DeepKE: A Deep Learning Based Knowledge Extraction Toolkit for Knowledge Base Population

Ningyu Zhang¹, Xin Xu¹, Liankuan Tao¹, Haiyang Yu¹, Hongbin Ye¹, Shuofei Qiao¹, Xin Xie¹, Xiang Chen¹, Zhoubo Li¹, Lei Li¹, Xiaozhuan Liang¹, Yunzhi Yao¹, Shumin Deng¹, Peng Wang¹, Wen Zhang¹, Zhenru Zhang², Chuanqi Tan², Qiang Chen², Feiyu Xiong², Fei Huang², Guozhou Zheng², Huajun Chen¹ ∗

¹ Zhejiang University & AZFT Joint Lab for Knowledge Engine
² Alibaba Group

http://deepke.zjukg.cn/

Abstract

We present an open-source and extensible knowledge extraction toolkit DeepKE, supporting complicated low-resource, document-level and multimodal scenarios in knowledge base population. DeepKE implements various information extraction tasks, including named entity recognition, relation extraction and attribute extraction. With a unified framework, DeepKE allows developers and researchers to customize datasets and models to extract information from unstructured data according to their requirements. Specifically, DeepKE not only provides various functional modules and model implementation for different tasks and scenarios but also organizes all components by consistent frameworks to maintain sufficient modularity and extensibility. We release the source code at GitHub¹ with Google Colab tutorials and comprehensive documents² for beginners. Besides, we present an online system³ for real-time extraction of various tasks, and a demo video⁴.

1 Introduction

As Information Extraction (IE) techniques develop fast, many large-scale Knowledge Bases (KBs) have been constructed. Those KBs can provide back-end support for knowledge-intensive tasks in real-world applications, such as language understanding (Che et al., 2021), commonsense reasoning (Lin et al., 2019) and recommendation systems (Wang et al., 2018). However, most KBs are far from complete due to the emerging entities and relations in real-world applications. Therefore, Knowledge Base Population (KBP) (Ji and Grishman, 2011) has been proposed, which aims to extract knowledge from the text corpus to complete the missing elements in KBs. For this target, IE is an effective technology that can extract entities and relations from raw texts and link them to KBs (Yan et al., 2021; Sui et al., 2021).

To date, a few remarkable open-source and long-term maintained IE toolkits have been developed, such as Spacy (Vasiliev, 2020) for named entity recognition (NER), OpenNRE (Han et al., 2019) for relation extraction (RE), Stanford OpenIE (Martínez-Rodríguez et al., 2018) for open information extraction, RESIN for event extraction (Wen et al., 2021) and so on (Jin et al., 2021). However, there are still several non-trivial issues that hinder the applicability of real-world applications.

Firstly, there are various important IE tasks, but most existing toolkits only support one task. Secondly, although IE models trained with those tools can achieve promising results, their performance may degrade dramatically when there are only a few training instances or in other complex real-world scenarios, such as encountering document-level and multimodal instances. Therefore, it is necessary to build a knowledge extraction toolkit facilitating the knowledge base population that supports multiple tasks and complicated scenarios: low-resource, document-level and multimodal.

In this paper, we share with the community a new open-source knowledge extraction toolkit called DeepKE (MIT License), which supports knowledge extraction tasks (named entity recognition, relation extraction and attribute extraction) in the standard supervised setting and three complicated scenarios: low-resource, document-level and multimodal settings. To facilitate usage, we design a unified framework for data processing, model training and evaluation. Developers and researchers can quickly customize their datasets and models for
various tasks without knowing too many technical
details, writing tedious glue code, or conducting
hyper-parameter tuning. We will provide mainte-
nance to meet new requests, add new tasks, and fix
bugs in the future. We highlight our major contri-
butions as follows:

• We develop and release a knowledge base
population toolkit that supports low-resource,
document-level and multimodal information
extraction.

• We offer flexible usage of the toolkit with suf-
ficient modularity as well as automatic hyper-
parameter tuning; thus, developers and re-
searchers can implement customized models
for information extraction.

• We provide detailed documentation, Google
Colab tutorials, an online real-time extraction
system and long-term technical support.

2 Core Functions

DeepKE is designed for different knowledge extrac-
tion tasks, including named entity recognition, rela-
tion extraction and attribute extraction. As shown
in Figure 1, DeepKE supports diverse IE tasks in
standard single-sentence supervised, low-resource
few-shot, document-level and multimodal settings,
which makes it flexible to adapt to practical and
complicated application scenarios.

2.1 Named Entity Recognition

As an essential task of IE, named entity recognition
(NER) picks out the entity mentions and classifies
them into pre-defined semantic categories given
plain texts. For instance, given the sentence “It was
one o’clock when we left Lauriston Gardens, and
Sherlock Holmes led me meet Gregson from Scot-
land Yard.”, NER models will predict that “Lau-
riston Gardens” as a location, “Sherlock Holmes”
and “Gregson” as persons, and “Scotland Yard”
as an organization. To achieve supervised NER,
DeepKE adopts the pre-trained language model
(Devlin et al., 2019) to encode sentences and make
predictions. DeepKE also implements NER in the
few-shot setting (including in-domain and cross-
domain) (Chen et al., 2022a) and the multimodal
setting.

2.2 Relation Extraction

Relation Extraction (RE), a common task in IE
for knowledge base population, predicts semantic
relations between pairs of entities from unstruc-
tured texts (Wu et al., 2021). To allow users to
customize their models, we adopt various mod-
els to accomplish standard supervised RE, includ-
ing CNN (Zeng et al., 2015), RNN (Zhou et al.,
2016), Capsule (Zhang et al., 2018a), GCN (Zhang
et al., 2018c, 2019), Transformer (Vaswani et al.,
2017) and BERT (Devlin et al., 2019). Meanwhile,
DeepKE provides few-shot and document-level

support for RE. For low-resource RE, DeepKE re-\\ implements\textsuperscript{5} KnowPrompt (Chen et al., 2022b), a recent well-performed few-shot RE method based on prompt-tuning. Note that few-shot RE is significant for real-world applications, which enables users to extract relations with only a few labeled instances. For document-level RE, DeepKE re-\\ implements DocuNet (Zhang et al., 2021) to extract inter-sentence relational triples within one document. Document-level RE is a challenging task that requires integrating information within and across multiple sentences of a document (Nan et al., 2020). RE is also implemented in the multi-modal setting described in Section 4.4.

2.3 Attribute Extraction
Attribute extraction (AE) plays an indispensable role in the knowledge base population. Given a sentence, entities and queried attribute mentions, AE will infer the corresponding attribute type. For instance, given a sentence “诸葛亮，字孔明，三国时期杰出的军事家、文学家、发明家。” (Liang Zhuge, whose courtesy name was Kong-ming, was an extraordinary strategist, litterateur and inventor in the Three Kingdoms period.), an entity “诸葛亮” (Liang Zhuge), and an attribute mention “三国时期” (Three Kingdoms period), DeepKE can predict the corresponding attribute type “朝代” (Dynasty). DeepKE adopts various models for AE (Table 1).

3 Toolkit Design and Implementation
We introduce the design principle of DeepKE as follows: 1) \textbf{Unified Framework}: DeepKE utilizes the same framework for various task objectives with respect to Data, Model and Core components; 2) \textbf{Flexible Usage}: DeepKE offers convenient training and evaluation with auto-hyperparameter tuning and the docker for operational efficiency; 3) \textbf{Off-the-shelf Models}: DeepKE provides pre-trained models (Chinese models with pre-defined schemas) for information extraction. We will introduce details of components in DeepKE and the unified framework in the following sections.

3.1 Data Module
The data module is designed for preprocessing and loading input data. The tokenizer in DeepKE implements tokenization for both English and Chinese

\textsuperscript{5}The code is re-organized in a unified format for flexible usage in DeepKE.
3.4 Framework Module

The framework module integrates three aforementioned components and different scenarios. It supports various functions, including data processing, model construction and model implementation. Meanwhile, developers and researchers can customize all hyper-parameters by modifying configuration files formatted as “*.yaml”, from which we apply Hydra\(^6\) to obtain users’ configuration. We also offer an off-the-shelf automatic hyperparameter tuning component. In DeepKE, we have implemented frameworks for all application functions mentioned in Section 2. For other future potential application functions, we have reserved interfaces for their implementation.

4 Toolkit Usage

4.1 Single-sentence Supervised Setting

All tasks, including NER, RE and AE, can be implemented in the standard single-sentence supervised setting by DeepKE. Every instance in datasets only contains one sentence. The datasets of these tasks are all annotated with specific information, such as entity mentions, entity categories, entity offsets, relation types and attributes.

4.2 Low-resource Setting

In real-world scenarios, labeled data may not be sufficient for deep learning models to make predictions for satisfying users’ specific demands. Therefore, DeepKE provides low-resource few-shot support for NER and RE, which is exceedingly distinctive. DeepKE offers a generative framework with prompt-guided attention to achieve in-domain and cross-domain NER. Meanwhile, DeepKE implements knowledge-informed prompt-tuning with synergistic optimization for few-shot relation extraction.

4.3 Document-Level Setting

Relations between two entities not only emerge in one sentence but appear in different sentences within the whole document. Compared to other IE toolkits, DeepKE can extract inter-sentence relations from documents, which predicts an entity-level relation matrix to capture local and global information.

4.4 Multimodal Setting

Multimodal knowledge extraction is supported in DeepKE. Intuitively, rich image signals related to texts are able to enhance context knowledge and help extract knowledge from complicated scenarios. DeepKE provides a Transformer-based multimodal entity and relation extraction method named IFAformer with prefix-based attention for multimodal NER and RE. Specifically, IFAformer simultaneously concatenates the textual and visual features in keys and values of the multi-head attention at each transformer layer, which can implicitly align multimodal features between texts and objects in text-related images\(^7\).

4.5 Online System & cnSchema-based Off-the-shelf Models

Besides this toolkit, we release an online system in http://deepke.zjukg.cn. As shown in Figure 3, we train our models in different scenarios with multilingual support (English and Chinese) and deploy them for online access. The system can be directly applied to recognize named entities, extract relations, classify attributes from plain texts, and visualizes extracted relational triples as knowledge graphs. The models are trained with the pre-defined schema (The system cannot extract knowledge out of the schema scope.) and offer flexible usage for users to obtain their customized models with their own schemas. Furthermore, DeepKE provides off-the-shelf extraction models with Chinese pre-trained language models (Cui et al., 2021b) based cnSchema\(^8\) supporting 28 entity types and 50 relation categories.

\(^6\)https://hydra.cc/


\(^8\)http://cnschema.openkg.cn/
Table 1: F1 Score (%) of the single-sentence, document-level and multimodal scenarios. * means these baselines are from other papers.

Table 2: F1 scores of in-domain low-resource NER on CoNLL-2003. * indicates low-resource entity types (100-shot).

Table 3: F1 scores of cross-domain few-shot NER (20-shot).

5 Experiment and Evaluation

5.1 Single-sentence Supervised Setting

The performance of the standard single-sentence supervised setting is reported in Table 1.

Named Entity Recognition We conduct NER experiments on two datasets: CoNLL-2003 (Sang and Meulder, 2003) for English and People’s Daily\(^9\) for Chinese. The English part of CoNLL-2003 contains four types of entities: persons (PER), locations (LOC), organizations (ORG) and miscellaneous (MISC). People’s Daily dataset is a Chinese dataset containing 45,518 entities classified into three categories PER, LOC and ORG. It is observed that DeepKE yields comparable performance with various encoders for these datasets. Meanwhile, DeepKE supports any English and Chinese NER datasets with BIO tags.

Relation Extraction We conduct RE experiments on the Chinese DuIE dataset\(^10\) with 10 relation categories. Each sample contains one sentence, one head entity, one tail entity in the sentence, their offsets, and the relation between them. We utilize six different neural networks in DeepKE for evaluation. Users can select models before training by changing only one hyper-parameter\(^11\). We report the performance of all models in Table 1.

Attribute Extraction The Chinese dataset for AE is from an online resource\(^12\). In each sample, one entity is annotated with its attribute type, value, and offset. Attributes in the dataset are classified into 6 categories. The training set contains 13,815 samples. The validation set contains 3,131 samples, and the test set includes 5,921 samples. Like RE, we leverage six neural models to extract attributes from the given sentence to evaluate DeepKE.

5.2 Low-resource Setting

We report the performance of the low-resource setting (NER and RE) in Table 2, 3, and 4.

Named Entity Recognition We conduct experiments on two datasets: CoNLL-2003 (Sang and Meulder, 2003) for English and People’s Daily\(^9\) for Chinese. The English part of CoNLL-2003 contains four types of entities: persons (PER), locations (LOC), organizations (ORG) and miscellaneous (MISC). People’s Daily dataset is a Chinese dataset containing 45,518 entities classified into three categories PER, LOC and ORG. It is observed that DeepKE yields comparable performance with various encoders for these datasets. Meanwhile, DeepKE supports any English and Chinese NER datasets with BIO tags.

Relation Extraction We conduct RE experiments on the Chinese DuIE dataset\(^10\) with 10 relation categories. Each sample contains one sentence, one head entity, one tail entity in the sentence, their offsets, and the relation between them. We utilize six different neural networks in DeepKE for evaluation. Users can select models before training by changing only one hyper-parameter\(^11\). We report the performance of all models in Table 1.

Attribute Extraction The Chinese dataset for AE is from an online resource\(^12\). In each sample, one entity is annotated with its attribute type, value, and offset. Attributes in the dataset are classified into 6 categories. The training set contains 13,815 samples. The validation set contains 3,131 samples, and the test set includes 5,921 samples. Like RE, we leverage six neural models to extract attributes from the given sentence to evaluate DeepKE.

5.2 Low-resource Setting

We report the performance of the low-resource setting (NER and RE) in Table 2, 3, and 4.

Named Entity Recognition We conduct experiments on both in-domain and cross-domain few-shot settings with LightNER (Chen et al., 2022a). Following Cui et al. (2021a), for the in-domain few-shot scenario, we reduce the number of training samples for certain entity categories by downsampling one dataset.

Attribute Extraction The Chinese dataset for AE is from an online resource\(^12\). In each sample, one entity is annotated with its attribute type, value, and offset. Attributes in the dataset are classified into 6 categories. The training set contains 13,815 samples. The validation set contains 3,131 samples, and the test set includes 5,921 samples. Like RE, we leverage six neural models to extract attributes from the given sentence to evaluate DeepKE.

5.2 Low-resource Setting

We report the performance of the low-resource setting (NER and RE) in Table 2, 3, and 4.
2003, we choose 100 “LOC” and 100 “MISC” as the low-resource entities and 2,496 “PER” and 3,763 “ORG” as the rich-resource entities. We leverage DeepKE to carry out the few-shot experiments and adopt BERT and BART with label-specific classifier layers as strong baselines denoted as LC-BERT and LC-BART. We also use template-based BART (Template.) (Cui et al., 2021a) as the competitive few-shot baseline. From Table 2, DeepKE outperforms other methods for both rich- and low-resource entity types, which illustrates that DeepKE has an outstanding performance on in-domain few-shot NER. In the cross-domain setting where the target entity categories and textual style are different from the source domain with limited labeled data available for training, we adopt the CoNLL-2003 dataset as an ordinary domain, and MIT Movie Review (Liu et al., 2013), MIT Restaurant Review (Liu et al., 2013) and Airline Travel Information Systems (ATIS) (Hakkani-Tür et al., 2016) datasets as target domains. The few-shot NER model in DeepKE is trained on CoNLL-2003 and fine-tuned on 20-shot target domain datasets (randomly sampled per entity category). We employ prototype-based Neigh.Tag. (Wiseman and Stratos, 2019), Example. (example-based NER) (Ziyadi et al., 2020), MP-NSP (Multi-prototype+NSP) (Huang et al., 2020), LC-BERT, LC-BART and Template. as competitive baselines. From Table 3, we notice that DeepKE achieves the most excellent few-shot performance.

### 5.3 Document-level Setting

DeepKE can extract intra- and inter-sentence relations among multiple entities within one document. We leverage a large-scale document-level RE dataset, DocRED (Ye et al., 2020), containing 3,053/1,000/1,000 instances for training, validation and testing, respectively. We use cased BERT-base and RoBERTa-large (Liu et al., 2019) as encoders. Compared with BERT-based and RoBERTa-based models, including Coref (Ye et al., 2020), and AT-LOP (Zhou et al., 2021), DeepKE appears the better or comparable performance than baselines as shown in Table 1.

### 5.4 Multimodal Setting

We report the performance of NER and RE in the multimodal scenario in Table 1.

### Named Entity Recognition

Multimodal NER experiments are conducted on Twitter-2017 (Lu et al., 2018) including texts and images from Twitter (2016-2017). The baselines for comparison are AdapCoAtt-BERT-CRF (Zhang et al., 2018b), ViLBERT (Lu et al., 2019) and UMT (Yu et al., 2020). We notice DeepKE can obtain a performance improvement compared with baselines.

### Relation Extraction

We use MNRE (Zheng et al., 2021b), a multimodal RE dataset containing sentences and images containing 23 relation categories. Previous SOTA models including BERT+SG (Zheng et al., 2021a), BERT+SG+Att (BERT+SG with attention calculating semantic similarity between textual and visual graphs) and MEGA (Zheng et al., 2021a), are leveraged for comparison. We further observe that DeepKE yields better performance than baselines.

### 6 Conclusion

In practical application, the knowledge base population struggles with low-resource, document-level and multimodal scenarios. To this end, we propose DeepKE, an open-source and extensible knowledge extraction toolkit. We conduct extensive experiments that demonstrate the models implemented by DeepKE can achieve comparable performance compared to some state-of-the-art methods. Besides, we provide an online system supporting real-time extraction (with the pre-defined schemas) without training. We will offer long-term maintenance to fix bugs, solve issues, add documents (tutorials) and meet new requests.
Broader Impact Statement

As noted in Manning (2022), linguistics and knowledge-based artificial intelligence were rapidly developing, and knowledge (explicit or implicit) as potential dark matter for language understanding still faces obstacles to acquisition and representation. To this end, IE technologies that aim to extract knowledge from unstructured data can serve as valuable tools to not only govern domain resources (e.g., medical, business) but also benefit deep language understanding and reasoning ability. Note that the proposed toolkit, DeepKE, can offer flexible usage in widespread IE scenarios with pre-trained off-the-shelf models. We hope to deliver the benefits of the proposed DeepKE to the natural language processing community.

Acknowledgments

We want to express gratitude to the anonymous reviewers for their kind comments. This work was supported by National Natural Science Foundation of China (No.62206246, 91846204 and U19B2027), Zhejiang Provincial Natural Science Foundation of China (No. LGG22F030011), Ningbo Natural Science Foundation (2021J190), and Yongjiang Talent Introduction Programme (2021A-156-G).

References


Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. 2010. Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of named entities. In Proceedings of SemEval@ACL Workshop.


Haoyang Wen, Ying Lin, Tuan Lai, Xiaoman Pan, Sha Li, Xudong Lin, Bin Zhou, Manling Li, Haoyu Wang, Hongming Zhang, Xiaodong Yu, Alexander Dong, Zhenhailong Wang, Yi Fung, Piyush Mishra, Qing Lyu, Didac Sureda, Brian Chen, Susan Windisch Brown, Martha Palmer, Chris Callison-Burch, Carl Vondrick, Jiawei Han, Dan Roth, Shih-Fu Chang, and Heng Ji. 2021. RESIN: A dockerized schema-guided cross-document cross-lingual cross-media information extraction and event tracking system. In Proceedings of NAACL-HLT.


A Toolkit Usage Details

In this section, we introduce how to use DeepKE exhaustively.

A.1 Build a Model From Scratch

Prepare the Runtime Environment  Users can clone the source code from the DeepKE GitHub repository and create a runtime environment. There are two convenient methods to create the environment. Users can choose to either leverage Anaconda or run the docker file provided in the repository. Besides, all dependencies can be installed by running pip install deepke directly. If developers would like to modify the source code of DeepKE, the following commands should be executed: running python setup.py install, modifying code and then running python setup.py develop. Users can also use corresponding datasets (e.g., default or customized datasets) to obtain specific information extraction models. All datasets need to be downloaded or uploaded in the folder named data.

<table>
<thead>
<tr>
<th>Word</th>
<th>Named Entity Tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>U.N.</td>
<td>B-ORG</td>
</tr>
<tr>
<td>official</td>
<td>O</td>
</tr>
<tr>
<td>Ekeus</td>
<td>B-PER</td>
</tr>
<tr>
<td>heads</td>
<td>O</td>
</tr>
<tr>
<td>for</td>
<td>O</td>
</tr>
<tr>
<td>Baghdad</td>
<td>B-LOC</td>
</tr>
<tr>
<td>.</td>
<td>O</td>
</tr>
<tr>
<td>Israel</td>
<td>B-LOC</td>
</tr>
<tr>
<td>Arafat</td>
<td>B-PER</td>
</tr>
<tr>
<td>s</td>
<td>O</td>
</tr>
<tr>
<td>flight</td>
<td>O</td>
</tr>
<tr>
<td>to</td>
<td>O</td>
</tr>
<tr>
<td>West</td>
<td>B-LOC</td>
</tr>
<tr>
<td>Bank</td>
<td>I-LOC</td>
</tr>
</tbody>
</table>

Table 5: Examples of the input format for NER.
Named Entity Recognition As shown in Table 5, the input data files with BIO tags for standard and few-shot NER contain two columns separated by a single space. Each word has been put on a separate line, and there is an empty line after each sentence. The two columns represent two items: the word and the named entity tag. Before training, all datasets with the formats mentioned above should be fed into NER models through the data loader. Developers can implement training and evaluation by running example code `run.py` to obtain a fine-tuned NER model, which will be used in the prediction period. For inference, users can run `predict.py` with a single sentence and obtain the output recognized entity mentions and types.

Relation Extraction The training input with the CSV format of standard RE is shown in Table 6. There are five components in the format, including a sentence, a relation, the head and tail entity of the relation, the head entity offset and the tail entity offset. For few-shot RE, one input sample, as shown in Figure 4, contains sentence tokens including words and punctuation, the head entity and tail entities with their mention names and position spans, and the relation between them. For example, an input of few-shot relation extraction instance is the format of `{"token": ["the", "dolphin", "uses", "its", "flukes", "for", "swimming", "and", "its", "flippers", "for", "steering", ",", ",", "h": ["dolphin", "pos": [1, 2], "t": ["flukes", "pos": [4, 5]], "relation": "Component-Whole(e2,e1)"] (h: head entity, t: tail entity, pos: position). The document-level RE training format is shown in Figure 5. One sample consists of a sample title, sentences separated into words and punctuation in one document, an entity set (including entity mentions, sentence IDs the entities are located in, entity position spans and entity types in the document) and a relation label set (including the head and tail entity IDs, relations and evidence sentence IDs). After training and validation, users can run the `predict` function given an input sentence with head and tail entity to obtain corresponding relations.

Attribution Extraction The input CSV files formatted as Table 7 should be given to train the attribution extraction (AE) model. One sample contains six components: a raw sentence, a queried attribute type, an entity and its offset, the entity’s corresponding attribute value and the attribute mention offset. After training, users will obtain a fine-tuned AE model, which can be leveraged to infer attributes. Given a sentence with an entity and a candidate attribute mention, the AE model will predict the attribute type with confidence. Note that all operations mentioned above are guided in the example code file `run.py` and `predict.py`.

A.2 Auto-Hyperparameter Tuning
To achieve automatic hyper-parameters fine-tuning, DeepKE adopts Weight & Biases, a machine learning toolkit for developers to reduce label-intensive hyper-parameter tuning. With DeepKE, users can visualize results and tune hyper-parameters automatically. Note that all metrics and hyper-parameter configurations can be customized to meet diverse settings for different tasks. For more details on automatic hyper-parameter tuning.
Table 8: Language supported in DeepKE.

<table>
<thead>
<tr>
<th>Task</th>
<th>Scenario</th>
<th>Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>NER</td>
<td>Supervised</td>
<td>Chinese</td>
</tr>
<tr>
<td></td>
<td>Few-shot</td>
<td>English, Chinese</td>
</tr>
<tr>
<td></td>
<td>Multimodal</td>
<td>English</td>
</tr>
<tr>
<td>RE</td>
<td>Supervised</td>
<td>Chinese</td>
</tr>
<tr>
<td></td>
<td>Few-shot</td>
<td>English</td>
</tr>
<tr>
<td></td>
<td>Multimodal</td>
<td>English</td>
</tr>
<tr>
<td></td>
<td>Document</td>
<td>English</td>
</tr>
<tr>
<td>AE</td>
<td>Supervised</td>
<td>Chinese</td>
</tr>
</tbody>
</table>

please refer to the official document\textsuperscript{14}.

A.3 Language Support

The current version of DeepKE supports English and Chinese implementation for three IE tasks, as shown in Table 8.

A.4 Notebook Tutorials

We provide Google Colab tutorials\textsuperscript{15} and jupyter notebooks in the GitHub repository as an exemplary implementation of every task in different scenarios. These tutorials can be run directly, thus, leading developers and researchers to have a whole picture of DeepKE’s powerful functions.

\textsuperscript{14}https://docs.wandb.ai
\textsuperscript{15}https://colab.research.google.com/drive/1vS8YJhJltzw3hpJczPt2400Azcs3ZpRi?usp=sharing

B Contributions

Ningyu Zhang from Zhejiang University, AZFT Joint Lab for Knowledge Engine, conducted the whole development of DeepKE and wrote the paper.

Xin Xu from Zhejiang University, AZFT Joint Lab for Knowledge Engine developed the standard NER and wrote the paper.

Liankuan Tao, Shuofei Qiao, Peng Wang, Haiyang Yu from Zhejiang University, AZFT Joint Lab for Knowledge Engine develop the standard RE and AE, the deepke python package, and documents and provides consistent maintenance.

Hongbin Ye from Zhejiang University, AZFT Joint Lab for Knowledge Engine developed the online system and constructed the online demo.

Xin Xie, Xiang Chen from Zhejiang University, AZFT Joint Lab for Knowledge Engine developed the few-shot relation extraction model KnowPrompt and the document-level relation extraction model DocuNet.

Zhoubo Li, Lei Li, Xiaozhuan Liang, Yunzhi Yao, Shumin Deng, Wen Zhang from Zhejiang University, AZFT Joint Lab for Knowledge Engine developed the Google Colab and proofread the paper.

Zhenru Zhang, Chuanqi Tan, Qiang Chen, Feiyu Xiong, Fei Huang from Alibaba Group, proofread the paper and advised the project.

Guozhou Zheng, Huajun Chen from Zhejiang University, AZFT Joint Lab for Knowledge Engine advised the project, suggested tasks, and led the research.