SOFTWARE

Transformers-sklearn: a toolkit for medical language understanding with transformer-based models

Feihong Yang†, Xuwen Wang†, Hetong Ma and Jiao Li*

From International Conference on Health Big Data and Artificial Intelligence 2020 Guangzhou, China. 29 October - 1 November 2020

Abstract

Background: Transformer is an attention-based architecture proven the state-of-the-art model in natural language processing (NLP). To reduce the difficulty of beginning to use transformer-based models in medical language understanding and expand the capability of the scikit-learn toolkit in deep learning, we proposed an easy to learn Python toolkit named transformers-sklearn. By wrapping the interfaces of transformers in only three functions (i.e., fit, score, and predict), transformers-sklearn combines the advantages of the transformers and scikit-learn toolkits.

Methods: In transformers-sklearn, three Python classes were implemented, namely, BERTologyClassifier for the classification task, BERTologyNERClassifier for the named entity recognition (NER) task, and BERTologyRegressor for the regression task. Each class contains three methods, i.e., fit for fine-tuning transformer-based models with the training dataset, score for evaluating the performance of the fine-tuned model, and predict for predicting the labels of the test dataset. transformers-sklearn is a user-friendly toolkit that (1) is customizable via a few parameters (e.g., model_name_or_path and model_type), (2) supports multilingual NLP tasks, and (3) requires less coding. The input data format is automatically generated by transformers-sklearn with the annotated corpus. Newcomers only need to prepare the dataset. The model framework and training methods are predefined in transformers-sklearn.

Results: We collected four open-source medical language datasets, including TrialClassification for Chinese medical trial text multi label classification, BC5CDR for English biomedical text name entity recognition, DiabetesNER for Chinese diabetes entity recognition and BIOSSES for English biomedical sentence similarity estimation. In the four medical NLP tasks, the average code size of our script is 45 lines/task, which is one-sixth the size of transformers’ script. The experimental results show that transformers-sklearn based on pretrained BERT models achieved macro F1 scores of 0.8225, 0.8703 and 0.6908, respectively, on the TrialClassification, BC5CDR and DiabetesNER tasks and a Pearson correlation of 0.8260 on the BIOSSES task, which is consistent with the results of transformers.

Conclusions: The proposed toolkit could help newcomers address medical language understanding tasks using the scikit-learn coding style easily. The code and tutorials of transformers-sklearn are available at https://doi.org/10.5281/

*Correspondence: li.jiao@imicams.ac.cn
†Feihong Yang and Xuwen Wang have contributed equally to this work
Institute of Medical Information and Library, Chinese Academy of Medical Sciences/Peking Union Medical College, 3rd Yabao Road, Beijing 100020, China

© The Author(s) 2021. Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article’s Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article’s Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/. The Creative Commons Public Domain Dedication waiver (http://creativecommons.org/publicdomain/zero/1.0/) applies to the data made available in this article, unless otherwise stated in a credit line to the data.
**Background**

Transformer is an attention-based architecture proposed by Vaswani et al. [1], which has been proved to be the state-of-the-art model by BERT [2] (i.e., Bidirectional Encoder Representations from Transformers), RoBERTa [3] (i.e., a Robustly Optimized BERT pre-training Approach), etc. With the development of natural language processing (NLP) technology, transformer-based models have emerged. To effectively utilize these models and evaluate their performance in downstream tasks, a Python library of transformer-based models, namely, transformers [4], has been developed by gathering state-of-the-art general purpose pre-trained models under a unified application program interface (API) together with an ecosystem of libraries. transformers has been reported to have been used in more than 200 research papers and included either as a dependency or with a wrapper in several popular NLP frameworks such as AllenNLP [5] and Flair [6].

scikit-learn [7], which is a Python module integrating a wide range of state-of-the-art machine learning algorithms for medium-scale supervised and unsupervised problems, is one of the most popular machine-learning toolkits. It is friendly to newcomers to apply it in machine learning tasks. Based on scikit-learn, Lemaître G et al. proposed imbalanced-learn [8] to provide various methods to cope with the imbalanced dataset problem frequently encountered in machine learning and pattern recognition. Szmyański P and Kajdanowicz T developed a Python library named scikit-multilearn [9] for performing multi label classification. Löning M et al. present sktime [10], which is a scikit-learn compatible the Python library with a unified interface for machine learning with time series. De Vazelhes W et al. implemented supervised and weakly supervised distance metric learning algorithms and wrapped them in a Python package named metric-learn [11]. These works made scikit-learn more powerful and efficient in specific domain tasks.

As known, the transformers toolkit is well designed and friendly to professional researchers and engineers. However, for newcomers who have no knowledge of transformers, it is still time-consuming to learn the background knowledge about transformers from scratch. scikit-learn is designed to make machine learning for easy use, but there is still a gap between machine learning and deep learning algorithms in scikit-learn.

To reduce the difficulty of getting started with transformer-based models and expand the capability of scikit-learn in deep learning, we combine the advantages of the transformers and scikit-learn toolkits and propose a Python toolkit named transformers-sklearn. The proposed toolkit aims to make transformer-based models convenient for beginners by wrapping the interfaces of transformers in only three APIs (i.e., fit, score, and predict). With transformers-sklearn, newcomers could use transformer-based models to address their NLP tasks, even though they had no previous knowledge of transformer. The users can pay more attention on the NLP task itself, with preparing the training dataset for fitting, the development dataset for scoring the model, and the test dataset for predicting.

The primary contributions of this paper are as follows. (1) We proposed transformers-sklearn, which makes transformer-based models for easy use and expands the capability of scikit-learn in deep learning methods. (2) To validate the performance of transformers-sklearn, experiments were conducted on four NLP tasks based on English and Chinese medical language datasets. We also compared transformers-sklearn with the widely used NLP toolkits such as transformers and UER [12]. (3) The code and tutorials of transformers-sklearn are available at https://doi.org/10.5281/zenodo.4453803.

**Methods**

In transformers-sklearn, there are three Python classes designed for classification, named entity recognition (NER), and regression tasks. Each class contains three methods, namely, fit, score, and predict.

**Python classes**

transformers-sklearn was implemented with three Python classes, which are BERTologyClassifier for the classification task, BERTologyNERClassifier for the named entity recognition (NER) task, and BERTologyRegressor for the regression task. BERTologyClassifier and BERTologyNERClassifier are subclasses of BaseEstimator and ClassifierMixin implemented by the scikit-learn toolkit. BERTologyRegressor is the subclass of BaseEstimator and RegressorMixin implemented by scikit-learn.

All classes could be customized by setting the values of multiple parameters. Among these parameters, model_type is used to specify which type of model initialization style should be used, and model_name_or_path is
used to specify which pre-trained model should be used. There are six model initialization types, namely, BERT, RoBERTa, XLNet [13], XLM [14], DistilBERT [15] and ALBERT [16]. All these models are implemented based on a transformer, but they differ in their data processing. More details about the parameters are shown in Table 1.

Class methods
The same as with the class methods of scikit-learn, three methods (i.e., fit, score, and predict) were implemented in each Python class of transformers-sklearn. The fit and score methods accept two parameters, which are \( X \) and \( y \). \( X \) is a container of sentences or documents, and \( y \) contains the corresponding labels. \( X \) and \( y \) could be one of the following Python data types: list, ndarray implemented by numpy [17], and DataFrame implemented by pandas [18]. The predict method only requires parameter \( X \).

The functions of the above class methods were as follows:

1. **Fit.** This method was used to fine-tune the customized pre-trained model following the configuration of the parameters in each class (i.e., BERTologyClassifier or BERTologyNERClassifier or BERTologyRegressor). In this method, the training set was automatically transformed to the specific format and then fed into the customized transformer-based model for fine-tuning.

2. **Score.** This method was used to evaluate the performance of the fine-tuned model. For example, in the classification task, this method would return the common evaluation indexes such as the precision, recall and F1-score for each type of label.

3. **Predict.** This method was used to predict the labels of a given dataset.

Traditionally, it is difficult for newcomers to address their NLP problems using the transformers-based methods. For instant, a user would like to apply the BERT model to address a binary classification task, thereafter, four steps were needed to fine-tune the pre-trained BERT model as follows:

1. **Data preparation.** The training set is transformed to a special format for the BERT model. The user needs to learn about the data processing of BERT.

2. **Model configuration.** The user customizes the model with fully understanding the architecture of BERT.

3. **Fine-tuning model.** The user determines epochs that are used to fine-tune the customized BERT.

4. **Saving fine-tuned model.** The user saves the fine-tuned model to the target path.

The four steps mentioned above increased the developmental difficulty for newcomers, and it is time-consuming for them to learn the necessary background knowledge. In our work transformers-sklearn, the four steps are implemented automatically in the fit method and transparent to users.

**Workflow**
As shown in Fig. 1, when facing an NLP task, the user first determines whether the transformer-based models could address the problem. If so, the user should choose one class from BERTologyClassifier, BERTologyNERClassifier and BERTologyRegressor, which could be customized by setting the parameters, depending on to which class the problem belongs. After customizing the chosen class, the user feeds the datasets into the fit method. Using the NER task as an example, the input data format is defined as Table 2. As shown the text field contains segmented texts to be labelled, and the label field contains the corresponding medical named entities obtained by manual annotations. Then, transformers-sklearn would conduct the fine-tuning process automatically. Finally, the user could evaluate the fine-tuned model using the score method or deploy the fine-tuned model in practice using the predict method. During the whole workflow,

| Table 1 | The common parameters of the Python classes in transformers-sklearn |
|---|---|
| **Name** | **Function** |
| model_type | Specifies which type of model initialization style should be used |
| model_name_or_path | Specifies which pre-trained model should be used |
| max_seq_length | Sets the max length of the sequence that could be accepted |
| per_gpu_train_batch_size | Sets the batch size per GPU |
| learning_rate | Sets the learning rate of the model |
| num_train_epochs | Sets the number of training epochs of the model |
| no_cuda | Sets whether the GPU is used for training or predicting |
it is possible for the user to dispense with understanding the internal mechanisms of the chosen transformer-based model.

Experiments
We conducted comparison experiments to validate the effectiveness of transformers-sklearn on multilingual medical NLP tasks. We selected three popular transformer-based model types from our package, i.e., BERT, RoBERTa, and ALBERT, and compared them with the original transformers and UER [12]. The pre-trained models of different model types can be downloaded automatically or manually from the community [19], as shown in Table 3. All experiments were conducted on four Tesla V100 16 GB GPUs with the initial number of training epochs set to 3, the learning rate set to 5e-5 and the other parameters set to their default values. The parameters such as the epochs and learning rate can be adjusted manually according to specific experiments.

Corpus
To assess the performance of transformers-sklearn on medical language understanding, we collected the following four English and Chinese medical datasets (TrialClassification, BC5CDR, DiabetesNER, and BIOSSES) from the NLP community as our experimental corpora. More details on the four datasets can be found in Table 4.

1. TrialClassification [20]. This dataset contains 38,341 Chinese clinical trial sentences and is labelled with 45 classes. It was developed for Chinese medical trial text multilabel classification.
2. BC5CDR [21]. This dataset is a collection of 1,500 PubMed titles and abstracts selected from the CTD-Pfizer corpus and was used in the BioCreative V chemical-disease relation task. It was developed for English biomedical text name entity recognition.
3. DiabetesNER [22]. The dataset contains more than 9,556 Chinese medical named entity identification samples. It was developed for Chinese diabetes entity recognition. We randomly selected 80% of the data for training and 20% of the data for testing.
4. BIOSSES [23]. This dataset is a corpus of 100 sentence pairs selected from the Biomedical Summarization Track Training Dataset in the biomi-

---

**Table 2** An example of the NER input data format in the BERTologyNERClassifier

| Input data field | Example |
|------------------|---------|
| Text             | [...["Naloxone","reverses","the","antihypertensive","effect","of","clonidine",""],...]] |
| Label            | [...["B-Chem","O","O","O","O","O","B-Chem","O"],...]] |

---

**Table 3** Pre-trained models and URLs

| Model                   | URL                                             |
|-------------------------|-------------------------------------------------|
| bert-base-chinese       | https://huggingface.co/bert-base-chinese       |
| bert-base-cased         | https://huggingface.co/bert-base-cased         |
| chinese-roberta-wwm-ext | https://huggingface.co/hfl/chinese-roberta-wwm-ext |
| roberta-base            | https://huggingface.co/roberta-base            |
| albert_chinese_base     | https://huggingface.co/voidful/albert_chinese_base |
| albert-base-v2          | https://huggingface.co/albert-base-v2          |
It was collected for English biomedical sentence similarity estimation. Here, we randomly selected 80% of the data for training and 20% of the data for testing.

Evaluation

Two types of evaluation indexes were used for scoring, which are the macro F1 and Pearson/Spearman correlation. For the macro F1, set \( n \) classes as \( C_1, C_2, \ldots, C_n \). The precision for each class was defined as \( P_i \), which equals the number of correct predictions \( C_i \) divided by the number of prediction \( C_i \). The recall for each class was defined as \( R_i \), which equals the number of correct predictions \( C_i \) divided by the number of predictions \( C_i \). Then, the macro F1 score of the tasks were calculated as follows:

\[
\text{Macro F1} = \left( \frac{1}{n} \right) \sum_{i=1}^{n} \frac{2 \times P_i \times R_i}{P_i + R_i}
\]

(1)

For the Pearson correlation, set \( y \) as the true value of given dataset and \( y_{\text{pred}} \) as the value predicted by the model. Then, the Pearson correlation was calculated as follows:

\[
\rho_{y,y_{\text{pred}}} = \frac{E(yy_{\text{pred}}) - E(y)E(y_{\text{pred}})}{\sqrt{E(y^2) - E^2(y)} \sqrt{E(y_{\text{pred}}^2) - E^2(y_{\text{pred}})}}
\]

(2)

Results

The performances of the BERT model implemented by transformers-sklearn, transformers and UER in the four medical NLP tasks are shown in Table 5. The transformers-sklearn toolkit achieved macro F1 scores of 0.8225, 0.8703 and 0.6908 in the TrialClassification, BC5CDR and DiabetesNER tasks, respectively, and a Pearson correlation of 0.8260 in the BIOSSES task, which are consistent with the results of transformers.

Tables 6 and 7 show the performances of the RoBERTa and ALBERT models, respectively. The RoBERTa model in transformers-sklearn achieved macro F1 scores of 0.8148, 0.8528, and 0.7068 in the TrialClassification, BC5CDR and DiabetesNER tasks, respectively, and a Pearson correlation of 0.3996 in the BIOSSES task. The ALBERT model in transformers-sklearn achieved macro F1 scores of 0.7142, 0.8422, and 0.6196 in the three respective tasks and a Pearson correlation of 0.1892 in the BIOSSES task.

As shown in Fig. 2, the entire code for BIOSSES implement is short and easy to use. The users could apply transformer-based models in the scikit-learn coding style with the help of our toolkit. In the four tasks, the average code load of our toolkit’s script is 45 lines/task, which is one-sixth the size of transformers’ script. In addition to the comparison of the number of lines of code, we also compared the running time of each model, as shown in Tables 5, 6, and 7, which indicated the high efficiency of transformers-sklearn.
### Discussion

#### Principal results

The proposed toolkit, transformers-sklearn, was proved to be easy to use for newcomers and could be used for transformer-based models as the scikit-learn coding style.

#### Limitations

Compared with transformers, the limitation of transformers-sklearn is its lack of flexibility. For example, within transformers-sklearn, it is impossible for users to extract any encoding or decoding layer of the transformer. In other words, users cannot determine which layer of transformer could act in the downstream tasks.

Furthermore, transformers-sklearn aims at making transformer-based models for easy use and expanding the capability of scikit-learn in deep learning methods. For advanced users, the transformers toolkit is better than our transformers-sklearn regarding flexibility.

#### Comparison to existing tools

Compared with prior toolkits, such as transformers and UER, transformers-sklearn is easy to get started using for newcomers with basic machine learning knowledge. The experimental results of the four medical NLP tasks showed that the BERT model in transformers-sklearn

---

### Table 6

| Name        | Score     | Second    | Lines of code | Pre-trained model          |
|-------------|-----------|-----------|---------------|---------------------------|
|             | Ours      | Transformers | Ours  | Transformers |                         |
| TrialClassification | 0.8148a  | 0.8231b   | 1206  | 1208 | 38  | 246  | chinese-roberta-wwm-ext |
| BC5CDR      | 0.8528a  | 0.8461a   | 460   | 504 | 41  | 309  | roberta-base             |
| DiabetesNER | 0.7068a  | 0.7184a   | 1445  | 1426| 63  | 309  | chinese-roberta-wwm-ext  |
| BIOSSES     | 0.3996b  | 0.3614b   | 36    | 17  | 41  | 246  | roberta-base             |

*a* The value of Macro F1, where the bolded one indicates the best performance  
*b* The value of Person correlation, where the bolded one indicates the best performance

### Table 7

| Name        | Score     | Second    | Lines of code | Pre-trained model          |
|-------------|-----------|-----------|---------------|---------------------------|
|             | Ours      | Transformers | Ours  | Transformers |                         |
| TrialClassification | 0.7142a  | 0.4504b   | 1062  | 1068 | 38  | 246  | albert_chinese_base      |
| BC5CDR      | 0.8422a  | 0.8523a   | 444   | 492 | 41  | 309  | albert-base-v2           |
| DiabetesNER | 0.6196a  | 0.6436a   | 1122  | 1253| 63  | 309  | albert_chinese_base      |
| BIOSSES     | 0.1892b  | 0.4394b   | 12    | 11  | 41  | 246  | albert-base-v2           |

*a* The value of Macro F1, where the bolded indicates the best performance  
*b* The value of Person correlation, where the bolded indicates the best performance

---

![Fig. 2](image)  

**Fig. 2** The code for BIOSSES within transformers-sklearn
obtained preferable performance while using much less code and comparable running time.

`transformers-sklearn` is based on `transformers`. We wrapped the powerful functions implemented by `transformers` and made them transparent to users. `transformers-sklearn` is also based on `scikit-learn`, which is popularly used in machine learning fields. Thus, the technique advantages of both `scikit-learn` and `transformers` were integrated in our toolkit.

**Conclusions**

In this paper, three Python classes including `BERTologyClassifier`, `BERTologyNERClassifier` and `BERTologyRegressor` and three methods of each class were developed in `transformers-sklearn`. To validate the effectiveness of `transformers-sklearn`, we applied the toolkit in four multilingual NLP tasks. The results showed that `transformers-sklearn` could effectively address the NLP problems in both Chinese and English if the pre-trained `transformer`-based model supported the language. The code and tutorials of `transformers-sklearn` are available at [https://doi.org/10.5281/zenodo.4453803](https://doi.org/10.5281/zenodo.4453803).

In future work, a keep-updating `transformers_sklearn` toolkit that combines flexibility and usability will be released, with supporting a wide range of medical language understanding tasks.

**Availability and requirements**

The datasets and software supporting the results of this article are available in the trueto/transformers-sklearn repository, [https://doi.org/10.5281/zenodo.4453803](https://doi.org/10.5281/zenodo.4453803).

**Authors’ contributions**

JL conducted the work. FY and JL designed the architecture of the proposed toolkit. FY implemented `transformers-sklearn`. FY, XW and JL analysed the results. FY, XW, HM and JL wrote and revised the manuscript. All authors read and approved the final manuscript.

**Funding**

This work has been supported by the Beijing Natural Science Foundation (Grant No. Z200016), CAS Innovation Fund for Medical Sciences (CFMS) (Grant No. 2018-I2M-AI-016), and the National Natural Science Foundation of China (Grant No. 61906214).

**Availability of data and materials**

The datasets and software supporting the results of this article are available in the trueto/transformers-sklearn repository, [https://doi.org/10.5281/zenodo.4453803](https://doi.org/10.5281/zenodo.4453803).

**Declarations**

**Ethics approval and consent to participate**

Not applicable.

**Consent for publication**

Not applicable.

**Competing interests**

The authors declare that they have no competing interests.

**Received** 21 February 2021 **Accepted** 1 March 2021

**Published** 30 July 2021

**Reference**

1. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser U, Polosukhin I. Attention is all you need. In: NIPS’17. Red Hook, NY, USA; 2017, p. 6000–6010.

2. Devlin J, Chang M, Lee K, Toutanova K. BERT: pre-training of deep bidirectional transformers for language understanding. In: NAACL-HLT;2019; 2019.

3. Liu Y, Ott M, Goyal N, et al. RoBERTa: a robustly optimized BERT pretraining approach. In: ArXiv 2019, abs/1907.11692.

4. Wolf T, Debut L, Sanh V, Chaumond J, Delangue C, Moi A, Cistac P, Rault T, Louf R, Funtowicz M et al. HuggingFace’s transformers: state-of-the-art natural language processing. ArXiv 2019, abs/1910.03771.

5. Gardner M, Grus J, Neumann M, Tafjord O, Peters ME, Schmitz M, Zettlemoyer L. AllenNLP: a deep semantic natural language processing platform. ArXiv 2018, abs/1803.07640.

6. Akbik A, Blythe D, Vollgraf R. Contextual string embeddings for sequence labeling. In: COLING2018:27th international conference on computational linguistics; 2018, p. 1638–1649.

7. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Müller A, Nothman J, Louppe G et al. Scikit-learn: machine learning in python. ArXiv 2012, abs/1201.0490.

8. Lermaitre G, Nogueira F, Aridas CK. Imbalanced-learn: a python toolbox to tackle the curse of imbalanced datasets in machine learning. ArXiv 2016, abs/1609.06570.

9. Szmaciński P, Kajdanowicz T. A scikit-based Python environment for performing multi-label classification. ArXiv 2017, abs/1702.01460.

10. Lönning M, Bagnall A, Ganesh S, Kazakov V, Lines J, Kiraly FJ. sktime: A Unified Interface for Machine Learning with Time Series. ArXiv 2019, abs/1909.07872.

11. de Vazelhes W, Carey CJ, Tang Y, Vauquier N, Bellet A. metric-learn: Metric Learning algorithms in python. ArXiv 2019, abs/1908.04710.

12. Zhao Z, Chen H, Zhang J, Zhao X, Liu T, Lu W, Chen X, Deng H, Ju Q, Du X. UER: An Open-source toolkit for pre-training models. In: Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP): system demonstrations; 2019; Hong Kong, China: Association for Computational Linguistics; 2018, p. 1638–1649.

**Abbreviation**

BERT: Bidirectional Encoder Representations from Transformers.

**Acknowledgements**

The authors would like to thank the open-source contributors for their early work on `transformers` and `scikit-learn`, so that `transformers-sklearn` can be designed and implemented.

**About this supplement**

This article has been published as part of BMC Medical Informatics and Decision Making Volume 21, Supplement 2 2021: Health Big Data and Artificial Intelligence. The full contents of the supplement are available at [https://bmcmedinformdecismak.biomedcentral.com/articles/supplements/volume-e-21-supplement-2](https://bmcmedinformdecismak.biomedcentral.com/articles/supplements/volume-e-21-supplement-2).
13. Yang Z, Dai Z, Yang Y, Carbonell J, Salakhutdinov R, Le QV. XLNet: generalized autoregressive pretraining for language understanding. ArXiv 2019, abs/1906.08237.
14. Lample G, Conneau A. Cross-lingual Language Model Pretraining. ArXiv 2019, abs/1901.0729.
15. Sanh V, Debut L, Chaumond J, Wolf T. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. ArXiv 2019, abs/1910.01108.
16. Lan Z, Chen M, Goodman S, Gimpel K, Sharma P, Soricut R. ALBERT: a Lite BERT for self-supervised learning of language representations. ArXiv 2019, abs/1901.11942.
17. NumPy. https://numpy.org/. Accessed 21 Aug 2020.
18. pandas: Python data analysis library. https://pandas.pydata.org/index.html. Accessed 21 Aug 2020.
19. Google Research. GitHub Repository. https://github.com/google-research/bert. Accessed 21 Aug 2020.
20. CHIP: Short text classification for clinical trial screening criteria. http://www.cips-chip.org.cn:8088/evaluation. Accessed 21 Aug 2020.
21. Wei C, Peng Y, Leaman R, Davis AP, Mattingly CJ, Li J, Wiegers TC, Lu Z. Overview of the BioCreative V chemical disease relation (CDR) task. In: Proceedings of the fifth biocreative challenge evaluation workshop. 2015: 154–166.
22. Cloud A. Alibaba Cloud Labeled Chinese Dataset for diabetes. https://tianchi.aliyun.com/dataset/dataDetail?dataId=22288. Accessed 21 Aug 2020.
23. Soğancıoğlu G, Öztürk H, Özgür A. BIOSSES: a semantic sentence similarity estimation system for the biomedical domain. Bioinformatics. 2017;33(14):49–58.

Publisher’s Note
Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.