# Article-enhanced Pre-trained Language Model for Legal Case Retrieval

First Author[1 [0000−1111−2222−3333]], Second Author[2,3 [1111−2222−3333−4444]], and Third Author[3 [2222−−3333−4444−5555]]

[1] Princeton University, Princeton NJ 08544, USA
[2] Springer Heidelberg, Europaplatz 3, 69115 Heidelberg, Germany lncs@springer.com
http://www.springer.com/gp/computer-science/lncs
[3] ABC Institute, Ruprecht-Karls-University Heidelberg, Heidelberg, Germany
{abc,lncs}@uni-heidelberg.de

**Abstract.** Legal case retrieval, which aims to find similar cases based on case facts, is a significant task in legal intelligence. In recent years, pre-trained models have been applied to legal case retrieval tasks; however, the constituent elements of legal cases have not yet been fully utilized. Compared to search engines, the retrieval of legal documents is more specialized, as legal texts are more structured and tend to be longer. Existing legal case retrieval methods leverage the semi-structured nature of legal texts by dividing legal documents into sections such as "facts," "holdings," and "judgment." To address these challenges, we proposed an **Article**-enhanced multi-defendant **L**egal **C**ase **R**etrieval framework called ArticleLCR, which incorporates legal provisions into the pre-training model to fully leverage the capabilities of the case retrieval model. Extensive experiments conducted on publicly available legal benchmark datasets demonstrate the effectiveness of our method in legal case retrieval tasks.

**Keywords:** Legal intelligence · Legal case retrieval · Article.

## 1 Introduction

Legal case retrieval is a critical task in intelligent judicial services. To uphold judicial justice and fairness, legal professionals must conduct extensive research and analysis of similar cases before handling a case. While ensuring the quality of their rulings, judges must also adhere to the principle of "consistent rulings for similar cases" [4]. In recent years, with the explosive growth of legal case documents, searching for relevant cases across the entire corpus has become increasingly challenging. As the internet continues to develop and the number of court cases gradually increases, the need for similar-case retrieval has become increasingly urgent. Currently, there are legal case retrieval websites available, including Bashou Case [5] and Beida Fabao [6], but there remains significant room for research regarding the task of retrieving similar cases.

[4] https://www.chinacourt.org/law/detail/2020/07/id/150187.shtml
[5] https://www.lawsdata.com/
[6] https://www.pkulaw.com/

Judicial documents are semi-structured texts, making it easier and more reliable to retrieve similar cases from them. As shown in Figure 1, judicial documents are generally divided into sections such as “facts,” “holdings,” and “judgment.” In criminal cases, the “judgment” section includes the applicable legal provisions, the charges, and the sentence; in civil cases, it includes the claims that were granted and those that were denied. We achieve the goal of retrieving similar cases by searching for these legal elements. In this example, the two cases are similar: both involve civil contract disputes, but one concerns a medical services contract dispute and the other a labor contract dispute.

| |
|---|
| Case 1: Plaintiff A sued Defendant Hospital over a *dental treatment dispute*, claiming the doctor lacked qualifications and failed to disclose risks, and sought RMB 77,402.80 in medical costs plus travel expenses. The court found the doctor was qualified and the hospital's deficiencies did not constitute fundamental breach. A's unilateral termination and treatment elsewhere were improper. The court ordered the hospital to refund RMB 9,414.80 and rejected other claims. |
| Case 2: Plaintiff B sued Defendant Company for unpaid wages and termination compensation, claiming an employment relationship from June 2012 to October 2014. The court found the company had ceased operations in 2010–2011, and the evidence (*labor contract*, attendance sheets, wage lists) was suspicious, self-made, and possibly collusive. The court rejected all claims. Case dismissed; fee borne by Plaintiff B. |

**Table 1.** An example of similar cases. Both cases involve civil contract disputes, but one concerns a medical services contract dispute and the other a labor contract dispute.

Existing research on precedent-based retrieval falls into three categories: methods that involve user participation [**?**], methods that focus on model pre-training [**?**], and methods that involve agents [**?**].

In fact, there are many technical and domain-specific challenges in addressing precedent-based retrieval tasks. These include how to define similarity among cases and how to retrieve similar cases based on the characteristics of the legal field.

To address this, we propose a case-law retrieval method called ArticleLCR. Specifically, we integrate legal provision prediction into the pre-training process. We have validated the feasibility of our proposed method on a case-law retrieval dataset. [7]

[7] Our code is available at `https://github.com/`

# 2 Related Works

## 2.1 Text Matching

Text matching tasks have been extensively studied, including matching similar exercises in education [1], matching resumes with job descriptions in the workplace [2], and retrieving text related to specific words in search engines. Text matching methods include traditional approaches such as TF-IDF and BM25. In recent years, the rise of pre-trained models and large language models has driven the development of text matching, such as using pre-trained models for dense retrieval and Retrieval-Augmented Generation with large language models [3].

Compared to other fields, case retrieval tasks are more specialized; for example, defining what constitutes “similarity” between cases and dealing with longer text in the query portion.

## 2.2 Legal Case Retrieval

# 3 PRELIMINARIES

## 3.1 Problem Formulation

Legal case retrieval task refers to finding cases relevant to a given query case from the candidate case set. Formally, given a query case q, and a set of candidate cases D, the task is to retrieve top-k relevant cases $D_q = d_1^*, d_2^*, \cdots, d_k^*$ from a large candidate pool.

# 4 ArticleLCR

## 4.1 overview

In this section, we describe our article-enhanced pre-trained lan-guage model for legal case retrieval in detail. In order to integrate the detail information of articles into the language model, we simulate the writing process of legal case documents and propose the ArticleLCR, which is shown in Figure 1. ArticleLCR mainly consists of three components, i.e., Fact Encoder, Reasoning Decoder, Decision Decoder and Article Decoder. Speciffcally, in the pretraining process, a BERT-like encoder is used to generate a vector representation of the Fact, and two shallow decoders are used to reconstruct the textof the Reasoning and Decision sections. Detailed descriptions about the ArticleLCR are as follows.

## 4.2 Fact Encoder

In the Fact Encoder, we first randomly replace some tokens of the Fact with a special token [MASK]. Only a small percentage of tokens is replaced, as the majority of the Fact need to be preserved.

In particular, We define the facts as $F = [f_1, f_2, \ldots, f_n]$ and the masked token set as $m(F)$, where $f$ denotes tokens and $n$ denotes the length of Fact. So the rest tokens can

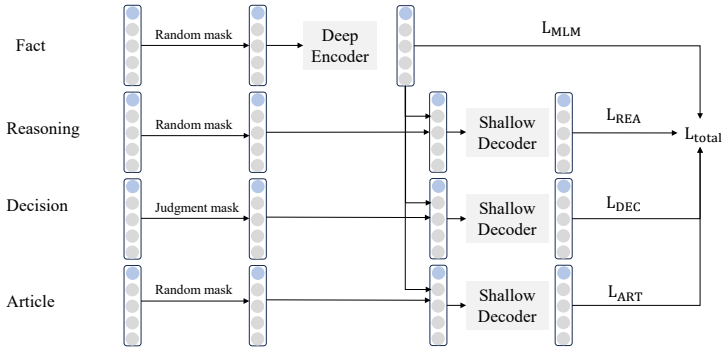



**Fig. 1.** The architecture of proposed ArticleLCR.

be represented as $F \backslash m(F)$. Then, the masked input $F_{\text{mask}}$ is transformed into a sentence vector by the Fact Encoder $\phi_F(\cdot)$. Following the previous work [9, 14], the final hidden state of the [CLS] token is taken as the representation of the entire sentence:

$$h_F = \phi_F\left(F_{mask}\right)$$

Similar to Bert, We input the masked facts into the encoder to recover $m(F)$ and compute mask language modeling (MLM) loss as one of our pre-training objectives. Specifically, the MLM loss $L_{MLM}$ is defined as:

$$L_{MLM} = -\sum_{x' \in m(F)} \log p\left(x' \mid F \backslash m(F)\right)$$

### 4.3 Reasoning Decoder

As mentioned above, the Reasoning section contains all the key legal elements of the case. Therefore, we design the Reasoning Decoder to model the logical relationships between Reasoning and Fact, aiming to improve the alignment between the focus of the encoder and the underlining key legal elements in case documents.

Specifically, we construct the Reasoning Decoder through polluting the original text of the Reasoning section $R = [r_1, r_2, \ldots, r_n]$. We split the Reasoning text into tokens and randomly select a subset $m(R)$ for masking. An aggressive mask ratio $(30\% - 60\%)$ is adopted for language reconstruction. In addition, the dense vector $h_F$ replaces the original [CLS] vector which is commonly used as the beginning of decoder inputs. The whole decoder input is formalized as:

$$\left[h_F, e_{r_1} + p_{r_1}, \ldots, e_{r_n} + p_{r_n}\right];$$

where $e_{r_k}$ and $p_{r_k}$ denote the embeddings and extra position embedding of $r_k$. In this way, the learning objective of the Reasoning Decoder is formulated as:

$$L_{REA} = -\sum_{x' \in m(R)} \log p\left(x' \mid [h_F, R \backslash m(R)]\right)$$

The Reasoning Decoder needs to rely on $h_F$ to recover the masked Reasoning section, which forces the $h_F$ to focus on useful information related to the Reasoning section in the facts. By this design, SAILER enhances attention to the key legal elements and captures the long dependencies between Facts and Reasoning.

### 4.4 Decision Decoder

Legal Judgment Prediction (LJP) aims to predict the judgment results of a case based on the fact description [11]. If two cases have the same charges and law articles, then their dense vectors should be closer in the vector space. In this part, the Legal Judgment Prediction is modeled as a text-to-text task by using Decision Decoder, which helps the vectors have a stronger discriminatory ability.

For Chinese legal cases, the Decision usually consists of the relevant law articles, charges, and terms of penalty. Given a Decision section $D = [d_1, d_2, \ldots, d_n]$, we mask it according to its specified format. Concretely, the Decision is usually formatted as follows: According to the Criminal Law of the People's Republic of China $[z_1]$, the defendant committed $[z_2]$ and was sentenced to $[z_3]$, where $z_1$ denotes the specific law articles, $z_2$ represents the charges, and $z_3$ is the term of penalty. We use $\mathcal{Z}$ to denote all masked tokens.

In some countries where the Decision does not have a specific format, we select a certain percentage of words with high TF-IDF [1] values to mask. For convenience, we still name the masked tokens as $\mathcal{Z}$.

The Decision Decoder also relies on the dense vector $h_F$ to recover the original decision. Specifically, the input of the Decision Decoder is:

$$\left[h_F, e_{d_1} + p_{d_1}, \ldots, e_{d_n} + p_{d_n}\right];$$

where $e_{d_k}$ and $p_{d_k}$ denote the embeddings and extra position embedding of $d_k$. The Decision Decoder is trained with the objective:

$$L_{DEC} = -\sum_{x' \in \mathcal{Z}} \log p\left(x' \mid [h_F, D \backslash \mathcal{Z}]\right)$$

In short, Decision Decoder cleverly models the Legal Judgment Prediction task to help generate discriminative vectors.

### 4.5 Article Decoder

We define the facts as $F = [f_1, f_2, \ldots, f_n]$ and the masked token set as $m(F)$, where $f$ denotes tokens and $n$ denotes the length of Fact.

$$h_F = \phi_F\left(F_{mask}\right) \tag{2}$$

$$L_{ART} = -\sum_{x' \in \mathcal{Z}} \log p\left(x' \mid [h_F, D \backslash \mathcal{Z}]\right)$$

### 4.6 Learning

During pre-training, we aim at optimizing the following training objectives:

$$L_{\text{Total}} = L_{MLM} + L_{REA} + L_{DEC} + L_{ART}$$

After pre-training, we drop both decoders and fine-tune the encoder. The purpose of fine-tuning is to make the query closer to related cases in the vector space compared to the irrelevant ones. Thus, given a query case $q$, let $d^+$and $d^-$be relevant and negative cases, the loss function $L$ is formulated as follows:

$$L\left(q, d^+, d_1^-, \ldots, d_n^-\right) = -\log \frac{\exp\left(s\left(q, d^+\right)\right)}{\exp\left(s\left(q, d^+\right)\right) + \sum_{j=1}^{n} \exp\left(s\left(q, d_j^-\right)\right)}$$

Following previous work [26, 41], the negative samples $d^-$are BM25 hard negatives. During training, it is computationally expensive to sample many negative samples for each query case. Therefore, in-batch negatives [16] are adopted for contrastive learning which to take full advantage of negative samples in the same batch. Specifically, if there are $B$ queries in a batch, each with one positive and $N$ negative examples, then we can obtain at most $(B-1)*(N+1)$ in-batch negative examples for each query during training.

## 5 Experiments

### 5.1 Datasets and Metrics

Statistical information for the LeCaRDv2 dataset is shown in the Table 2

### 5.2 Baselines

Traditional Retrieval Models

- BM25 [29] is a highly effective sparse retriever based on exactword matching.
- QL [40] is another representative traditional retrieval modelbased on dirichlet smoothing.

• Generic Pre-trained Models

| DATASETS | LeCaRD | CAIL2022-LCR | COLIEE2020 | COLIEE2021 | LeCaRDv2 |
|---|---|---|---|---|---|
| Language | chinese | chinese | english | english | |
| # Train query case | - | - | 520 | 650 | |
| # Train candidate cases/query | - | - | 200 | 4,415 | |
| # Test query case | 107 | 130 | 130 | 250 | |
| # Test candidate cases/query | 100 | 100 | 200 | 4,415 | |
| Avg. length per case document | 8,275 | 2,707 | 3,232 | 1,274 | |
| # Avg. relevant cases per query | 10.33 | 11.53 | 5.15 | 4.73 | |

**Table 2.** Statistics of benchmark datasets.

– BERT [9] is a multi-layer transformer encoder, which is astrong baseline in ad-hoc retrieval tasks.
– BERT_xs is a criminal law-speciffc BERT pre-trained with663 million Chinese criminal judgments.
– RoBERTa [18] is an enhanced version of BERT with substan-tially enlarged training data. Compared with BERT, RoBERTais pre-trained only with MLM task.

– Lawformer [34] is the ffrst Chinese pre-trained languagemodel in the legal domain, which focuses on processing longdocuments.
– LEGAL-BERT [5] is a pre-trained language model using alarge number of English legal documents, achieving state-of-the-art results in multiple tasks.

• Retrieval-oriented Pre-trained Models

– Condenser [14] is designed for ad-hoc dense retrieval. It uti-lizes skip connections to aggregate text information into densevectors.
– coCondenser [15] adds an unsupervised corpus-level con-trastive loss to Condenser, which can effectively warm up thevector space.
– SEED [21] theoretically analyzes the insufffciency of the au-toencoder architecture for dense retrieval. It uses a weak de-coder to enhance the encoder training process.
– CoT-MAE [33] designs efffcient data construction approachesto train asymmetric encoder-decoder architectures.
– RetroMAE [19] proposes enhanced decoding which makes itharder to reconstruct text and achieves state-of-the-art perfor-mance on ad-hoc retrieval.
– SAILER： A Pre-trained Text Encoder for the Legal Domain.

### 5.3 Implementation Details

### 5.4 Overall Performance

### 5.5 Ablation Study

### 5.6 Hyperparameter Analysis

The Table 5 shows the impact of different mask rate ratios on model performance.

| Model | LeCaRD | | | | | CAIL2022-LCR | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Precision | Recall | F1_score | NDCG@10 | NDCG@30 | Precision | Recall | F1_score | NDCG@10 | NDCG@30 |
| Traditional Retrieval Models | | | | | | | | | | |
| BM25 | 0.8916 | 0.1748 | 0.2922 | 0.7115** | 0.8172 | 0.8477** | 0.2018** | 0.3259** | 0.7303** | 0.8304* |
| QL | 0.8897 | 0.1737 | 0.2906 | 0.7157** | 0.8373 | 0.8538** | 0.2004** | 0.3246** | 0.7535** | 0.8545 |
| Generic Pre-trained Models | | | | | | | | | | |
| Chinese BERT | 0.6654** | 0.1263** | 0.2123** | 0.5252** | 0.5374** | 0.6600** | 0.1487** | 0.2427** | 0.5604** | 0.5618** |
| Chinese RoBERTa | 0.8841 | 0.1778 | 0.2960 | 0.7438** | 0.7897** | 0.9046 | 0.2180 | 0.3513 | 0.8043** | 0.8518 |
| BERT_xs | 0.7159** | 0.1377** | 0.2309** | 0.5695** | 0.5751** | 0.6462** | 0.1369** | 0.2259** | 0.5236** | 0.5206** |
| Lawformer | 0.8056** | 0.1552** | 0.2603** | 0.6216** | 0.6362** | 0.7908** | 0.1820** | 0.2935** | 0.6908** | 0.6988** |
| Retrieval-oriented Pre-trained Models | | | | | | | | | | |
| Condenser | 0.8280** | 0.1632** | 0.2727** | 0.6469** | 0.7125** | 0.8092** | 0.1892** | 0.3067** | 0.6925** | 0.7402** |
| coCondenser | 0.8411** | 0.1648** | 0.2756** | 0.6719** | 0.7404** | 0.8138** | 0.1931** | 0.3121** | 0.7063** | 0.7607** |
| SEED | 0.8411** | 0.1575** | 0.2653** | 0.6721** | 0.7330** | 0.8369** | 0.1906** | 0.3105** | 0.7365** | 0.7815** |
| COT-MAE | 0.8467** | 0.1567** | 0.2644** | 0.6815** | 0.7089** | 0.8446** | 0.1996** | 0.3229** | 0.7274** | 0.7311** |
| RetroMAE | 0.8505** | 0.1675** | 0.2799** | 0.6876** | 0.7326** | 0.8531** | 0.1974** | 0.3206** | 0.7378** | 0.7770** |
| SAILER | 0.9028 | 0.1902 | 0.3142 | 0.7979 | 0.8485 | 0.8938 | 0.2310 | 0.3671 | 0.8319 | 0.8660 |
| ArticleLCR | 0.44 | 0.44 | 0.44 | - | 0.93 | | | | | |

**Table 3.** Zero-shot performance of various baselines. */** denotes that SAILER performs signiffcantly better than baselines at p < 0.05/0.01 level using the ffsher randomization test. Best method in each column is marked bold.

| Method | NDCG@10 | NDCG@20 | NDCG@30 |
|---|---|---|---|
| SAILER | 0.7979 | 0.8226 | 0.8485 |
| SAILER_share | 0.7869 | 0.8009 | 0.8300 |
| w/o Decision Decoder | 0.7226 | 0.7442 | 0.7884 |
| w/o Reasoning Decoder | 0.7385 | 0.7607 | 0.7956 |
| w/o Both | 0.6479 | 0.6684 | 0.6867 |

**Table 4.** Ablation study on LeCaRD under zero-shot setting.Best results are marked bold.

| Enc | Dec | NDCG@10 | NDCG@20 | NDCG@30 |
|---|---|---|---|---|
| 0 | 15 | 0.7542 | 0.7746 | 0.8173 |
| 0 | 30 | 0.7685 | 0.7846 | 0.8202 |
| 0 | 45 | 0.7739 | 0.7905 | 0.8262 |
| 0 | 60 | 0.7740 | 0.7858 | 0.8285 |
| 15 | 15 | 0.7778 | 0.7950 | 0.8275 |
| 15 | 30 | 0.7835 | 0.7995 | 0.8343 |
| 15 | 45 | 0.7979 | 0.8226 | 0.8485 |
| 15 | 60 | 0.7865 | 0.8052 | 0.8412 |
| 30 | 15 | 0.7946 | 0.8048 | 0.8361 |
| 30 | 30 | 0.7952 | 0.8089 | 0.8378 |
| 30 | 45 | 0.7915 | 0.8001 | 0.8409 |
| 30 | 60 | 0.7830 | 0.8052 | 0.8445 |

**Table 5.** Impact of mask rate (%) on LeCaRD. Best results aremarked bold.

### 5.7 Case Study

The weight of a keyword represents the importance of a token. in similar cases, the more critical the word, the greater its weight. Experiments have validated the accuracy of the keywords. Figure 2 illustrates the attention weights of tokens in a case.

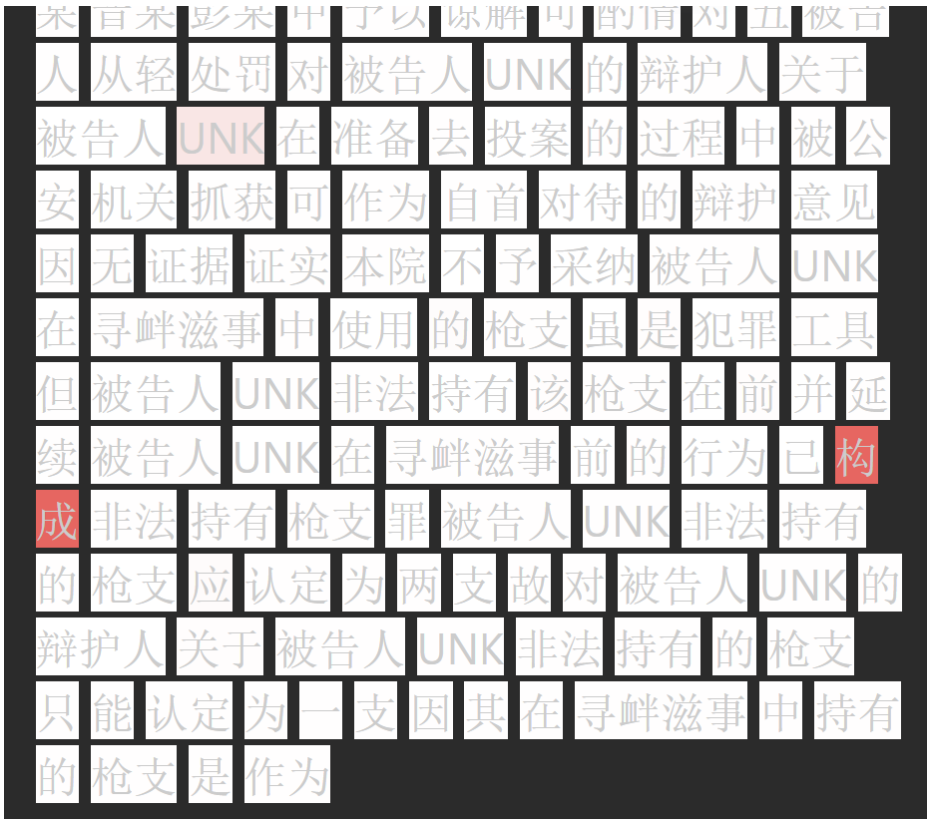



**Fig. 2.** Visualization of attention weight.

### 5.8 Visual Analysis

To further investigate SAILER’s discriminative ability, we used t-SNE to visualise the vector distributions of legal cases. Specifically, we selected six types of offences that are easily confused, including intentional assault, provocation, brawling, theft, robbery and fraud. For each offence, we randomly sampled a certain number of cases from the pre-training corpus and generated case vectors using off-the-shelf pre-trained models under a zero-shot setting. As shown in Figure 5, SAILER demonstrates strong discriminative capability. Chinese BERT consistently groups cases involving different offences together;

whilst Chinese RoBERTa, although capable of distinguishing between theft, robbery and fraud, confuses intentional assault, provocation and brawling. By reconstructing the original text, SEED is able to learn more information and generate uniformly distributed vectors. However, the boundaries between similar offences are too blurred, which is detrimental to legal case retrieval. In contrast, SAILER can more accurately distinguish between vectors representing different charges without any supervised data, indicating that it has acquired rich legal knowledge through our pre-training objectives.

By mapping the representations of cases involving different offences onto a two-dimensional vector space using t-SNE, cases that cluster together are found to be more similar. Experiments validate the interpretability of the proposed method.

### 5.9 ArticleLCR4LawyerRec

We are attempting to use case-based retrieval to recommend solicitors. Specifically, solicitors who have been involved in similar cases are presented as recommendations.

## 6 Conclusion

In this article, we propose ArticleLCR, article-enhanced pre-trained language model for legal case retrieval. Specifically, first, we incorporate legal articles and their detailed texts. Then, we mask a corresponding proportion of the legal articles as if they were from pre-trained corpora. Finally, we incorporate the legal articles into the pre-training process through the legal article prediction task to enhance the performance of proposed model. The effectiveness of our proposed method was verified on the benchmark dataset.

**Acknowledgments.** A bold run-in heading in small font size at the end of the paper is used for general acknowledgments, for example: This study was funded by X (grant number Y).

**Disclosure of Interests.** It is now necessary to declare any competing interests or to specifically state that the authors have no competing interests. Please place the statement with a bold run-in heading in small font size beneath the (optional) acknowledgments[8], for example: The authors have no competing interests to declare that are relevant to the content of this article. Or: Author A has received research grants from Company W. Author B has received a speaker honorarium from Company X and owns stock in Company Y. Author C is a member of committee Z.

## References

1. Liu, Q., Huang, Z., Huang, Z., Liu, C., Chen, E., Su, Y., Hu, G.: Finding similar exercises in online education systems. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 1821–1830 (2018)
2. Qin, C., Zhu, H., Xu, T., Zhu, C., Jiang, L., Chen, E., Xiong, H.: Enhancing person-job fit for talent recruitment: An ability-aware neural network approach. In: The 41st international ACM SIGIR conference on research & development in information retrieval. pp. 25–34 (2018)

[8] If EquinOCS, our proceedings submission system, is used, then the disclaimer can be provided directly in the system.

3. Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., Cao, Y.: ReAct: Synergizing reasoning and acting in language models. In: International Conference on Learning Representations (ICLR) (2023)