


Article

Tibetan Sentence Boundaries Automatic Disambiguation Based on Bidirectional Encoder Representations from Transformers on Byte Pair Encoding Word Cutting Method

Fenfang Li , Zhengzhang Zhao, Li Wang and Han Deng

College of Computer Science & Engineering, Northwest Normal University, Lanzhou 730070, China; 2022222306@nwnu.edu.cn (Z.Z.); 2021222195@nwnu.edu.cn (L.W.); 2021222276@nwnu.edu.cn (H.D.)

* Correspondence: lifenfang@nwnu.edu.cn

Abstract: Sentence Boundary Disambiguation (SBD) is crucial for building datasets for tasks such as machine translation, syntactic analysis, and semantic analysis. Currently, most automatic sentence segmentation in Tibetan adopts the methods of rule-based and statistical learning, as well as the combination of the two, which have high requirements on the corpus and the linguistic foundation of the researchers and are more costly to annotate manually. In this study, we explore Tibetan SBD using deep learning technology. Initially, we analyze Tibetan characteristics and various sub-word techniques, selecting Byte Pair Encoding (BPE) and Sentencepiece (SP) for text segmentation and training the Bidirectional Encoder Representations from Transformers (BERT) pre-trained language model. Secondly, we studied the Tibetan SBD based on different BERT pre-trained language models, which mainly learns the ambiguity of the shad (“|”) in different positions in modern Tibetan texts and determines through the model whether the shad (“|”) in the texts has the function of segmenting sentences. Meanwhile, this study introduces four models, BERT-CNN, BERT-RNN, BERT-RCNN, and BERT-DPCNN, based on the BERT model for performance comparison. Finally, to verify the performance of the pre-trained language models on the SBD task, this study conducts SBD experiments on both the publicly available Tibetan pre-trained language model TiBERT and the multilingual pre-trained language model (Multi-BERT). The experimental results show that the F1 score of the BERT (BPE) model trained in this study reaches 95.32% on 465,669 Tibetan sentences, nearly five percentage points higher than BERT (SP) and Multi-BERT. The SBD method based on pre-trained language models in this study lays the foundation for establishing datasets for the later tasks of Tibetan pre-training, summary extraction, and machine translation.

Keywords: sentence boundary disambiguation; Tibetan; pre-trained language model; BERT (BPE); shad (“|”)



Citation: Li, F.; Zhao, Z.; Wang, L.; Deng, H. Tibetan Sentence Boundaries Automatic Disambiguation Based on Bidirectional Encoder Representations from Transformers on Byte Pair Encoding Word Cutting Method. *Appl. Sci.* **2024**, *14*, 2989. <https://doi.org/10.3390/app14072989>

Academic Editor: Jose Machado

Received: 22 February 2024

Revised: 27 March 2024

Accepted: 28 March 2024

Published: 2 April 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Sentence boundary disambiguation (SBD) is a fundamental task in natural language processing (NLP), which is crucial for understanding the structure and semantics of sentences [1]. Humans are good at their languages and can quickly determine the location of sentence boundaries when reading a passage based on linguistic conventions or grammar. Humans cannot only determine sentence boundaries through punctuation but also rely on information about sentence meaning to assist in sentence boundary disambiguation. However, very often, when our task requires a large amount of data, sentence segmentation by manual methods is labor intensive. It can lead to inconsistent segmenting results due to differences in each person’s understanding of sentences [2].

From a more macroscopic point of view, studying Tibetan clauses is of great significance in promoting the overall development of the Tibetan NLP field. Through the research and improvement of clause technology, we can promote the progress of other NLP

tasks in the Tibetan language field and improve the automation level of Tibetan information processing. This helps promote the dissemination and communication of Tibetan culture and supports the informatization and modernization of Tibetan areas. Tibetan clause-splitting technology also shows many application prospects in practical application scenarios. Whether in news reporting, social media, or literature, Tibetan clause-splitting technology can help people process and analyze Tibetan texts more efficiently and improve work efficiency and accuracy. With the continuous development and improvement of the technology, it is believed that Tibetan clause-splitting technology will play a more significant role in more fields in the future. The Tibetan clause task has essential applications and significance in Tibetan NLP. It supports subsequent NLP tasks and is crucial in several practical application scenarios. With the deepening of research and the advancement of technology, the Tibetan clause task will play an even more critical role in the future [3].

Tibetan is an ancient language belonging to the Tibetan–Burmese branch of the Sino-Tibetan language family, in which the SBD task is even more challenging because sentence boundaries are often unclear. Most existing Tibetan sentences are labeled by rules and statistical learning methods and then proofread by hand. Still, the efficiency could be better in the case of large amounts of data or high data demand. Therefore, batch data processing using deep learning techniques is essential. By training the model on a large amount of Tibetan text data, the model learns the syntactic structure and patterns of Tibetan text to determine sentence boundaries accurately [4]. At this stage, NLP-based research includes basic tasks such as sentence segmentation, word segmentation, linguistic annotation, and syntactic analysis, as well as downstream tasks such as machine translation, dialog generation, information extraction, and summary generation. Most of these tasks require sentences as input units, so the merits of SBD tasks directly affect the efficiency of downstream tasks [5]. Tibetan has its unique script, phonetics, grammatical features, and grammatical rules, and is a phonetic script. These foundations are still in use in the modern Tibetan script. The Tibetan language is a phonetic script spelled by phonetics. The modern Tibetan language consists of 30 consonant letters and four vowel letters [6].

Units separated by a tsheg (""") in Tibetan are called syllables, and a Tibetan syllable corresponds to a word in English. A Tibetan syllable may have at least one consonant and up to seven phonemes. The seven parts are root, prefix, superfix, subfix, vowel, suffix, and postfix. Each letter in the syllable is called a component, and components are the constituent parts of a syllable and equal to characters in English. In Tibetan encoding, the character is the basic unit of the computer display, printing, and counting. Tibetan has both horizontal and vertical spelling, and this two-way spelling is a significant feature of Tibetan [7].

In traditional Tibetan grammar, a sentence is a linguistic unit consisting of two or more syllabic words that can express the name of a transaction connected with a grammatical auxiliary or an ordinary dummy word to express the distinction of the transaction. Tibetan sentences contain various dummy words, among which gerunds mainly express semantic relations between noun components and verbs. Tibetan sentences can be divided into long sentences (compound sentences) and short sentences (single sentences). Nouns with grammatical auxiliaries form short sentences, while long sentences can express a complete meaning and usually have a dummy word at the end of the sentence to indicate the end of the sentence.

A complete sentence in Chinese or English has a distinct end punctuation mark, usually a period, question mark, exclamation point, semicolon, or ellipsis to indicate the end of a sentence. In Tibetan, due to the problem of the partitive class of end-of-sentence dots and intra-sentence dots, the end-of-sentence dots cannot be used as sentence clauses or syn-copation markers [8]. For example, “བཀྲ་ཤིས་རྒྱལ་སྐྱོད་ནི་དཔལ་ལྷན་དང་། ① སང་མ། ② གཏུགས། ③ རྩེ། ④ འཁོར་ལོ། ⑤ རྒྱལ་མཚན། ⑥ བུམ་པ། ⑦ གསེར་ཉ་བཅས་སོ། ⑧” (the Eight Auspicious Treasures are the auspicious knot, the incredible lotus, the precious umbrella, the conch, the gold-lipped block, the precious vase, and the golden fish.) In the “Auspicious Eight Treasures”, the shad at ①~⑦ is an intra-sentence shad, and only the shad at ⑧ is an end-of-sentence shad. When

divided into shad, the sentence will be divided into several non-sentence units such as “བཀྲ་ཤིས་རྒྱལ་པོ་འཕྲུལ་པའི་ཐུགས་རྒྱུ་ལྟར་དུ་དང་།” and “པད་མ་”. The problem of concatenating end-of-sentence and intra-sentence shad in Tibetan has caused significant difficulties in automatic clause splitting [9].

In addition, Tibetan is rich in the phenomenon of linguistic partitions; for example, in the dictionary samples listed in the above literature, there are several noun-verb partitions such as “རྒྱུ་”, “མིང་”, and “སྐོར་” [10]. The text does not propose any corresponding disambiguation strategy or method, which leads to the fact that when splitting a sentence; the noun-verb partitions will be taken for the verb of the sentence end, resulting in the splitting of many incorrect sentences [11].

We now provide an example of a sentence ending in “g-” without “shad” and discuss other uses of “shad”. In Tibetan, sentences ending in “g-” usually denote statements or commands and often do not need “shad” to mark the end of the sentence. Note the following example: “དེ་ལ་འདི་ཡག་པོ་འདུག་” (This is a great book). In this example, the sentence ends with “ཡག་” indicating a complete declarative sentence without the need for “shad”. As for other uses of “shad”, in addition to marking sentence structure, it can also be used to enumerate elements in a list. In this case, “shad” may appear after each enumerated item to help the listener or reader distinguish between different items. For example, note the following: “གཅིག་། གཉིས་། གསུམ་།” (one, two, three.) In this enumeration example, “shad” appears after each number, indicating that three elements are enumerated.

To summarize, shad in Tibetan is a multi-functional auxiliary that plays various roles in a sentence, such as marking structure, linking clauses, and enumerating elements. By understanding these uses, we can better analyze and understand the complex sentence structure of Tibetan. The functions of standard punctuation in modern Tibetan are shown in Table 1; this study focuses on Tibetan SBD based on the shad (“།”).

Table 1. Modern Tibetan punctuation marks and their functions.

Punctuation Marks	Function
Tsheg (“་”)	The Tibetan tsheg is used to divide syllables.
Shad (“།”)	The Tibetan shad indicates the discourse’s pause, turning point, or end.
Double Shad (“།།”)	The application of double shad is the same as the shad, but it emphasizes that a passage or sentence has ended.
Quadruple Shad (“།།།།”)	The quadruple shads are used in chapters, volumes, and book endings. It belongs to Tibetan stylistic symbols.
Rinchenspung Shad (“༏”)	The left end of the sentence begins with a Rinchenspung shad when it is less than three letters.
Snake-Shaped Shad (“༏”)	In a long Tibetan text, when the last syllable of a sentence is at the first position of a line, the syllable is separated by a snake-shaped shad.
Double Ornament (“༏༏”)	The Tibetan double ornament is mainly used at the beginning of articles and books to indicate the start of the text.

With the development of the global Tibetan language community and the popularization of digitization technology, the data resources of Tibetan text are gradually enriched, providing favorable conditions for developing Tibetan pre-training technology [12]. In recent years, some research has been attempted to pre-train Tibetan, and some results have been achieved. The core idea of the pre-trained language model is to use unlabeled text data to pre-train the model so that the model learns the intro, mosaic structure, and language pattern. Currently, models such as ELMo [13], GPT [14], and BERT [15] have become benchmark models in NLP. However, the development and application of pre-training techniques still face many challenges for low-resource languages like Tibetan [16].

In the pre-trained language model, the training unit is fundamental, and many tasks use the word as the minimum processing unit, and a word is a token. In Chinese text encoding, we can use “character” as the encoding unit, and in English processing, we can use “letter” or “word” as the primary encoding unit. The use of “letters” as the coding unit is prone to the problem of long coding length, which leads to a high computational cost of the model. When using “word” as the basic unit, the number of “words” is fixed, and words beyond the dictionary’s scope cannot be processed, resulting in the instability of the word list. In deep learning model training, the word list is too extensive, which leads to computational overload; most models discard low-frequency words during training or fix the word list size [17]. Although such a setup reduces the computational load, it also encounters the Out of Vocabulary (OOV) problem. The simplest solution to the OOV problem is to label these words as Unknown (UNK) uniformly. However, too many words labeled “UNK” during training will affect the model’s generalization ability. Based on this, people began to study different methods to solve this problem, namely, the subword cut algorithm. Subword refers to the subunit in a word, which is a method of dividing words or phrases into smaller units [18], with a granularity between words and letters. For example, “subword” can be divided into two subwords: “sub” and “word” [19]. Commonly used encoding methods are Byte Pair Encoding (BPE) [20], byte-level BPE (BBPE) [21], Wordpiece [22], and Sentencepiece [23].

Aiming at the current research status of Tibetan SBD, the need for downstream task dataset establishment, and Tibetan grammar semantic analysis, this study explores the automatic recognition of Tibetan sentence boundaries based on the BERT pre-trained language model combined with the classical deep learning model through which the computer learns the meaning of the sequence composed of each shad (“|”) and the text preceding it, which transforms the SBD problem into a binary classification problem to determine whether the current shad (“|”) is an actual sentence end marker. Since the current publicly available pre-trained language models for Tibetan are generally effective on the SBD task we first train a BERT pre-trained language model for Tibetan in this study based on BPE and Sentencepiece and verify the performance of the model by combining CNN [24], RNN [25], RCNN [26], and DPCNN [27] on the two pre-trained language models. Meanwhile, to verify the effectiveness of the data in this study, experiments are also conducted on the publicly available TiBERT [28] model and Multilingual BERT (Multi-BERT) [29], which proves the effectiveness of the SBD data in this study on multiple pre-trained language models, as well as the suitability of Tibetan SBD tasks for the BERT pre-trained language model based on the BPE word cutting approach. Meanwhile, to reduce the model’s computational complexity and training cost, the experiment selects the number of token sequences in the window before the punctuation mark to participate in the training and testing. Finally, it proves the high efficiency of the BERT(BPE) deep learning-based model in the Tibetan SBD task.

2. Related Work

Most NLP technologies develop first with English, then German, French, Chinese, and then low-resource languages such as Tibetan. The research for SBD tasks lies in new features and models that effectively distinguish between bounded and unbounded. For languages such as English, where NLP research has been conducted earlier, most SBD is based on machine learning methods such as Decision Tree (DT) [30], Multilayer Perceptron (MLP) [31], Hidden Markov Model (HMM) [32], maximum entropy (ME) [33], Conditional Random Field (CRF) [34], and other models [35].

NLP techniques for languages such as English provide ideas and references for the study of Tibetan, and the study of English SBD is mainly concerned with determining whether an abbreviated period (“.”) is an actual sentence boundary or just an abbreviation representing a particular position. Scholars have conducted studies based on rule-based and statistical learning methods to address this issue. Read et al. [36] counted 75,000 English scientific abstracts in which 54.7–92.8% of the sentence dots appeared at the end,

about 90% indicated the end of the sentence, 10% indicated abbreviation, and 0.5% both. Mikheev [37], based on determining whether the words to the left of a potential sentence boundary or the right side of a word are an abbreviation or a proprietary name, uses a set of rules to implement SBD. Mikheev [38] combines the approach in the literature [37] with lexical tagging methods for supervised learning, including tagging for sentence endings, to reduce the error rate. Kiss and Strunk [39] propose a completely unsupervised system called PUNKT. The system is rooted in recognizing acronyms by finding collocation keys between candidate words and sentence points. Riley [40] proposed a DT classifier to determine whether acronyms mark sentence boundaries. It utilizes the probability of a word being the end or beginning of a sentence, word length, and word case as features to perform SBD. Reynar et al. [41] used supervised ME learning to study SBD by considering sentence segmentation as a disambiguation task with good results. Gillick et al. [42] used SVM to study SBD in English by using a large amount of training data through an SVM model to determine the function of abbreviated periods in English.

For Tibetan, rule-based research, machine learning, and a combination of both have been applied to the study of SBD. A rule-based approach based on auxiliary suffixes to detect sentence boundaries in Tibetan was proposed by Zhao et al. [43], which provided a preliminary analysis and exploration of the syntactic features of Tibetan legal texts. Ren and An [44] proposed an SBD method for constructing three lexicons (ending word lexicon, non-ending word lexicon, and unique word lexicon), which transforms the SBD problem into a query of to which lexicon the word to the left of a shad ("།") belongs. Cai [45] proposed a verb-centered binary SBD method based on maximum entropy. First, the maximum entropy model detects Tibetan sentences by grammar rules and thesaurus and then further identifies ambiguous sentences. Based on the literature [45], Li et al. [46] proposed a rule and maximum entropy-based approach. The ambiguous sentence boundaries are first identified using Tibetan boundary word lists. Then, the maximum entropy model is used to identify the ambiguous sentence boundaries that the rules cannot recognize. This is the first time that rules and machine learning methods have been combined for the Tibetan SBD task. Ma et al. [47] proposed a Tibetan SBD based on linguistic tagging. Firstly, the text is segmented into words and lexically tagged; then, the text is scanned; when scanning a shad ("།") or a double shad ("།།"), it determines whether the word to the left of the shad ("།") or double shad ("།།") is a conjunction or not and whether the position of the word is a noun, a numeral, or a state word, if yes, the model will continue scanning; otherwise, sentence segmentation is performed. Zhao et al. [48] studied the SBD method for Modern Tibetan auxiliary verb endings, which first identifies the auxiliary verb to the left of the shad ("།"), then determines whether the auxiliary verb to the left is a verb by the auxiliary verb, and finally considers whether the syllable number of the sentence is more significant than seven and segments it from the location of the shad ("།"). Zha and Luo [49] extracted Tibetan sentences by reverse search of function word position and suffix lexical properties. The method improved the efficiency of Tibetan sentence extraction and identified 11 lexemes that mark the end of a sentence. Que et al. [50] investigated the problem of automatic recognition of tight cuneiforms ("།") in Tibetan based on rules and SVM. The method first builds a feature vocabulary using terminal words and tight wedge characters and then uses SVM for classification. Wan [51] investigated the rule-based SBD problem for Tibet by analyzing the concepts and properties of Tibetan sentences and statistically studying the forms of Tibetan sentence endings and sentence-final lexemes.

The above studies used rule-based, statistical learning, and a combination of the two to solve the SBD problem in modern Tibetan from different perspectives. However, the corpora used in the above studies are oriented to different research focuses and have not been publicized. Among them, the rule-based approach requires researchers to have a high level of linguistic and grammatical foundation, and they need to construct the corresponding word and lexical dictionaries that can identify the sentence boundaries in advance; at the same time, the rule-based study needs to perform the tasks of participle and linguistic annotation, and based on the principle of error amplification in the deep learning model,

the performance of participle and lexical labeling performance has a certain degree of influence on SBD research.

3. Tibetan SBD Based on BERT (BPE)

In this study, Tibetan sentence boundary disambiguation based on BERT (BPE) is investigated, which mainly includes the exploration of the BPE-based Tibetan subword method, the training of the BERT (BPE) pre-trained language model, and the disambiguation of Tibetan sentence boundaries based on BERT and its improved model, and this study chooses BERT-DPCNN as an example of the improved model to be introduced. Figure 1 is the structure of the BERT-DPCNN model.

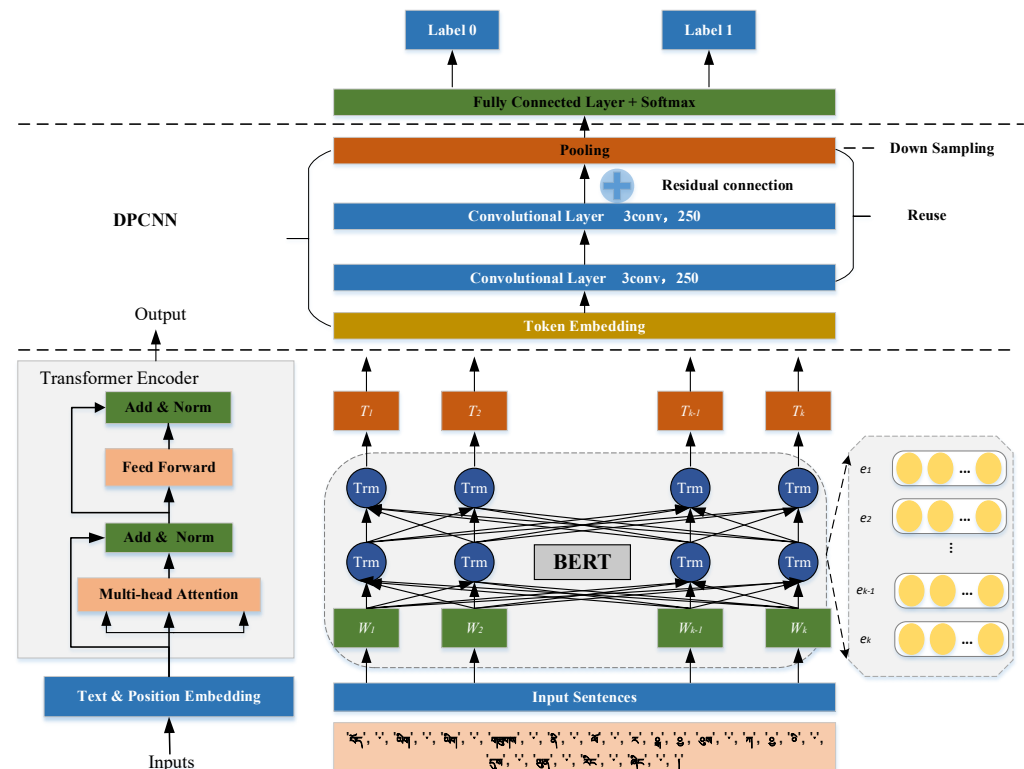


Figure 1. Architecture of the BERT-DPCNN model.

3.1. Subword Method of BPE

In the process of NLP training, the input unit of the training text is crucial, and there are differences in the impact of different input units on the results. The commonly used subword methods in the BERT pre-trained language model are BPE, Sentencepiece Wordpiece, etc., and the subword methods are closely related to linguistic features. In this study, we researched the Tibetan BERT pre-trained language model based on two subword methods, BPE and Sentencepiece. BPE is one of the essential coding methods in NLP, and its effectiveness has been proved by the most powerful language models such as GPT-2, RoBERTa, XLM, FlauBERT, and so on. Since it was found during the preliminary attempts that the Wordpiece subword method would have lost information such as vowels, resulting in incomplete information of Tibetan after word cutting, and the units generated by the Sentencepiece subword method are longer, this study uses the BPE subword method during the research of Tibetan pre-trained language model. BPE, designed by Gage in 1994, is a statistically based sequence compression algorithm designed to solve the string compression problem and is widely used in subword slicing in NLP.

The basic flow of the BPE subword slicing method is as follows:

Step 1: Slice each word into a sequence of characters and add a special ending symbol to each character.

Step 2: Calculate word frequency: count the number of occurrences of each character sequence.

Step 3: Merge characters: find a pair of neighboring character sequences with the highest word frequency and merge them into a new character sequence. At the same time, update the word frequency table and recalculate the number of times each character sequence occurs.

Step 4: Repeat step 3 until the specified number of subwords is reached or all the character sequences in the word frequency table can no longer be merged.

Step 5: Final slicing: the initial word is sliced according to the final vocabulary list to obtain the subword sequences.

The pseudo-code for BPE word cutting is as Algorithm 1:

Algorithm 1 Byte-pair encoding

```

1: Input: a set of string  $D$ , target vocab size  $k$ 
2: procedure BPE( $D, k$ )
3:    $V \leftarrow$  all unique characters in  $D$ 
4:   while  $|V| < k$  do Merge tokens
5:      $t_L, t_R \leftarrow$  Most frequent bigram in  $D$ 
6:      $t_{New} \leftarrow t_L + t_R$  Make a new token
7:      $V \leftarrow V + [t_{New}]$ 
8:     Replace each occurrence of  $t_L, t_R$  in  $D$  with  $t_{New}$ 
9:   end while
10:  return  $V$ 
11: end procedure

```

This study has tried three subword methods: BPE, Sentencepiece, and Wordpiece. Table 2 is an example of three different subword methods. From Table 2, we can see that the average length of the sequence after Sentencepiece cutting is the longest, the length of Wordpiece cutting is the shortest, and many of them are single characters, while the length of the BPE is between Wordpiece and Sentencepiece. Meanwhile, we found that the result of the WordPiece cut word has the problem of losing some of the building blocks, so, in this study, we trained the BERT model for both the Sentencepiece and BPE word cutting methods.

Table 2. Tokenization results of a Tibetan sentence under three subword methods.

	Original Text:
Raw Text	<p>བོད་ཡིག་ཡིག་གཟུགས་ནི་ལོ་རྒྱུས་ཀྱི་རྒྱུ་ཡུན་རིང་ཞིང་། རིག་གནས་ཀྱི་དོན་སྤྲིང་ཟབ་ལ་སྦྱ་ཅལ་གྱི་ཉམས་འགྱུར་ཕུན་སུམ་ཚྩ་གས་ བོ་བཅས་ལྟན་ པའི་ཡིག་རིགས་ཤིག་ཡིན།</p>
	English Translation: Tibetan Calligraphy Has a Long History, Deep Cultural Significance, and Rich Artistic Style.
BPE	<p>['བོད', ',', 'ཡིག', ',', 'ཡིག', ',', 'གཟུགས', ',', 'ནི', ',', 'ལོ', ',', 'རྒྱུ', 'ས་', ',', 'ཀྱི', 'འ', 'རྒྱུ', 'ཡུན', ',', 'རིང', ',', 'ཞིང', ',', '།', 'རིག', ',', 'གནས', ',', 'ཀྱི', 'དོན', ',', 'སྤྲིང', ',', 'ཟབ', ',', 'ལ་', ',', 'སྦྱ', 'ཅལ', ',', 'ཀྱི', 'ཉམས', ',', 'འགྱུ', 'ར་', ',', 'ཕུན', ',', 'སུམ', ',', 'ཚྩ', 'གས', ',', 'བོ', ',', 'བཅས', ',', 'ལ་', 'རྟ', 'ན', ',', 'པའི', ',', 'ཡིག', ',', 'རིགས', ',', 'ཤིག', ',', 'ཡིན', '།']</p>
WordPiece	<p>['བ', 'ོ', 'ད', ',', 'ཡ', 'ི', 'ག', ',', 'ཡ', 'ི', 'ག', ',', 'གཟ', 'ུ', 'གས', ',', 'ན', 'ི', 'ལ', 'ོ', 'ར', 'ྒྱ', 'ས', ',', 'ཀ', 'ྱི', 'ད', 'ུ', 'ས', ',', 'ཡ', 'ུ', 'ན', ',', 'ར', 'ི', 'ང', ',', '།', 'འ', 'ི', 'ང', '།', 'ར', 'ི', 'ག', ',', 'གན', 'ས', ',', 'ཀ', 'ྱི', 'ད', 'ོ', 'ན', ',', 'ས', 'ྤྲི', 'ང', ',', 'ཟབ', ',', 'ལ', ',', 'ས', 'ྦྱ', 'ར', 'ུ', 'ལ', ',', 'ག', 'ྱི', 'ཉམ', 'ས', ',', 'འག', 'ུ', 'ར', ',', 'པ', 'ུ', 'ན', 'ི', 'ས', 'ུ', 'མ', 'ི', 'ཚྩ', 'ོ', 'གས', 'ི', 'བ', 'ོ', 'བཅས', 'ི', 'ལ', 'ང', 'ན', 'ི', 'པའ', 'ི', 'ཡ', 'ི', 'ག', 'ི', 'ར', 'ི', 'གས', 'ི', 'ཤ', 'ི', 'ག', 'ི', 'ཡ', 'ི', 'ན', '།']</p>
SentencePiece	<p>['བོད་ཡིག', 'ཡིག་གཟུགས', 'ནི་ལོ་', 'རྒྱུས་', 'ཀྱི', 'རྒྱུ་ཡུན་རིང་', 'ཞིང་', '།', 'རིག་གནས་ཀྱི', 'དོན་སྤྲིང་', 'ཟབ་', 'ལ་', 'སྦྱ་ཅལ་གྱི', 'ཉམས་འགྱུར་', 'ཕུན་སུམ་ཚྩ་གས་བོ་', 'བཅས་ལྟན་པའི', 'ཡིག་རིགས་', 'ཤིག་ཡིན', '།']</p>

3.2. BERT Model Training

Devlin et al. [15] proposed a Transformer-based Bidirectional Encoder Model (BERT) in 2019, masking several words with unique tokens before predicting them and pre-processing the bidirectional representation of unlabeled text by jointly regulating left and right up and down in all layers. In this study, the Tibetan BERT pre-trained language model based on two subword methods, Sentencepiece and BPE, is applied in the SBD model. The Tibetan SBD task is investigated based on the BERT + deep learning model. The BERT uses the encoder part of the Transformer, which consists of multiple layers of bi-directional Transformer [52] encoder stacked together. Each layer of the Transformer's encoder consists of a multi-head attention mechanism and a feed-forward network with layer normalization. The feed-forward network consists of two fully connected layers and a nonlinear activation function RELU; Figure 2 shows the structure of the multi-head attention mechanism.

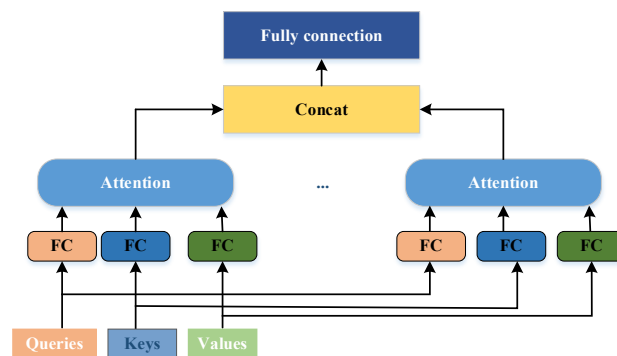


Figure 2. Multi-head attention mechanism.

The multi-head attention mechanism is an extension of the attention mechanism, which independently calculates attention weights in multiple representation subspaces, allowing the model to learn different attention patterns in different subspaces, thereby improving the model's representation ability. The attention mechanism creates three vectors before input: Query vector (Q), Key vector (K), and Value vector (V). Attention is computed in three general steps:

(1) Calculate the weights; firstly, Q and all K calculate the weights using the similarity method.

$$f(Q, K_i) = \begin{cases} Q^T K_i \\ Q^T W K_i \end{cases} \quad (1)$$

Normalization, using the Softmax function to normalize $f(Q, K_i)$:

(2) Sum the normalized weights with V weighting to obtain Attention:

$$Attention = (Q, K, V) = \sum_i a_i V_i \quad (2)$$

where Q , K , and V denote the query, index, and the value obtained from the query, respectively, all obtained from the input word vectors by linear transformation, and then the value of the Softmax function is solved by Equation (2), after calculating the attentional weights of each position to the other positions, the self-attention mechanism will generate a weighted vector for each position based on these weights, i.e., multiplying each position with the attentional weights of the other positions, and then summing up these products to obtain the weighted vector for that position, i.e., the attentional representation of that position [53].

In the multi-head attention mechanism, i.e., for a given query Q , essential K , and value V , each attention head h_i ($i = 1, 2, \dots, h$) is computed as follows:

$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V) \quad (3)$$

The multi-head attention mechanism is calculated as follows:

$$MultiHead(Q, K, V) = concat(head_1, head_2, \dots, head_n)W^Q \quad (4)$$

In this study, the specific process of sentence boundary disambiguation based on the Tibetan BERT pre-trained language model is as follows:

(1) Sentences are first segmented, and [CLS] markers are added to the beginning of each sentence, and [SEP] markers are added to the end of each sentence.

(2) The labeled sentences are input to the embedding layer: marker embedding, fragment embedding, and position embedding. Token embedding adds [CLS] tokens to the beginning of each sentence while separating each sentence fragment by [SEP]; fragment embedding is used to differentiate between multiple sentence fragments given, mapping the tokens to EA in odd-indexed sentences, and mapping the tokens to EB in even-indexed sentences. Positional embedding encodes the positional information of words in the input, as shown in Equations (5) and (6).

$$PE_{(pos, 2i)} = \sin(pos/10,000^{2i/d}) \quad (5)$$

$$PE_{(pos, 2i+1)} = \cos(pos/10,000^{2i/d}) \quad (6)$$

d denotes the dimension of the embedding, $2i$ denotes the even dimension of the embedding dimension, and $2i + 1$ is the odd dimension.

(3) The BERT model accepts these inputs, and after the model is trained, the context embedding representation of each token is the output. The [CLS] tokens in front of each sentence are entered into the output vector after modeling as the sentence vector representation of that sentence.

(4) Input the acquired sentence vector representation into the Softmax layer. Finally, for a simple binary classifier to determine whether the input text is to be disconnected from the current position or not, the judgment is shown in Equation (7).

$$Y_l = \sigma(W_0 R_i + b_0) \quad (7)$$

In this study, in addition to studying the Tibetan SBD based on the BERT model, we also compare the SBD based on the BERT + deep learning model, which is presented as an example of DPCNN.

3.3. Introduction to the DPCNN Model

Early deep learning models mainly include Convolutional Neural Networks (CNNs) [24] and Recurrent Neural Networks (RNN) [25]. TextRNN and TextCNN are model architectures for multi-label text classification problems. The TextRNN model adopts a Bi-directional Long Short-Term Memory (Bi-LSTM), where the input of the latter time step depends on the output of the previous time step, which cannot be processed in parallel and affects the overall process speed. TextCNN mainly relies on sliding windows to extract the features, which are limited in their ability to model long distances and are insensitive to the order of speech. Based on TextCNN, researchers propose the text recurrent convolutional neural network (TextRCNN [26]), in which the function of feature extraction in the convolutional layer is replaced by RNN, i.e., the feature extraction of TextCNN is replaced by an RNN. The advantage of RNNs is that it can better capture contextual information, which is conducive to capturing the semantics of long texts. Therefore, the overall structure becomes an RNN + pooling layer called RCNN. Based on TextCNN, Deep Pyramid Convolutional Neural Networks (DPCNN [27]) have been proposed, which is strictly the first word-level widely effective deep text classification convolutional neural network, which can also be understood as a more effective deep CNN.

As shown in the upper part of Figure 2, the DPCNN model mainly consists of region embedding, two equal-length convolutions, the Block region, and the residual connections represented by the plus sign. Among them, the Block region contains one-half pooling and

two equal-length convolutional layers, which is why the model is called the deep pyramid model. The main advantage of DPCNN is its ability to extract long-distance textual dependencies by continually deepening the network. This study investigates BERT-DPCNN as an example of an improved BERT+ deep learning model.

3.4. Sentence Boundary Disambiguation

Figure 2 is the network structure diagram of this study based on the BERT-DPCNN model as an example of studying the Tibetan SBD based on the BERT + deep learning model. After training on the BERT model to obtain the representation of each token, it is inputted into the deep learning model for the second stage of training. Finally, the label of each clause is obtained by the softmax function. Training data may contain several clauses, assuming that each clause is represented by S . The clause where the shad ("།") containing the end-of-sentence marker is labeled as a positive sample, and the other labels are negative samples. Suppose the clause S contains n tokens (where n is determined according to the length of (S)), then S is a sequence consisting of w_1, w_2, \dots, w_e . In this study, we use $p(k|S; \theta)$ to denote the probability that the shad ("།") at the end of the current clause fragment is the true clause token k , where θ is a parameter in the network.

After obtaining the vector representation x_i of the syllable w_i , a linear transformation will be performed with a \tanh linear activation function to send the result to the next layer.

$$y_i^{(1)} = \tanh(W^{(2)}x_i + b^{(2)}) \quad (8)$$

After all, the token is represented as a vector, and this study uses a maximum pooling layer to convert clauses of different lengths into fixed-length vectors that capture the information of the entire sentence.

$$y^{(2)} = \max_{i=1, \dots, n} y_i^{(1)} \quad (9)$$

The max function is the maximum function and the k_{th} element of $y^{(2)}$ is the maximum of the k_{th} element of $y_i^{(1)}$. The pooling layer uses the output of the loop structure as input. The model ends with the output layer, which is defined as follows:

$$y^{(3)} = W^{(3)}y^{(2)} + b^{(3)} \quad (10)$$

Finally, the output number $y^{(3)}$ is converted into a probability using the softmax function, i.e., the probability that the current clause's shad ("།") is an accurate end-of-sentence marker.

$$p_i = \frac{\exp(y_i^{(3)})}{\sum_{k=1}^n \exp(y_k^{(3)})} \quad (11)$$

4. Experimental Setup and Results

4.1. Experimental Setting

This study uses an NVIDIA V-100 GPU to train the model. The manufacturer of NVIDIA V-100 GPU is NVIDIA Corporation. The company is headquartered in Santa Clara, CA, USA. When the model is trained, each parameter's parameter settings are shown in Table 3. This study uses the Tibetan BERT pre-trained language model based on BEP and Sentencepiece subword methods. The checkpoints are saved each time the model is trained, and finally, the training ends when the loss does not drop for 1000 consecutive batches on the validation set, and the model is tested on the model with the highest performance checkpoints.

tokens before the shad (“|”) is preserved during training, the sequence is as follows: “ལྷོ་གཤམ་ལྷོ་པ་རེད།”. Figure 3 shows the statistics of clause lengths after shad (“|”) separation. According to the related research on determining the end position of a sentence by analyzing the words near the punctuation mark and the lexical properties of the words, in this study, we set the window sizes in the experiments to reduce the training cost, which are 3, 5, 10, and 15, respectively.

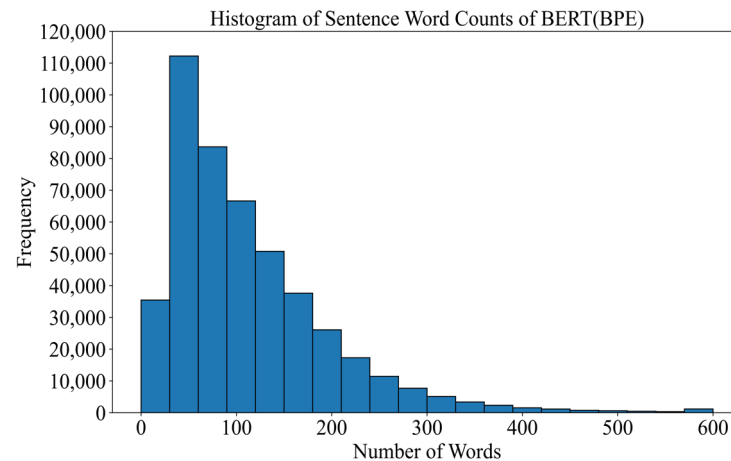


Figure 3. Frequency statistics of sentence length corresponding to the BERT (BPE) in this study.

4.3. Evaluation Metrics

To evaluate the model’s performance, the experiments in this study choose three necessary indicators for evaluating the deep learning model: precision, recall, and F1-score. Due to the large gap between the total number of positive and negative samples in the data and considering that the imbalance of the samples will lead to bias in the results, the weighted average of these three indicators is finally chosen to measure the model’s efficiency. In the experimental results listed subsequently, P, R, and F1 represent precision, recall, and F1-score, respectively.

4.4. Experimental Results

This study uses pre-trained language models to explore the Tibetan SBD method. To verify the reliability of the experimental data in the sentence boundary disambiguation task, the pre-trained language model of Tibetan BERT based on two subword methods, BPE and Sentencepiece, was firstly trained, and the experiments of Tibetan sentence boundary disambiguation were carried out on BERT (BPE), BERT (SP), the publicly available TiBERT model and Multi-BERT model. Meanwhile, related experiments were conducted on BERT combined with classical deep learning models (BERT-CNN, BERT-RNN, BERT-RCNN, BERT-DPCNN). In this study, the parameter settings for training the BERT pre-trained language model are consistent with Google’s publicly available BERT base parameter settings, and the Tibetan SBD experiments are conducted on the trained models.

4.4.1. Tibetan Sentence Boundary Disambiguation Based on BERT (BPE) and BERT (Sentencepiece)

In this study, we first validate the Tibetan sentence boundary disambiguation based on BERT (BPE) and BERT (Sentencepiece), and the experimental results are shown in Tables 4 and 5.

Table 4. Tibetan SBD based on BERT (BPE) pre-trained language models.

BERT (BPE)	Window = 3			Window = 5			Window = 10			Window = 15		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
BERT	94.91	94.92	94.9	94.62	94.59	94.55	95.19	95.2	95.19	94.91	94.89	94.9
BERT-CNN	94.94	94.93	94.9	94.76	94.73	94.69	95.32	95.32	95.32	94.15	94.16	94.15
BERT-RNN	94.82	94.81	94.78	94.9	94.9	94.88	95.12	95.12	95.1	94.73	94.7	94.71
BERT-RCNN	95.04	95.05	95.03	94.81	94.79	94.76	95.24	95.24	95.23	94.74	94.75	94.74
BERT-DPCNN	94.3	94.3	94.27	94.94	94.95	94.94	95.26	95.26	95.24	94.49	94.5	94.48

Table 5. Tibetan SBD based on BERT(SP) pre-trained language models.

BERT (SP)	Window = 3			Window = 5			Window = 10			Window = 15		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
BERT	88.55	88.58	88.56	90.21	90.26	90.20	90.60	90.63	90.61	91.47	91.49	91.47
BERT-CNN	88.57	88.55	88.56	90.51	90.51	90.51	90.88	90.88	90.88	90.61	90.64	90.62
BERT-RNN	88.74	88.75	88.75	90.34	90.36	90.35	94.47	90.51	90.47	45.23	38.54	37.33
BERT-RCNN	88.36	88.38	88.37	90.49	90.53	90.50	90.88	90.91	90.88	90.57	90.59	90.51
BERT-DPCNN	88.75	88.77	88.76	90.35	90.36	90.35	91.34	91.37	91.34	91.15	91.16	91.15

Tables 4 and 5 show that the performance of Tibetan SBD based on BERT (BPE) is about 5% higher than that of Tibetan SBD based on BERT (Sentencepiece). In the experiments of Tibetan SBD based on BERT (BPE), the highest F1 value is 95.32%, and the lowest is 94.9%; for BERT (BPE), on the BERT-CNN model, the highest F1 score is found when the window is 10, followed by the performance of the BERT-RCNN model when the window is 3, with an F1 score of 95.03%, and the performance of the model based on BERT (BPE) + deep learning model is 95.03%. The metrics differ when the model and window are different. However, the overall is around 95%. Table 5 shows the Tibetan SBD based on the BERT (SP) model practiced in this study, and the overall F1 of the Tibetan SBD performance based on BERT (SP) is between 88.76% and 91.47%. All the performances tend to increase during the process of increasing the window. Still, the maximum value is 3–4 percentage points different from BERT (BPE). Tables 4 and 5 show that the window change has a more noticeable effect on BERT (SP). At the same time, the performance of BERT(BPE) is more stable with the increase in the window, which also verifies the scientific and referable nature of the related literature in determining whether the shad (“|”) is an accurate end-of-sentence marker or not by analyzing the words on the left side of the shad (“|”) and the linguistic properties of the words.

4.4.2. Tibetan SBD Based on Publicly Available BERT Pre-Trained Language Models

To compare the effectiveness of the pre-trained language models in this study, the Tibetan SBD based on the publicly available pre-trained language model TiBERT and the multilingual model Multi-BERT are produced, and the Tibetan SBD based on the BERT+deep learning (BERT-CNN, BERT-RNN, BERT-RCNN, BERT-DPCNN) model is also compared with the performance. During the study of SBD based on a pre-trained language model, it is found that there are also Tibetan elements present in Multi-BERT, namely, “།”, “༎”, “ང”, “ག”, “ཆ”, “ད”, “ན”, “བ”, “མ”, “ཨ”, “ར”, “ལ”, “ས”, “ཤ”, “ེ”, “ཱ”, “ུ”, “ེ”, “ུ”, “ུ”, “ུ”, and “ུ”, which contain the commonly used Tibetan end-of-sentence punctuation mark “།” and the syllable separator “༎”. Multi-BERT cuts words based on Wordpiece, while TiBERT cuts words based on Sentencepiece. The experimental results based on TiBERT and Multi-BERT are shown in Tables 6 and 7.

Table 6. Sentence boundary disambiguation in Tibetan based on TiBERT.

TiBERT	Window = 3			Window = 5			Window = 10			Window = 15		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
TiBERT	89.6	89.58	89.59	91.45	91.47	91.46	90.85	90.88	90.86	90.31	90.36	90.32
TiBERT-CNN	90.02	90	90.01	90.27	90.25	90.26	90.38	90.39	90.39	90.43	90.47	90.44
TiBERT-RNN	90.05	90.05	90.05	90.38	90.4	90.39	90.38	90.41	90.39	91.34	91.32	91.33
TiBERT-RCNN	90.01	89.98	89.99	90.4	90.35	90.37	90.42	90.42	90.42	90.63	90.6	90.62
TiBERT-DPCNN	90.06	90.04	90.05	90.28	90.23	90.25	90.84	90.86	90.85	90.34	90.35	90.34

Table 7. Sentence boundary disambiguation in Tibetan based on Multi-BERT.

Multi-BERT	Window = 3			Window = 5			Window = 10			Window = 15		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
BERT	89.48	89.53	89.44	89.44	89.49	89.42	89.01	89.04	88.93	39.94	63.19	48.94
BERT-CNN	88.35	88.30	88.12	89.96	89.96	89.86	88.78	88.84	88.77	39.94	63.19	48.94
BERT-RNN	88.68	88.73	88.63	90.64	90.68	90.64	90.43	90.44	90.34	69.75	70.67	69.42
BERT-RCNN	89.26	89.31	89.27	89.59	89.63	89.54	91.05	91.07	91.00	73.34	37.32	20.95
BERT-DPCNN	88.66	88.72	88.66	89.45	89.45	89.45	90.32	90.29	90.17	13.55	36.81	19.8

As can be seen from Table 6, the performance of the TiBERT-based Tibetan SBD method is close to that of BERT (SP), with an overall F1 of 89.99–91.33%. The performance of the TiBERT-based pre-trained language model is more stable when the window is changed. The TiBERT model has the most considerable F1 value at a window of 5, 91.46%, followed by the TiBERT-RNN at an F1 value when the window is 15, which is 91.33%, and under the same window setting, the change of the model has a more subtle effect on the accuracy. This also proves that the Tibetan-based BERT pre-training model will have room for improvement after changing the subword method and training corpus. Expanding the size of the Tibetan pre-trained language model corpus is necessary to train an efficient Tibetan pre-trained language model.

This study explores the existing publicly available Multi-BERT pre-trained language model for Tibetan sentence boundary disambiguation, and the experimental results are shown in Table 7. Table 6 shows that the performance of Tibetan SBD based on Multi-BERT is partially close to that of Tibetan SBD based on Sentencepiece. From Table 7, it can be found that the Multi-BERT pre-trained model can recognize a part of Tibetan sentence boundaries. The five models show a trend of increasing and then decreasing evaluation indexes, such as F1, as the window increases, and the performance based on BERT-RCNN is the optimal value among all the experiments, reaching 91% when the window is 10. The experimental results show that in the experiments based on Multi-BERT models, the effect of BERT-based combined with deep learning models is improved over BERT alone.

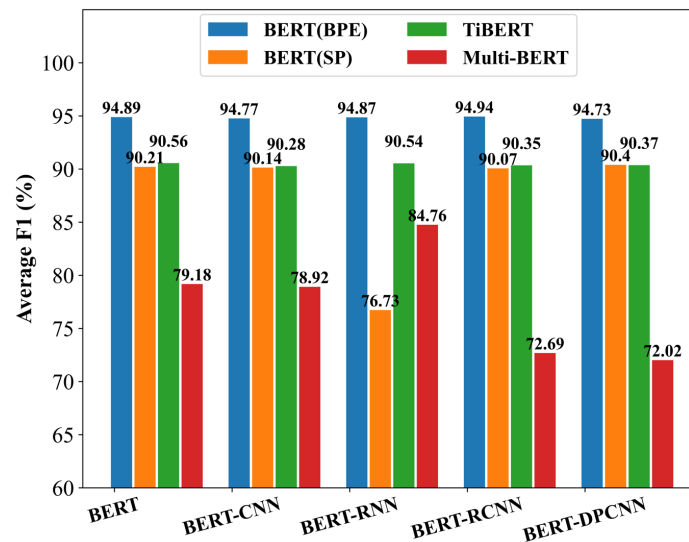
4.4.3. Comprehensive Performance Comparison of Tibetan SBD under Four Pre-Trained Language Models

(1) Comparison of average metrics

Table 8 and Figure 4 compare the average metrics on the four pre-trained language models to compare the performance of Tibetan SBD.

Table 8. Average values on the four pre-trained language models.

Average Values	BERT(BPE)			BERT(SP)			TiBERT			Multi-BERT		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
BERT	94.91	94.9	94.89	90.21	90.24	90.21	90.55	90.57	90.56	76.97	82.81	79.18
BERT-CNN	94.79	94.79	94.77	90.14	90.14	90.14	90.28	90.28	90.28	76.76	82.57	78.92
BERT-RNN	94.89	94.88	94.87	79.70	77.04	76.73	90.54	90.55	90.54	84.88	85.13	84.76
BERT-RCNN	94.96	94.96	94.94	90.08	90.10	90.07	90.37	90.34	90.35	85.81	76.83	72.69
BERT-DPCNN	94.75	94.75	94.73	90.40	90.42	90.4	90.38	90.37	90.37	70.50	76.32	72.02

**Figure 4.** Mean F1 values of SBD under four pre-trained language models.

From Table 8 and Figure 4, it can be seen that there are specific differences in the average SBD indicators of the five models on the four pre-trained language models: BERT (BPE), BERT (SP), TiBERT, and Multi BERT. The performance of BERT (BPE) and TiBERT pre-trained language models is relatively stable, with the highest average performance on the BERT (BPE) model, followed by the TiBERT model, and the lowest average performance on Multi-BERT. BERT (BPE) outperforms other models by four percentage points. Table 7 shows that on the Multi-BERT model, the performance indicators become unstable with the increase in windows. We know that, during text segmentation and pre-training, other pre-trained language models include all Tibetan letters (30 consonants and four vowels). At the same time, the vocabulary in Multi-BERT only contains a portion of Tibetan letters. Therefore, during training, when segmenting the corpus, a lot of “[UNK]” will be generated, which significantly impacts understanding the text’s meaning.

(2) True Positive Comparison

In the experiments of this study, the data of the Tibetan test machine is 46,564 entries. After segmentation by the shad (“|”), a total of 126,762 labeled data units are obtained, including 46,564 positive samples (labeled as “1”) and 80,198 negative samples (labeled as “0”). The focused task of SBD is to determine the probability that a shad (“|”) with a clause function is recognized as correct. Therefore, this experiment compares the number of True Positive samples for each model in data prediction based on the BERT(BPE) and TiBERT (public Tibetan pre-trained language model), as shown in Table 9.

Table 9. True Positive samples of Tibetan SBD based on BERT(BPE) and TiBERT with four window sizes.

True Positive	Window = 3		Window = 5		Window = 10		Window = 15	
	BERT (BPE)	TiBERT	BERT (BPE)	TiBERT	BERT (BPE)	TiBERT	BERT (BPE)	TiBERT
BERT	42,553	40,128	41,693	40,909	43,071	40,388	43,655	39,589
BERT-CNN	42,175	40,409	41,815	40,629	43,548	40,316	42,188	39,995
BERT-RNN	42,145	40,259	42,333	40,205	42,564	39,985	43,666	41,365
BERT-RCNN	42,677	40,537	42,044	40,915	42,849	40,466	42,847	40,921
BERT-DPCNN	41,790	40,431	42,880	40,835	42,683	40,579	42,236	40,290

From Table 9, it can be seen that the predicted number of TP samples of different models is consistent with Tables 4 and 6, primarily, under the same window, the BERT(BPE) syncope performs better than the TiBERT model, which is in line with the experimental data in the previous study, and at the same time, for the same pre-trained language model, the number of TP samples in the process of window change is also consistent with the results in the previous table, so that the BERT model trained by using the BPE sub-word method is better suited for the Tibetan SBD task, and this provides a preprocessing method for subsequent downstream tasks and pre-trained language model datasets to be constructed.

4.4.4. Experimental Result of Sequence Labeling SBD

This study conducts sequence labeling experiments on HMM, CRF, Bi-LSTM, and Bi-LSTM-CRF models. The experimental results are shown in Table 10. A, P, R, and F1 represent the accuracy, precision, recall, and F1 score.

Table 10. Experimental result of sequence labeling SBD.

Models		HMM			CRF			Bi-LSTM			Bi-LSTM-CRF		
Labels	P	R	F1	P	R	F1	P	R	F1	P	R	F1	
B	81.22	91.59	86.10	95.27	94.07	94.67	71.60	97.00	82.38	90.53	96.91	93.61	
E	81.14	90.79	85.70	95.27	94.09	94.68	77.78	96.37	86.08	90.52	96.85	93.58	
M	99.68	99.24	99.46	99.79	99.83	99.81	99.88	98.81	99.34	99.89	99.63	99.76	
Avg/Total	99.04	98.96	98.99	99.63	99.63	99.63	99.00	98.74	98.82	99.56	99.54	99.55	

Table 10 shows the SBD results based on the sequence labeling methods. We know that when the sequence labeling methods for SBD are trained, most syllables are labeled with label M, so there is a problem of label imbalance. We can see from Table 10 that the F1 of label M is the highest among the four models, followed by label E, and label B is the worst. It can be seen from Table 4 that the average F1 of the three labels under the four models ranges from 98.82% to 99.63%, but the accuracy of the three labels varies greatly. Since label M is the most significant proportion of labels in the training data, its F1 is the highest, ranging from 99.34% to 99.81%, label B from 82.38% to 94.67%, and label E from 85.7 to 94.68%. This study focuses on SBD, and we should pay more attention to the evaluation index of the label E. For the metric of label E, CRF and the Bi-LSTM-CRF have the best effect among the four models, and their performance is approximately 94.68%, followed by Bi-LSTM, which is 86.08%, and the F1 of HMM is the lowest, which is 85.7%. It can be seen that among the sequence labeling methods based on deep learning, the CRF Method is better than Bi-LSTM, and Bi-LSTM-CRF is much higher than Bi-LSTM. We can conclude that CRF can improve the performance of sequence labeling SBD. It can be seen that for the sequence labeling method to realize SBD, the evaluation results show significant differences among labels due to the imbalance of labels.

4.4.5. English Sentence Boundary Disambiguation Based on BERT and Multi-BERT

In the previous study, the Tibetan SBD was verified on the Tibetan BERT model trained in this study and the publicly available Tibetan BERT model, and from the results, it can be seen that the BERT (BPE) trained in this study has the best performance, followed by the publicly available pre-trained language model for Tibetan, TiBERT, and the BERT (SP) trained in this study. In contrast, the SBD for Tibetan based on Multi-BERT has the worst experimental results in the previous study. To verify the generalizability of the pre-trained language models on the task of sentence boundary disambiguation, this study experiments with the English SBD based on the BERT and Multi-BERT, and the window sizes were set to 3 and 5. The experimental results are shown in Tables 11 and 12.

Table 11. English sentence boundary disambiguation based on BERT (uncased).

BERT (Uncased)	Window = 3			Window = 5			Average		
	P	R	F1	P	R	F1	P	R	F1
BERT	98.04	98.37	98.09	98.32	98.43	98.13	98.18	98.40	98.11
BERT-CNN	98.17	98.36	98.09	98.30	98.46	98.21	98.24	98.41	98.15
BERT-RNN	98.16	98.4	98.13	98.21	98.44	98.17	98.19	98.42	98.15
BERT-RCNN	98.06	98.35	98.08	98.28	98.44	98.20	98.17	98.40	98.14
BERT-DPCNN	98.16	98.39	98.09	98.18	98.43	98.18	98.17	98.41	98.14

Table 12. English sentence boundary disambiguation based on Multi-BERT.

Multi-BERT (Uncased)	Window = 3			Window = 5			Average		
	P	R	F1	P	R	F1	P	R	F1
BERT	98.21	98.41	98.14	98.34	98.45	98.16	98.28	98.43	98.15
BERT-CNN	98.18	98.39	98.15	98.29	98.44	98.18	98.24	98.42	98.17
BERT-RNN	98.17	98.4	98.12	98.23	98.44	98.19	98.20	98.42	98.16
BERT-RCNN	98.16	98.38	98.14	98.26	98.43	98.14	98.21	98.41	98.14
BERT-DPCNN	98.14	98.39	98.13	98.2	98.41	98.15	98.17	98.40	98.14

From Tables 11 and 12, we can see that when BERT and Multi BERT are introduced for English sentence boundary disambiguation, they achieve more than 98% disambiguation effect, respectively. Meanwhile, we find that the F1 score is slightly higher than BERT when Multi BERT is used for sentence boundary disambiguation, and the indexes are slightly improved when the window is increased from 3 to 5. Under the BERT pre-trained language model, all the sentence boundary disambiguation methods based on the improved BERT model (i.e., BERT-CNN, BERT-RNN) have slightly improved than the BERT model, in which the F1 value of the BERT-CNN is the highest. Under the Multi BERT pre-trained language model, the performance of SBD based on the BERT model is higher than that based on the improved BERT model (i.e., BERT-CNN and BERT-RNN), so from the results of the English SBD experiments, it can be seen that the English SBD performance is higher than Tibetan SBD performance because there is ambiguous punctuation in English as “.”, from the previous introduction, it is known that for the English punctuation “.”, 54.7–92.8% of them appear at the end of the sentence, about 90% indicate the end of the sentence, 10% indicate an abbreviation, and 0.5% both, so it is easier to distinguish whether it is an end-of-sentence marker or not while in Tibetan, the structure of Tibetan characters is complex, the function of a shad (“|”) is more diverse, and there are marking phrases, fragments, and end-of-sentence markers in many cases, so the problem of SBD in Tibetan has been a matter of concern and deserves more investigation.

5. Discussion

This study explores Tibetan SBD based on pre-trained language models, which solves some problems that traditional methods cannot solve, but there are still some shortcomings. Compared with mainstream languages such as Chinese and English, the data resources of Tibetan are relatively limited. This leads to the problem that deep learning models may

need more data during training, affecting SBD accuracy. Deep learning-based Tibetan SBD models are often trained for specific domains or datasets, and their generalization ability could be improved. The model's performance may be affected when applied to other domains or datasets.

6. Conclusions and Future Work

Building a high-quality training corpus has become increasingly important for the application and development of the “pre-training + fine-tuning” model in the field of NLP. Since it is costly to build the dataset manually, it has become a trend to build it through computer modeling. In this study, to improve the efficiency of the Tibetan sentence boundary disambiguation model, we first analyze the characteristics of existing Tibetan pre-trained language models on the SBD task, then train the Tibetan BERT pre-trained model based on different subword methods and finally validate the performance of the Tibetan SBD on four different pre-trained language models, namely, BERT (BPE), BERT (SP), TiBERT, and Multi-BERT. The performance of SBD on pre-trained + four classical deep learning models with different windows is studied. The experimental results show that the method in this study has some generality in SBD tasks, and the BPE subword approach is more suitable for the Tibetan SBD task. In later work, the corpus size will be expanded further. The open corpus will be supplemented for more in-depth validation of the model, and at the same time, we will try to apply the method of this study in the data preprocessing stage of Tibetan machine translation, summary extraction, text generation, and pre-trained language models. The future improvement directions mainly include two parts. Firstly, we aim to establish a larger Tibetan corpus, including text data from different fields and styles. Secondly, a deeper understanding of the rules of Tibetan language in terms of grammar structure, language features, and the use of punctuation can provide more accurate evidence for SBD.

Author Contributions: Conceptualization, F.L. and Z.Z.; methodology, F.L.; software, Z.Z.; validation, Z.Z. and L.W.; formal analysis, H.D.; investigation, L.W.; resources, F.L.; data curation, Z.Z.; writing—original draft preparation, F.L.; writing—review and editing, F.L. and H.D.; visualization, L.W.; supervision, H.D.; project administration, Z.Z.; funding acquisition, F.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (Grant 62162056, 62266037), Major science and technology projects of Gansu province (23ZDGA009), Science and Technology Commissioner Special Project of Gansu province (23CXGA0086), Gansu Provincial Department of Education: Industry Support Program Project (2022CYZC-12), and Northwest Normal University Young Teachers Research Ability Enhancement Program Project (NWNLU-LKQN2019-28).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained within the article.

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

References

1. Kaur, J.; Singh, J. Deep Neural Network Based Sentence Boundary Detection and End Marker Suggestion for Social Media Text. In Proceedings of the 2019 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS), Greater Noida, India, 18–19 October 2019.
2. Liu, Y.; Shriberg, E.; Stolcke, A.; Hillard, D.; Ostendorf, M.; Harper, M. Enriching speech recognition with automatic detection of sentence boundaries and disfluencies. *IEEE Trans. Audio Speech Lang. Process.* **2006**, *14*, 1526–1540.
3. Hua, Q.C.R.; Zhao, H.X. Dependency Parsing of Tibetan Compound Sentence. *J. Chin. Inf. Process.* **2016**, *30*, 224–229.
4. Rou, T.; Se, C.J.; Cai, R.J. Semantic Block Recognition Method for Tibetan Sentences. *J. Chin. Inf. Process.* **2019**, *33*, 42–49.
5. Sun, N.; Du, C. News text classification method and simulation based on the hybrid deep learning model. *Complexity* **2021**, *2021*, 8064579. [[CrossRef](#)]
6. Minaee, S.; Kalchbrenner, N.; Cambria, E.; Nikzad, N.; Gao, J. Deep learning based text classification: A comprehensive review. *ACM Comput. Surv.* **2020**, *54*, 1–40. [[CrossRef](#)]

7. Wan, F.; He, X. Tibetan Syntactic Parsing based on Syllables. In Proceedings of the International Conference on Mechatronics & Industrial Informatics, Zhuhai, China, 30–31 October 2015; Volume 31, pp. 753–756.
8. Garrett, E.J. *Evidentiality and Assertion in Tibetan*; University of California: Los Angeles, CA, USA, 2001.
9. Garrett, E.; Hill, N.W.; Kilgariff, A.; Vadlapudi, R.; Zadoks, A. The contribution of corpus linguistics to lexicography and the future of Tibetan dictionaries. *Rev. D'études Tibétaines* **2015**, *32*, 51–86.
10. Meelen, M.; Hill, N. Segmenting and POS tagging Classical Tibetan using a memory-based tagger. *Himal. Linguist.* **2017**, *16*, 64–89. [[CrossRef](#)]
11. Lobsang, G.; Lu, W.; Honda, K.; Wei, J.; Dang, J. Tibetan vowel analysis with a multi-modal Mandarin-Tibetan speech corpus. In Proceedings of the 2016 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA), Jeju, Republic of Korea, 13–15 December 2016; IEEE: New York, NY, USA, 2016.
12. Wan, F.C.; Yu, H.Z.; Wu, X.H.; He, X.Z. Tibetan Syntactic Parsing for Tibetan-Chinese Machine Translation. In Proceedings of the 2014 International Conference on Artificial Intelligence and Industrial Application, (AIIA2014), Hong Kong, China, 12–14 March 2014.
13. Peters, M.; Neumann, M.; Iyyer, M.; Gardner, M.; Zettlemoyer, L. Deep contextualized word representations. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT2018), New Orleans, LA, USA, 1–6 June 2018; pp. 2227–2237.
14. Radford, A.; Narasimhan, K.; Salimans, T.; Sutskever, I. Improving language understanding by generative pre-training. *Comput. Lang.* **2017**, *4*, 212–220.
15. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL2019), Minneapolis, MN, USA, 2–7 June 2019; pp. 4171–4186.
16. Liang, L.; Tian, F.; Sun, B.W. Current status of Tibetan sentiment analysis and cross language analysis. In Proceedings of the 2018 6th International Conference on Machinery, Materials and Computing Technology (ICMMCT 2018), Jinan, China, 2–3 June 2018; Volume 152, p. 324.
17. Luong, T.; Sutskever, I.; Le, Q.; Vinyals, O.; Zaremba, W. Addressing the Rare Word Problem in Neural Machine Translation. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, Beijing, China, 26–31 July 2015; pp. 11–19.
18. Jean, S.; Cho, K.; Memisevic, R.; Bengio, Y. On using very large target vocabulary for neural machine translation. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing (ACL-IJCNLP), Beijing, China, 26–31 July 2015; pp. 1–10.
19. Sennrich, R.; Haddow, B.; Birch, A. Neural Machine Translation of Rare Words with Subword Units. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Berlin, Germany, 7–12 August 2016.
20. Al-Rfou, R.; Choe, D.; Constant, N.; Guo, M.; Jones, L. Character-Level Language Modeling with Deeper Self-Attention. In Proceedings of the AAAI Conference on Artificial Intelligence (AAAI2019), Honolulu, HI, USA, 27 January–1 February 2019; pp. 3159–3166. [[CrossRef](#)]
21. Wang, C.; Cho, K.; Gu, J. Neural machine translation with byte-level subwords. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 9154–9160.
22. Kudo, T. Subword regularization: Improving neural network translation models with multiple subword candidates. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL2018), Melbourne, Australia, 15–20 July 2018; pp. 66–75.
23. Kudo, T.; Richardson, J. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations (EMNLP2018), Brussels, Belgium, 31 October–4 November 2018; pp. 66–71.
24. Kim, Y. Convolutional neural networks for sentence classification. In Proceedings of the 19th Empirical Methods in Natural Language Processing (EMNLP2014), Doha, Qatar, 25–29 October 2014; pp. 1746–1751.
25. Liu, P.; Qiu, X.; Huang, X. Recurrent neural network for text classification with multi-task learning. In Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, Phoenix, AZ, USA, 12–17 February 2016.
26. Lai, S.; Xu, L.; Liu, K.; Zhao, J. Recurrent convolutional neural networks for text classification. In Proceedings of the AAAI Conference on Artificial Intelligence, Austin, TX, USA, 25–30 January 2015; Volume 29. [[CrossRef](#)]
27. Johnson, R.; Zhang, T. Deep Pyramid Convolutional Neural Networks for Text Categorization. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics—ACL2017 (Volume 1: Long Papers), Vancouver, BC, Canada, 30 July–4 August 2017; pp. 562–570.
28. Sun, Y.; Liu, S.S.; Deng, J.J.; Zhao, X.B. TiBERT: Tibetan pre-trained language model. In Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics (SMC2022), Prague, Czech Republic, 9–12 October 2022; pp. 2956–2961.
29. Pires, T.; Schlinger, E.; Garrette, D. How multilingual is multilingual BERT? In Proceedings of the Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; pp. 4996–5001.
30. Song, Y.Y.; Lu, Y. Decision tree methods: Applications for classification and prediction. *Shanghai Arch. Psychiatry* **2015**, *27*, 130–135. [[PubMed](#)]

31. Altay, O.; Varol Altay, E. A novel hybrid multilayer perceptron neural network with improved grey wolf optimizer. *Neural Comput. Appl.* **2023**, *35*, 529–556. [[CrossRef](#)]
32. Boeker, M.; Hammer, H.L.; Riegler, M.A.; Halvorsen, P.; Jakobsen, P. Prediction of schizophrenia from activity data using hidden Markov model parameters. *Neural Comput. Appl.* **2023**, *35*, 5619–5630. [[CrossRef](#)]
33. Huang, J.; Zweig, G. Maximum entropy model for punctuation annotation from speech. In Proceedings of the 7th International Conference on Spoken Language Processing, ICSLP2002—INTERSPEECH 2002, Denver, CO, USA, 16–20 September 2002.
34. Sutton, C.; McCallum, A. An introduction to conditional random fields. *Found. Trends Mach. Learn.* **2010**, *4*, 267–373. [[CrossRef](#)]
35. Palmer, D.D.; Hearst, M.A. Adaptive multilingual sentence boundary disambiguation. *Comput. Linguist.* **1997**, *27*, 241–267.
36. Read, J.; Drizan, R.; Oepen, S.; Solberg, L.J. Sentence boundary detection: A long solved problem? In Proceedings of the 24th International Conference on Computational Linguistics, Mumbai, India, 8–15 December 2012; pp. 985–994.
37. Mikheev, A. Tagging sentence boundaries. In Proceedings of the 1st Meeting of the North American Chapter of the Association for Computational Linguistics, Seattle, WA, USA, 29 April–4 May 2000; pp. 264–271.
38. Mikheev, A. Periods capitalized words, etc. *Comput. Linguist.* **2002**, *28*, 289–318. [[CrossRef](#)]
39. Kiss, T.; Strunk, J. Unsupervised multilingual sentence boundary detection. *Comput. Linguist.* **2006**, *32*, 485–525. [[CrossRef](#)]
40. Riley, M.D. Some applications of tree-based modelling to speech and language. In Proceedings of the DARPA Speech and Natural Language Workshop, Association for Computational Linguistics, Philadelphia, PA, USA, 21–23 February 1989; pp. 339–352.
41. Reynar, J.C.; Ratnaparkhi, A. A maximum entropy approach to identifying sentence boundaries. In Proceedings of the 5th Conference on Applied Natural Language Processing, Washington, DC, USA, 31 March–3 April 1997; pp. 16–19.
42. Gillick, D. Sentence Boundary Detection and the Problem with the U.S. Human Language Technologies. In Proceedings of the Conference of the North American Chapter of the Association of Computational Linguistics, Boulder, CO, USA, 31 May–5 June 2009; pp. 241–244.
43. Zhao, W.N.; Liu, H.D.; Yu, X.; Wu, J.; Zhang, P. The Tibetan Sentence Boundary Identification based on Legal Texts. In Proceedings of the National Symposium on Computational Linguistics for Young People (YWCL 2010), Wuhan, China, 11–13 October 2010; p. 7.
44. Ren, Q.J.; An, J.C.R. Research on Automatic Recognition Method of Tibetan Sentence Boundary. *China Comput. Commun.* **2014**, *316*, 62–63.
45. Cai, Z.T. Research on the Automatic Identification of Tibetan Sentence Boundaries with Maximum Entropy Classifier. *Comput. Eng. Sci.* **2012**, *34*, 187–190.
46. Li, X.; Cai, Z.T.; Jiang, W.B.; Lv, Y.J.; Liu, Q. A Maximum Entropy and Rules Approach to Identifying Tibetan Sentence Boundaries. *J. Chin. Inf. Process.* **2011**, *25*, 39–44.
47. Ma, W.Z.; Wanme, Z.X.; Nima, Z.X. Method of Identification of Tibetan Sentence Boundary. *J. Tibet. Univ.* **2012**, *27*, 70–76.
48. Zhao, W.N.; Yu, X.; Liu, H.D.; Li, L.; Wang, L.; Wu, J. Modern Tibetan Auxiliary Ending Sentence Boundary Detection. *J. Chin. Inf. Process.* **2013**, *27*, 115–120.
49. Zha, X.J.; Luo, B. Tibetan Sentence Extraction Method Based on Feature of Function Words and Sentence Ending Words. *J. Northwest Minzu Univ.* **2018**, *39*, 39–43+62.
50. Que, C.Z.M.; Hua, Q.C.R.; Cai, R.D.Z.; Xia, W.J. Tibetan Sentence Boundary Recognition Based on Mixed Strategy. *J. Inn. Mong. Norm. Univ. (Nat. Sci. Chin. Ed.)* **2019**, *48*, 400–405.
51. Wan, M.L.Z. *A Tibetan Sentence Boundary Recognition Method Based on Part of Speech at the End of Sentence*; Normal University: Xining, China, 2016.
52. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In Proceedings of the Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017 (NIPS2017), Long Beach, CA, USA, 4–9 December 2017; pp. 5998–6008.
53. Zhang, X.; Qiu, X.; Pang, J.; Liu, F.; Xingwei, L.I. Dual-axial self-attention network for text classification. *Sci. China-Inf. Sci.* **2021**, *64*, 76–86+145. [[CrossRef](#)]
54. Tsering, S.; Dorla. Study on the construction of Tibetan sentence segmentation dataset under scarcity language resources. *Plateau Sci. Res.* **2022**, *6*, 85–94.
55. Koehn, P. Europarl: A Parallel Corpus for Statistical Machine Translation. In Proceedings of the 10th Machine Translation Summit Proceedings of Conference, Phuket, Thailand, 12–16 September 2005; pp. 79–86.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.