YATO: Yet Another deep learning based Text analysis Open toolkit

Zeqiang Wang[1]*, **Yile Wang**[2]*, **Jiageng Wu**[1], **Zhiyang Teng**[3], **Jie Yang**[1]†

[1]Zhejiang University
[2]Institute for AI Industry Research (AIR), Tsinghua University
[3]Nanyang Technological University

zeqiangwang.medicalai@outlook.com   wangyile@air.tsinghua.edu.cn
jiagengwu@zju.edu.cn   zhiyang.teng@ntu.edu.sg   jieynlp@gmail.com

Abstract

We introduce YATO, an open-source, easy-to-use toolkit for text analysis with deep learning. Different from existing heavily engineered toolkits and platforms, YATO is lightweight and user-friendly for researchers from cross-disciplinary areas. Designed in a hierarchical structure, YATO supports free combinations of three types of widely used features including 1) traditional neural networks (CNN, RNN, *etc*.); 2) pre-trained language models (BERT, RoBERTa, ELECTRA, *etc*.); and 3) user-customized neural features via a simple configurable file. Benefiting from the advantages of flexibility and ease of use, YATO can facilitate fast reproduction and refinement of state-of-the-art NLP models, and promote the cross-disciplinary applications of NLP techniques. The code, examples, and documentation are publicly available at `https://github.com/jiesutd/YATO`. A demo video is also available at `https://www.youtube.com/playlist?list=PLJ0mhzMcRuDUlTkzBfAftOqiJRxYTTjXH`.

1 Introduction

Large language models (LLMs) such as GPT-3 (Brown et al., 2020), ChatGPT (OpenAI, 2022), and LLaMA (Touvron et al., 2023a,b) have gained significant progress in natural language processing (NLP), showing strong abilities to understand text and competitive performance across various NLP tasks. However, these models are either close-source or difficult to fine-tune due to the high computational costs, which makes them inconvenient for academic research or practical implementation. Alternatively, traditional neural models, such as recurrent neural networks (RNN, Hochreiter and Schmidhuber, 1997), convolutional neural networks (CNN, LeCun et al., 1989), and pre-trained language models (PLMs, Devlin et al., 2019; Liu

et al., 2019; Clark et al., 2020) have been widely studied and utilized for text understanding. These models benefit from large-scale training data and can be quickly fine-tuned toward specific usages. Recent works also show they can offer useful guidance to LLMs (Xu et al., 2023). Therefore, small open-source deep learning models are important in current NLP systems, especially in computation and data resource-limited scenarios.

However, due to the complexity of the deep learning model architecture, it is challenging to implement methods or reproduce results from the literature. The different implementations of these models can lead to unfair comparisons or misleading results. Most existing frameworks were designed for professional developers, which brings additional obstacles for less experienced users, especially for researchers with less or no artificial intelligence (AI) background (Zacharias et al., 2018; Zhang et al., 2020; Johnson et al., 2021). In addition, these frameworks seldom support user-defined features required for various domain applications (*e.g.*, in medical named entity recognition, customized lexicons can be supplemented as external features, such that additional labels are tagged as features when a word occurs in the lexicon). For non-expert, cross-domain users, customizing models via source code with additional features is complex. To promote interdisciplinary applications of cutting-edge NLP techniques, it is necessary to build a flexible, user-friendly, and effective text representation framework that supports a wide range of deep learning architectures and customized domain features.

There exist several text analysis toolkits in the NLP community. CoreNLP (Manning et al., 2014) and spaCy (Honnibal and Montani, 2017) offer pipelines for many traditional NLP tasks, while the performance is sometimes under-optimal due to the use of less powerful models. AllenNLP (Gardner et al., 2017) and flairNLP (Akbik et al., 2019) utilize pre-trained models while they do not support

---

*Equal contribution.
†Corresponding author.

System	Lightweight	PLM	Neural Models	User-Defined Features	Configurable w/o Coding	SOTA Performance	Reference
CoreNLP	✓	✗	✗	✗	✗	✗	(Manning et al., 2014)
spaCy	✓	✓	✓	✗	✗	✗	(Honnibal and Montani, 2017)
AllenNLP	✓	✓	✓	✗	✗	✓	(Gardner et al., 2017)
FlairNLP	✓	✓	✓	✗	✗	✓	(Akbik et al., 2019)
NCRF++	✓	✗	✓	✓	✓	✓	(Yang and Zhang, 2018)
FairSeq	✗	✓	✓	✗	✗	✓	(Ott et al., 2019)
Transformers	✗	✓	✓	✗	✗	✓	(Wolf et al., 2020)
PaddleNLP	✗	✓	✓	✗	✗	✓	(Contributors, 2021)
EasyNLP	✗	✓	✓	✗	✗	✓	(Wang et al., 2022a)
YATO	✓	✓	✓	✓	✓	✓	**This paper**

Table 1: Comparison between existing popular text analysis libraries and our proposed YATO.

user-defined features. FairSeq (Ott et al., 2019) is designed for sequence-to-sequence tasks like machine translation and document summarization. Transformers (Wolf et al., 2020) offers implementation for various tasks by using state-of-the-art models across different modalities, while it is heavily engineered. PaddleNLP (Contributors, 2021) and EasyNLP (Wang et al., 2022a) are specifically designed for industrial application and commercial usage, which are not lightweight for research purposes. The above toolkits are mostly developed for professional AI researchers or engineers, where heavy coding effort is necessary during model development and deployment. The learning curve is steep to fully leverage these toolkits for cross-disciplinary researchers (*e.g.*, medical, financial) who need to build models with lightweight code.

This paper presents a toolkit, **YATO** (**Y**et **A**nother deep learning based **T**ext analysis **O**pen toolkit), for researchers looking for a convenient way of building state-of-the-art models for two most popular types of NLP tasks: sequence labeling (*e.g.*, Part-of-Speech tagging, named entity recognition) and sequence classification (*e.g.*, sentiment analysis, document classification). YATO is built on NCRF++ (Yang and Zhang, 2018), a popular neural sequence labeling toolkit with over 250+ citations from research papers, 1,900+ stars and 120+ merged pull requests on GitHub as of Oct. 2023. NCRF++ has been utilized in many cross-disciplinary research projects, including medical (Yang et al., 2020) and finance (Wan et al., 2021). YATO retains its strengths, integrates advanced pre-trained language models, and adds capabilities for sequence classification and data visualization.

2 Highlights of YATO

Table 1 lists the comparison of YATO and popular existing text analysis libraries. The highlights of YATO include:

● **Lightweight.** YATO focuses on two fundamental while popular NLP tasks: sequence labeling and sequence classification, covering many downstream applications such as information extraction, sentiment analysis, text classification, *etc*. Different from the heavily engineered libraries, YATO is concise and lightweight with less library dependence. It can be fast developed and deployed in various environments, making it a user-friendly toolkit for less experienced users.

● **Flexible.** Most of the existing libraries do not support the combination of various neural features. By using YATO, users can customize their models through free combinations of various neural models, including traditional neural networks (CNN, RNN) and state-of-the-art PLMs, as well as hand-crafted features for domain adaptation. YATO also supports various inference layers, including attention pooling, softmax, conditional random field (CRF), and *nbest* decoding.

● **Configurable.** To minimize the effort of coding, all the model developments on YATO can be easily conducted by editing the configuration file. YATO will load the configuration file and construct the deep learning models following the configurations.

● **Easy to Use.** YATO is built based on PyTorch[1] and has been released on PyPI[2], the installation can be done through *pip install ylab-yato*. For non-AI users, editing a configuration file to build deep learning models is simple and intuitive. For AI users, YATO provides various modularized functions for professional development.

● **High Performance.** In extensive experiments on sequence labeling and classification tasks, YATO proves that it can achieve state-of-the-art performance on most tasks and datasets. YATO offers flexibility in terms of hardware resources, supporting both GPU and CPU for training and inference

---

[1] https://pytorch.org/.
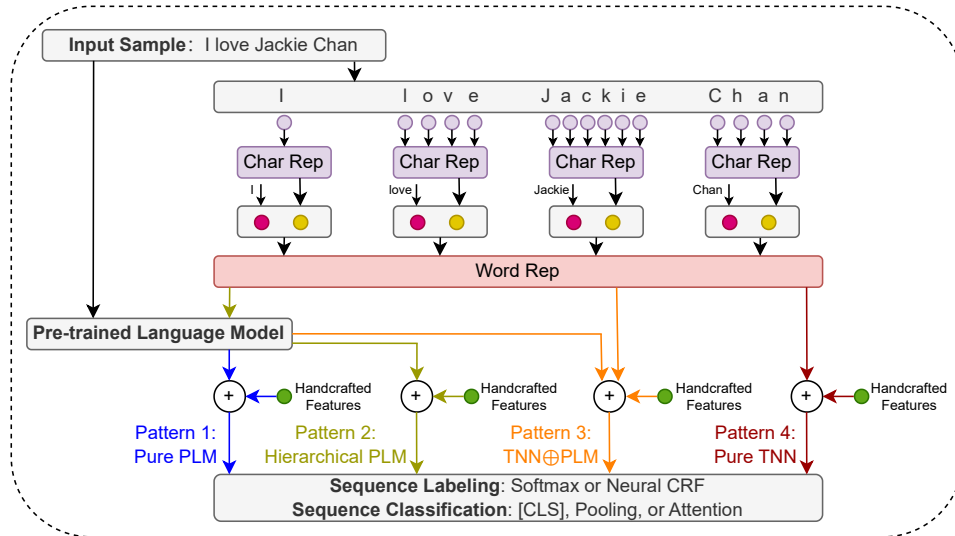[2] https://pypi.org/.

Figure 1: The overall architecture of YATO. The purple, red, yellow, and green circles represent character embeddings, word embeddings, character sequence representation, and custom handcrafted features, respectively.

processes. It provides the ability to specify the desired device configuration, facilitating efficient utilization of multiple GPUs on a single server.

• **Visualization.** YATO offers the interface for visualizing text attention, which can help users further interpret and analyze the results.

3 Architecture of YATO

YATO is designed *hierarchically* to support flexible combinations of character-level features, word-level features, and pre-trained language models[3], as well as handcrafted features. As illustrated in Figure 1, YATO supports four patterns to represent text as embeddings and with flexible choices on adding handcrafted features and inference layers.

3.1 Text Representations

Pure Pre-trained Language Model. YATO enables the initialization of parameters with pre-trained language models such as BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), and ELECTRA (Clark et al., 2020), and fine-tunes them on training data. Leveraging the rich knowledge inside the PLM, they have demonstrated strong performance on downstream tasks. To better leverage the models with domain-specific knowledge, YATO also supports pre-trained models designed for specific tasks, such as SciBERT (Beltagy et al., 2019), BioBERT (Lee et al., 2020) and others.

Hierarchical Pre-trained Language Model. The

hierarchical pre-trained language model in YATO differs from the conventional notion of hierarchy, which typically describes relationships between word, sentence, and document structures. Instead, it signifies the ordinal relation between the traditional neural network and the pre-trained language model. Specifically, YATO supports using both word sequence features and the pre-trained language model representations in a hierarchical way, where the word and character features can be explicitly encoded in advance and used as the input for the pre-trained language model.

Traditional Neural Network (TNN) & Pre-trained Language Model. In contrast to the hierarchical combination, we can use the word sequence features directly before the final prediction layer, combined with the representation brought from the PLMs. Such a feature-based approach is also used in ELMo (Peters et al., 2018) and BERT (Devlin et al., 2019), which shows close performance while does not require fine-tuning the pre-trained models.

Pure Traditional Neural Network. Besides using the transformer-based pre-trained models, we also support traditional neural models such as RNN, CNN, and BiLSTM. Compared with Transformer, these models usually have fewer parameters and are also shown effective for sequence modeling (Ma and Hovy, 2016; Lample et al., 2016; Yang et al., 2018), especially when the training data is limited.

3.2 Handcrafted Features and Inference

Handcrafted Features . YATO provides feature embedding modules to encode any handcrafted fea-

---

[3]It supports all available pre-trained models from https://huggingface.co/models.

133

```
#Dataloader
train_dir=The path of train dataset
dev_dir=The path of development dataset
...
#Model
use_crf=True/False
use_char=True/False
char_seq_feature=GRU/LSTM/CNN/False
word_seq_feature=GRU/LSTM/CNN/FeedFowrd/False
low_level_transformer=pretrain language model
high_level_transformer=PLM from huggingface
bilstm=True/False
...
#Handcrafted Features
feature=[POS] emb_size=20 emb_dir=your POS embedding
feature=[Cap] emb_size=20 emb_dir=your Cap embedding
...
#Hyperparameters
sentence_classification=True/False
status=train/decode
iteration=epoch number
batch_size=batch size
optimizer=SGD/Adagrad/adadelta/rmsprop/adam/adamw
learning_rate=learning rate
...
#Prediction
raw_dir=The path of decode file
decode_dir=The path of the decode result file
nbest=0 (for labeling)/1 (for classification)
...
```

Table 2: A sample of a configuration file.

tures and the corresponding feature embeddings can be concatenated to the text representation on any patterns. Users can add feature embeddings by formatting the text following YATO instructions without any coding effort.

Inference Layers. The encoded text embeddings from the text representation structure are fed into a softmax or CRF (Lafferty et al., 2001) layer for sequence labeling tasks, and YATO also supports *nbest* decoding to generate more candidate label sequences with probabilities for further optimization. For sequence classification tasks, a classifier head over [CLS] representation, or pooling/attention layer on all word representations is used, and a visualization tool is also available in YATO to visualize the word importance in the attention layer.

4   The Overall Workflow and Usage

The overall workflow of YATO comprises three primary components to facilitate solving downstream tasks: configuration preparation, model training, and model decoding. Users can leverage this modular pipeline to tackle a wide range of applications.
Configuration Preparation. Users can specify the

dataset, model, optimizer, and decoding through the configuration file, as shown in Table 2. In particular, different patterns can be customized by setting the values of "high_level_transformer", "low_level_transformer", "bilstm", and different word sequence representations can be easily designed through specifying "char_seq_feature" and "word_seq_feature". Handcrafted features can be added through "feature".

Model Training. YATO enables efficient training of high-performance models for sequence labeling and sentence classification with minimal code and configuration file specification. For example, users can train a competitive named entity recognition or text classification model using a few lines of code and a configuration file. The simplicity of the YATO interface allows rapid prototyping and experimentation for these fundamental NLP tasks. The framework was designed in batch computation which can fully utilize the power of GPUs.

Model Decoding. Similar to model training, simple file configuration can be used to enable YATO. Besides the greedy decoding, YATO also supports *nbest* decoding, *i.e.*, which decodes label sequences with the top n probabilities by using the Viterbi decoding in neural CRF layers. The *nbest* results can serve as important resources for further optimizations, *e.g.*, reranking (Yang et al., 2017).

5   Experiments

5.1   Datasets and Main Results

To evaluate our framework, we evaluated 8 datasets that cover sequence labeling and classification tasks in both English and Chinese, including named entity recognition (NER) on CoNLL2003 (Tjong Kim Sang and De Meulder, 2003), OntoNotes (Hovy et al., 2006) and MSRA (Levow, 2006); CCG supertagging on CCG-Bank (Hockenmaier and Steedman, 2007); sentiment analysis on SST2, SST5 (Socher et al., 2013), and ChnSentiCorp (Tan and Zhang, 2008).

Table 3 and Table 4 demonstrate that YATO can reproduce both classical and state-of-the-art deep learning models on most sequence labeling and classification tasks. For some results such as BERT on CoNLL, the originally reported 92.4 F1 score by Devlin et al. (2019) may not be achieved with current libraries, as discussed in previous literature (Stanislawek et al., 2019; Gui et al., 2020). Overall, YATO achieves the best performance on MSRA, OntoNotes 4.0, CCG supertagging, and

Model	CoNLL 2003		OntoNotes 5.0		MSRA		Ontonotes 4.0		CCG supertagging	
	YATO	Ref.	YATO	Ref.	YATO	Ref.	YATO	Ref.	YATO	Ref.
CCNN+WLSTM+CRF	**91.26**	91.11[a]	81.53	-	92.83	-	74.55	-	93.80	-
BERT-base	91.61	92.4[b]	84.68	85.54[e]	**95.81**	94.71[f]	**80.57**	79.93[f]	**96.14**	92.16[h]
RoBERTa-base	**90.23**	90.11[c]	**86.28**	86.2[c]	**96.02**	95.1[g]	**80.94**	80.37[e]	96.16	-
ELECTRA-base	**91.59**	91.5[d]	85.25	87.6[d]	96.03	-	80.47	-	96.29	-

Table 3: Results for sequence labeling tasks. **Bold** represents that YATO's re-produce is more accurate. [a](Yang et al., 2018) [b](Devlin et al., 2019) [c](Liang et al., 2020) [d](Shelmanov et al., 2021) [e](Wang et al., 2022b) [f](Liu et al., 2021) [g](Li et al., 2022) [h](Durrani et al., 2020)

Model	SST2		SST5		ChnSentiCorp	
	YATO	Ref.	YATO	Ref.	YATO	Ref.
CCNN+WLSTM+CRF	87.61	-	43.48	-	88.22	-
BERT-base	**93.00**[†]	92.7[†a]	**53.48**	53.2[d]	**95.96**	95.3[g]
RoBERTa-base	92.55[†]	94.8[†b]	51.99	56.4[e]	**96.04**	95.6[h]
ELECTRA-base	94.72[†]	95.1[†c]	**55.11**	54.8[f]	**95.96**	94.5[g]

Table 4: Results of sequence classification tasks. **Bold** represents that YATO re-produce is more accurate. † denotes the results of the dev set. [a](Devlin et al., 2019) [b](Zaheer et al., 2020) [c](Clark et al., 2020) [d](Munikar et al., 2019) [e](Sun et al., 2020) [f](Xia et al., 2022) [g](Cui et al., 2021) [h](Xin et al., 2020)

Patterns	SST5	CoNLL 2003
1. Pure PLM	53.48	**91.61**
2. Hierarchical PLM	53.77	90.52
3. TNN⊕ PLM	**54.84**	90.47
4. Pure TNN	43.48	91.26

Table 5: Performances of different training patterns.

	NCBI-disease	Yidu-s4k
Pure PLM	84.23	82.73
+Handcrafted Features	**84.85** (Cap)	82.98 (Lexicon2)
	84.78 (Lexicon1)	**83.46** (Lexicon3)

Table 6: Handcraft features. Lexicon1 (English medical glossary), Lexicon2 (medical word list from THUOCL), and Lexicon3 (Chinese medical glossary).

ChnSentiCorp. The compatibility and reproducibility across different models and tasks demonstrate that YATO can serve as a platform for reproducing and comparing different methods from classical neural models to state-of-the-art PLMs.

5.2 Comparison of Different Patterns

Table 5 shows the performance of four different model patterns on both sequence labeling and classification tasks (one dataset for each task). The combination of Hierarchical PLM and TNN⊕PLM (patterns 2 and 3) outperforms pure models (patterns 1 and 4) on SST5. However, pure PLM achieves the best performance on the CoNLL 2003 NER dataset. These results demonstrate that complex models are not always better than simple models, and a flexible framework is necessary for providing various model candidates.

5.3 Results by Using Handcraft Features

To demonstrate the effectiveness of encoding handcraft features in domain application, Table 6 shows

the comparison results on two medical NER tasks, the NCBI-disease (Doğan et al., 2014) for English and Yidu-S4K (CCKS, 2019) for Chinese. Experiments on NCBI-disease apply two types of features, capitalization and English medical lexicon from the Chinese-English mapping medical glossary[4]. Experiments on Yidu-s4k dataset employ two medical lexicons as handcrafted features: the medical glossary of THUOCL (Han et al., 2016) (THU Open Chinese Lexicon) and the same medical glossary sourced from the web[4] while in Chinese version. Results show that handcraft features can improve the model performance in the medical domain.

5.4 Comparison with Transformers

The aforementioned results show that we can achieve the reported values across various tasks by using YATO. We further use tasks from GLUE benchmark (Wang et al., 2018) and compare with the results by using Huggingface Transform-
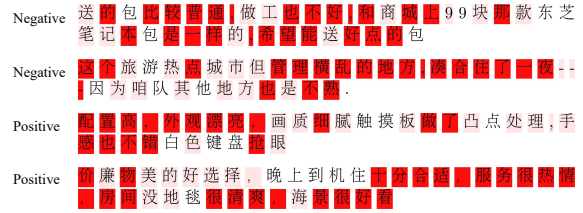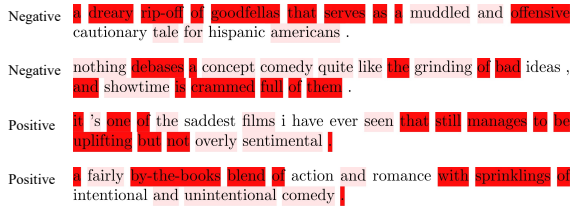
---

[4] http://medtop10.com

Figure 2: Visualization of attention weights. Different degrees of background color reflect the distributions of attention in words or characters.

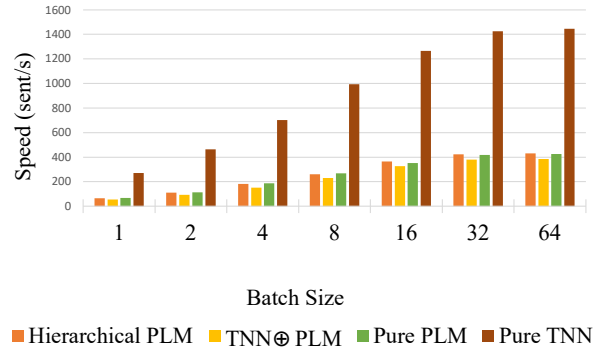| Tasks (Metric)	Transformers	YATO
MRPC (Accuracy)	84.07	**85.78**
QQP (Accuracy)	90.71	**90.81**
MNLI (Matched Accuracy)	83.91	**84.21**
QNLI (Accuracy)	**90.66**	90.28
SST2 (Accuracy)	92.32	**93.00**
RTE (Accuracy)	**65.70**	63.90
CoLA (Correlation Coefficient)	56.53	**58.00**
Average	80.55	**80.85**

Table 7: Results of fine-tuning BERT-base-uncased model on GLUE benchmark.

Figure 3: Decoding speed for the four patterns on different batch sizes. Tested on NVIDIA RTX 2080Ti GPU.

ers (Wolf et al., 2020), which is one of the most popular libraries. Table 7 shows the results by using BERT-base-uncased model, the values of Huggingface Transformers are sourced from the corresponding github page[5]. YATO achieves comparable and overall better performance than that of Huggingface Transformers by using default settings.

5.5 Visualization of Attention Map

Beyond performance, YATO provides a visualization tool for taking the list of words and the corresponding weights as input to generate Latex code for visualizing the attention-based result. Figure 2 provides visualization examples of attention on sentiment prediction tasks. Words or characters with sentiment polarities can be automatically extracted and highlighted using our YATO module. As shown in this table, words that have a high impact on the sentiment are highlighted. This visualization module can improve the interpretability of deep learning models in our toolkit.

5.6 Efficiency Analysis

YATO is implemented using a fully batch computing approach, making it quite efficient in both model training and decoding. With the help of GPU and large batches, models built on YATO can be decoded efficiently. Figure 3 shows the

decoding speed of four patterns at different batch sizes. The Pure TNN model has the best inference speed (1400 sentences/s) with a batch size of 64. The decoding speed of Transformer-based models decreases to around 400 sentences/s, showing the trade-off between efficiency and performance. Overall, without using external optimization techniques, YATO has a competitive decoding speed.

6 Conclusion

YATO is an open-source toolkit for text analysis that supports various combinations of state-of-the-art deep learning models and user-customed features, with high flexibility and minimum coding effort. YATO is maintained by core developers from YLab (https://ylab.top/). It aims to help AI researchers build state-of-the-art NLP models and assist non-AI researchers in conducting cross-disciplinary research with advanced NLP techniques. Given the success of its predecessor, NCRF++, we believe that YATO will greatly promote the applications of NLP in various cross-disciplinary fields and reduce disparities of AI application in these areas. In the future, we plan to integrate advanced LLMs and customize modules that support modeling time series, multimodal features, and specific features for various domains.

---

[5] https://github.com/huggingface/transformers/tree/main/examples/pytorch/text-classification

136

Limitations

Our proposed text analysis toolkit mainly focuses on discriminative style tasks, where most of them are treated as token-level or sentence-level classification tasks. Recent studies show that the generative style language models such as GPT (Radford et al., 2018), BART (Lewis et al., 2019), and T5 (Raffel et al., 2020) can also show promising zero-shot and few-shot results by adding user-defined prompts or instructions as external inputs, we leave this as our future work.

References

Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019. Flair: An easy-to-use framework for state-of-the-art nlp. In *Proc. of NAACL, (Demonstrations)*.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: Pretrained language model for scientific text. In *EMNLP*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

CCKS. 2019. China conference on knowledge graph and semantic computing 2019 named entity recognition of chinese electronic medical record. https://www.sigkg.cn/ccks2019/en/.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *Proc. of ICLR*.

PaddleNLP Contributors. 2021. Paddlenlp: An easy-to-use and high performance nlp library. https://github.com/PaddlePaddle/PaddleNLP.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, and Ziqing Yang. 2021. Pre-training with whole word masking for Chinese BERT. *IEEE/ACM Trans. Audio, Speech and Lang. Proc.*, 29:3504–3514.

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

Rezarta Islamaj Doğan, Robert Leaman, and Zhiyong Lu. 2014. Ncbi disease corpus: a resource for disease name recognition and concept normalization. *Journal of biomedical informatics*, 47:1–10.

Nadir Durrani, Hassan Sajjad, Fahim Dalvi, and Yonatan Belinkov. 2020. Analyzing individual neurons in pre-trained language models. In *Proc. of EMNLP*.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2017. Allennlp: A deep semantic natural language processing platform.

Tao Gui, Jiacheng Ye, Qi Zhang, Zhengyan Li, Zichu Fei, Yeyun Gong, and Xuanjing Huang. 2020. Uncertainty-aware label refinement for sequence labeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2316–2326, Online. Association for Computational Linguistics.

Shiyi Han, Yuhui Zhang, Yunshan Ma, Cunchao Tu, Zhipeng Guo, Zhiyuan Liu, and Maosong Sun. 2016. Thuocl: Tsinghua open chinese lexicon. *Tsinghua University*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Julia Hockenmaier and Mark Steedman. 2007. CCGbank: A Corpus of CCG Derivations and Dependency Structures Extracted from the Penn Treebank. *Computational Linguistics*, 33(3):355–396.

Matthew Honnibal and Ines Montani. 2017. spacy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing.

Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. OntoNotes: The 90% solution. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 57–60, New York City, USA. Association for Computational Linguistics.

Kyle P Johnson, Patrick J Burns, John Stewart, Todd Cook, Clément Besnier, and William JB Mattingly. 2021. The classical language toolkit: An nlp framework for pre-modern languages. In *Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing: System demonstrations*, pages 20–29.

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

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California. Association for Computational Linguistics.

Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. 1989. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Gina-Anne Levow. 2006. The third international Chinese language processing bakeoff: Word segmentation and named entity recognition. In *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*, pages 108–117, Sydney, Australia. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Linyang Li, Yong Dai, Duyu Tang, Zhangyin Feng, Cong Zhou, Xipeng Qiu, Zenglin Xu, and Shuming Shi. 2022. Markbert: Marking word boundaries improves chinese bert. *arXiv preprint arXiv:2203.06378*.

Chen Liang, Yue Yu, Haoming Jiang, Siawpeng Er, Ruijia Wang, Tuo Zhao, and Chao Zhang. 2020. Bond: Bert-assisted open-domain named entity recognition with distant supervision. In *Proc. of SIGKDD*.

Wei Liu, Xiyan Fu, Yue Zhang, and Wenming Xiao. 2021. Lexicon enhanced Chinese sequence labeling using BERT adapter. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5847–5858, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv e-prints*, page arXiv:1907.11692.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074, Berlin, Germany. Association for Computational Linguistics.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proc. of ACL: System Demonstrations*.

Manish Munikar, Sushil Shakya, and Aakash Shrestha. 2019. Fine-grained sentiment classification using bert. In *2019 Artificial Intelligence for Transforming Business and Society (AITB)*, volume 1, pages 1–5. IEEE.

OpenAI. 2022. Chatgpt: https://chat.openai.com/.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proc. of NAACL-HLT: Demonstrations*.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Artem Shelmanov, Dmitri Puzyrev, Lyubov Kupriyanova, Denis Belyakov, Daniil Larionov, Nikita Khromov, Olga Kozlova, Ekaterina Artemova, Dmitry Dylov, and Alexander Panchenko. 2021. Active learning for sequence tagging with deep pre-trained models and bayesian uncertainty estimates. pages 1698–1712.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proc. of EMNLP*.

Tomasz Stanislawek, Anna Wróblewska, Alicja Wójcicka, Daniel Ziembicki, and Przemyslaw Biecek. 2019. Named entity recognition - is there a glass ceiling? In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 624–633, Hong Kong, China. Association for Computational Linguistics.

Zijun Sun, Chun Fan, Qinghong Han, Xiaofei Sun, Yuxian Meng, Fei Wu, and Jiwei Li. 2020. Self-explaining structures improve nlp models. *arXiv preprint arXiv:2012.01786*.

138

Songbo Tan and Jin Zhang. 2008. An empirical study of sentiment analysis for chinese documents. *Expert Systems with Applications*, 34(4):2622–2629.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proc. of NAACL*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023a. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Xingchen Wan, Jie Yang, Slavi Marinov, Jan-Peter Calliess, Stefan Zohren, and Xiaowen Dong. 2021. Sentiment correlation in financial news networks and associated market movements. *Scientific reports*, 11(1):3062.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Chengyu Wang, Minghui Qiu, Taolin Zhang, Tingting Liu, Lei Li, Jianing Wang, Ming Wang, Jun Huang, and Wei Lin. 2022a. Easynlp: A comprehensive and easy-to-use toolkit for natural language processing.

Shuhe Wang, Xiaoya Li, Yuxian Meng, Tianwei Zhang, Rongbin Ouyang, Jiwei Li, and Guoyin Wang. 2022b. *k*nn-ner: Named entity recognition with nearest neighbor search. *arXiv preprint arXiv:2203.17103*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proc. of EMNLP: System Demonstrations*.

Mengzhou Xia, Mikel Artetxe, Jingfei Du, Danqi Chen, and Ves Stoyanov. 2022. Prompting electra: Few-shot learning with discriminative pre-trained models. *arXiv preprint arXiv:2205.15223*.

Ji Xin, Raphael Tang, Jaejun Lee, Yaoliang Yu, and Jimmy Lin. 2020. DeeBERT: Dynamic early exiting for accelerating BERT inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2246–2251, Online. Association for Computational Linguistics.

Canwen Xu, Yichong Xu, Shuohang Wang, Yang Liu, Chenguang Zhu, and Julian McAuley. 2023. Small models are valuable plug-ins for large language models. *arXiv preprint arXiv:2305.08848*.

Jie Yang, Shuailong Liang, and Yue Zhang. 2018. Design challenges and misconceptions in neural sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3879–3889, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Jie Yang, Liqin Wang, Neelam A Phadke, Paige G Wickner, Christian M Mancini, Kimberly G Blumenthal, and Li Zhou. 2020. Development and validation of a deep learning model for detection of allergic reactions using safety event reports across hospitals. *JAMA Network Open*, 3(11):e2022836–e2022836.

Jie Yang and Yue Zhang. 2018. NCRF++: An open-source neural sequence labeling toolkit. In *Proc. of ACL, System Demonstrations*.

Jie Yang, Yue Zhang, and Fei Dong. 2017. Neural reranking for named entity recognition. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 784–792.

Jan Zacharias, Michael Barz, and Daniel Sonntag. 2018. A survey on deep learning toolkits and libraries for intelligent user interfaces. *arXiv preprint arXiv:1803.04818*.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. *Advances in NeurIPS*.

Yue Zhang, Yile Wang, and Jie Yang. 2020. Lattice lstm for chinese sentence representation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:1506–1519.