Technical Report of UER-py

A Datasets

We use a range of English and Chinese datasets to test the performance of UER. For English datasets, we use GLUE benchmark (Wang et al., 2019)[1]. Since test sets of GLUE benchmark are not available, we use development sets for evaluation[2]. For Chinese datasets, we use 1) ERNIE benchmark, which contains Chnsenticorp, LCQMC (Liu et al., 2018), XNLI[3], MSRA-NER, and nlpcc-dbqa[4]. They are respectively sentiment analysis, question pair matching, natural language inference, sequence labeling, and document-based question answering datasets. More information of these datasets can be found in ERNIE's github project[5]; 2) Douban book review (Qiu et al., 2018)[6], Shopping review, and Tencent news review. The first two are 2-way classification datasets (positive and negative), and the last one is a 3-way classification dataset (positive, neutral, and negative); 3) Three NER datasets: WeiboNER, ResumeNER, and Ontonote4NER. Detailed introduction to these NER datasets can be found in the work of Zhang and Yang (2018).

Table 1 lists the statistics of 11 Chinese datasets. We use BERTTokenizer, which tokenizes Chinese text sequence into characters. We can not distribute WeiboNER, ResumeNER, and Ontonote4NER datasets for their licences. Other datasets can be downloaded at `https://share.weiyun.com/5LQcJJP`

Dataset	N	l	$\|v\|$
Book review	20K/10K/100K	36	6863
Tencent review	370K/50K/10K	17	9815
Shopping	20K/10K/10K	50	5172
ChnSentiCorp	9600/1200/1200	105	6014
nlpcc-dbqa	181882/40996/81536	52	11208
LCQMC	238766/8802/12500	22	5835
XNLI	390702/2490/5010	49	7643
MSRA-NER	20865/2319/4637	46	4349
WeiboNER	1350/270/270	55	3127
Ontonote4NER	15724/4300/4346	31	3363
ResumeNER	3821/462/477	32	1792

Table 1: Dataset statistics. N: Number of samples in train set, dev set, and test set. l: Average length. $|V|$: Vocabulary size.

B Influence of corpora

This section explores the influence exerted by corpora to pre-training models. Chinese Google BERT is trained on Wikipedia. We load Google's pre-trained model and train it upon other corpora, including People daily news corpus and WebQA corpus. News is beneficial for NER tasks due to more information of people and organizations included; WebQA and datasets of ERNIE benchmark (except MSRA-NER) are basically generated by web users. Therefore, we use People daily news corpus on four NER datasets and use WebQA corpus on ERNIE benchmark (except MSRA-NER). Table 2 and 3 compare our pre-trained models with Google BERT. We can observe that suitable corpora can largely improve the performance of pre-training models on downstream datasets.

C Fine-tuning strategies

This section demonstrates the effectiveness of semi-supervised fine-tuning strategy. We firstly load Google's model and train it on downstream datasets with MLM target. NSP target is removed since some samples are too short to be divided

[1] https://gluebenchmark.com/

[2] HuggingFace also uses development sets for evaluation and reports the results on its github project.

[3] https://github.com/facebookresearch/XNLI

[4] http://tcci.ccf.org.cn/conference/2016/dldoc/evagline2.pdf

[5] https://github.com/PaddlePaddle/LARK/tree/develop/ERNIE

[6] https://embedding.github.io/evaluation/

Dataset	BERT	People daliy news
MSRA-NER	92.7	94.4(+1.7)
WeiboNER	64.6	67.5(+2.9)
Ontonote4NER	76.4	78.9(+2.5)
ResumeNER	94.4	95.1(+0.7)

Table 2: Comparison of Google BERT and BERT pre-trained on People daliy news corpus. Four NER datasets are used as evaluation benchmarks.

Dataset	BERT	WebQA
LCQMC	77.5	78.8(+1.3)
XNLI	86.6	87.4(+0.8)
Chnsenticorp	94.3	95.4(+1.1)
DBQA	94.6	95.0(+0.4)

Table 3: Comparison of Google BERT and BERT pre-trained on WebQA corpus. Four datasets (ERNIE benchmark except MSRA-NER) are used as evaluation benchmarks.

into multiple parts. And then models are fine-tuned in supervised manner. Table 4 shows the results on five datasets. We can observe that the implementation of a semi-supervised fine-tuning strategy further improves the results over Google BERT, which provides a new strong baseline for these datasets.

Dataset	BERT	Semi-supervised fine-tuning
Tencent review	84.2	84.7
Book review	87.5	88.1
Shopping	96.3	96.8
ChnSentiCorp	94.3	95.6
MSRA-NER	92.7	93.4

Table 4: Comparison of different fine-tuning strategies on five datasets.

D Speed

The models are run on CPU E5-2699, 256GB Mem, trained on Nvidia Tesla P40 with CUDA 9.0.176 and cuDNN 7.0.5.

Training efficiency is a major concern for pre-training models, since pre-training usually requires days to weeks. UER supports 3 training modes, CPU, single GPU, and distributed multi-GPU. BERT is used as the benchmark model to test speed. Distributed mode is supported by NCCL. Table 5 shows the number of tokens processed per second. We can observe that adding GPUs and machines can largely speed up the training process. Three machines (24 GPUs in total) are able to process over 80 thousand tokens per second. Given that BERT-base of Google is trained by 1 million steps and each step contains 128,000 tokens, it takes around 18 days to reproduce BERT's experiment by UER with 24 GPUs.

#machine	#GPU/machine	#tokens/s
1	1	7050
1	2	13071
1	4	24695
1	8	44300
3	8	84386

Table 5: The speed of the BERT on distributed mode.

E Qualitative evaluation of context-dependent word embedding

In this section, we qualitatively evaluate pre-trained models by finding words' nearest neighbours. For traditional word embedding models such word2vec (Mikolov et al., 2013) and GloVe (Pennington et al., 2014), each word is attached with a fixed vector. However, the meaning of a polysemous word may depend on its context. To capture context information, hidden states in pre-trained model can be used to represent a word. To this end, each token in corpus is represented by an unique vector. ELMO (Peters et al., 2018) calculates the vector for each token and uses the vector to find other tokens (in entire corpus). While in this work, we propose a vocabulary-based retrieval method for context-dependent word embedding.

E.1 Finding nearest neighbours by word-based model

Google only provides a character-based BERT model for Chinese. In this work, we pre-train a word-based BERT model on Wikipedia. Jieba[7] is used to do word segmentation. The model is trained by 500k steps with batch size of 256 and sequence length of 192. Figure 1 illustrates the way of finding a word's nearest neighbours given certain context: We first feed the sentence into BERT encoder, and take the hidden state vector as the word's context-dependent vector. Then we replace **Geely** with other words in the vocabulary, and calculate their context-dependent vector representations similarly. Finally, we could obtain vectors of all words in certain context. We calculate the cosine value of word vectors to find word's top-n nearest neighbours.

[7] https://github.com/fxsjy/jieba

Sentence	Character-based Model		Word-based Model	
	Neighbour	Similarity	Neighbour	Similarity
福特汽车公司宣布出售旗下高端汽车沃尔沃予中国浙江省的吉利汽车。 (Ford Motor Company announces the sale of its high-end Volvo to **Geely** Automobile in Zhejiang Province, China.)	福利(Fuly)	0.932	沃尔沃(Volvo)	0.771
	吉德(Jide)	0.917	永利(Yongli)	0.745
	吉普(Jeep)	0.915	天安(Tianan)	0.741
	宾利(Bentley)	0.909	仁和(Renhe)	0.740
	福特(Ford)	0.904	金牛座(Taurus)	0.732
主要演员有扎克·布拉夫、约翰·麦吉利、朱迪·雷耶斯等。 (The main actors are Zach Bluff, John Mc**Geely**, Judy Reyes, etc.)	科利(Coley)	0.979	玛利(Mary)	0.791
	杰利(Jerry)	0.979	米格(MiG)	0.768
	福利(Fuly)	0.978	韦利(Willy)	0.767
	莫利(Moly)	0.978	马力(Mary)	0.763
	宾利(Bentley)	0.977	安吉(Anji)	0.761
这是一个吉利的征兆。 (This is a sign of **auspiciousness**.)	吉祥(lucky)	0.900	仁德(kindness)	0.749
	吉安(good fortune)	0.873	光彩(glorious)	0.743
	吉凶(good or bad)	0.866	愉快(happy)	0.736
	吉田(Yoshida)	0.863	永元(Yongyuan)	0.736
	吉德(Gide)	0.857	仁和(Renhe)	0.732

Table 6: 吉利(Geely) has different meanings in different contexts, thus produce different nearest neighbours.

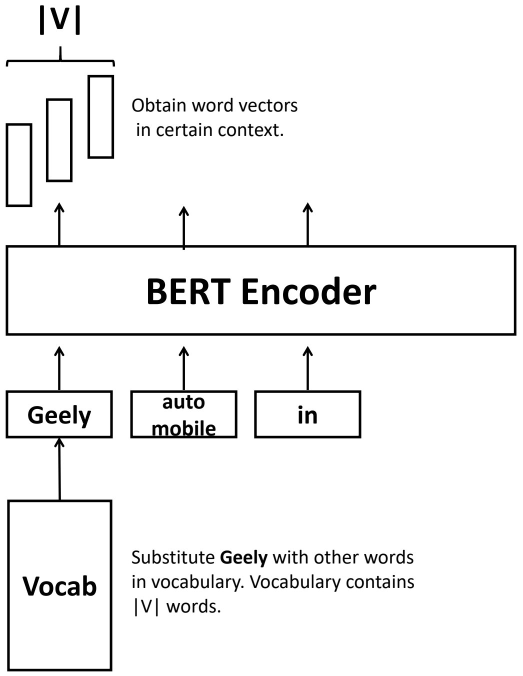
|V|
Obtain word vectors in certain context.
BERT Encoder
Geely
auto mobile
in
Vocab
Substitute Geely with other words in vocabulary. Vocabulary contains |V| words.

Figure 1: The illustration of vocabulary-based retrieval method for context-dependent word embedding.

E.2 Finding nearest neighbours by character-based model

Besides word-based model, we attempt to use character-based BERT model to find nearest neighbours of a target word. For a character-based model, each character corresponds to a hidden state. Since a word may consist of multiple characters, we take the average of characters' hidden states to obtain word vector in certain context. The remaining steps are identical with word-based models.

E.3 Case studies

Compared to traditional word embedding, context-dependent word embedding can disambiguate word meanings according to the context. For example, 吉利(Geely) is a polysemous Chinese word with three frequently-used meanings: 1) a car brand; 2) the name of a person; 3) auspiciousness. Table 6 illustrates nearest neighbors of 吉利(Geely) given different contexts.

In sentence 1, "Geely" refers to a car brand. We can observe that character-based model and word-based model perform well. "Jeep", "Bentley", "Ford", "Volvo", and "Taurus" are car brands. "Yongli", "Tianan", and "Renhe" (returned by word-based model) are polysemous words, who possess auspicious meanings and are also used as companies' names.

In sentence 2, "Geely" refers to a person's name. It can be seen that both models can capture the information that the "Geely" is the name of a person through the context. Character-based model tends to return words that share characters with the target word. For example, all names returned by character-based model include 利(ly), which is the second character of 吉利(Geely). The words returned by word-based model are more diverse.

In sentence 3, ”Geely” refers to auspiciousness. All words returned by character-based model include 吉(ji), which is the first character of 吉利(Geely). Word-based model performs well in this case. The returned words are reasonable. ”Yongyuan” and ”Renhe” stand for Chinese auspicious words.

References

Xin Liu, Qingcai Chen, Chong Deng, Huajun Zeng, Jing Chen, Dongfang Li, and Buzhou Tang. 2018. Lcqmc: A large-scale chinese question matching corpus. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1952–1962.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Yuanyuan Qiu, Hongzheng Li, Shen Li, Yingdi Jiang, Renfen Hu, and Lijiao Yang. 2018. Revisiting correlations between intrinsic and extrinsic evaluations of word embeddings. In *CCL2018*, pages 209–221. Springer.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In the Proceedings of ICLR.

Yue Zhang and Jie Yang. 2018. Chinese ner using lattice lstm. *arXiv preprint arXiv:1805.02023*.