018.
4. Ablimit, M.; Parhat, S.; Hamdulla, A.; Zheng, T.F. Multilingual Language Processing Tool for Uyghur, Kazak and Kirghiz. In Proceedings of the Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, Kuala Lumpur, Malaysia, 12–15 December 2017.
5. Rena, A.; Li, X.; Ainiwaner, T. Research on Uyghur stem extraction method based on hybrid method. *J. Comput. Appl.* **2015**, *32*, 112–114.
6. Saidiyaguli, E.; Xiang, L.; Zong, C.; Askar, H. A Multi-Strategy Approach to Uyghur Stemming. *J. Chin. Inf. Process.* **2015**, *29*, 204–210.
7. Ulan, N.; Rahmotola, M.; Askar, H. The Method of Kazakh Word Lemmatization Based on N-gram Model. *Comput. Knowl. Technol.* **2017**, *13*, 160–162.
8. Gulinazi; Sun, T.L.; Hu, X.D. Kazakh Text Classification Based on Improved SV-NN. *J. Northwest. Norm. Univ. Nat. Sci. Ed.* **2018**, *50*, 63–70.
9. Kaibierhan, M. Research on Stem Extraction of Kirgiz Based on Hybrid Method. Master's Thesis, Xinjiang Normal University, Urumqi, China, 2020.
10. Halidanmu, A.; Yong, C.; Yang, L.; Maosong, S. Uyghur morphological segmentation with bidirectional GRU neural networks. *J. Tsinghua Univ. (Sci. Technol.)* **2017**, *57*, 1–6.
11. Li, J.; Fan, Q.N.; Zhang, K. Keyword Extraction Based on tf/idf for Chinese News Document. *Wuhan Univ. J. Nat. Sci.* **2007**, *12*, 917–921. [CrossRef]
12. Hasan, K.S.; Ng, V. Conundrums in Unsupervised Keyphrase Extraction: Making Sense of the State-of-the-Art. In Proceedings of the 23rd International Conference on Computational Linguistics, Beijing, China, 23–27 August 2010; pp. 365–373.
13. Blei, D.M.; Ng, A.Y.; Jordan, M.I. Latent Dirichlet allocation. *J. Mach. Learn. Res.* **2003**, *3*, 993–1022.
14. Zhang, K.; Xu, H.; Tang, J.; Li, J. Keyword Extraction Using Support Vector Machine. In Proceedings of the Seventh International Conference on Web-Age Information Management, Hong Kong, China, 17–19 June 2006.
15. Irfan, R.; Khan, S.; Qamar, A.M.; Bloodsworth, P.C. Refining Kea++ Automatic Keyphrase Assignment. *J. Inf. Sci.* **2014**, *40*, 446–459. [CrossRef]
16. Mihalcea, R.; Tarau, P. TextRank: Bringing order into text. *EMNLP* **2004**, *4*, 404–411.
17. Liu, Z.; Huang, W.; Zheng, Y.; Sun, M. Automatic keyphrase extraction via topic decomposition. In Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, Cambridge, MA, USA, 9–11 October 2010; pp. 366–376.
18. Pay, T.; Lucci, S. Automatic keyword extraction: An ensemble method. In Proceedings of the 2017 IEEE International Conference on Big Data, Boston, MA, USA, 11–14 December 2017; pp. 4816–4818.

19. Peng, L.; Bin, W.; Zhi, W.S. Tag-TextRank: A Webpage Keyword Extraction Method Based on Tags. *J. Comput. Res. Dev.* **2012**, *49*, 2344–2351.
20. Su, X.K. Study on Extraction of Uyghur Keywords in Public Opinion Analysis. Master's Thesis, Xinjiang University, Urumqi, China, 2015.
21. Ghalip, A.; Li, X. Uyghur Keyword Extraction and Text Classification Based on TextRank Algorithm and Mutual Information Similarity. *Comput. Sci.* **2016**, *43*, 43–47.
22. Hu, B.Y.; Gulila, A. Research on the Kazakh Network Hot Keywords Extraction Method. *Comput. Appl. Softw.* **2017**, *34*, 45–50.
23. Imam, S.; Parhat, R.; Hamdulla, A.; Li, Z. Performance Analysis of Different Keyword Extraction Algorithms for Emotion Recognition from Uyghur Text. *J. Tsinghua Univ. (Sci. Technol.)* **2017**, *57*, 270–273.
24. Besharati, S.; Veisi, H.; Darzi, A.; Saravani, S.H.H. A hybrid statistical and deep learning based technique for Persian part of speech tagging. *Iran J. Comput. Sci.* **2021**, *4*, 35–43. [CrossRef]
25. Alkhwiter, W.; Al-Twairesh, N. Part-of-speech tagging for Arabic tweets using CRF and Bi-LSTM. *Comput. Speech Lang.* **2020**, *65*, 101138. [CrossRef]
26. Jin, C.; Shi, Z.; Li, W.; Guo, Y. Bidirectional LSTM-CRF Attention-based Model for Chinese Word Segmentation. *arXiv* **2021**, arXiv:2105.09681.
27. Zhang, T.; Xu, R. Performance Comparisons of Bi-LSTM and Bi-GRU Networks in Chinese Word Segmentation. In Proceedings of the 5th International Conference on Deep Learning Technologies, Qingdao, China, 23–25 July 2021.
28. Wu, G.; Tang, G.; Wang, Z.; Zhang, Z.; Wang, Z. An Attention-Based BiLSTM-CRF Model for Chinese Clinic Named Entity Recognition. *IEEE Access* **2019**, *7*, 113942–113949. [CrossRef]
29. Graves, A.; Fernandez, S.; Schmidhuber, J. Bidirectional LSTM Networks for Improved Phoneme Classification and Recognition. In Proceedings of the 15th International Conference on Artificial Neural Networks: Formal Models and Their Applications, Warsaw, Poland, 11–15 September 2005.
30. Wang, C.; Chen, J.; Wu, X. Dictionary Chinese Word Segmentation research a method combined with CRFs. In Proceedings of the 5th International Conference on Computer Sciences and Convergence Information Technology, Seoul, Republic of Korea, 30 November–2 December 2010.
31. Le, Q.V.; Miklov, T. Distributed Representations of Sentences and Documents. *arXiv* **2014**, arXiv:1405.4053v2.
32. Zhou, J.Z.; Cui, X.H. Keyword extraction method based on word vector and TextRank. *Appl. Res. Comput.* **2019**, *36*, 97–100.
33. Ma, H.; Wang, S.; Li, M.; Li, N. Enhancing graph-based keywords extraction with node association. In Proceedings of the 12th International Conference on Knowledge Science, Engineering and Management, Athens, Greece, 28–30 August 2019.
34. Mihalcea, R.; Tarau, P.; Figa, E. PageRank on semantic networks, with application to word sense disambiguation. In Proceedings of the 20th International Conference on Computational Linguistics, Geneva, Switzerland, 23–27 August 2004.
35. Sardar, P.; Mijit, A.; Askar, H. A robust morpheme sequence and Convolutional Neural Network based Uyghur and Kazakh short text classification. *Information* **2019**, *10*, 387.
36. Sardar, P.; Mijit, A.; Askar, H. Keyword Extraction of Uyghur-Kazakh Texts based on Stem Units. *Comput. Eng. Sci.* **2020**, *42*, 131–137.