IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 29, 2018, accepted June 17, 2018, date of publication July 26, 2018, date of current version August 20, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2860058

Chinese Language Processing Based on Stroke Representation and Multidimensional Representation

HANG ZHUANG[1], (Student Member, IEEE), CHAO WANG[1], (Senior Member, IEEE), CHANGLONG LI[1], (Student Member, IEEE), YIJING LI[2], QINGFENG WANG[3,4], AND XUEHAI ZHOU[1]

[1]Department of Computer Science, University of Science and Technology of China, Hefei 230026, China
[2]Suzhou Industry Park Kuatang Experimental Primary School, Suzhou 215122, China
[3]Department of Software Engineering, University of Science and Technology of China, Hefei 230026, China
[4]Department of Computer Science and Technology, Southwest University of Science and Technology, Mianyang 621010, China

Corresponding author: Hang Zhuang (zhuangh@mail.ustc.edu.cn) and Xuehai Zhou (xhzhou@ustc.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61379040, in part by the Anhui Provincial Natural Science Foundation under Grant 1608085QF12, in part by the Suzhou Research Foundation under Grant SYG201625, in part by the Youth Innovation Promotion Association of CAS under Grant 2017497, and in part by the Fundamental Research Funds for the Central Universities under Grant WK2150110003.

ABSTRACT With the development of deep learning and artificial intelligence, deep neural networks are increasingly being applied for natural language processing tasks. However, the majority of research on natural language processing focuses on alphabetic languages. Few studies have paid attention to the characteristics of ideographic languages, such as the Chinese language. In addition, the existing Chinese processing algorithms typically regard Chinese words or Chinese characters as the basic units while ignoring the information contained within the deeper architecture of Chinese characters. In the Chinese language, each Chinese character can be split into several components, or strokes. This means that strokes are the basic units of a Chinese character, in a manner similar to the letters of an English word. Inspired by the success of character-level neural networks, we delve deeper into Chinese writing at the stroke level for Chinese language processing. We extract the basic features of strokes by considering similar Chinese characters to learn a continuous representation of Chinese characters. Furthermore, word embeddings trained at different granularities are not exactly the same. In this paper, we propose an algorithm for combining different representations of Chinese words within a single neural network to obtain a better word representation. We develop a Chinese word representation service for several natural language processing tasks, and cloud computing is introduced to deal with preprocessing challenges and the training of basic representations from different dimensions.

INDEX TERMS Chinese word representation, stroke-based word representation, multidimensional word representation, convolutional neural networks, natural language processing, word similarity, text classification, automatic text summarization.

I. INTRODUCTION

Natural language processing (NLP) refers to techniques that allow a machine, by analyzing data, to extract information from context and represent the input information in a different way [2]. Natural language processing using a Chinese corpus is usually referred to as Chinese information processing. Chinese is one of the oldest languages in the world and has the largest number of users who treat it as their mother tongue. Chinese characters (known as hanzi) developed from hieroglyphics and currently comprise the only widely used ideographic language (the Chinese characters adopted for use in the Japanese and Korean languages are known as kanji and hanja, respectively).

There is still a massive gap between the NLP of the Chinese language and that of alphabetic languages. In a written language system, a logogram of an ideographic language (Chinese) is a written character that represents a word or a phrase and thus has meaning in itself. By contrast,

in alphabets and syllabaries, individual written characters represent sounds rather than concepts. These characters do not necessarily have meanings in themselves but instead are combined to construct words and phrases that have meanings. Moreover, there are many more levels and dimensions in the Chinese language than alphabetic languages. For instance, the Chinese language can be divided into sentences, words, characters, Pinyin (pronunciation of a character or word), radical components and strokes, while an alphabetic language can be divided only into sentences, words, pronunciation, etymons, and characters. In addition, the pronunciation of a word in an alphabetic language is strongly related to its spelling. The character is the basic unit in an alphabetic language; by contrast, each Chinese character is composed of strokes, which are, in turn, combined into radicals before they become a character. Furthermore, a combinational stroke can be further split into basic strokes.

We simply counted the numbers of characters and words in English, Russian and Chinese. In addition, we considered the number of strokes in Chinese. As is widely known, there are 26 characters (52 characters if the upper- and lower-case versions of each letter are considered) in the English language and 33 characters in the Russian language. The number of words is difficult to count because it changes over time. There are more than 410,000 words in the first edition of the Oxford English Dictionary and approximately 616,500 entries in the Merriam-Webster Collegiate Dictionary. According to the Wikipedia entry on the Explanatory Dictionary of the Living Great Russian Language, which is a major explanatory dictionary for the Russian language, this dictionary contains approximately 200,000 words and 300,000 proverbs. In addition, we referred to the Kangxi Dictionary, which contains more than 49,000 Chinese characters, and the thesaurus compiled by Beijing Guoan Information Equipment Co., Ltd., which collects more than 91,000 Chinese characters. However, only 8,105 Chinese characters are included in the Table of General Standard Chinese Characters, which is typically sufficient for daily use. Similarly, there are more than 374,000 Chinese words in the Sogou Chinese thesaurus and more than 326,000 Chinese words in the Google Chinese thesaurus. We take the number of Chinese words in The Chinese Dictionary, which contains entries for 384,897 Chinese words, as the number of Chinese words. As shown in table 1, although a Chinese character is called a character, it is not a 'character' in the same sense of quantity and meaning as in alphabetic languages. There are 5 basic strokes in the Chinese language and 31 strokes in total in simplified Chinese. The logarithm of the number of each element in each language is shown in Figure 1, from which we can see that the number of Chinese characters is not of the same order of magnitude as the number of English characters or Russian characters. Instead, the number of Chinese strokes is much more similar to the numbers of English characters and Russian characters.

TABLE 1. Numbers of characters and words in English, Russian and Chinese and number of strokes in Chinese.

Number of	English	Russian	Chinese
Strokes	/	/	31
Characters	26	33	91,251
Words	616,500	500,000	384,897

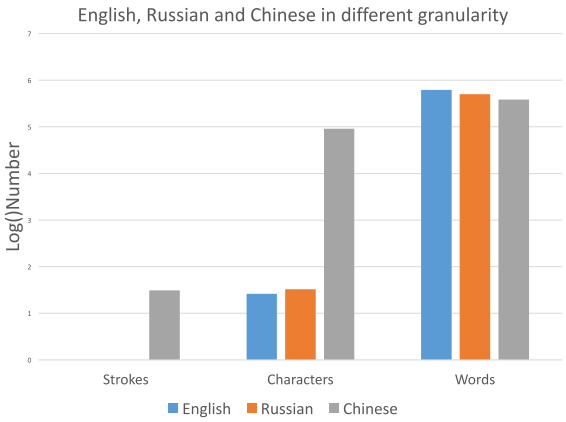


FIGURE 1. Logarithms of the numbers of each element in English, Russian and Chinese.

In some works on Chinese language processing (such as those of Chen *et al.* [3], Zhang *et al.* [4] and Kang *et al.* [5]), Chinese words and Chinese characters have been taken as inputs to learn a representation of Chinese words. From Figure 1, we can easily see that the characters of English represent a fine-grained level of division, whereas the character level is a coarse-grained level of division in Chinese. Consequently, Chinese language processing at either the word level or the character level still suffers from data sparsity and challenges related to rare words. The existing datasets are far from sufficient to cover all characters, let alone all words. In addition, they are blind to character component information and ignore the relations between the characters themselves.

As an example, consider the financial news headline shown in Figure 2. The top row presents a headline from a financial newspaper, whose English translation is given at the bottom of the example. The second row shows the pronunciation of

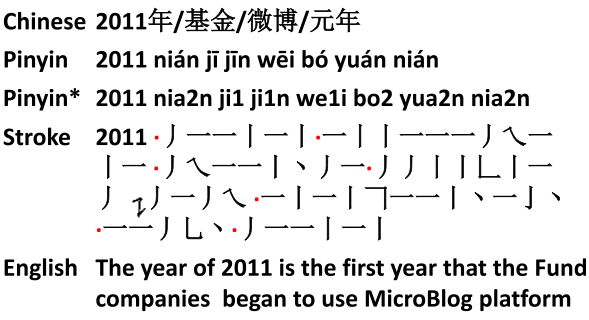


FIGURE 2. Illustration of the Chinese language: A headline of a financial newspaper from a dataset for text classification tasks. The Chinese words, characters, Pinyin, strokes and English translation are listed.

the sentence, called Pinyin, which is the official Romanization system for Standard Chinese. The third row is another format of Pinyin in which numbers are used to represent the tones. This format can be more easily processed and stored by a computer. The fourth row is the stroke sequence of the Chinese sentence comprising the news headline. The red dots divide the stroke sequence into segments that belong to different Chinese characters. Each square-dimensioned symbol in the first row is a distinct Chinese character glyph. Unlike in English, there is no obvious division between words in Chinese. Therefore, a Chinese sentence needs to be segmented into words. The techniques for grouping adjacent characters into words are called Chinese Word Segmentation (CWS) techniques.

To our knowledge, Zhang textitet al. [6] were the first to use Pinyin in combination with a neural network. Although Pinyin represents only the pronunciation of Chinese characters, these authors applied Pinyin in analogy to the English character model for text classification and achieved some progress. However, they were predominantly concerned with the commonalities among different languages [7]. On the other hand, there are too many Chinese characters or words that are pronounced in the same way. As shown in Figure 3, there are 165 Chinese characters with the same Pinyin *yi* when the tone is ignored. All those characters are included in the Modern Chinese Dictionary, which is an official and widely used dictionary. Even more characters with the Pinyin *yi* can be found in the Kangxi Dictionary. The Pinyin system consists of five tones, including the neutral tone, for each Pinyin. In this example, 90 of these Chinese characters have the same Pinyin *yi* at the fourth tone, which is known as the falling tone. All of the homonyms shown in Figure 4, which were selected from among the 165 characters with the same Pinyin *yi*, have different meanings from each other. The main meaning of each Chinese character is listed to its right. This is a common phenomenon in Chinese linguistics. Thus, Pinyin can provide only a partial representation of Chinese characters. If Pinyin is taken as the only input, homonyms with different meaning such as the words shown in Figure 4 will be easily confused and difficult or impossible to distinguish.

一	弋	伊	衣	医	依	祎	咿	椅	浟	铱	猗	揖
壹	椅	欹	毉	漪	噫	噫	繄	匜	仪	圯	夷	貤
詍	诒	迤	饴	怡	宜	荑	栘	咦	贻	姨	栘	眙
胰	宧	庡	蛇	移	痍	遗	颐	椸	疑	嶷	簃	彝
鸃	乙	已	以	苡	佁	尾	矣	苢	迤	蚁	舣	酏
倚	扆	椅	顗	蛾	旖	踦	齮	乂	弋	亿	义	艺
刈	忆	艾	仡	议	屹	亦	衣	异	抑	苅	杙	呓
邑	貤	佚	译	枍	易	峄	鈘	佾	泆	侇	怿	诣
驿	绎	枻	轶	昳	食	狊	奕	弈	疫	斁	羿	挹
喡	益	浥	悒	谊	埸	勩	逸	翊	翌	暘	嗌	鴃
肄	裛	裔	意	溢	缢	蓺	蝪	鹬	廙	瘗	潩	嫕
鷁	镒	饐	毅	鐿	鹝	熠	殪	瞖	螘	儗	劓	燚
		繶	翳	臆	瘞	翼	藹	镱	瘞	懿		

FIGURE 3. One hundred sixty-five Chinese characters with the same pronunciation *yi* (ignoring the tone) in the Modern Chinese Dictionary. Of these characters, 90 have exactly the same pronunciation, *yi* with a falling tone.

Chinese Character	English Meaning	Chinese Character	English Meaning
壹	one	遗	lose
衣	clothes	疑	doubt
医	medical	已	already
椅	chair	蚁	ant
依	depend	义	righteousness
宜	suitable	议	discuss
移	shift	异	different
姨	aunt	益	benefit

FIGURE 4. Selected Chinese characters with the same pronunciation *yi* but completely divergent meanings.

Unlike English characters, Chinese characters are logograms, of which over 80% represent phonosemantic compounds, with the semantic component conferring a broad spectrum of meaning and the phonetic component suggesting the sound. This is probably the reason why the approach based on Pinyin works. In addition, most Chinese character glyphs can represent the pronunciation themselves. Furthermore, there are no two Chinese characters with the exact same glyph. Therefore, the use of Chinese character glyphs is much better than the use of only the pronunciation of Chinese characters. Moreover, we can delve deeper by decomposing each character at a more granular level. In Chinese linguistics, most Chinese characters can be decomposed into several character components, and each character component can be further decomposed into several strokes. The right part of the first example in Figure 5 illustrates the decomposition

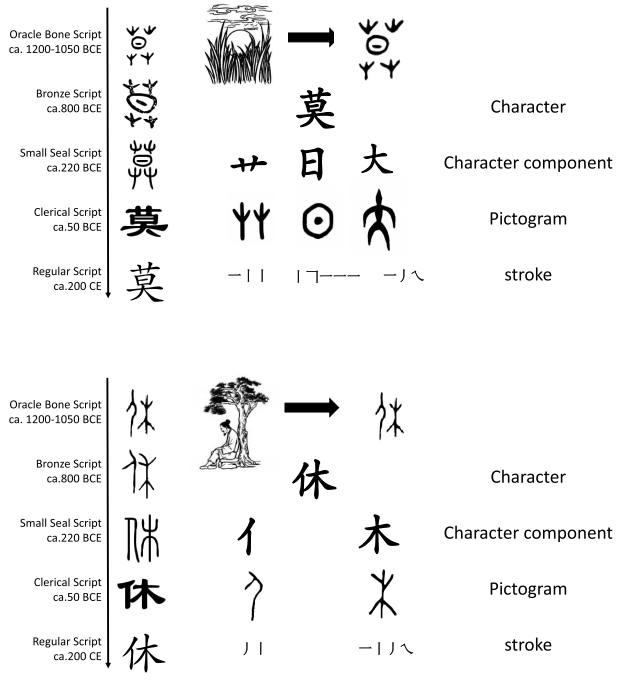


FIGURE 5. Decomposition of the Chinese characters Mo and Xiu. There are five scripts of the character listed on the left with name and formation time of each script. The regular script of that character is split into character components and strokes. The corresponding pictograms of each component are listed below.

of a Chinese character glyph. The left part of shows the same Chinese character, whose original meaning is ''dusk'', in different fonts. This character can be decomposed into 3 Chinese character components and consists of 10 strokes in total. As depicted by the pictograms on the right, the first character component means ''grass'', and the second character component means ''sun''. The third component is a deformed version of the first character component and also means ''grass''. These three character components together convey the meaning that ''the sun falls into the bushes'', whose meaning is exactly ''dusk''. Similarly, a second example is shown for the character whose meaning is ''rest''. This Chinese character can be decomposed into 2 Chinese character components, which represent ''man'' and ''tree''. The two character components together represent a man resting against a tree.

Some researchers [8]–[10] believe that character components are the most basic semantic unit of Chinese and that the character component level is sufficiently deep for learning a Chinese character representation. However, it would be difficult to compile all of the possible character components from the diverse variants of Chinese character radicals and components. Moreover, the intricate hierarchical relationship in the Chinese component structure makes it extremely difficult to accurately split Chinese characters into character components. By contrast, a stroke is a line or combination of lines that forms a part of a Chinese character glyph. Although at first glance, a character may look like it has 5 individual lines, it may in fact consist of only 4 strokes. This can occur because a stroke, unlike a line, is created without removing the pen from the paper. A detailed introduction to this concept was presented in our previous work [11]. We regard the strokes as the smallest units of Chinese character glyphs. Character-level distributed word representations are very useful for capturing implied information and have been successfully applied in a variety of NLP tasks, especially in alphabetic languages. Probably because of its peculiarities and drastic differences from alphabetic languages, Chinese has not attracted much attention in top NLP research forums. Inspired by the recent success of deep learning at the character level [6], [12], [13], we delve deeper, to the stroke level, for Chinese language processing. To the best of our knowledge, our work [1] is the first to treat strokes as the basic units for Chinese language processing. In a recent previous work [14] from Alibaba, a Chinese word vector representation based on simplified strokes has been proposed. Some researchers [3], [15] have realized that Chinese word embeddings learned with respect to different dimensions carry different information. Chen's work [3] is limited to words and characters, while [15] takes character components into consideration. However, strokes [1] and Pinyin [6] are far different from words and characters and thus can be further used in the joint learning of Chinese word embeddings.

In this paper, we learn a Chinese character embedding by exploiting the deeper information contained in Chinese characters and try to combine the information carried by strokes, Pinyin, characters, and words. We propose an algorithm for combining different features obtained by learning representations based on different dimensions in order to produce a better word embedding. The relevant algorithms can be executed in parallel at both the algorithm level and the data level. Furthermore, a series of experiments designed to verify the effectiveness of the proposed approach are reported. Based on the proposed method of stroke-level representation learning, we have developed a service for Chinese text classification and automatic text summarization. Additionally, we have published a stroke-enhanced Chinese character embedding that can be easily applied to other Chinese NLP tasks. An extended set of ROUGE evaluation criteria based on word similarity theory that considers semantics has been proposed. In addition, we are the first to construct a correspondence table between Chinese characters and stroke sequences and have developed a method of translating Chinese characters into stroke sequences.

II. RELATED WORKS

Word similarity, text classification, and automatic text summarization are all classic tasks in the natural language processing community. Word similarity is also known as the phenomenon of semantically related words (synonyms) and refers to the prediction of whether two words are semantically related. This task focus on relationships at the word level. The goal of text classification is to assign a predefined label to each document; thus, it is also known as document classification [16]. In the text classification task, the label of a document depends on the labels of the sentences that make up that document. Therefore, we regard text classification as a sentence-level task. The purpose of automatic text summarization is to extract information from an original text and compress it to provide users with a concise text description. A good summarization system should gain an understanding of the whole text and reorganize the extracted information to generate a coherent, informative and significantly shorter summary that conveys the important information about the original text [17], [18]. Through these three tasks, we can verify the effectiveness of stroke-based representation learning and multidimensional representation learning at the word, sentence, and document levels.

A. CHINESE CHARACTER EMBEDDING

In Zhang and Yanns work [6], Chinese is represented at the Pinyin level for text classification; thus, Chinese is effectively treated as an alphabetic language. Although working at the Pinyin level might be a viable approach, using strokes seems more reasonable from a linguistic point of view. In particular, Pinyin represents only pronunciation, which is arguably further separated from semantic meaning than strokes are. Suns work [9] and Shis work [8] operate at the radical level, with the precise meaning approaching the character component level. Compared with Chinese characters, character components represent a more fine-grained division. However, their granularity is still relatively coarse compared with that

of strokes. Radicals and character components are similar to Chinese characters in that they also face the problem of insufficient datasets. Some previous works have tried to apply the methods developed for alphabetic languages to Chinese in order to make the Chinese language more uniform with other languages. Some of them provide inspiration for a more fine-grained exploration, but their granularity is still not fine enough. Thus, we try to make Chinese language processing more universal by adopting an even more fine-grained approach.

B. TEXT CLASSIFICATION

Text classification is one of the most classic tasks in natural language processing. The aim of this task is to assign a predeïAned label to each document. A common approach to text classification is to use Bag-of-Words descriptors [19], N-grams [20], and the term frequency-inverse document frequency (TF-IDF) [21] as features and traditional models such as SVM [22] and Naïve Bayes [23] models as classifiers. Recently, however, many researchers [6], [12] have directed their attention to deep learning models, particularly convolutional neural networks, which have enabled significant progress in computer vision [24] and speech recognition. A convolutional neural network, as originally invented by LeCun [25] for computer vision, is a model that uses convolution kernels to extract local features. Analyses have shown that convolutional neural networks are effective for NLP tasks [26], [27]. Consequently, a convolutional neural network is selected as a major component of our model for text classification.

C. AUTOMATIC TEXT SUMMARIZATION

Depending on the method used, automatic text summarization can be divided into extractive summarization and abstractive summarization. Since the frequency of a certain vocabulary term in a document can reflect its importance to a certain extent, some works [28], [29] have used the probability of finding a word in a given sentence as the score for that word and have summed the probabilities of all words contained in a sentence as the score for that sentence. Works such as [30]–[32] have used highly scalable Bayesian topic models to model the topic relevance probability of a vocabulary term itself. Some research [33] has also considered the use of implicit semantic analysis or other matrix decomposition techniques to obtain low-dimensional implicit semantic representations. Research has shown that implicit semantic relationships affect the accuracy of automatic text summarization. Thus, we apply our stroke-based word embedding for the automatic text summarization task.

D. CONVOLUTIONAL NEURAL NETWORK

A convolutional neural network, as proposed by LeCun in the 1990s [25], [34], is a hierarchical neural network that extracts local features by convolving the input with a group of kernel filters. Convolutional neural networks involve many more connections than weights. The architecture itself imposes a form of regularization. In addition, a convolutional neural network automatically provides some degree of translation invariance. This particular kind of neural network assumes that we wish to learn filters, in a data-driven fashion, as a means to extract features describing the inputs. The convolutional feature maps thus obtained are then subsampled (in a process referred to as pooling) and filtered out to the next layer. In the following, we will briefly introduce the CNN algorithm.

In a convolution layer, the previous layer's feature maps are convolved with trainable kernels and fed through an activation function to produce the output feature map [35]. Each output map may be generated through convolutions with multiple input maps. In general, we have

$$x_j^l = f(\sum_{i \in M_j} x_i^{l-1} * k_{ij}^l + b_j^l) \tag{1}$$

Here, $x_i^l \in \mathbb{R}^{M_l \times M_l}$ represents the i^{th} feature map in the l^{th} layer, and $k_{ij}^l \in \mathbb{R}^{K_l \times K_l}$ represents the j^{th} kernel filter in the l^{th} layer related to the i^{th} map in the $(l-1)^{th}$ layer. M_j represents the selection of input maps from the $(l-1)^{th}$ layer related to the j^{th} map in the l^{th} layer. The output activation function $f(\cdot)$ is commonly chosen to be the logistic (sigmoid) function $f(x) = (1 + e^{-\beta x})^{-1}$ or the hyperbolic tangent function $f(x) = atanh(bx)$. Each output map is given an additive bias b; b_j^l denotes the bias for the j^{th} map in the l^{th} layer.

A subsampling layer produces downsampled versions of the input maps. If there are N input maps, then there will be exactly N output maps, although the output maps will be smaller. More formally,

$$x_j^l = f(\beta_j^l down(x_j^{(l-1)}) + b_j^l) \tag{2}$$

where $down(\cdot)$ represents a subsampling function. Typically, this function will sum over each distinct $n \times n$ block in an input map; consequently, the output map will be n times smaller in both spatial dimensions.

III. STARTING QUESTIONNAIRES

To verify the rationality of our proposed method, we designed a questionnaire-based experiment. The questionnaires were designed based on the existing knowledge base of the participating students. The questionnaires were divided into three main types of questions and two different types of knowledge. The questions were character spelling examination questions based on Pinyin, strokes or both. The characters used in the questions included characters already learned in class and characters to be learned in the future, which the students may not have already known during testing. We distributed the questionnaires to students in grade four and grade six who represents two stages of Chinese character learning. There are 293 students from grade four and 94 students from grade six. However, only 373 students participated in all three surveys.

With the assistance of Li Yijing, one of the authors of this article who is a teacher from elementary school, we conducted three questionnaires for primary school students in

the same class within two weeks. All the three sets of questionnaires contain the content and format described above. Because the students were trained to spell Chinese characters based on Pinyin in their daily learning, the accuracies achieved on the Pinyin-based questions in all three questionnaires were similar. However, spelling Chinese characters based on stroke sequences was a new experience for the tested elementary school students. Consequently, an accuracy of only 45% was achieved on the stroke-based questions in the first questionnaire, which was much lower than the corresponding accuracies achieved on the other two questionnaires. Therefore, we discarded all results from the first questionnaire and took the averaged results from the last two questionnaires as the final results, which are shown in Figure 6. Although the students were not very familiar with spelling words based on stroke sequences, they still achieved higher accuracies when using the stroke-based method than when using the method based on Pinyin. When confronted with untaught words, the students' accuracy with the Pinyin-based method dropped by more than 20 percent compared with that for learned words, while the accuracy achieved with the multidimensional method dropped by only 2 percent. Additionally, the accuracy of the stroke-based method was more than 35% higher than the accuracy of the Pinyin-based method. From these results, we can conclude that the stroke-based approach is much more useful when spelling unknown characters and that it is also effective to combine Pinyin and strokes.

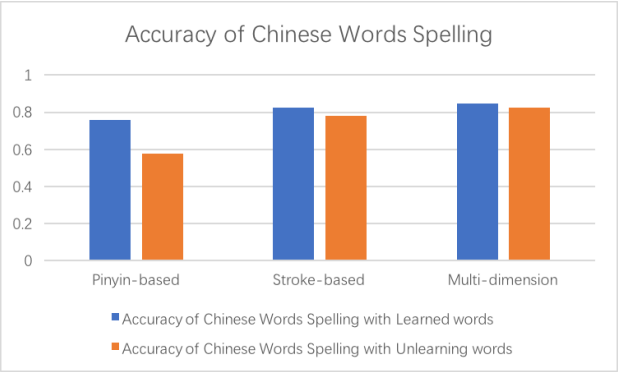


FIGURE 6. Accuracy of Chinese character spelling based on two types of knowledge and three types of questions.Two types of knowledge refer to learned words and unlearned words. Three types of questions refer to Pinyin-based spelling, Stroke-based spelling and multi-dimension based spelling.

IV. DESIGN OF THE STROKE-BASED NATURAL LANGUAGE PROCESSING SERVICE

In this section, we present our stroke-based model for learning a Chinese character embedding. Then, we use this stroke-based character embedding as the foundation for building a new service for natural language processing tasks.

A. STROKE-BASED CHINESE CHARACTER EMBEDDING

Due to the unique features of Chinese encoding, we need to translate Chinese characters into stroke sequences. Then, a stroke embedding will be learned by training a neural network, thereby introducing more structural features of Chinese characters than can be captured by one-hot vectors. Chinese-character-to-stroke-sequence translation plays an important role in the stroke-based Chinese character embedding training process. With the learned stroke vectors, each character can be represented by a matrix consisting of stroke vectors. A convolutional neural network was directly applied to such character matrices to generate a character embedding. Several activation functions were interspersed throughout to generate a character embedding with a fixed length.

There are some differences between character components and radicals. Although Chinese characters do not consist of letters, different structural parts may be formed into different Chinese characters. The most basic unit that has a meaning in itself is called a Chinese character component. Each Chinese character consists of one or several components. Radicals serve as indexing components in a Chinese dictionary; therefore, each Chinese character has only one radical. In other words, a radical is a type of character component. Therefore, we consider only character components.

A Chinese-character-to-stroke-sequence table was built that contains 20,877 Chinese characters, covering all commonly used Chinese characters and the Chinese characters that appear in the datasets we used. Translation can be easily performed based on this table, which is the cornerstone of this work. A one-hot vector is one possible choice for a stroke vector. However, a one-hot representation cannot reflect the semantic relations between strokes. To generate a better stroke representation, Erik Huangs framework, which is illustrated in Figure 7, was adopted to train the stroke vectors. Since our goal was to obtain stroke vectors, a small change was made in the framework to satisfy the necessary requirements. We replaced each document with a table of similar character components, which contained pairs of character components with the same semantic meaning. The model used the strokes in each character to compute a score for that character, called $Score_c$. Then, the strokes in each character component were replaced by the other similar character component in the corresponding pair to compute another score, called $Score_s$. The more similar the two components are, the smaller the difference between $Score_c$ and $Score_s$ is expected to be. The pretraining model used a back-propagation algorithm to generate the stroke vectors.

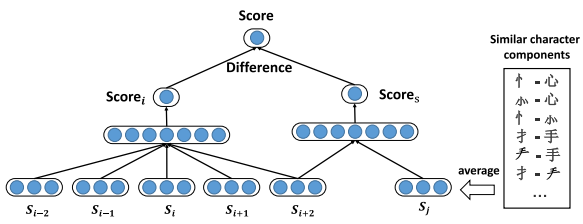


FIGURE 7. Stroke embedding learning considering similar character components.

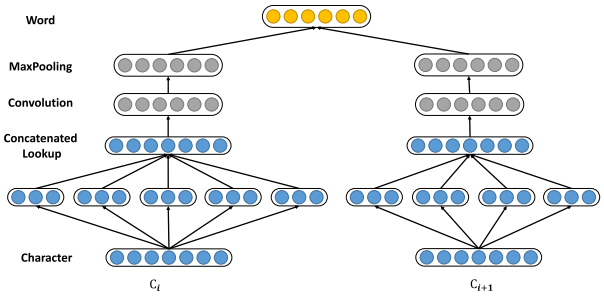


FIGURE 8. Architecture of the stroke-based Chinese character embedding learning model.

As illustrated in Figure 8, each C_i represents a Chinese character. The model leverages the Chinese-character-to-stroke-sequence table as a lookup table to map each Chinese character to several stroke vectors. The number of stroke vectors depends on the number of strokes the character contains. All of the stroke vectors in a character form a matrix that represents that character. Each stroke vector occupies one column in the matrix.

The main component in our model is the convolutional model, which simply computes several convolutions between the input and the convolution operator. Unlike in the convolution calculations performed in image processing with a fixed image size, different characters contain different numbers of strokes. However, all matrices have the same height D, which is the length of each stroke vector. The character size can be represented as $(D \times L)$. The model also considers several windows with different widths but the same height D for the convolution operator. The window size can be represented as $(D \times W_i)$. Every filter yields several results in the form of matrices with only one dimension. The number of results depends on the width of the window and the width of the character matrix. Due to the unfixed character width, the model uses pooling to make the length of the character vectors uniform. There are several kinds of pooling, such as max pooling over time, k-max pooling, and chunk-max pooling. In our model, chunk-max pooling is used in the pooling layer. Suppose that the window widths are denoted by w_i. The width of the widest window should be $w_{max} = max w_1, ..., w_n$, where n is the number of filters. C_i is the number of chunk partitions in the filter with a width of w_i. Therefore, the size of the character vector is $\sum_i^n C_i$.

Some special situations arise due to the unfixed character matrix size. If $l < w_i + C_i - 1$, the convolution calculation cannot be performed with a filter of width w_i and C_i partitions. The most common method of addressing this situation is to fill the necessary dummy elements with zeros. However, zero is too far away from the real values in this case. Therefore, a solution is proposed as follows:

① When w_i satisfies $w_i \leq l < w_i + C_i - 1$

For one or two windows, if the value l satisfies $w_i \leq l < w_i + C_i - 1$, the maximum possible number of partitions is $(w_i - l - 1)$. To preserve the fixed character vector length, C_i results are needed. Therefore, we generate $(C_i + l - w_i + 1)$ more results to fill this need by cyclically repeating the results.

② When w_i satisfies $l < w_i + C_i - 1$

When too many filters cannot satisfy $l \geq w_i + C_i - 1$ or any filter of width w_i satisfies $l < w_i$, we need to shrink the window size. To simplify this problem, we consider a unified number of partitions, C. The character vector size should now be $n * C$, which is equal to $\sum_i^n C_i$. We reselect the top n widths if l is larger than n. Otherwise, the widths could be 1 to l. We cyclically reuse the results and then combine the same results together.

The majority of Chinese words are composed of two characters. However, a Chinese word may also be composed of only one character or more than two characters. To simplify the problem of the number of characters per word, we average the vectors of all characters that form a Chinese word to obtain the corresponding word embedding vector.

B. CHINESE TEXT CLASSIFICATION SERVICE

To generate a better stroke-based Chinese character embedding, we translated Chinese characters into stroke sequences and trained stroke vectors using similar character components. Thus, we obtained a representation of the corresponding relationships between Chinese characters and strokes in the form of well-trained stroke vectors. To apply this stroke-based Chinese character embedding for text classification, we first translate new raw data into stroke sequences and then train and use a text classification module composed of a 6-layer convolutional network and a 2-layer fully connected neural network. Although the purpose is to apply the character embedding for text classification, the first layer in the text classification model takes stroke vectors as the input. The window size for the first layer depends on the average stroke length of the characters. The model is trained with the stroke sequence format of the original training data and generates labels for the newly input data as its output. Meanwhile, the newly arriving data in stroke sequence format will be appended to the training dataset. When enough new data have been appended to the training dataset, this will trigger a new round of training with the whole dataset. In our experiment, we set the retraining threshold to 5%, which means that it will become increasingly more difficult to trigger a new round of training as new data accumulate; this can reduce the time consumed for training the model.

As illustrated in Figure 9, we use general news text, which is always written in the form of Chinese characters, as the input in this architecture. A translation module is used to translate the Chinese characters into Chinese strokes. These strokes are used in the core module and are also appended to the training dataset. After enough newly arriving news data have been appended to the training dataset, the core module of the text classification service will be retrained.

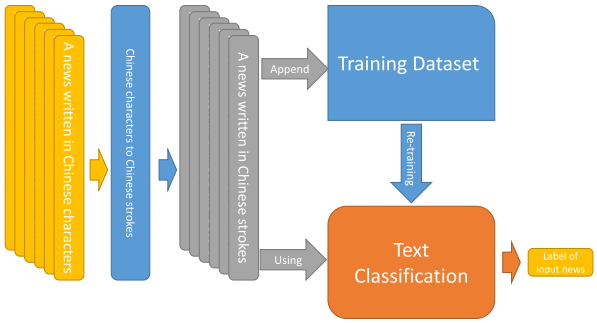


FIGURE 9. Changes in the state of the data in the architecture of the text classification service from raw data to the label of input data.

C. CHINESE AUTOMATIC TEXT SUMMARIZATION

Unlike in the text classification task, in automatic text summarization, it is necessary to generate a text that can be read. Therefore, in the text summarization architecture, we do not replace all Chinese characters with Chinese strokes. The decoder part of this architecture remains the same as what it should be to perform decoding. Before the encoder, the translation module used in the text classification task is still used to translate Chinese characters into Chinese strokes. However, for this task, the stroke vectors of each Chinese character are connected into a matrix. Three convolutional neural network layers act on these matrices that represent Chinese characters to generate the Chinese character embedding. The Chinese character embedding sequences calculated from the stroke vectors are fed into a recurrent neural network as input.

The architecture for automatic text summarization is as shown in Figure 10. When a sentence or document written in Chinese characters is provided as input, the translation module translates the Chinese characters into Chinese strokes as shown in Figure 9. Each stroke is represented by a vector as learned using the architecture shown in Figure 7, and these vectors are combined into a character matrix. A convolutional neural network and max pooling are used to transform the character matrices into the character embedding as shown in Figure 8. We adopt two deep architectures. The first one does not use local context during the decoding. We use a recurrent neural network as the encoder and its last hidden state as the input to the decoder, as shown by the gray blocks in Figure 10. The other architecture does consider context during decoding. We use the combination of all hidden states of the encoder as the input to the decoder, as shown by the green blocks in Figure 10. For the recurrent neural network, we adopt the gated recurrent units (GRUs) proposed by Chung *et al.* [36], which have been proven to be comparable to LSTM [37]. All parameters of the two architectures are randomly initialized, and ADADELTA [38] is used to update the learning rate. After the model is trained, beam search is used to generate the beset summaries during the decoding process.

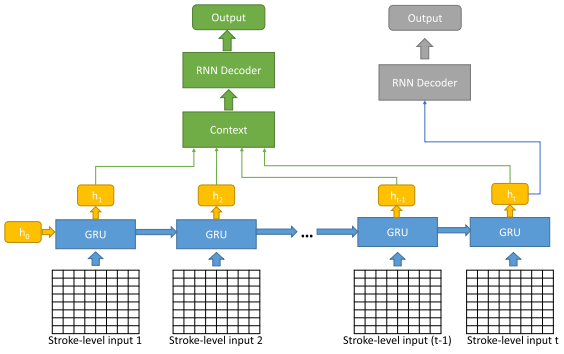


FIGURE 10. Architecture of the automatic text summarization service. In this figure, there two different architectures using different decoder frameworks, with and without context.

V. MULTIDIMENSIONAL REPRESENTATION LEARNING ALGORITHM

To learn as many feature dimensions as possible, we designed a Chinese word representation learning algorithm that can be executed in parallel. A common Chinese corpus is typically made up of sequences of Chinese characters and symbols (T_n), which can be regarded as the original dataset. The original dataset can be transformed into a segmented dataset (W_n) composed of words identified by means of Chinese word segmentation techniques. The original dataset can also be transformed into Pinyin form (P_n) and stroke form ($Str_(n)$). With these various types of datasets, we can independently train different word embeddings. In the word embedding training process for each dimension, the dataset can be divided into multiple segments for parallel training, as shown in algorithm 1.

After all embeddings based on the different dimensions have been trained, the final Chinese word representation is obtained by combining these embeddings from bottom to top with regard to their hierarchical levels relative to Chinese characters. We first combine the embedding based on Pinyin with the embedding based on strokes, which considered as the bottom level. The ensemble operation preserves the vector size of the word embedding. Then, the result of this first step is combined with the embedding based on Chinese characters after a convolution operation. Similarly, the embedding based on Chinese words is finally combined with the intermediate results.

As shown in Figure 11, the embedding of strokes and characters of Pinyin are at the bottom of this model. In practice, another form of Pinyin which use numbers to represent tones. Whats more, the strokes are also represented by letters. Thence, for the lowest granularity representation we use a unified symbol system including characters, numbers and other symbols. The number of characters of Pinyin for a single Chinese character is no more than 7 even with the tones. Therefore, in the experiment of this article, the Pinyin sequence is connected to the stroke sequence as input to the convolutional layer. Similar to the stroke-based embedding, we get a Chinese character embedding based on stroke sequence and Pinyin sequence of the Chinese character.

Algorithm 1 Algorithm for the Joint Learning of a Multidimensional Chinese Word Representation

Input: T_n: Training dataset consisting of sequences of Chinese characters and symbols; $S(x)$: Chinese word segmentation function that groups suitable characters into words; $P(x)$: Transformation function that transforms Chinese characters into Pinyin; $Str(x)$: Transformation function that transforms Chinese characters into stroke sequences; V: Vector size;

Output: $E_M(V)$

for each *Sample* in T_n **do**
  $S(Sample)$ to generate W_n
  $P(Sample)$ to generate P_n
  $Str(Sample)$ to generate Str_n
end for
for each dataset (T_n, W_n, P_n, Str_n), train a respective word representation **do**
  Train $E_c(V)$ with T_n
  Train $E_w(V)$ with W_n
  Train $E_p(V)$ with P_n
  Train $E_{str}(V)$ with Str_n
end for
$E_s(V) = Ensemble(E_p, E_{str})$
$E_m(V) = Ensemble(Conv(E_s), E_c)$
$E_l(V) = Ensemble(Conv(E_m), E_w)$
$E_M(V) = Conv(E_l)$

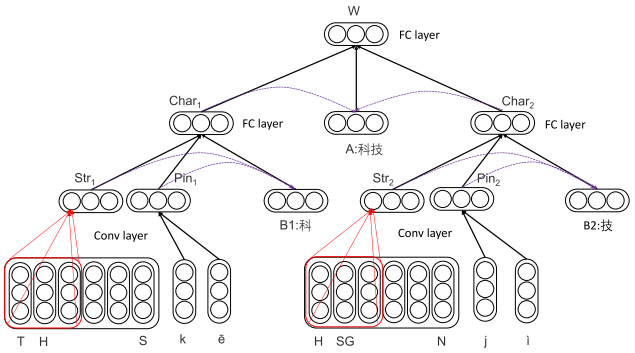


FIGURE 11. The architecture for multidimensional representation learning.

The output of the convolutional neural network will be integrated with the Chinese character embedding trained using the raw data. The most common method is to take the average of the two embedding. We try to use a fully connected layer instead of the average method to handle low-frequency Chinese characters and similar characters in glyph or pronunciation. Similarly, a fully connected layer is used in the word-level ensemble to deal with words out of vocabulary and the order of characters.

VI. EXPERIMENT

In this section, we report a series of experiments conducted to verify the effectiveness of the proposed stroke-based Chinese word representation and multidimensional Chinese word representation. In the next section, we introduce an idea for expanding the ROUGE criteria.

A. DATASETS AND EXPERIMENTAL SETTINGS

1) WORD SIMILARITY

We selected the two standard datasets used in Chen's article [3] and one small dataset built based on the same corpus for the word similarity computation. The corpus of news articles originates from The Peoples Daily and contains 31 million words. The vocabulary size is 105 thousand words and 6 thousand characters, which is far smaller than our character table but covers 96% of the characters in the national standard GB2312 character set.

2) TEXT CLASSIFICATION

For the text classification task, we chose the public dataset THUCNews as the basic dataset. THUCNews is based on Sina News historical data and includes 740,000 news articles in 14 classes.

As illustrated in table 2, we chose 4 of these classes: entertainment, sports, finance, and technology. From each category, 10,000 news articles were chosen for training, and 5,000 news articles were chosen for testing. Compared with the complete dataset, the selected dataset is not large, but the amount of data of each category is more balanced.

TABLE 2. Selected THUCNews dataset.

Class	Training Data	Testing Data
Entertainment	10,000	5,000
Sports	10,000	5,000
Finance	10,000	5,000
Technology	10,000	5,000
Sum	40,000	20,000

3) AUTOMATIC TEXT SUMMARIZATION

For the automatic text summarization task, we chose a large-scale Chinese short text summarization dataset [39] constructed from the Chinese microblogging website Sina Weibo. This corpus contains over 2 million real Chinese short texts, with short summaries provided by the author of each text and 10,666 tagged short summaries.

4) STROKE SELECTION AND PROCESSING

Generally speaking, there are 32 strokes in modern Chinese, as shown in Figure 12. There are some similar stroke pairs that can be represented using the same symbol. For instance, the fifth stroke in the first row and the last one in the first row can be regarded as the same stroke by ignoring the little hook. In this way, we ultimately compressed these 32 strokes down to 23 for use in our experiment, thereby allowing each stroke to be represented by a different letter of the English alphabet. We considered three ways to express strokes, which differ mainly in how the original English letters are handled. Through simple verification, we found that ignoring the original English letters is the simplest and most effective approach.

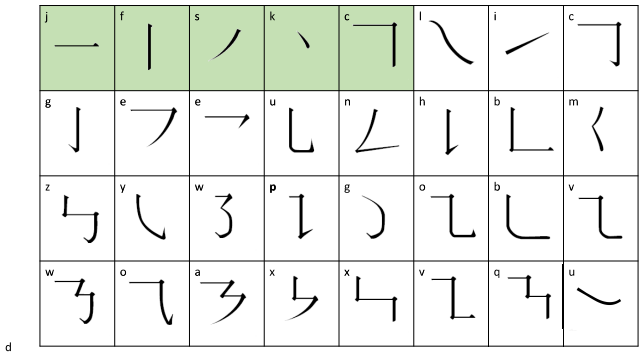


FIGURE 12. The 32 strokes used in modern Chinese, including 5 basic strokes (the first five with green background) and 26 composite strokes. All the 32 strokes are encoded with letters. And there are 9 pairs of strokes with same letter codes totally.

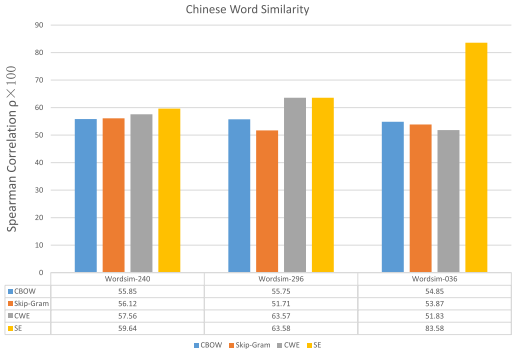


FIGURE 13. Results for the Chinese word similarity task obtained with cbow, skip-gram, CWE, and stroke-based embedding.

B. WORD SIMILARITY COMPUTATION

For this task, all models are required to compute the semantic relatedness of given word pairs. The correlations between the results of the models and human judgments are reported as the model performance. For this evaluation, we selected three datasets: wordsim-240, wordsim-296, and wordsim-36. In wordsim-240, there are 240 pairs of Chinese words and human-labeled relatedness scores. Of the 240 word pairs, 233 appear in the training corpus, and the remaining 7 word pairs consist of new words. In wordsim-296, 280 of the word pairs appear in the training corpus, and the remaining 16 pairs consist of new words. The two datasets described above were also used in Chen's article [3]. In addition, we built another dataset, wordsim-36, with 36 selected word pairs and human-labeled relatedness scores. Of these 36 word pairs, 34 appear in the training corpus, and the remaining 2 word pairs consist of new words.

From the evaluation results obtained on the wordsim datasets, we observe that stroke based embedding(SE) significantly outperforms the baseline methods on all three datasets. The wordsim-36 dataset consists of word pairs selected by considering similar character components. From the evaluation results obtained on wordsim-36, we can conclude that there is still a great gap that stroke-based Chinese character embedding can fill.

C. TEXT CLASSIFICATION

For this task, we used the same news data in different formats to train different core models. In addition, different translation models were naturally used. We turned off the trigger process to eliminate the overhead incurred for retraining, and we introduced exactly the same news data in exactly the same order as the input to the service for each different core model.

As shown in Figure 14, we observed that the average length of a single Chinese character in the stroke format is much longer than that in the Pinyin format. For the stroke format, we reset the filter window size for the first two layers of the core model from 7 to 15 to suit the longer input. However, we kept the window size of 7 for the Pinyin-format input.

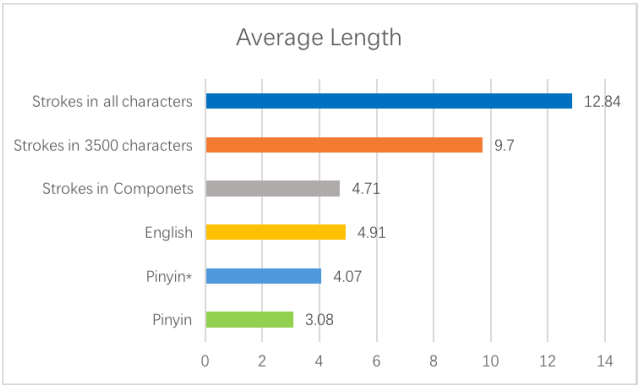


FIGURE 14. Average lengths of pinyin characters in two different formats, average lengths of strokes in Chinese component, first level Chinese characters and all Chinese characters we collected, and average lengths of characters in English words.

In our experimental environment, we obtained the results shown in Figure 15. From the evaluation results obtained for Chinese text classification, we can observe that the result for the stroke format is much better than that for the Pinyin format. We believe that the number of units in each layer and the size of the filter window strongly influence the results and that the small change applied to the filter window size here is still far from the optimal parameter for the model 15.

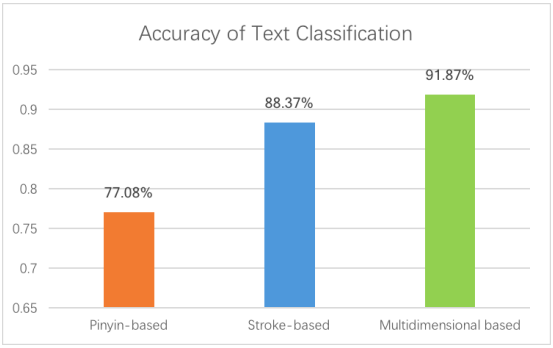


FIGURE 15. The Chinese corpora are translated into Pinyin format and stroke format and then applied to the text classification model. Also, multidimensional representation learned with the joint learning algorithm applied to the text classification task. Finally, we got the accuracy of text classification using the same dataset.

Also, the multidimensional representation based text classification get the best result. Although we obtained a better result with the stroke format than with the Pinyin format, the results did not fully meet our expectations. We will need to further optimize the core model in our future work. For the multidimensional representation learning, a neural-dynamic distributed scheme [40], [41] is introduced to guarantee the superiority, effectiveness and accuracy of our multidimensional representation learning algorithm.

D. AUTOMATIC TEXT SUMMARIZATION

For the automatic text summarization task, we used a quarter of the main dataset content (approximately 600,000 pairs) as training data to train a supervised learning model for summary generation. As shown in Table 3, a higher score indicates greater relevance between a short text and its corresponding summary. Thus, the pairs with scores of 3, 4 and 5 show high relevance in their summaries. These summaries are highly informative, concise and significantly shorter than the original text. We adopted these pairs, which constitute more than 80% of the data in Part II, as the testing dataset.

TABLE 3. Large-scale Chinese short text summarization dataset. The main contents of LCSTS consist of 2,400,591 short text summary pairs. There are 10,666 pairs with human-labeled scores ranging from 1 to 5 that indicate the relevance between the short text and the corresponding summary, where '1' denotes 'least relevant' and '5' denotes 'most relevant'.

Human Score	Part II	Part III
1	942	165
2	1,039	216
3	2,019	227
4	3,128	301
5	3,538	197
Sum	10,666	1,106

Two approaches were used, as in [39]: a Chinese-character-based method and a Chinese-word-based method. We extended both methods to the stroke level, and we constructed two deep architectures, with and without contextual consideration. Ultimately, we tested four methods based on two architectures, as shown in Table 4.

TABLE 4. The ROUGE-1, ROUGE-2, and ROUGE-L results obtained at the word level, the character level, the stroke-based word level, and the stroke-based character level using the two deep architectures with and without context.

	R-1	R-2	R-L
GRU-Word	0.173	0.081	0.156
GRU-Char	0.211	0.087	0.182
GRU-StrbasedWord	0.158	0.073	0.143
GRU-StrbasedChar	0.203	0.084	0.175
ATTN-Word	0.267	0.158	0.237
ATTN-Char	0.293	0.171	0.276
ATTN-StrbasedWord	0.258	0.143	0.222
ATTN-StrbasedChar	0.287	0.156	0.269

In the character-based method, we took Chinese characters as the input and reduced the vocabulary size to 4,000 characters, which covered more than 99.9% of the Chinese characters appearing in the text. In the word-based method, the text was segmented into Chinese words by using jieba [42]. The vocabulary was limited to 50,000 words, which is much larger than the number of commonly used Chinese vocabulary words. To generate summaries after the model was trained, beam search was used, with the size of the beam set to 10. For evaluation, we adopted the ROUGE metrics [43], which have been proven to be strongly correlated with human evaluations to a certain degree. ROUGE-1, ROUGE-2, and ROUGE-L were used, as in most text summarization tasks. The results are listed in Table 4.

From Figure 16, we can see that all three ROUGE results for the stroke-level-based methods are close to those for the word- and character-level methods. Still, the stroke-level performance did not exceed that of the original methods, as in the other two tasks. First, we used the stroke-level approach only to generate the embedding of the Chinese characters or words and then returned to the original method. This means that only a small part of the task was modified to consider the stroke level, which is still far from the target for this task. Second, the input to the encoder was based on the stroke level, but the output of the decoder was based on the original character- or word-level natural language model; consequently, the encoder input and the decoder output were not well matched with each other. Finally, automatic text summarization is widely regarded as a highly difficult problem. We can treat the word similarity task as a word-level task and text classification as a sentence-level task, but automatic text summarization must be treated as a higher-level task. Thus, it is more complicated than the other two.

VII. ROUGE-SN

ROUGE (Recall-Oriented Understudy for Gisting Evaluation) [43], [44] is an automatic summarization evaluation method for English summarization systems. We must translate Chinese words and characters into letters and numbers to use this system. Moreover, the basic ROUGE system relies on n-grams to evaluate the summaries, which requires absolute matching, as shown in formula 3.

$$ROUGE - N = \frac{\sum_{S \in Ref} \sum_{\eta \in S} C_m(\eta)}{\sum_{S \in Ref} \sum_{\eta \in S} C(\eta)} \tag{3}$$

The η represents n-gram which also applied in all formulas below in this section. From the word similarity task, we know that there are many synonyms and antonyms that can be used in place of each other to express the same meaning in a sentence. Take the summarization dataset as an example. The summaries in pairs with scores of 1 or 2 are highly abstracted and contain many words that do not appear in the original text, which means that a human evaluation will not yield a high ROUGE score. Therefore, we take the word similarity into consideration. As shown in formula 4, we take every Chinese word in the reference summaries and find the most similar word as the matching one. The similarity of exactly the same word will be 1; consequently, this metric will be the same as ROUGE-N if the generated summary is same as the reference one. Obviously, the result for $ROUGE - SN_0$ will

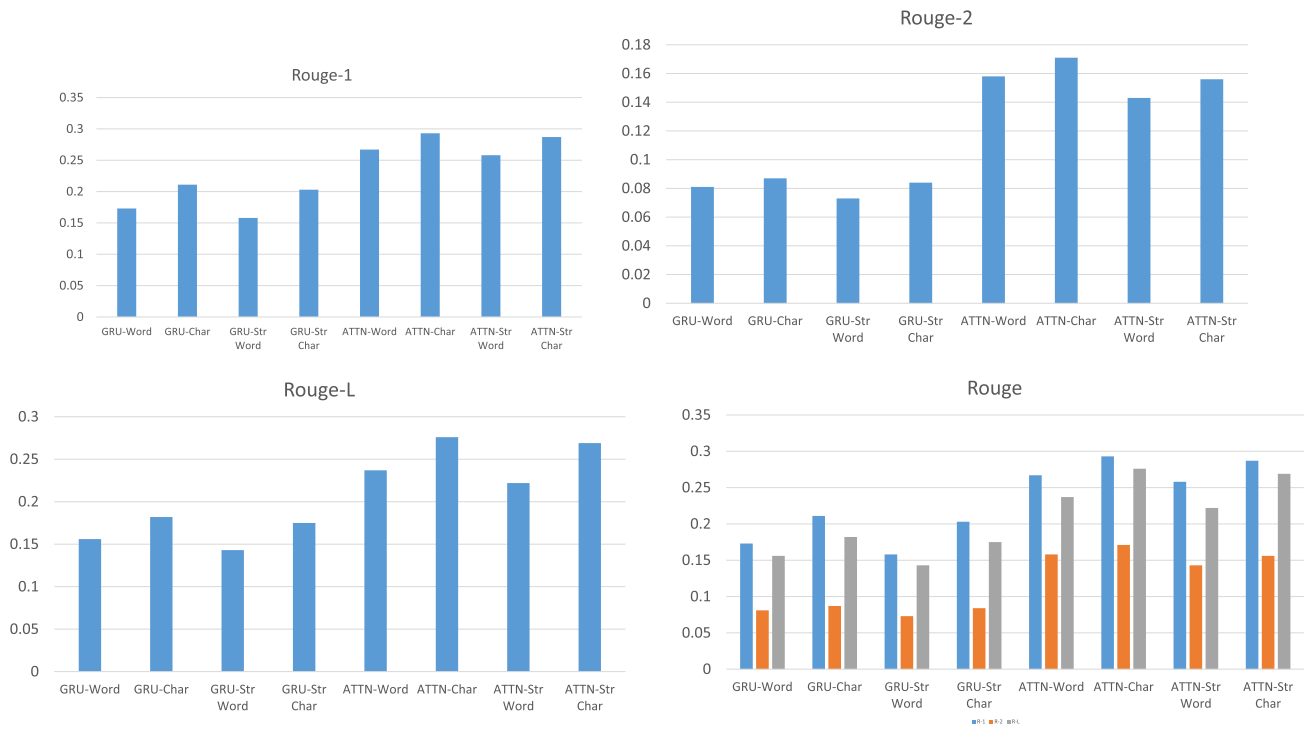


FIGURE 16. ROUGE-1, ROUGE-2 and ROUGE-L results obtained with the four methods using the two deep architectures.

generally be larger than that for ROUGE-N.

$$ROUGE - SN_0 = \frac{\sum_{S \in Ref} \sum_{\eta \in S} C(\eta) P_S}{\sum_{S \in Ref} \sum_{\eta \in S} C(\eta)} \tag{4}$$

However, in some cases, an irrelevant word will be identified as the most similar one. Therefore, a threshold is needed to remove instances in which we cannot find any words in the generated summary that are similar to a given word in the reference summary. Thus, only words with similarities higher than the selected threshold T will be used to calculate the result. This is shown in formula 5. In short, ROUGE-SN is an extension of ROUGE-N that considers the word similarity.

$$ROUGE - SN = \frac{\sum_{S \in Ref} \sum_{\eta \in S} C_{P_s \geq T}(\eta) P_S}{\sum_{S \in Ref} \sum_{\eta \in S} C(\eta)} \tag{5}$$

Finally, we take the average result for each task without stroke-level learning as the baseline and take the average result for each task based on stroke-level learning to represent the performance of our approach to determine the relative advantage of the latter. From the results, we find that the stroke-level representation is superior to the original for the word similarity and text classification task, while the performance achieved with the stroke-level representation is almost comparable to the state of the art for automatic text summarization. When ROUGE-SN is used as the evaluation standard, the stroke-level approach achieves a higher score than the original methods.

VIII. CONCLUSION AND FUTURE WORK

We proposed a method of leveraging stroke-level information for learning a continuous representation of Chinese characters. To the best of our knowledge, our work [1] is the first to treat strokes as the basic units of characters for Chinese language processing. In this paper, we proposed an algorithm for combining different features learned from different dimensions of Chinese words. To speed up training, the relevant algorithms can be executed in parallel at both the algorithm level and the data level. Our stroke-based model is capable of capturing the semantic relations between characters from a fine-grained level of expression. The effectiveness of our method has been verified on Chinese word similarity assessment, text classification, and automatic text summarization. The experimental results show that our method outperforms widely accepted embedding learning algorithms. More work needs to be done to take full advantage of the proposed multidimensional Chinese word representation in the future. Furthermore, we have developed a service based on the stroke-based word representation that can be easily applied to other Chinese language processing tasks. Moreover, we have expanded the ROUGE evaluation system by considering the influence of similar words on the evaluation of automatic text summarization results.

REFERENCES

[1] H. Zhuang, C. Wang, C. Li, Q. Wang, and X. Zhou, "Natural language processing service based on stroke-level convolutional networks for Chinese text classification," in *Proc. IEEE Int. Conf. Web Services (ICWS)*, Jun. 2017, pp. 404–411.

[2] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *J. Mach. Learn. Res.*, vol. 12 pp. 2493–2537, Aug. 2011.
[3] X. Chen, L. Xu, Z. Liu, M. Sun, and H.-B. Luan, "Joint learning of character and word embeddings," in *Proc. IJCAI*, 2015, pp. 1236–1242.
[4] M. Zhang, Y. Zhang, W. Che, and T. Liu, "Chinese parsing exploiting characters," in *Proc. 51st Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, 2013, pp. 125–134.
[5] M. Kang, T. Ng, and L. Nguyen, "Mandarin word-character hybrid-input neural network language model," in *Proc. 12th Annu. Conf. Int. Speech Commun. Assoc.*, 2011, pp. 625–628.
[6] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 649–657.
[7] X. Zhang and Y. LeCun. (2017). "Which encoding is the best for text classification in Chinese, English, Japanese and Korean?" [Online]. Available: https://arxiv.org/abs/1708.026571708.02657
[8] X. Shi, J. Zhai, X. Yang, Z. Xie, and C. Liu, "Radical embedding: Delving deeper to Chinese radicals," in *Proc. 53rd Annu. Meeting Assoc. Comput. Linguistics 7th Int. Joint Conf. Natural Lang. Process.*, vol. 2, 2015, pp. 594–598.
[9] Y. Sun, L. Lin, N. Yang, Z. Ji, and X. Wang, "Radical-enhanced Chinese character embedding," in *Proc. Int. Conf. Neural Inf. Process.* Beijing, China: Association for Computational Linguistics, 2014, pp. 279–286.
[10] Y. Li, W. Li, F. Sun, and S. Li. (2015). "Component-enhanced Chinese character embeddings." [Online]. Available: https://arxiv.org/abs/1508.06669
[11] H. Zhuang, C. Li, and X. Zhou, "CCRS: Web service for Chinese character recognition," in *Proc. IEEE Int. Conf. Web Services (ICWS)*, San Francisco, CA, USA, Jul. 2018.
[12] Y. Kim, Y. Jernite, D. Sontag, and A. M. Rush, "Character-aware neural language models," in *Proc. AAAI*, 2016, pp. 2741–2749.
[13] R. Jozefowicz, O. Vinyals, M. Schuster, N. Shazeer, and Y. Wu. (2016). "Exploring the limits of language modeling." [Online]. Available: https://arxiv.org/abs/1602.02410
[14] S. Cao *et al.*, "Cw2vec: Learning Chinese word embeddings with stroke n-gram information," in *Proc. AAAI Conf. Artif. Intell.*, 2018.
[15] J. Yu, X. Jian, H. Xin, and Y. Song, "Joint embeddings of Chinese words, characters, and fine-grained subcharacter components," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2017, pp. 286–291.
[16] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proc. 25th Int. Conf. Mach. Learn.*, 2008, pp. 160–167.
[17] E. Hovy and C.-Y. Lin, "Automated text summarization and the summarist system," in *Proc. Workshop Held*. Baltimore, MD, USA: Association for Computational Linguistics, Oct. 1998, pp. 197–214.
[18] D. Das and A. F. Martins, "A survey on automatic text summarization," *Literature Surv. Lang. Statist. II Course CMU*, vol. 4, pp. 192–195, Nov. 2007.
[19] Z. S. Harris, "Distributional structure," *Word*, vol. 10, nos. 2–3, pp. 146–162, 1954.
[20] W. B. Cavnar *et al.*, "N-gram-based text categorization," in *Proc. 3rd Annu. Symp. Document Anal. Inf. Retr. (SDAIR)*, Ann Arbor, MI, USA, 1994, no. 2, pp. 161–175.
[21] K. S. Jones, "A statistical interpretation of term specificity and its application in retrieval," *J. Documentation*, vol. 28, no. 1, pp. 11–21, 1972.
[22] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," in *Proc. Eur. Conf. Mach. Learn.*, in Lecture Notes in Computer Science: Lecture Notes in Artificial Intelligence, vol. 1398. Berlin, Germany: Springer, 1998, pp. 137–142.
[23] A. McCallum *et al.*, "A comparison of event models for naive bayes text classification," in *Proc. Workshop Learn. Text Categorization (AAAI)*, vol. 752, 1998, no. 1, pp. 41–48.
[24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.
[25] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
[26] A. Severyn and A. Moschitti, "UNITN: Training deep convolutional neural network for twitter sentiment classification," in *Proc. 9th Int. Workshop Semantic Eval. (SemEval)*, 2015, pp. 464–469.
[27] B. Zhang, J. Su, D. Xiong, Y. Lu, H. Duan, and J. Yao, "Shallow convolutional neural network for implicit discourse relation recognition," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2015, pp. 2230–2235.
[28] A. Nenkova and L. Vanderwende, "The impact of frequency on summarization," Microsoft Res., Redmond, WA, USA, Tech. Rep. MSR-TR-2005, 2005, vol. 101.
[29] L. Vanderwende, H. Suzuki, C. Brockett, and A. Nenkova, "Beyond SumBasic: Task-focused summarization with sentence simplification and lexical expansion," *Inf. Process. Manage.*, vol. 43, no. 6, pp. 1606–1618, 2007.
[30] H. Daumé, III, and D. Marcu, "Bayesian query-focused summarization," in *Proc. 21st Int. Conf. Comput. Linguistics, 44th Annu. Meeting Assoc. Comput. Linguistics*, 2006, pp. 305–312.
[31] A. Haghighi and L. Vanderwende, "Exploring content models for multi-document summarization," in *Proc. Hum. Lang. Technol., Annu. Conf. North Amer. Chapter Assoc. Comput. Linguistics*, 2009, pp. 362–370.
[32] A. Celikyilmaz and D. Hakkani-Tur, "A hybrid hierarchical model for multi-document summarization," in *Proc. 48th Annu. Meeting Assoc. Comput. Linguistics*. Baltimore, MD, USA: Association for Computational Linguistics, 2010, pp. 815–824.
[33] X. Wan, J. Yang, and J. Xiao, "Manifold-ranking based topic-focused multi-document summarization," in *Proc. IJCAI*, vol. 7, 2007, pp. 2903–2908.
[34] Y. LeCun *et al.*, "Handwritten digit recognition with a back-propagation network," in *Proc. Adv. Neural Inf. Process. Syst.*, 1990, pp. 396–404.
[35] J. Bouvrie, "Notes on convolutional neural networks," *Neural Nets*, vol. 11, p. 18, 2006.
[36] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Gated feedback recurrent neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 2067–2075.
[37] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. (2014). "Empirical evaluation of gated recurrent neural networks on sequence modeling." [Online]. Available: https://arxiv.org/abs/1412.3555
[38] M. D. Zeiler. (2012). "ADADELTA: An adaptive learning rate method." [Online]. Available: https://arxiv.org/abs/1212.5701
[39] B. Hu, Q. Chen, and F. Zhu. (2015). "LCSTS: A large scale Chinese short text summarization dataset." [Online]. Available: https://arxiv.org/abs/1506.05865
[40] L. Jin, S. Li, X. Luo, Y. Li, and B. Qin, "Neural dynamics for cooperative control of redundant robot manipulators," *IEEE Trans. Ind. Informat.*, to be published.
[41] L. Jin, S. Li, B. Hu, M. Liu, and J. Yu, "Noise-suppressing neural algorithm for solving time-varying system of linear equations: A control-based approach," *IEEE Trans. Ind. Informat.*, to be published.
[42] J. Sun, "'Jieba' Chinese word segmentation tool," 2012.
[43] C.-Y. Lin and E. Hovy, "Automatic evaluation of summaries using n-gram co-occurrence statistics," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics Hum. Lang. Technol.*, vol. 1. Baltimore, MD, USA: Association for Computational Linguistics, 2003, pp. 71–78.
[44] C. Flick, "ROUGE: A package for automatic evaluation of summaries," in *Proc. Workshop Text Summarization Branches Out*, 2004, p. 10.

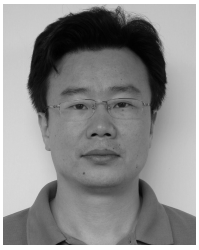
HANG ZHUANG (S'15) received the B.S. degree in computer science from the University of Science and Technology of China in 2012, where he is currently pursuing the Ph.D. degree with the Embedded System Lab. He has authored or co-authored over 20 publications, technical reports, and patents. His research interests are focused on cloud computing, machine learning, parallel and distributed computing, and natural language processing.

CHAO WANG (SM'17) received the B.S. and Ph.D. degrees from the the Department of Computer Science, University of Science and Technology of China, in 2006 and 2011, respectively. He is currently an Associate Researcher with the Embedded System Lab, Suzhou Institute, University of Science and Technology of China, Suzhou, China. He has authored over 60 publications and patents, including publications in the IEEE TC, ACM TCBB, and TACO, and through the FPGA conference. His research interests are focused on multicore and reconfigurable computing. He is a member of ACM and a Senior Member of CCF. He serves as the Publicity Chair for HiPEAC 2015 and ISPA 2014 and as a Guest Editor for TCBB and IJPP. He is currently an Editorial Board Member of MICPRO and IET CDT.

CHANGLONG LI (S'17) received the B.S. degree in computer science from the University of Science and Technology of China in 2012, where he is currently pursuing the Ph.D. degree with the Embedded System Lab. He has authored or co-authored over 20 publications, technical reports, and patents. His research interests are focused on mobile cloud computing and big data.

YIJING LI received the B.A. degree in Chinese literature from Jiaxing University, Jiaxing, China, in 2012, and the M.A. degree in Chinese education from Anhui Normal University, Wuhu, China, in 2014. She is currently with the Suzhou Industry Park Kuatang Experimental Primary School. She is a member of the Suzhou Calligraphers Association.

QINGFENG WANG received the M.S. degree in computer engineering from the Southwest University of Science and Technology, Mianyang, China, in 2014. She is currently pursuing the Ph.D. degree with the School of Software Engineering, University of Science and Technology of China. Her research interests include computer-aided diagnosis, machine learning, class imbalance learning, and distributed databases.



XUEHAI ZHOU is currently the Executive Dean of the School of Software Engineering, University of Science and Technology of China, and a Professor with the School of Computer Science. He serves as a General Secretary of the Steering Committee on Computer College Fundamental Lessons and the Technical Committee on Open Systems of the China Computer Federation. He has led many national 863 projects and NSFC projects. He has published over 100 international journal and conference articles in the areas of software engineering, operating systems, and distributed computing systems. He is a member of ACM and a Senior Member of CCF.