Does Chinese BERT Encode Word Structure?

Yile Wang[1,2,3], **Leyang Cui**[1,2,3] and **Yue Zhang**[2,3]
[1]Zhejiang University
[2]School of Engineering, Westlake University
[3]Institute of Advanced Technology, Westlake Institute for Advanced Study
{wangyile,cuileyang}@westlake.edu.cn
yue.zhang@wias.org.cn

Abstract

Contextualized representations give significantly improved results for a wide range of NLP tasks. Much work has been dedicated to analyzing the features captured by representative models such as BERT. Existing work finds that syntactic, semantic and word sense knowledge are encoded in BERT. However, little work has investigated word features for character-based languages such as Chinese. We investigate Chinese BERT using both attention weight distribution statistics and probing tasks, finding that (1) word information is captured by BERT; (2) word-level features are mostly in the middle representation layers; (3) downstream tasks make different use of word features in BERT, with POS tagging and chunking relying the most on word features, and natural language inference relying the least on such features.

1 Introduction

Large scale pre-trained models such as BERT (Devlin et al., 2019) have been widely used in NLP, giving improved results in a wide range of downstream tasks, including dependency parsing (Zhou and Zhao, 2019), summarization (Liu and Lapata, 2019) and reading comprehension (Xu et al., 2019). To better understand the reason behind their effectiveness, a natural question is what knowledge can be learned during the self-supervised pre-training stage. Some previous works prove that BERT effectively captures syntactic information (Goldberg, 2019), semantic information (Jawahar et al., 2019), commonsense knowledge (Cui et al., 2020), and factual knowledge (Petroni et al., 2019) without fine-tuning on task-specific datasets, which explains its effectiveness on related tasks.

BERT has also gained success in Chinese NLP (Cui et al., 2019; Sun et al., 2019; Zhang et al., 2019). However, relatively little work investigate knowledge learned by Chinese BERT. Different from English, Chinese sentences are written as sequences of characters without explicit word boundary. Yet Chinese BERT is character-based, where the contextualized representations are built on each character. For traditional Chinese NLP, word segmentation is considered an essential pre-processing step (Cai and Zhao, 2016; Xu and Sun, 2016). In neural modeling, Li et al. (2019) argue that character-based model consistently outperforms word-based model for Chinese, because the former can alleviate the over-fitting and out-of-vocabulary issue. In this paper, we investigate whether Chinese BERT encodes word structure features.

We aim to answer the following three research questions. First, how much word information is captured by character-based Chinese BERT? Second, out of 12 representation layers of a BERT encoder, which layers encode the most word-level features? Third, the connection between word-level features embedded in BERT representations and the performance of downstream tasks such as named entity recognition (NER) and natural language inference (NLI). Intuitively, some tasks rely on word features more than other tasks. By analyzing word knowledge inside BERT after fine-tuning on each task, we can gain empirical evidence on how the task is solved by BERT.

We take two main approaches for analyzing BERT. First, BERT is based on Transformer (Vaswani et al., 2017), a multi-layer multi-head self-attention-network architecture, where the embedding of each

Proceedings of the 28th International Conference on Computational Linguistics, pages 2826–2836
Barcelona, Spain (Online), December 8-13, 2020

character is calculated by neural attention (Bahdanau et al., 2015) over all the characters in a sequence. Inspired by Clark et al. (2019), we look into the attention weight distribution of each head in each layer, in order to understand whether some attention head exhibit salient word-level patterns in the attention targets. Second, we take Chinese word segmentation as a probing task (Conneau et al., 2018; Liu et al., 2019), training a linear classifier based on the BERT representation at each Transformer layer, so that word-level information contained in the layer can be quantified through segmentation performance.

Results on two Chinese datasets with varying segmentation standards consistently show that word information is captured by BERT representation. There exist attention heads that focus on the start and end characters in each word, word unigrams and bigrams, as well as word boundary patterns. In addition, word information is captured mostly in the middle layers of Transformer, allowing light-weight probing layers to achieve segmentation F1 score around 90% on both segmentation datasets. Finally, we find that different Chinese tasks require different levels of word information, with fine-tuning on tasks such as POS tagging and chunking significantly improving the probing task, while fine-tuning on tasks such as NLI significantly decreasing probing accuracies.

To our knowledge, we are the first to investigate word structure knowledge in Chinese BERT. Our code has been released at `https://github.com/ylwangy/BERT_zh_Analysis`.

2 BERT

BERT (Devlin et al., 2019) consists of multi-layer Transformer (Vaswani et al., 2017) blocks. Formally, given a sentence $s = c_1, c_2, ..., c_n$, each c_i is first transformed into input vector e_i by summing up token embeddings E_{c_i}, segment embeddings SE_{c_i}, which distinguish the sentence location, and position embeddings PE_{c_i}, which indicates character position:

$$e_i = E_{c_i} + SE_{c_i} + PE_{c_i} \tag{1}$$

The vectors $e_1, ..., e_n \in \mathbb{R}^{n \times d}$ are taken as input to the first layer in a Transformer encoder, which consists of L layers. Now for each layer, denote the input as E. E is then transformed into vectors for queries Q^m, keys K^m, and values V^m via linear mappings, $\{Q^m, K^m, V^m\} \in \mathbb{R}^{n \times d_k}$:

$$Q^m, K^m, V^m = EW_Q^m, EW_K^m, EW_V^m, \tag{2}$$

where $\{W_Q^m, W_K^m, W_V^m\} \in \mathbb{R}^{d \times d_k}$ are trainable parameters, $m \in [1, 2, ..., M]$ represent the m-th attention head. M parallel attention functions are applied to produce M hidden states $\{H^1, ..., H^M\}$:

$$\begin{aligned} \alpha^m &= softmax(\tfrac{Q^m K^{m\top}}{\sqrt{d_k}}) \\ H^m &= \alpha^m V^m \end{aligned} \tag{3}$$

α^m is the attention distribution for the m-th head and $\sqrt{d_k}$ is a scaling factor. Finally, multi-head hidden states are concatenated to obtain a hidden representation $\hat{h}_i$ of each character c_i:

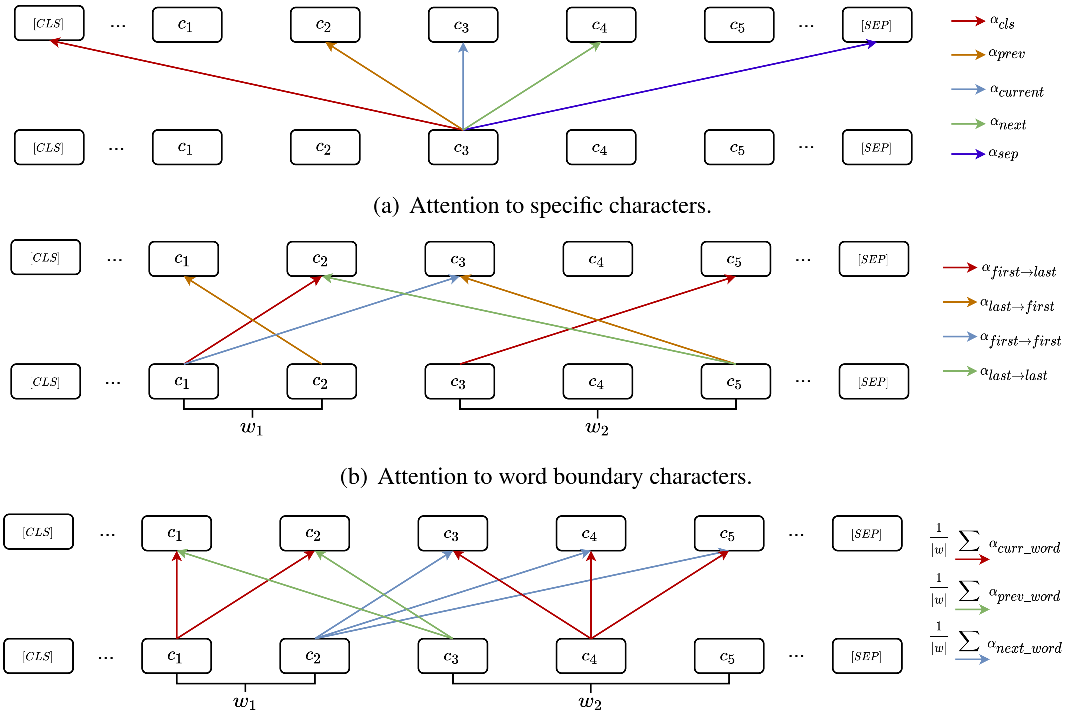
$$\hat{h}_i = [H_i^1, ..., H_i^M] \tag{4}$$

$\hat{h}_i$ are then fed to a multi-layer perceptron for computing the final outputs h_i for the layer. Feed-forward connections and layer normalization are also applied, the detail of which can be found in Vaswani et al. (2017). We denote the output of the l-th layer as h_i^l ($l \in [1, 2, ..., L]$).

Given a corpus $\{s_t = c_1, c_2, ..., c_{n_t}\}|_{t=1}^T$, the masked language model objective is to minimize the loss of predicting the randomly chosen masked character c_{mask_j} ($j \in [1, 2, ..., n_t]$) by its representation h_{mask_j} in the last layer L:

$$L_{MLM} = -\sum_{t=1}^{T} \sum_{j=1}^{n_t} \log p(E_{c_{mask_j}} | h_{mask_j}^L), \tag{5}$$

where E is the token embedding table in Eq. 1.



(a) Attention to specific characters.

(b) Attention to word boundary characters.

(c) Averaged attention to curruent and surrounding words.

Figure 1: Different attention weight distribution patterns.

During pre-training, special tokens [CLS] and [SEP] are added to indicate the beginning of a sentences and the separation of two sentences, respectively. We conduct our experiments on BERT-base-Chinese[1], which has 12 layers, 12 heads and 768 hidden size.

3 Attention Distribution Analysis

We analyze the distribution of attention weight α in Eq. 3 across different layers and heads. Specifically, we compute the attention weight from characters to several specific characters as well as the attention distribution from characters to surrounding words. The structures are illustrated by Figure 1.[2]

For each character, we first investigate its attention weights to several specific characters, including the character itself, the previous character, the next character, and two special tokens [CLS] and [SEP], as shown in Figure 1(a). Formally, the attention from c_i to c_j in a certain head is defined as in Eq. 3:

$$\alpha_{c_i \to c_j} = \frac{exp(Q_i K_j^\top / \sqrt{d_k})}{\sum_l exp(Q_i K_l^\top / \sqrt{d_k})}, \tag{6}$$

where c_j can be the previous, current, or next character c_{i-1}, c_i, c_{i+1}, and the token [CLS] or [SEP], respectively.

We further investigate attention to characters at word boundary locations. Formally, for each c_i that is the first character of some word w_k, we first consider the attention weights to the last characters of the current word and the first character of the next word:

$$\begin{aligned} &\alpha_{c_i:first\to last} = \alpha_{c_i \to c_j}, s.t.\ w_k = \{c_i, ..., c_j\} \\ &\alpha_{c_i:first\to first} = \alpha_{c_i \to c_j}, s.t.\ w_k = \{c_i, ...\}, w_{k+1} = \{c_j, ...\} \end{aligned} \tag{7}$$

Similarly, for each c_j that is the last character of some word w_k, we consider the attention weights to

[1] https://github.com/google-research/bert

[2] We mainly show the difference of the attention patterns for character c_i, and omit the query, key and value transformations.

(a) Results on CTB9.

Layers	Specific Characters					First&Last Character of Words			
	Curr	Next	Prev	CLS	SEP	F→L	L→F	F→ F_1	L→$\text{L}_{\text{-1}}$
#1	9.0	**62.0**(5)	35.7	35.3	12.6	15.7	13.7	7.2	7.7
#2	7.6	60.8	62.7	60.8	12.1	26.7	28.7	19.4	21.9
#3	38.7	31.5	61.5	**65.5**(4)	25.2	**64.7**(10)	**59.7**(11)	7.7	9.1
#4	44.5	61.5	38.0	21.0	61.4	38.0	23.3	16.0	23.6
#5	7.9	13.4	17.7	25.8	80.1	19.7	39.7	14.2	12.7
#6	12.6	36.7	33.5	9.8	57.3	25.8	56.4	20.9	18.3
#7	5.4	37.4	**91.9**(4)	4.1	78.1	60.9	38.2	**33.3**(1)	25.2
#8	6.7	52.8	34.4	8.7	90.4	35.3	33.8	24.1	**26.8**(3)
#9	9.9	18.7	12.1	13.8	86.4	35.1	55.3	9.3	13.1
#10	18.5	9.2	9.4	5.6	88.7	42.2	37.6	9.0	10.5
#11	12.8	32.0	15.8	11.0	88.1	19.5	24.2	5.9	8.32
#12	**55.1**(2)	7.1	3.3	1.6	**90.7**(6)	8.0	8.7	7.3	6.6

(b) Results on PKU.

Layers	Specific Characters					First&Last Character of Words			
	Curr	Next	Prev	CLS	SEP	F→L	L→F	F→ F_1	L→$\text{L}_{\text{-1}}$
#1	8.6	58.6	34.1	34.7	11.8	13.7	10.9	6.3	6.2
#2	6.7	58.9	61.1	62.0	11.8	23.8	22.3	17.0	22.5
#3	36.0	34.9	62.8	**68.7**(4)	23.1	**58.4**(10)	50.3	5.3	9.2
#4	43.4	64.1	39.4	20.0	56.5	36.7	17.4	14.9	21.6
#5	9.7	14.3	17.8	26.3	74.9	16.4	33.1	12.4	10.2
#6	13.0	41.5	33.8	10.1	56.4	24.8	**50.5**(5)	18.7	18.5
#7	5.8	38.1	**91.7**(4)	4.2	82.3	55.1	34.9	**29.5**(1)	22.3
#8	6.5	**60.1**(5)	38.2	7.0	**91.0**(11)	36.7	31.4	19.3	**24.7**(3)
#9	11.0	22.7	13.6	12.5	83.7	28.3	51.1	7.4	11.0
#10	16.4	10.7	11.8	6.4	88.7	39.5	34.5	7.4	9.4
#11	16.8	36.4	17.2	6.8	90.3	18.8	22.4	4.3	7.2
#12	**60.3**(2)	7.5	2.7	2.2	87.3	5.6	6.2	4.7	4.9

Table 1: Character-to-character attention distribution. The numbers j in the parentheses denotes the j-th head. F, L, F_1, and $\text{L}_{\text{-1}}$ denote first character of current word, last character of current word, first character of next word, and last character of previous word, respectively.

the first character of the current word and the last character of the previous word:

$$\begin{aligned}\alpha_{c_j:last\rightarrow first} &= \alpha_{c_j\rightarrow c_i}, s.t.\ w_k = \{c_i, ..., c_j\}\\ \alpha_{c_j:last\rightarrow last} &= \alpha_{c_j\rightarrow c_i}, s.t.\ w_{k-1} = \{..., c_i\}, w_k = \{..., c_j\}\end{aligned} \tag{8}$$

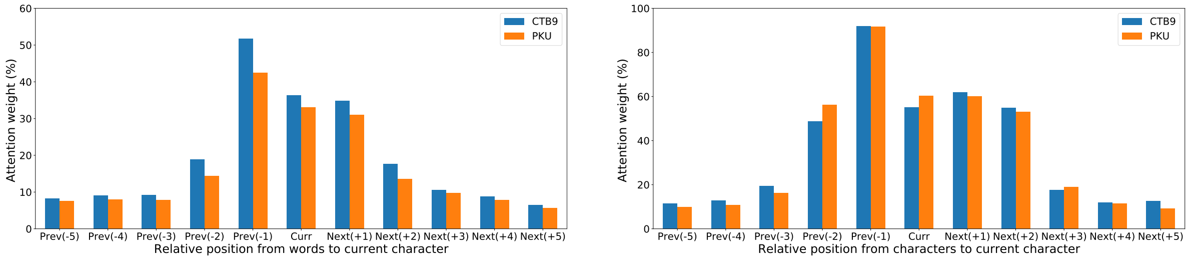
Figure 1(b) shows one example, where w_1 and w_2 are two adjacent words, c_1, c_2 are the first and last character of the word w_1, and c_3, c_5 are the first and last character of the word w_2, respectively. For this example, we consider the attentions $\alpha_{c_1\rightarrow c_2}$, $\alpha_{c_3\rightarrow c_5}$, $\alpha_{c_1\rightarrow c_3}$ (Eq. 7), and $\alpha_{c_2\rightarrow c_1}$, $\alpha_{c_5\rightarrow c_3}$, $\alpha_{c_5\rightarrow c_2}$ (Eq. 8).

In addition to character-to-character attention weights, we consider character-to-word attention by taking the average of attention weights to characters that belong to specific words. Formally, we define the attentions from a character c_j in word w_k to it previous, current and next words as follows:

$$\begin{aligned}\alpha_{c_j:prev\_word} &= \frac{1}{|w_{k-1}|}\sum_{c_i\in w_{k-1}} \alpha_{c_j\rightarrow c_i}\\ \alpha_{c_j:curr\_word} &= \frac{1}{|w_k|}\sum_{c_i\in w_k} \alpha_{c_j\rightarrow c_i}\\ \alpha_{c_j:next\_word} &= \frac{1}{|w_{k+1}|}\sum_{c_i\in w_{k+1}} \alpha_{c_j\rightarrow c_i},\end{aligned} \tag{9}$$

where $|w|$ denotes the length of word w.

Figure 1(c) shows one example, where the α_{curr_word} for c_1 is the average of $\alpha_{c_1\rightarrow c_1}$ and $\alpha_{c_1\rightarrow c_2}$, because c_1 belongs to the words w_1 composed of characters c_1 and c_2. α_{next_word} for c_1 is the average of $\alpha_{c_1\rightarrow c_3}$, $\alpha_{c_1\rightarrow c_4}$ and $\alpha_{c_1\rightarrow c_5}$.



(a)

(b)

Figure 2: Character-to-word and character-to-characters attention distribution with different positions.

3.1 Datasets

There exist different word segmentation criteria for the same sentence, which mainly differ by the segmentation granularity. We select two corpora with golden word segmentation labels for computing attention distribution: Chinese Treebank (Xue et al., 2005) 9.0 (CTB9) and PKU (Emerson, 2005), averaging the attention weights across all the characters for each attention heads.

3.2 Results

Character-to-Character Attention. Table 1 show the largest values among all the 12 attention heads in each layer of BERT. We use head i-j to denote the j-th attention head in the i-th layer. The average sentence lengths of the CTB9 and PKU datasets are 26.3 and 35.8, respectively. As a result, random baseline for the two datasets are 3.8% (1/26.3) and 2.7% (1/35.8), respectively.

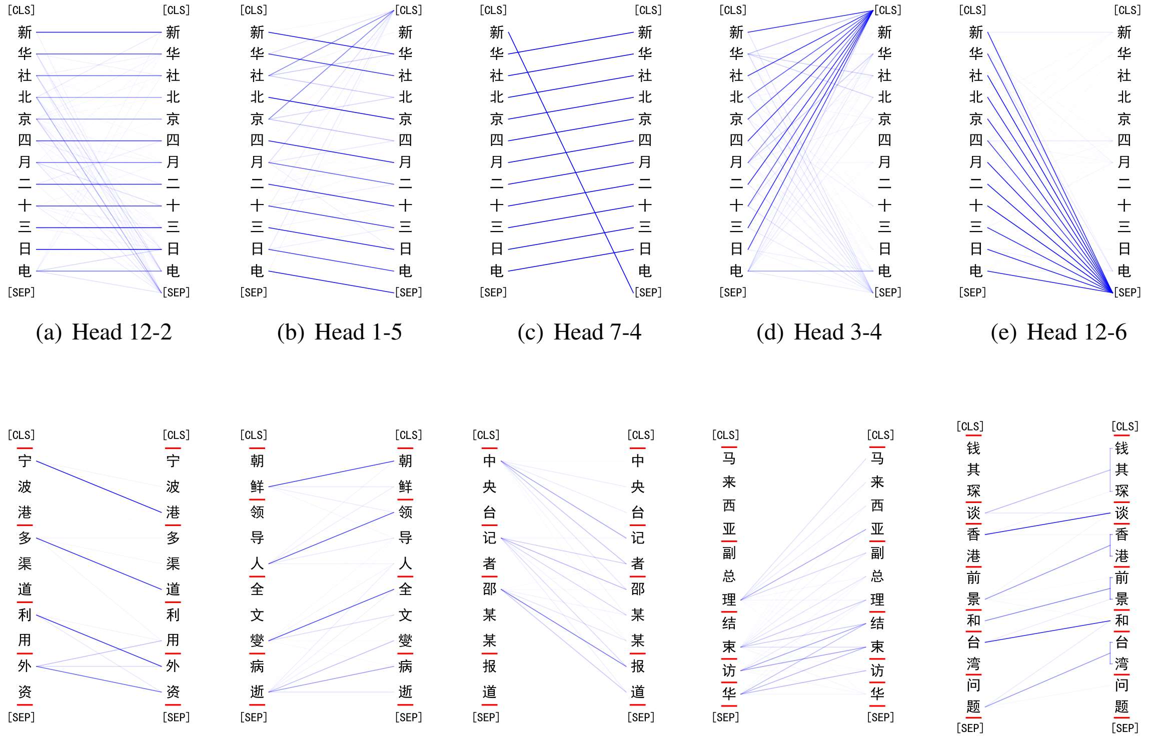
First, the Specific Characters columns of the two tables show weights calculated by Eq. 6. There are specific heads with strong attention weights to both the current character and its neighboring characters. For both datasets, attention to the previous character can reach 92%, while to the next character is 60%. This shows that the previous character plays a very important role in certain BERT representation layers. Attention weights to [SEP] and [CLS] are also significantly stronger than the random baseline, with those to the [SEP] node being above 90%. This shows that BERT is highly sensitive to sentence boundaries. Finally, results are quite consistent between CTB9 and PKU.

The First&Last Characters of Words columns of the two tables show attention weights calculated by Eqs. 7 and 8. First, attention between the first and last characters of the same word can reach 50% to 60%, which shows that word information is captured by BERT representation. In addition, attention between consecutive words can reach 20% to 30%, which shows that word n-gram information is also learned. Word internal attention is higher than cross-word attention, which can be because word n-grams are more sparse and play less role in BERT.

Finally, the strongest weights mostly occur in the middle layers (3$\sim$8), which suggests that word information from BERT concentrates in those layers. Our probing task in Section 4 confirms this.

Our findings are in line with those by Clark et al. (2019), who observe that different heads in English BERT models can put strong attention weights on different dependency relations. In addition, Jawahar et al. (2019) shows that syntax in BERT are in relatively lower layers while semantics are in higher layers. Word information can be regarded as a lexical level syntax feature, and thus our finding is similar.

Character-to-Word Attention. Figure 2(a) shows the results of the best-performing heads calculated using Eq 9. In addition to the current, previous and next words, we also show the results for other words within a window size of 5. We can find that attention to the previous word is stronger compared with that to the current word and the next word. Attention weights decrease when the target is increasingly far from the current characters, which shows that neighboring context is more important in BERT representation concerning words. This is consistent with character-to-character trends (Figure 2(b)). Values for CTB9 are relatively larger than those for PKU mainly because the sentences are shorter, but the main observations are consistent.


[CLS] 新 华 社 北 京 四 月 二 十 三 日 电 [SEP]
[CLS] 宁 波 港 多 渠 道 利 用 外 资 [SEP]
[CLS] 朝 鲜 领 导 人 全 文 燮 病 逝 [SEP]
[CLS] 中 央 台 记 者 邵 某 某 报 道 [SEP]
[CLS] 马 来 西 亚 副 总 理 结 束 访 华 [SEP]
[CLS] 钱 其 琛 谈 香 港 前 景 和 台 湾 问 题 [SEP]

(a) Head 12-2 (b) Head 1-5 (c) Head 7-4 (d) Head 3-4 (e) Head 12-6

(f) Head 3-10 (g) Head 3-11 (h) Head 7-1 (i) Head 8-3 (j) Head 6-5

Figure 3: Examples of attention distribution visualization for sentences on CTB9. Head i-j denotes the j-th attention head in the i-th layer. The darkness of blue lines reflect the values of attentions. The red lines denotes the word boundary.

3.3 Case Study

We visualize the best-performing heads in Table 1 using a few sentences on CTB9. The top row of Figure 3 shows the sentence "新华社(Xinhua Agency)北京(Beijing)四月(April)二十三日(23th)电(news)". These heads consistently attend to one of the specific characters. For example, the head 1-5 attends to the next characters, with the last character "电(news)" attending to the [SEP] token. The head 7-4 attends to the previous character, with the first character "新(Novel)" attending to the [SEP] token as well.

Figures 3(f) to 3(i) show more examples that the heads attend to word boundary characters. For head 3-11, most of the last characters tend to pay attention to the first character of the same words. As for head 8-3, the last characters attend to the last characters of the previous words. There are exceptions to the general trends above. Take head 3-10 for example, in the sentence "宁波港(Ningbo Port)多渠道(multi-channel)利用(use)外资(overseas investment)", the character "利(benefit)" puts the most attention on the character "外(outside)", which is out of the corresponding word "利用(use)".

Figure 3(j) shows the attention distribution in the sentence "钱其琛(Qichen Qian)谈(talk)香港(Hong Kong)前景(prospect)和(and)台湾(Taiwan)问题(issue)" for head 6-5. Each character puts the most attention on characters in the previous word. For example, the character "谈(talk)" focuses mainly on characters in the word "钱其琛(Qichen Qian)", and the character "题(question)" puts most attention to characters in the word "台湾(Taiwan)". This indicates that the head 6-5 takes most of the information from the previous words to generate contextualized character representation.

4 Probing Task

We probe the contextualized representation of each character for Chinese Word Segmentation (CWS) (Xue, 2003). In particular, CWS can be treated as a character-level sequence labeling task, where the label set includes B, M, E, and S (which stand for the beginning, middle, ending of word and single

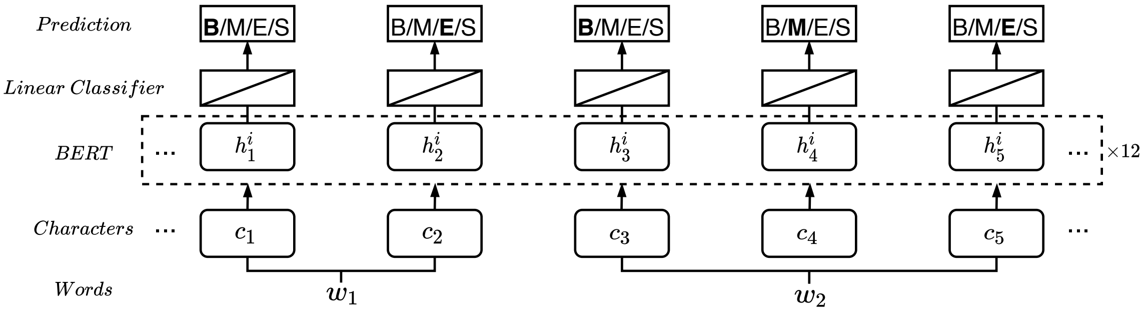


Figure 4: Word segmentation probing model.

Data	Train			Dev			Test		
	#Sent	#Word	#Char	#Sent	#Word	#Char	#Sent	#Word	#Char
CTB9	93.5K	1.68M	2.62M	9.28K	142K	217K	13.0K	230K	361K
PKU	17.1K	1.01M	1.66M	1.90K	99.8K	163K	1.94K	104K	172K
MSR	78.2K	2.12M	3.63M	8.69K	246K	417K	3.98K	106K	184K
CITYU	47.7K	1.29M	2.14M	5.30K	158K	258K	1.49K	40.9K	67.6K
AS	638K	4.88M	7.49M	70.8K	563K	876K	14.4K	122K	197K

Table 2: Statistics of datasets.

character word, respectively). We directly use the fixed hidden representations in each layer as the input, on which a trainable linear classifier is built, as Figure 4 shows. We use local classifier rather than the conditional random fields (Lafferty et al., 2001), in order to focus on information extracted from hidden representations directly. The intuition is that if a simple linear classifier can predict the segmentation labels, we can reasonably conclude that the model has word structure features.

Formally, in the representations of layer l, the label probability distribution of the character c_j is calculated as follows:

$$P(y\prime|h^l_j) = softmax(Wh^l_j) \tag{10}$$

where $y\prime \in \{\text{B, M, E, S}\}$, W is the linear transformation parameters and h^l_j is the j-th hidden representation in l-th layer.

4.1 CWS Settings and Results

We experiment with the CTB 9.0 dataset[3], and widely-used SIGHAN 2005 benchmarks[4] including four subsets (i.e, PKU, MSR, CITYU and AS). We split the CTB 9.0 dataset into train/dev/test set following Shao et al. (2017). Statistics of the datasets are shown in Table 2. The standard word segmentation F1 score is used for evaluation. Our model is implemented using NCRF++ (Yang et al., 2018). In particular, we use Adam (Kingma and Ba, 2015) as our optimization method, with a learning rate of 2e-5, a dropout rate of 0.1, and the number of training epochs being 3. The parameters are selected using the development set.

Table 3 shows the main results. In general, layers 3∼8 give relatively higher performances, with F1 score close to or over 90% for all the datasets. The best-performing layers are layers 4 and 7. Existing state-of-the-art neural segmentors trained on the datasets give F1 score of 96.5, 96.1, 97.4, 97.2, and 96.2 on CTB9, PKU, MSR, CITYU and AS datasets, respectively (Shao et al., 2017; Meng et al., 2019). From the above results we can see that word information is strongly captured by BERT in the middle layers. This is consistent with findings of Section 3.

4.2 Tuning on Downstream Tasks

To further analyze the influence of downstream tasks to word segmentation by fine-tuning the model using different downstream tasks including NER, chunking, POS tagging and five tasks in the Chinese

[3] https://catalog.ldc.upenn.edu/LDC2016T13

[4] http://sighan.cs.uchicago.edu/bakeoff2005/

Layers	CTB9	PKU	MSR	CITYU	AS	Layers	CTB9	PKU	MSR	CITYU	AS
#1	80.70	75.70	79.01	75.38	78.75	#7	92.01	88.68	**89.69**	91.05	**92.04**
#2	88.91	85.21	86.31	86.33	87.73	#8	91.91	88.59	89.40	91.03	91.97
#3	91.54	88.43	88.66	90.57	91.24	#9	91.65	88.29	89.22	90.84	91.58
#4	**92.20**	**88.74**	89.37	**91.26**	91.93	#10	91.19	88.05	89.07	90.44	91.14
#5	92.14	88.53	89.42	90.87	91.89	#11	90.97	87.75	88.84	89.86	90.81
#6	92.00	88.36	89.56	90.80	91.93	#12	89.68	85.80	87.40	88.20	89.72

Table 3: Word segmentation results of our probing model.

Models (Best Layer)	CTB9	PKU	MSR	CITYU	AS	∆ (*Avg.*)
BERT-base	92.20	88.74	89.69	91.26	92.04	-
NER-finetune	92.37	89.22	89.93	91.49	92.25	⇑ 0.27
Chunking-finetune	94.43	91.01	90.45	93.15	93.74	⇑ 1.77
POS Tagging-finetune	**94.60**	**91.16**	**90.61**	**93.42**	**93.86**	⇑ 1.94
CMNLI-finetune	90.67	88.64	87.91	89.69	90.59	⇓ 1.28
TNEWS-finetune	92.05	88.57	89.15	90.99	91.92	⇓ 0.25
AFQMC-finetune	92.04	88.85	89.14	91.30	92.04	⇓ 0.11
WSC-finetune	92.14	88.82	89.30	91.33	92.03	⇓ 0.06
CSL-finetune	92.17	88.80	89.29	91.37	92.17	⇓ 0.02

Table 4: Performance of the word segmentation probing task after model finetuning.

Language Understanding Evaluation (CLUE) benchmark (Xu et al., 2020), then we execute the probing task again. Intuitively, word information is encoded by downstream tasks if the fine-tuned model performs better on word segmentation probing task, or vice versa. The tasks and datasets we use are as follows:

• **NER**. We use the OntoNotes 4.0 (Weischedel et al., 2011) as the named entity recognition dataset.

• **Chunking**. CTB 4.0 is used as the chunking dataset. We split the data into train/dev/test set following Lyu et al. (2016).

• **POS Tagging**. CTB 5.0 is used as the part-of-speech tagging dataset. The dataset split is the same as in Shao et al. (2017).

• **CMNLI** (Chinese Multi-Genre of Natural Language Inference) is a CLUE task to predict the relationship (neutral, entailment or contradiction) between two sentences.

• **TNEWS** is a short sentence classification dataset[5] in CLUE, where the task is to assign a title to each news. The category of labels includes finance, technology, sports, etc.

• **AFQMC** (Ant Financial Question Matching Corpus) comes from Ant Technology Exploration Conference (ATEC) Developer competition[6]. The CLUE task is to predict whether two sentences are semantically similar.

• **WSC** (Winograd Schema Challenge) (Levesque et al., 2012) Chinese datastet is a co-reference resolution task in CLUE.

• **CSL** (Chinese Scientific Literature) dataset[7] contains Chinese paper abstracts and their keywords. The CLUE task is to recognize whether given keywords are correct to the corresponding paper, where some noise keywords are generated by using TF-IDF value.

The results are shown in Table 4. We observe 5 trends for different datasets. First, for POS tagging and chunking, CWS probing is strongly improved after fine-tuning. These two datasets are the mostly connected with words, and joint segmentation models have been investigated (Zhang and Clark, 2010; Zheng et al., 2013; Lyu et al., 2016). Second, for NER, the performance slightly improved. The task sees debates on whether word information is useful, where segmentation error can outweight word features (He and Wang, 2008; Liu et al., 2010). Third, for WSC and CSL the performance did not change. Fourth, for the text classification tasks TNEWS and AFQMC the performance slightly decreases. This may because word features do not help much for these tasks. Last, the performance decreases sharply for CMNLI, which shows that the semantics heavy task does not make use of word information and also

[5] `https://github.com/fatecbf/toutiao-text-classfication-dataset/`

[6] `https://dc.cloud.alipay.com/index#/topic/intro?id=3`

[7] `https://github.com/CLUEbenchmark/LightLM`

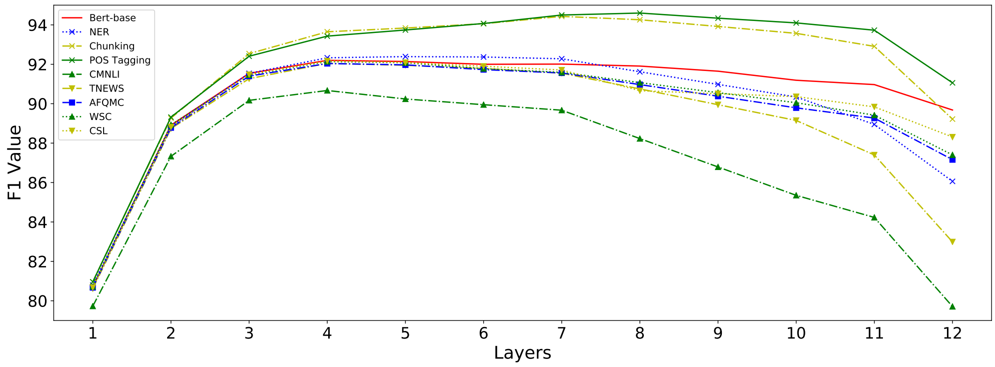


Figure 5: Layerwise word segmentation performance for BERT and fine-tuned models.

worsen the performance of CWS probing model.

We show the layer-wise word segmentation results on CTB9 for different models in Figure 5. A silent finding is that for those tasks that benefit from word information, the best-performing layers move up from the middle layers. In contrast, for the CMNLI task, the best-performing layers move down from the middle layers. This shows that useful information can be brought closer to the final prediction layer during fine-tuning, or that training signals influence the top layers more strongly. For the other tasks, the best-performing layers are still 3∼8.

5 Related Work

Knowledge from Pre-trained Models. Peters et al. (2018) find that ELMo encodes syntactic and semantic features at different layers. To analyze the syntactic features inside BERT, Goldberg (2019) use BERT to predict the masked verb, and compare assigned scores between the correct verb and the incorrect verb. Petroni et al. (2019) demonstrate that BERT is able to recall factual knowledge using pre-defined cloze templates. While all the above work has been conducted on English, little work analyzes the linguistic or structural knowledge from Chinese pre-trained model. We thus fill a gap in the literature.

Attention Analysis. Clark et al. (2019) visualize the attention patterns from BERT, finding different behaviors from different attention heads. Htut et al. (2019) evaluate syntactic knowledge by computing the maximum spanning tree on BERT's attention to recover dependency trees. Kovaleva et al. (2019) analyze the attention distribution for BERT fine-tuned across different tasks, pointing out that redundancy exists in different heads. Our work is similar in finding the patterns by making use of attention and fine-tuning. However, except token-to-token attention patterns, we also investigate attentions to specific characters for finding the potential word structure of Chinese.

Probing Method. Conneau et al. (2018) introduce 10 probing tasks, uncovering linguistic properties that a sentence encoder captures. Liu et al. (2019) design probing tasks on the contextualized representations from pre-trained models and investigate the linguistic knowledge it encodes. Ian et al. (2019) design edge probing tasks to investigate the sub-sentential structure of contextualized word embeddings. Hewitt and Manning (2019) propose structural probing methods and find that syntax trees are embedded implicitly in ELMo and BERT. Our method is similar in analyzing the contextualized representations directly. However, we select the Chinese word segmentation as our probing task for each character-level contextualized representation.

6 Conclusion

We investigated the capability of Chinese BERT for capturing the word structure using two different methods. First, analyzing the attention distribution for different patterns, we find that some of the attention heads can capture the word structure implicitly. Second, using a word segmentation probing task for the contextualized representation inside the model, we find that a simple linear classifier performs well in the middle layers. By using our probing method, we find evidence that different Chinese tasks rely on different degrees of word information, with NLI relying the least on word features.

Acknowledgements

We thank all the anonymous reviewers for their thoughtful comments and suggestions. We gratefully acknowledge funding from the National Natural Science Foundation of China (NSFC No.61976180) and a research grant from Rxhui Inc. Yue Zhang is the corresponding author.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In Yoshua Bengio and Yann LeCun, editors, *ICLR*.

Deng Cai and Hai Zhao. 2016. Neural word segmentation learning for Chinese. In *ACL*, pages 409–420.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? an analysis of BERT's attention. In *ACL Workshop*, pages 276–286.

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. In *ACL*, pages 2126–2136.

Yiming Cui, Ting Liu, Wanxiang Che, Li Xiao, Zhipeng Chen, Wentao Ma, Shijin Wang, and Guoping Hu. 2019. A span-extraction dataset for Chinese machine reading comprehension. In *EMNLP-IJCNLP*, pages 5886–5891.

Leyang Cui, Sijie Cheng, Yu Wu, and Yue Zhang. 2020. Does BERT Solve Commonsense Task via Commonsense Knowledge? *arXiv e-prints*, page arXiv:2008.03945, August.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, pages 4171–4186.

Thomas Emerson. 2005. The second international Chinese word segmentation bakeoff. In *SIGHAN Workshop*.

Yoav Goldberg. 2019. Assessing BERT's Syntactic Abilities. *arXiv e-prints*, page arXiv:1901.05287.

Jingzhou He and Houfeng Wang. 2008. Chinese named entity recognition and word segmentation based on character. In *SIGHAN Workshop*.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *NAACL*, pages 4129–4138.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *ACL*, pages 3651–3657.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the dark secrets of BERT. In *EMNLP-IJCNLP*, pages 4365–4374.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, page 282–289.

Hector J. Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In *KR*, page 552–561.

Xiaoya Li, Yuxian Meng, Xiaofei Sun, Qinghong Han, Arianna Yuan, and Jiwei Li. 2019. Is word segmentation necessary for deep learning of Chinese representations? In *ACL*, pages 3242–3252.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *EMNLP-IJCNLP*, pages 3730–3740.

Zhangxun Liu, Conghui Zhu, and Tiejun Zhao. 2010. Chinese named entity recognition with a sequence labeling approach: Based on characters, or based on words? In *ICIC*, page 634–640.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019. Linguistic knowledge and transferability of contextual representations. In *NAACL*, pages 1073–1094.

Chen Lyu, Yue Zhang, and Donghong Ji. 2016. Joint word segmentation, pos-tagging and syntactic chunking. In *AAAI*, pages 3007–3014.

Yuxian Meng, Wei Wu, Fei Wang, Xiaoya Li, Ping Nie, Fan Yin, Muyu Li, Qinghong Han, Xiaofei Sun, and Jiwei Li. 2019. Glyce: Glyph-vectors for chinese character representations. In *NeurIPS*, pages 2746–2757. Curran Associates, Inc.

Phu Mon Htut, Jason Phang, Shikha Bordia, and Samuel R. Bowman. 2019. Do Attention Heads in BERT Track Syntactic Dependencies? *arXiv e-prints*, page arXiv:1911.12246.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *NAACL*, pages 2227–2237.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *EMNLP-IJCNLP*, pages 2463–2473.

Yan Shao, Christian Hardmeier, Jörg Tiedemann, and Joakim Nivre. 2017. Character-based joint segmentation and pos tagging for chinese using bidirectional rnn-crf. In *IJCNLP*, pages 173–183.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R. Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. In *ICLR*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS*, pages 5998–6008.

Ralph Weischedel, Sameer Pradhan, Lance Ramshaw, Martha Palmer, Nianwen Xue, Mitchell Marcus, Ann Taylor, Craig Greenberg, Eduard Hovy, Robert Belvin, et al. 2011. Ontonotes release 4.0. *LDC2011T03, Philadelphia, Penn.: Linguistic Data Consortium*.

Jingjing Xu and Xu Sun. 2016. Dependency-based gated recursive neural network for Chinese word segmentation. In *ACL*, pages 567–572.

Hu Xu, Bing Liu, Lei Shu, and Philip Yu. 2019. BERT post-training for review reading comprehension and aspect-based sentiment analysis. In *NAACL*, pages 2324–2335.

Liang Xu, Xuanwei Zhang, Lu Li, Hai Hu, Chenjie Cao, Weitang Liu, Junyi Li, Yudong Li, Kai Sun, Yechen Xu, Yiming Cui, Cong Yu, Qianqian Dong, Yin Tian, Dian Yu, Bo Shi, Jun Zeng, Rongzhao Wang, Weijian Xie, Yanting Li, Yina Patterson, Zuoyu Tian, Yiwen Zhang, He Zhou, Shaoweihua Liu, Qipeng Zhao, Cong Yue, Xinrui Zhang, Zhengliang Yang, and Zhenzhong Lan. 2020. CLUE: A Chinese Language Understanding Evaluation Benchmark. *arXiv e-prints*, page arXiv:2004.05986.

Naiwen Xue, Fei Xia, Fu-Dong Chiou, and Marta Palmer. 2005. The penn chinese treebank: Phrase structure annotation of a large corpus. *Natural language engineering*, pages 207–238.

Nianwen Xue. 2003. Chinese word segmentation as character tagging. In *ROCLING-IJCLCLP*, pages 29–48.

Jie Yang, Shuailong Liang, and Yue Zhang. 2018. Design challenges and misconceptions in neural sequence labeling. In *COLING*, pages 3879–3889.

Yue Zhang and Stephen Clark. 2010. A fast decoder for joint word segmentation and POS-tagging using a single discriminative model. In *EMNLP*, pages 843–852.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: Enhanced language representation with informative entities. In *ACL*, pages 1441–1451.

Xiaoqing Zheng, Hanyang Chen, and Tianyu Xu. 2013. Deep learning for Chinese word segmentation and POS tagging. In *EMNLP*, pages 647–657.

Junru Zhou and Hai Zhao. 2019. Head-driven phrase structure grammar parsing on Penn treebank. In *ACL*, pages 2396–2408.