Injecting Descriptive Meta-information into Pre-trained Language Models with Hypernetworks

Wenying Duan[1*], *Xiaoxi He*[2*], *Zimu Zhou*[3], *Hong Rao*[1], *Lothar Thiele*[2]

[1]Nanchang University
[2]ETH Zurich
[3]Singapore Management University

wenyingduan@ncu.edu.cn, hex@ethz.ch, zimuzhou@smu.edu.sg, raohong@ncu.edu.cn, thiele@ethz.ch

Abstract

Pre-trained language models have been widely adopted as backbones in various natural language processing tasks. However, existing pre-trained language models ignore the descriptive meta-information in the text such as the distinction between the title and the mainbody, leading to over-weighted attention to insignificant text. In this paper, we propose a hypernetwork-based architecture to model the descriptive meta-information and integrate it into pre-trained language models. Evaluations on three natural language processing tasks show that our method notably improves the performance of pre-trained language models and achieves the state-of-the-art results on keyphrase extraction.

Index Terms: descriptive meta-information, hypernetworks, pre-trained language model

1. Introduction

Pre-trained language models like BERT [1], RoBERTa [2], ALBERT [3] etc. have been successful in various natural language processing (NLP) tasks. After being pre-trained on large-scale language corpora, these models can be used as backbones for different downstream tasks such as Text Classification, Machine Reading Comprehension (MRC), Textual Entailment and Named Entity Recognition. For a given downstream task, the backbones only need to be fine-tuned with a few additional layers without substantial task-specific architecture modifications.

Despite their easy adoption and superior performance, these pre-trained language models only exploit *plain* text features such as word embedding and sentence embedding. That is, they treat the input text equally and ignore the inherent context in the text such as the distinction between the title and the mainbody, or semantic roles *e.g.*, who, what and how [4, 5]. Ignorance of such information can lead these deep learning based models to pay over-weighted attention to insignificant words [6, 7].

To this end, we integrate *descriptive meta-information* into pre-trained language models to improve their performance on downstream tasks. That is, instead of treating all text equally, we differentiate between *meta-text* and *main-text*, where the meta-text contains descriptive meta-information of the main-text. The descriptive meta-information for text, *e.g.*, titles, keywords and abstracts, is known to assist humans to comprehend complex text. Such information is also easily accessible in the input data of many NLP tasks. For example, the data input for text classification and keyphrase extraction can be considered as pairs of title (*meta-text*) and mainbody (*main-text*), where the title is often a high-level summary of the mainbody. Similarly, the input text for MRC is pairs of question (*meta-text*) and mainbody (*main-text*), where the question provides guidance to find answers in the mainbody.

Prior studies [8, 9, 10] model descriptive meta-information in text via *attention* mechanisms. R-Net [9] encodes questions and passages separately for MRC, applies attention to obtain question-aware context representations, and uses self-attention to further refine question-aware representations. SQLA [8] adopts a hierarchical attention network to extract both question-aware context and passage-aware context for MRC tasks. Attention mechanism has also been utilized in modeling the title and full-text (title and mainbody) for keyphrase generation. Title-Guided Net [10] encodes titles and full-text separately and applies attention-based matching to aggregate the descriptive meta-information from the title for each token in the full-text.

All the attention-based approaches [8, 9, 10] require training an end-to-end model, which often performs worse than using state-of-the-art pre-trained language models [1, 2, 3] as backbones, especially with limited training data. Moreover, as we empirically show in Sec. 3, directly applying attention-based methods on top of state-of-the-art pre-trained language models may even decrease the accuracy due to parameter redundancy [11, 12]. We argue that a naive integration of attention-based schemes into pre-trained language models is sub-optimal as the core of the transformer blocks in pre-trained language models is already based on multi-head attention [13]. Therefore, an alternative mechanism that *(i)* models descriptive meta-information and *(ii)* works in synergy with pre-trained language models is vital to further improve the performance on many NLP tasks.

In this paper, we model the descriptive meta-information via *hypernetworks* [14]. A hypernetwork is a neural network that generates the weights for another neural network. Based on the hypernetwork, we design a new architecture built upon existing pre-trained language models that extracts the descriptive meta-information in the meta-text for more effective processing of the main-text. As illustrated in Fig. 1, both meta- and main-text are first encoded via a pre-trained language model into embedding. The embedding of the meta-text is then fed into a *hyper encoding layer* to generate the weights of the *infer encoding layer*. Afterwards, the infer encoding layer converts the embedding of the main-text into representations for task-specific fine-tuning. Our proposed architecture coincides with how humans process the meta- and main-text: the interpretation of the main-text is dependent on the meta-text, where the low-level knowledge on both the meta- and main-text comes from the universal language ability (pre-trained language models).

Our main contributions and results are as follows.

- To the best of our knowledge, this is the first study that models descriptive meta-information in NLP tasks via

[*]Authors contributed equally.

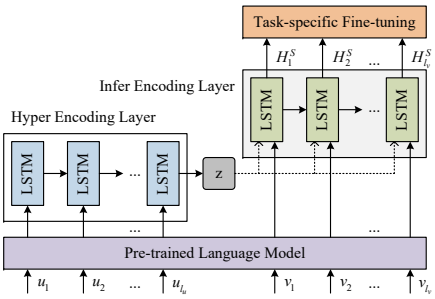

Figure 1: *Overview of our method. Meta-text* $u_1, u_2, \cdots, u_{l_u}$ *and main-text* $v_1, v_2, \cdots, v_{l_v}$ *are encoded by a pre-trained language model. The embeddings of meta-text are input into a hyper encoding layer to output descriptive meta-information* z*, which generates the parameters for the infer encoding layer. The embeddings of the main-text are fed into the infer encoding layer to generate the fused representation,* $H^s_1, H^s_2, \cdots, H^s_{l_v}$*, which is input into the task-specific layers for fine-tuning.*

hypernetworks. It is also the first attempt to model descriptive meta-information with BERT-like pre-trained models. Our proposed architecture seamlessly integrates descriptive meta-information modeling into pre-trained language models, and allows the meta-text to explicitly affect the processing of the main-text.

- Evaluations on three NLP tasks across nine benchmarks show that our hypernetwork-based architecture outperforms the state-of-the-art attention based mechanisms in modeling descriptive meta-information and works in synergy with mainstream pre-trained language models. Particularly, our model achieves a high inference accuracy on keyphrase extraction benchmarks, exceeding the state-of-the-art pre-trained models by 4.7%, 2.8%, and 4.3% on Inspec, SE-2010, and SE-2107.

2. Our Method

We represent the *meta-text* and *main-text* in NLP tasks as a text-pair $\langle u, v\rangle$ where u and v are the token sequences of the meta- and main-text, respectively, and u contains *descriptive meta-information* about v, *i.e.*, u provides a descriptive overview of the information in v. We take advantage of the existing *pre-trained language model* to encode both u and v into their embeddings. The embeddings are then fed into a hypernetwork to learn the descriptive meta-information. Specifically, the embedding of u is input into a *hyper encoding layer* and the embedding of v is input into an *infer encoding layer*, where the output of the hyper encoding layer generates the weights for the infer encoding layer. We elaborate on each component below.

2.1. Pre-trained Language Model Encoder

We use BERT [1] or other transformer-based pre-trained language model as the encoder to embed each token into a fixed-length vector. Specifically, given a text-pair (u, v), where u and v are sequences of tokens for the meta- and main-text, where the lengths of u and v are l_u and l_v, we obtain the embedding of u and v as follows:

$$\mathrm{H}^{\mathrm{uv}} = PTM\left([\mathrm{u};\mathrm{v}]\right) \tag{1}$$

$$\mathrm{H}^{u}, \mathrm{H}^{v} = Split(\mathrm{H}^{\mathrm{uv}}) \tag{2}$$

where *PTM* is a pre-trained language model. We use BERT [1], RoBERTa [2] and ALBERT [3] in our evaluation. $[\cdot;\cdot]$ denotes concatenation, and $Split(\cdot)$ means splitting, *i.e.*, $\mathrm{H}^{\mathrm{uv}} \in \mathbb{R}^{(l_u+l_v)\times N}$ is split into $H^u \in \mathbb{R}^{l_u \times N}$ and $H^v \in \mathbb{R}^{l_v \times N}$, where N is the hidden size of the pre-trained language model.

The pre-trained language models encode rich hierarchy of linguistic information *e.g.*, syntactic and semantic features into the representation of u and v [15]. However, since u and v are concatenated as the input without differentiating meta- and main-text, the output representations fail to model the impact of u on v, *i.e.*, the descriptive meta-information, which motivates us to explicitly model the descriptive meta-information via a hypernetwork, as described next.

2.2. Hyper Encoding Layer

The hyper encoding layer takes the embedding H^u of u as input to generate the weights of the infer encoding layer for modeling v. Specifically, we first apply a long short-term memory (LSTM) unit to compress H^u into a vector z, where $z \in \mathbb{R}^{N_z}$. We set N_z as a small constant (16 or 32 in this paper).

$$z = \mathit{LSTM}\left(\mathrm{H}^{\mathrm{u}}\right) \tag{3}$$

where z is the descriptive meta-information of text-pair (u, v). Then the weights of infer encoding layer are generated as:

$$W_{Ih} = W_{hz} z \tag{4}$$

$$W_{Ix} = W_{xz} z \tag{5}$$

$$W_{Ib} = W_{bz} z \tag{6}$$

where W_{hz}, W_{xz} and W_{bz} are learnable parameters. $W_{Ih} \in \mathbb{R}^{N_{Ih}}$, $W_{Ix} \in \mathbb{R}^{N_{Ix}}$, and $W_{Ib} \in \mathbb{R}^{N_{Ib}}$ are the weights of infer encoding layer, which is also an LSTM unit (see Sec. 2.3). We make the following notes on the hyper encoding layer.

- Since the weights of the infer encoding layer are dependent on z, the descriptive meta-information contained in the meta-text is explicitly modeled and conveyed to the downstream tasks.
- The LSTM unit is chosen to account for the varied length of H^u. The processing of meta-text by LSTM is consistent with human reading behavior.
- For a different text-pair (u, v), the weights of the corresponding infer encoding layer also differ. Thus, our model can be considered as an instance-wise dynamic network where the hyper encoding layer is at inference time but the parameters of the infer encoding layer adapts according to the input instances. Instance-wise dynamic networks are effective in improving the representation of neural networks in various applications [16].

2.3. Infer Encoding Layer

The infer encoding layer is another LSTM unit. The standard formulation of a LSTM is given by:

$$\begin{aligned} f_t &= \sigma\left(W_{fh} h_{t-1} + W_{fx} x_t + b_f\right) \\ i_t &= \sigma\left(W_{ih} h_{t-1} + W_{ix} x_t + b_i\right) \\ \tilde{c}_t &= \tanh\left(W_{ch} h_{t-1} + W_{cx} x_t + b_c\right) \\ c_t &= f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \\ o_t &= \sigma\left(W_{oh} h_{t-1} + W_{ox} x_t + b_o\right) \\ h_t &= o_t \odot \tanh\left(c_t\right) \end{aligned} \tag{7}$$

where H^t_v from H_v is the input x_t at time step t, h_t is the hidden state at t, σ is sigmoid function, W_{fh}, W_{fx}, W_{ih}, W_{ix}, W_{ch}, W_{cx}, W_{oh}, W_{ox}, b_f , b_i , b_c and b_o are learnable parameters.

As mentioned in Sec. 2.2 , all the parameters of the infer encoding layers are generated by W_{Ih}, W_{Ix}, and W_{Ib}:

$$\begin{aligned}(W_{fh}, W_{ih}, W_{ch}, W_{oh}) &= \text{Chunk}\,(W_{Ih}) \\ (W_{fx}, W_{ix}, W_{cx}, W_{ox}) &= \text{Chunk}\,(W_{Ix}) \\ (b_f, b_i, b_c, b_o) &= \text{Chunk}\,(W_{Ib})\end{aligned} \tag{8}$$

where W_{fh} , W_{ih} , W_{ch} , $W_{oh} \in \mathbb{R}^{N_h \times N_h}$, $N_h \times 4N_h = N_{Ih}$ and W_{fx} , W_{ix} , W_{cx} , $W_{ox} \in \mathbb{R}^{N_h \times N_x}$,$N_h \times 4N_x = N_{Ix}$ and b_f , b_i , b_c , $b_o \in \mathbb{R}^{N_h}$, $4N_h = N_{ib}$. Chunk is the operation to split a tensor into a specific number of equal-sized chunks.

2.4. Putting It Together

From Sec. 2.2 and Sec. 2.3, the embedding H^u of the meta-text generates the descriptive meta-information z, which generates the parameters for the infer encoding layer that takes the embedding H^v of the main-text as input. We implement the above hypernetwork via LSTM units, *i.e.*,

$$H^S = LSTM(H_v; z) \tag{9}$$

where H^S is the fused representation of the input text-pair $\langle u, v\rangle$ for downstream tasks. The embeddings H^u and H^v are encoded by pre-trained language models as in [1, 2, 3].

During training, only the parameters of hyper encoding layer, infer encoding layer and task-specific layers will be updated jointly by gradient descent.

3. Evaluation

We experiment with three popular NLP tasks: text classification, machine reading comprehension, and keyphrase extraction. As introduced in Sec. 2, our method transforms an input text-pair into a representation H^S, which can be then fed into a task-specific layer for fine-tuning as in standard pre-trained language models. The input text is tokenized dependent on the pre-trained language model, with a maximum length of 384 for machine reading comprehension and 512 for the other two tasks. We set the hidden size of hyper encoding layer and infer encoding layer to 512. All models are implemented in PyTorch. We present the detailed setups and results for each task below.

3.1. Text Classification

Datasets. We experiment on four datasets: THUNews [17], AGnews[18], INEWS[1], and ToutiaoNews[2]. THUNews is a widely used Chinese news classification benchmark. Each instance is labeled with one of the 14 news categories (finance, technology, sports, etc.). The task is to predict the category each instance belongs to. AGnews is similar to THUNews but is a English dataset for news classification. INEWS is a Chinese news dataset for sentiment classification (positive, negative or neutral). ToutiaoNews contains Chinese news published by TouTiao. An instance in the former three datasets is in the form of $\langle title, passage\rangle$ whereas an instance in ToutiaoNews is $\langle title, keyphraseset\rangle$. Table 1 summarizes their statistics.

Setups. The representation H^S is directly passed to a fully connected layer to obtain the class logits. The training objectives are Cross-Entropy and we use Adam with weight decay regularization as the optimizer. The learning rate is set to 2e-5 with weight decay of 0.01. We choose BERT [1], RoBERTa [2] and ALBERT [3] as baselines. Specifically, BERT-large-uncased[3], RoBERTa-large[4], ALBERT-xlarge[5] are chosen for English corpus while BERT-large-zh[3], RoBERTa-wwm-ext-large [19] and ALBERT-xlarge-zh[5] for Chinese corpus.

Table 1: *Statistics of text classification datasets.*

Corpus	Class	Train	Dev	Test
THUNews	14	33.3k	4.2k	4.2k
ToutiaoNews	15	267.8k	57k	57k
AGnews	4	120k	N/A	7.6k
INEWS	3	5.4K	1k	1k

Results. Table 2 summarizes the classification accuracy of different pre-trained language models without and with hypernetworks for incorporating the descriptive meta-information in the text pairs. For the same pre-trained language model, the classification accuracy increases by 0.18% to 8% when adding the hypernetwork. For all the four datasets, the highest classification accuracy is achieved when the hypernetwork is integrated into the pre-trained language model.

Table 2: *Performance of text classification.*

Method	INEWS	THU News	Toutiao News	AGnews
BERT	74.14%	90.36%	85.53%	94.55%
RoBERTa	75.91%	90.32%	86.98%	94.98%
ALBERT	75.0%	90.29%	85.67%	94.97%
BERT+Hypernet (ours)	80.13%	93.47%	89.21%	95.06%
RoBERTa+Hypernet (ours)	80.91%	**93.56%**	**90.05%**	**95.36%**
ALBERT+Hypernet (ours)	**80.22%**	93.50%	89.34%	95.14%

3.2. Machine Reading Comprehension

Datasets. We experiment with two MRC benchmark datasets: SQuAD 1.1 [20] and SQuAD 2.0 [21]. SQuAD 1.1 contains over 100,000 question-answer pairs from more than 500 articles. The task is to predict the answer text span in the passage given a question and a passage containing the answer. SQuAD 2.0 combines the 100,000 questions in SQuAD 1.1 with over 50,000 new, unanswerable questions. For SQuAD 2.0, the task is to not only answer questions when possible, but also decide whether there is an answer in the paragraph.

Setups. In this task, the representation H^S is passed to a fully connected layer to get the start logits s and end logits e of all tokens. We use cross-entropy as the training objective. Total span extraction loss is the sum of a cross-entropy for the start and end positions. We use Adam with weight decay regularization as the optimizer and set the learning rate of 3e-5 with weight decay of 0.01. In addition to the naive BERT and its variants as baselines, we also experiment with the following methods.

- R-Net [9]: a state-of-the-art method for MRC, which uses a gated attention-based recurrent network to learn the matching between the question and the passage and a self-attention layer to learn the global information of the passage. We use pre-trained language models to obtain the representation of the question and the passage. Then, the representation of the question and the passage are fed into R-Net to predict the answer span.

[1] https://github.com/CLUEbenchmark/CLUE

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

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

[4] https://huggingface.co/transformers

[5] https://github.com/google-research/ALBERT

- SLQA [8]: a novel hierarchical attention model for MRC. It applies co-attention and self-attention to focus on the correct answer span. SLQA is integrated with pre-trained language models in the same way as R-Net.

Results. Table 3 lists the Exact Match (EM) and F1 score of different methods. Comparing the results on the same pre-trained language model, our approach outperforms the attention-based R-net and SLQA. SLQA and R-Net sometimes perform even worse than the naive pre-trained language models. For example, the EM and F1 score of SLQA+BERT are lower than BERT on both SQuAD 1.1 and SQuAD 2.0. This might be explained by parameter redundancy. Research [11, 12] has shown that there are redundant parameters in pre-trained language models that adopt multi-head attention. Integrating attention-based R-Net or SLQA with the transformer-based language models adds more attention, which may exacerbate the parameter redundancy problem. Among all the methods, RoBERTa+Hypernet performs the best. This is expected because RoBERTa-large is a more powerful pre-trained language model than BERT-large-uncased and ALBERT-xlarge on MRC task [1, 2, 3].

Table 3: *Performance of machine reading comprehension.*

Method	SQuAD 1.1 (EM/F1)	SQuAD 2.0 (EM/F1)
BERT	84.1*/90.9*	78.7*/81.9*
RoBERTa	88.9/94.6	86.5*/89.4*
ALBERT	86.4*/92.9*	84.1*/87.9*
BERT+R-Net [9]	84.9/91.9	81.9/85.1
RoBERTa+R-Net [9]	88.7/94.5	87.1/89.7
ALBERT+R-Net [9]	86.8/92.9	83.6/86.5
BERT+SLQA [8]	84.3/90.7	77.0*/80.2*
RoBERTa+SLQA [8]	86.7/92.1	86.9/89.9
ALBERT+SLQA [8]	86.5/92.8	84.2/88.0
BERT+Hypernet (ours)	87.4/93.3	84.5/86.4
RoBERTa+Hypernet (ours)	**89.7/95.2**	**87.2/90.1**
ALBERT+Hypernet (ours)	89.2/94.8	85.8/87.3

* result from the published paper; otherwise from our implementation.

3.3. Keyphrase Extraction

Datasets. We test on three keyphrase extraction datasets: Inspec [22], SemEval-2010 [23] and SemEval-2017 [24]. Inspec contains title and abstracts from 2000 scientific articles. SE-2010 consists of 284 articles published by ACM and SE-2017 contains 500 open access scientific articles by ScienceDirect. Table 4 summarizes the statistics of the datasets.

Table 4: *Statistics of keyphrase extraction datasets.*

Corpus	Train	Dev	Test
Inspec	1k	0.5k	0.5k
SE-2010	0.13K	14	0.1k
SE-2017	0.35k	50	0.1k

Setups. In this task, the representation H^S is passed to a fully connected layer with a conditional random field (CRF) to get the label scores of all tokens. We include CRF as part of the task-specific layer because adding the CRF significantly improve the performance of keyphrase extraction [25]. The training objective here is to maximize the log-likelihood. We use Adam with weight decay regularization as the optimizer, and set the learning rate of 2e-3 with weight decay of 0.01.

In addition to the naive pre-trained language models[25], we also experiment with the Title-Guided Network (TG) [10], a state-of-the-art sequence-to-sequence model which uses a title-guided encoder for keyphrase generation. Specifically, we integrate the encoder of TG (the main contribution of TG) into the pre-trained language model. We first use the language model to obtain the representations of the title and the full-text (including title and the mainbody). Then these representation vectors are fed into TG to predict the label of each token.

Results. Table 5 shows the F1 score of different methods. Compared with using the pre-trained language models alone, adding both TG and our hypernetwork notably improves the performance on all datasets. Our hypernetwork achieves more notable improvement, validating the advantage of the hypernetwork over the attention-based approach. The results also demonstrate employing hypernetwork helps the model work better with very small training dataset, which is critical for most NLP tasks as large-scale annotated data is unavailable [4].

Table 5: *Performance of keyphrase extraction.*

Method	Inspec	SE-2010	SE-2017
BERT+BiLSTM-CRF[25]	0.591	0.330	0.522
SciBERT+BiLSTM-CRF[25]	0.593	0.357	0.521
RoBERTa+BiLSTM-CRF[25]	0.595	0.278	0.508
ALBERT+BiLSTM-CRF	0.591	0.346	0.524
BERT+TG-CRF [10]	0.595	0.334	0.530
SciBERT+TG-CRF [10]	0.597	0.363	0.535
RoBERTa+TG-CRF [10]	0.604	0.297	0.521
ALBERT+TG-CRF [10]	0.601	0.346	0.522
BERT+Hypernet-CRF (ours)	0.620	0.349	0.546
SciBERT+Hypernet-CRF (ours)	0.621	**0.367**	0.544
RoBERTa+Hypernet-CRF (ours)	**0.623**	0.348	0.533
ALBERT+Hypernet-CRF (ours)	0.621	0.353	**0.547**

Please note that [25] did not report the results on ALBERT.

4. Conclusion

In this paper, we propose a novel hypernetwork-inspired architecture to integrate descriptive meta-information into pre-trained language models. We explicitly differentiate the meta- and the main-text and design an LSTM-based hypernetwork to model meta-text-dependent processing of the main-text. Evaluations on text classification, machine language comprehension, and keyphrase extraction show that our method notably outperforms the state-of-the-art pre-trained language models and existing attention-based descriptive meta-information modeling schemes. One future direction of our work is to extend to multi-modal data *e.g.*, text-speech pairs, and exploit hypernetworks to integrate descriptive meta-information into multi-modal pre-training models [26, 27, 28] for spoken language understanding, visual-linguistic, and other tasks.

5. Acknowledgement

Wenying Duan and Hong Rao's work was supported in part by the Natural Science Foundation of China (NSFC) (Grant 81960325) and the Association of Fundamental Computing Education in Chinese Universities (Grant 2021-AFCEC-055). Part of Xiaoxi He and Lothar Thiele's work was supported by the Swiss National Science Foundation in the context of the NCCR Automation. Zimu Zhou's research was supported by the Singapore Ministry of Education (MOE) Academic Research Fund (AcRF) Tier 1 grant. Zimu Zhou is the corresponding author.

6. References

[1] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[2] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.

[3] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "Albert: A lite bert for self-supervised learning of language representations," in *Proceedings of the International Conference on Learning Representations*, 2020.

[4] Z. Zhang, Y. Wu, H. Zhao, Z. Li, S. Zhang, X. Zhou, and X. Zhou, "Semantics-aware bert for language understanding," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020, pp. 9628–9635.

[5] A. Traylor, C. Chen, B. Golshan, X. Wang, Y. Li, Y. Suhara, J. Li, C. Demiralp, and W.-c. Tan, "Enhancing review comprehension with domain-specific commonsense," *arXiv preprint arXiv:2004.03020*, 2020.

[6] P. K. Mudrakarta, A. Taly, M. Sundararajan, and K. Dhamdhere, "Did the model understand the question?" in *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2018, pp. 1896–1906.

[7] R. Jia and P. Liang, "Adversarial examples for evaluating reading comprehension systems," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 2021–2031.

[8] W. Wang, M. Yan, and C. Wu, "Multi-granularity hierarchical attention fusion networks for reading comprehension and question answering," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2018, pp. 1705–1714.

[9] W. Wang, N. Yang, F. Wei, B. Chang, and M. Zhou, "Gated self-matching networks for reading comprehension and question answering," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2017, pp. 189–198.

[10] W. Chen, Y. Gao, J. Zhang, I. King, and M. R. Lyu, "Title-guided encoding for keyphrase generation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2019, pp. 6268–6275.

[11] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter," *arXiv preprint arXiv:1910.01108*, 2020.

[12] K. Clark, U. Khandelwal, O. Levy, and C. D. Manning, "What does bert look at? an analysis of bert's attention," in *Proceedings of the ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, 2019, pp. 276–286.

[13] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proceedings of the International Conference on Neural Information Processing Systems*, 2017, pp. 6000—-6010.

[14] D. Ha, A. Dai, and Q. V. Le, "Hypernetworks," *arXiv preprint arXiv:1609.09106*, 2016.

[15] G. Jawahar, B. Sagot, and D. Seddah, "What does bert learn about the structure of language?" in *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 3651–3657.

[16] Y. Han, G. Huang, S. Song, L. Yang, H. Wang, and Y. Wang, "Dynamic neural networks: A survey," *arXiv preprint arXiv:2102.04906*, 2021.

[17] S. Maosong, L. Jingyang, G. Zhipeng, Z. Yu, Z. Yabin, S. Xiance, and L. Zhiyuan, "Thuctc: An efficient chinese text classifier," *GitHub Repository*, 2016.

[18] G. M. D. Corso, A. Gulli, and F. Romani, "Ranking a stream of news," in *Proceedings of the International Conference on World Wide Web*, 2005, pp. 97–106.

[19] Y. Cui, W. Che, T. Liu, B. Qin, S. Wang, and G. Hu, "Revisiting pre-trained models for Chinese natural language processing," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing: Findings*, 2020, pp. 657–668.

[20] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "SQuAD: 100,000+ questions for machine comprehension of text," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2016, pp. 2383–2392.

[21] P. Rajpurkar, R. Jia, and P. Liang, "Know what you don't know: Unanswerable questions for SQuAD," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2018, pp. 784–789.

[22] A. Hulth, "Improved automatic keyword extraction given more linguistic knowledge," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2003, pp. 216–223.

[23] S. N. Kim, O. Medelyan, M.-Y. Kan, and T. Baldwin, "Semeval-2010 task 5: Automatic keyphrase extraction from scientific articles," in *Proceedings of the International Workshop on Semantic Evaluation*, 2010, pp. 21–26.

[24] I. Augenstein, M. Das, S. Riedel, L. Vikraman, and A. McCallum, "Semeval 2017 task 10: Scienceie-extracting keyphrases and relations from scientific publications," *arXiv preprint arXiv:1704.02853*, 2017.

[25] D. Sahrawat, D. Mahata, H. Zhang, M. Kulkarni, A. Sharma, R. Gosangi, A. Stent, Y. Kumar, R. R. Shah, and R. Zimmermann, "Keyphrase extraction as sequence labeling using contextualized embeddings," in *Advances in Information Retrieval*, 2020, pp. 328–335.

[26] W. Su, X. Zhu, Y. Cao, B. Li, L. Lu, F. Wei, and J. Dai, "VL-BERT: pre-training of generic visual-linguistic representations," in *Proceedings of the International Conference on Learning Representations*, 2020.

[27] J. Lu, D. Batra, D. Parikh, and S. Lee, "Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32, 2019.

[28] Y.-S. Chuang, C.-L. Liu, H.-Y. Lee, and L. shan Lee, "Speechbert: An audio-and-text jointly learned language model for end-to-end spoken question answering," *arXiv preprint arXiv:1910.11559*, 2020.