Intent-DQD: An Intent-based and Annotation-free Method for Duplicate Question Detection in CQA Forums

Yubo Shu$^1$, Hansu Gu$^2$, Peng Zhang$^1$, Tun Lu$^1$, Ning Gu$^1$

$^1$Fudan University, Shanghai, China
$^2$Seattle, United States

ybshu20@fudan.edu.cn, hansug@acm.org
{zhangpeng_, lutun, ninggu}@fudan.edu.cn

Abstract

With the advent of large language models (LLMs), Community Question Answering (CQA) forums offer well-curated questions and answers that can be utilized for instruction-tuning, effectively training LLMs to be aligned with human intents. However, the issue of duplicate questions arises as the volume of content within CQA continues to grow, posing a threat to content quality. Recent research highlights the benefits of detecting and eliminating duplicate content. It not only enhances the LLMs’ ability to generalize across diverse intents but also improves the efficiency of training data utilization while addressing concerns related to information leakage. However, existing methods for detecting duplicate questions in CQA typically rely on generic text-pair matching models, overlooking the intent behind the questions. In this paper, we propose a novel intent-based duplication detector named Intent-DQD that comprehensively leverages intent information to address the problem of duplicate question detection in CQA. Intent-DQD first leverages the characteristics in CQA forums and extracts training labels to recognize and match intents without human annotation. Intent-DQD then effectively aggregates intent-level relations and establishes question-level relations to enable intent-aware duplication detection. Experimental results on fifteen distinct domains from both CQADupStack and Stack Overflow datasets demonstrate the effectiveness of Intent-DQD. Reproducible codes and datasets will be released upon publication of the paper.

1 Introduction

With the rapid development of web technology, Community Question Answering (CQA) forums have emerged as crucial platforms for individuals to seek solutions to their problems. These forums serve as high-quality knowledge bases, offering well-curated questions and answers that can be effectively utilized for instruction-tuning for large language models (LLMs). Instruction tuning plays a vital role in the Reinforcement Learning from Human Feedback (RLHF) process employed in building systems like ChatGPT and Claude (Ouyang et al., 2022; Bai et al., 2022). The purpose of instruction tuning is to train LLMs to better follow human intents. For instance, when a human posts a question to an LLM, instruction tuning helps the LLM learn to provide an answer instead of merely imitating the act of asking a question. As the volume of content in CQA forums continues to increase, the issue of duplicate questions becomes more prevalent, thereby posing a significant challenge to content quality.

Recent research emphasizes the significance of detecting and removing duplicate content in the instruction-tuning of LLMs. It promotes generalization across diverse intents (Hernandez et al., 2022), reduces overlap between training and test sets (Lee et al., 2022), mitigates information leakage risks (Kandpal et al., 2022), enhances learning stability, and improves training data efficiency. Inspiringly, Zhou et al. (2023) discovers that even a carefully selected set of 1,000 unique questions can effectively train an LLM to follow intents and achieve competitive performance when compared to OpenAI’s GPT3-DaVinci003 model.

Despite the impressive results of previous studies in detecting duplication Liang et al. (2019); Xia et al. (2021); Peng et al. (2022), these methods do not consider intents in question, which may lead to sub-optimal results when deduplicating content. Figure 1 shows two question pairs that illustrate the importance of considering intents in duplication detection for instruction-tuning. From a generic text-pair matching perspective, both the exemplar duplicate and neutral pairs are partially semantically matched. Despite being mismatched in the non-intent part, the first pair is labeled as duplica-
I am making a space shooter game. How to fire bullets in the direction a ship is facing?

I have a space shooter game I am making. I want to play a sound every time a user hits the button fire.

Java non-static variable reference Error. I Googled resolution suggesting the variable isn't initialized yet static variable reference

rized as follows:

detect duplicate questions.

question pair level, based on inter and intra similarities in CQA forums.

Finally, we design a similarity matrix on the learn intent representations in a low-dimensional contrastive learning (Khosla et al., 2020) and properly match intents without human annotation. Based on matched intent pairs, we leverage supervised contrastive learning (Khosla et al., 2020) and properly learn intent representations in a low-dimensional space. Finally, we design a similarity matrix on the question pair level, based on inter and intra similarities between multiple intents for each question to detect duplicate questions.

The key contributions of this paper are summarized as follows:

• We propose a method to aggregate intent-level relations and obtain question-level relations to detect duplicate questions with multiple intents.

• The experimental results on two datasets covering fifteen domains demonstrate that our method can leverage question intent and achieve better performance over previous methods.

2 Annotation-free Label Collection

The primary challenge is the absence of labels for (1) intent recognition, which involves determining whether a sentence expresses an intent or not; (2) intent matching, which involves classifying whether the relationship between intents is matched or mismatched. In order to overcome this challenge, we introduce strategies for mining labels without human annotation by leveraging the characteristics of CQA forums.

2.1 Labels for Intent Recognition

The intent of a question refers to the user’s goal or purpose in asking it. We have found that in most cases, the question title serves as a concise representation of what users are asking, which can be considered as intent. However, only using titles may lead to noise when titles are too short or do not provide enough information, such as “Order list” and “What should I do?”. Inspired by previous research (Beyer et al., 2020), we additionally use regular expression patterns to match more accurate intents and eliminate overly generic intents only with pronouns, such as "how to do it/this/that". More information about used regex patterns is in Appendix A.1.

2.2 Labels for Intent Matching

After recognizing intents, there exist three types of links which can be used to derive valid labels for intent matching. Consider a common situation where a pair of questions is labeled as duplication in CQA forums, as shown in Figure 2, we can extract semantically matched intent pairs from inter, intra and mixed-links:

• Intra-link Inside one question, the intent in the title and the intent in the body can be linked. As the title and the body are expected to be consistent, their intents are likely semantically the same.
Find number of lines in csv without reading it

How to get line count of a large file cheaply in Python?

Figure 2: An example of mining matched intents through inter, intra, mixed-links in a pair of duplicate questions.

- **Inter-link** Between a pair of duplicate questions, intents in their titles can be duplicate as well.

- **Mixed-link** We can combine intra-links and inter-links to create mixed-links and further generate more training samples for intent matching.

Besides question pairs, it is worth noting that the intra-link can also work even within a single question. Similarly, we extract negative intent pairs from questions labeled as "related but not duplicate" in CQA forums by utilizing inter and mixed links. These negative intent pairs often exhibit lexical similarity but possess distinct semantic meanings. Consequently, pairs formed by these intents can serve as hard negative labels for training.

3 Method

After label collection, we proceed to present our method for detecting duplicate questions. IntentDQD consists of three key components: intent recognition module, intent matching module, and duplicate question detection module. Figure 3 shows the overall structure of our method. In the intent recognition module, we exploit the inner features of intent sentences to augment recognition accuracy. In the intent matching module, we leverage supervised contrastive learning (Khosla et al., 2020) and properly learn intent representations in a low-dimensional space. In the duplicate question detection module, we design a similarity matrix on the question pair level to integrate inter and intra-similarities between multiple intents.

3.1 Intent Recognition

To ensure the accuracy of intent recognition, we incorporate three types of features in addition to transformer-based language models. We first focus on keywords that match tags in the QA forum. Matches such as "Tensorflow" or "PyTorch" indicate the mention of important concepts that are recognized by the QA forum, and are thus helpful for the model to learn if they are related to intent detection. We build a keyword vocabulary from all tags in the forum and use a binary flag to indicate whether the current word matches the tag. Additionally, considering the sentence "What is the difference between Concept A and Concept B?", we can classify the sentence as intent without knowing the specific concepts. So, we add part-of-speech (POS) information to help the model utilize the linguistic features of a sentence. Lastly, we use sentence position in the question as a feature because some users may prefer to put intents at the start or the end of a post. We implement all three features as extra embeddings added on top of existing text token embedding as illustrated in Figure 3(b).

3.2 Intent Matching

To distinguish matched or mismatched intents, we develop a dedicated intent matching layer based on contrastive learning. After the label collection process, we have collected both matched intents pairs (positive samples) and mismatched intent pairs (negative samples). We then feed these intent pairs to calculate the loss function of supervised contrastive learning. More specifically, \( \text{IntSet} \) represents a dataset of intents, and \( \text{intent}_k \) is the k-th intent in the dataset. Inside a training batch, we define: \( P(k) \) contains positive intents.
of \( \text{intent}_k \); and \( N(k) \) contains negative intents of \( \text{intent}_k \); \( A(k) \equiv N(k) \cup P(k) \setminus \text{intent}_k \) is a union set excluding \( \text{intent}_k \). For \( \text{intent}_k \), the supervised contrastive loss \( l \) is formulated as:

\[
l = -\frac{1}{|P(k)|} \sum_{p \in P(k)} \log \frac{\exp(\text{sim}(I_k, I_p)/\tau)}{\sum_{a \in A(k)} \exp(\text{sim}(I_k, I_a)/\tau)}
\]

where \( I \) denotes the encoded vector of a intent, and \( \text{sim} \) donates cosine similarity function and \( \tau \in \mathbb{R}^+ \) is a scalar temperature parameter. During the training process, optimizing the loss function for each intent in \( \text{IntSet} \) will decrease the distance between positive intents in the embedding space and increase the distance between negative intents.

We use the intent embeddings from contrastive learning to match intents within a pair of questions. As shown in figure 3, given two questions \( Q_A \) and \( Q_B \), we encode the intents of \( Q_A \) as a list of intent embeddings \( \mathbf{I}_1^A, \mathbf{I}_2^A, \ldots, \mathbf{I}_M^A \) and intents of \( Q_B \) as \( \mathbf{I}_1^B, \mathbf{I}_2^B, \ldots, \mathbf{I}_N^B \). The \( M \) and \( N \) represent the number of intents in \( Q_A \) and \( Q_B \), respectively (e.g., \( M = 2 \) and \( N = 3 \) for the case in the figure). We match intents within each question and across questions, generating the following intent similarity matrices:

\[
R^{AA} = [\mathbf{I}_1^A, \mathbf{I}_2^A, \ldots, \mathbf{I}_M^A] \cdot [\mathbf{I}_1^A, \mathbf{I}_2^A, \ldots, \mathbf{I}_M^A]^T
\]

\[
R^{BB} = [\mathbf{I}_1^B, \mathbf{I}_2^B, \ldots, \mathbf{I}_N^B] \cdot [\mathbf{I}_1^B, \mathbf{I}_2^B, \ldots, \mathbf{I}_N^B]^T
\]

\[
R^{AB} = [\mathbf{I}_1^A, \mathbf{I}_2^A, \ldots, \mathbf{I}_M^A] \cdot [\mathbf{I}_1^B, \mathbf{I}_2^B, \ldots, \mathbf{I}_N^B]^T
\]

The first matrix \( R^{AA} \in \mathbb{R}^{M \times M} \) is used to match intents within \( Q_A \). Similarly, the second \( R^{BB} \in \mathbb{R}^{N \times N} \) is used to match intents within \( Q_B \). The third \( R^{AB} \in \mathbb{R}^{M \times N} \) is used to match intents between \( Q_A \) and \( Q_B \). The idea behind matching intent in this way is to provide a comprehensive view of question-level relation identification. In the next module, we will introduce how to aggregate these intent-level relations into question-level relations for duplication detection.

### 3.3 Duplicate Question Detection

It is worth noting that a question can have one or multiple intents. Therefore, in the duplicate question detection module, we combine the above matrices and calculate a holistic similarity matrix \( S \) between \( Q_A \) and \( Q_B \). The shape of \( S \) is \( M \times N \). The similarity score at the \( i \)-th row and \( j \)-th column, \( S_{ij} \), represents not only an intent-level relation between \( \text{intent}_i^A \) and \( \text{intent}_j^B \), but also the degree to which this intent-level relation contributes to the question-level relation. For example, if \( \text{intent}_i^A \) and \( \text{intent}_j^B \) are semantically matched, its contribution degree to \( S_{ij} \) depends on two situations: (1)
intent_A is also matched with other intents in Q_A, indicating that Q_A is a focused question, which makes the match between intent_A and intent_B more significant for duplication. (2) intent_A is mismatched with other intents in Q_A, meaning that the match between intent_A and intent_B has a minor contribution for duplication. For this purpose, when calculating S_ij, we take into account the inner intent similarity matrices R^{AA} and R^{BB} with R^{AB}. The explicit formula of S_ij is below:

\[ S_{ij} = \frac{\sum_{m=1}^{M} \sum_{n=1}^{N} R^{AA}_{mn} \cdot R^{AB}_{mn} \cdot R^{BB}_{nj}}{(\sum_{m=1}^{M} R^{AA}_{mn}) \cdot (\sum_{n=1}^{N} R^{BB}_{nj})} \]  

After calculating \( S \in \mathbb{R}^{M \times N} \), as shown in Figure 3, we obtain the final feature \( f \) by concatenating the max pooling of \( S \) with the interaction of question embedding \( h_A \) and \( h_B \) from the bi-encoder:

\[ f = [h_A : h_B : |h_A - h_B| : \text{MaxPooling}(S)] \]

We use a linear layer followed by a Softmax to make the final prediction:

\[ \hat{y} = \text{softmax}(\text{ReLU}(Wf + b)) \]

where \( \hat{y} \) is the prediction of probability in \{duplicate, neutral\}. For duplicate question detection, the ultimate goal is to optimize the standard binary cross-entropy loss function.

4 Experiment Setting

4.1 Datasets

We introduce two datasets covering fifteen different domains to evaluate our method. The first dataset is CQADupStack, and the second dataset comes from the Stack Overflow data dump. Table 1 shows the statistics of two datasets.

Table 1: The statics of datasets

<table>
<thead>
<tr>
<th></th>
<th>CQADupStack</th>
<th>SODup</th>
</tr>
</thead>
<tbody>
<tr>
<td># of question pairs</td>
<td>48090</td>
<td>100000</td>
</tr>
<tr>
<td># of intents</td>
<td>259588</td>
<td>467816</td>
</tr>
<tr>
<td># of non-intents</td>
<td>404373</td>
<td>595488</td>
</tr>
<tr>
<td>intent rate</td>
<td>39.1%</td>
<td>44.0%</td>
</tr>
<tr>
<td>average intents per question</td>
<td>3.12</td>
<td>3.56</td>
</tr>
</tbody>
</table>

CQADupStack dataset was released by Ahasanuzzaman et al. (2016) and has become a popular benchmark for CQA research (Nakov et al., 2017). The dataset includes data from 12 CQA sites in Stack Exchange, such as Android, English, Game, Mathematics, Physics and Unix. We use duplicate questions and neutral questions in the dataset and randomly split them into training, validation, and test sets with a ratio of 8:1:1. However, the dataset does not include the largest forum Stack Overflow (SO) as the authors stated that SO data was too large to process at that time. To overcome the limitation, we have constructed an additional dataset, the Stack Overflow duplicate questions dataset (SODup).

SODup dataset. Stack Overflow is a widely used CQA forum that focuses on code-related questions. All questions and question links are available at the official data dump.\(^1\) We downloaded the data dump and then filtered questions according to the three most popular programming languages: JavaScript, Python and Java. We extract the duplicate/neutral question relations based on the LinkTypeId field of PostLinks table: (1) Duplicate relation: If a question pair has LinkTypeId=3, it means that the pair is marked as "Duplicate" by forum users. Therefore, we recognize the question pairs having LinkTypeId=3 as duplicate relations. (2) Neutral relation: Instead of just using random question pairs which are too simple for classification, we also gather indirectly referenced questions as neutral relations which serve as hard neutrals. Specifically, a question pair with LinkTypeId=1 means that one question references the other. However, some of the question pairs with a LinkTypeId=1 in the data dump are indeed duplicates in reality due to the delay of user annotation. To increase the probability of being true neutrals, we use questions with indirect references as neutral relations. The ratio of random and indirectly referenced questions is 1:1 , and the ratio of duplicate and neutral relations is also 1:1 . We adopt the same setting 8:1:1 for training, validation and test split in the SODup dataset.

4.2 Training Details

In order to avoid label leakages, there is no overlap on question_ids between the test dataset for duplication question detection and the dataset for intent recognition and matching. We train intent recognition and duplicate question detection with a batch size of 32. The batch size for contrastive learning in intent matching is 16. We set a maximum

\(^1\)https://archive.org/details/stackexchange
Table 2: Overall performance of duplicate question detection.

<table>
<thead>
<tr>
<th></th>
<th>SODup</th>
<th>CQADupStack</th>
</tr>
</thead>
<tbody>
<tr>
<td>BiMPM</td>
<td>0.847</td>
<td>0.870</td>
</tr>
<tr>
<td>MFAE</td>
<td>0.853</td>
<td>0.869</td>
</tr>
<tr>
<td>SBERT</td>
<td>0.868</td>
<td>0.873</td>
</tr>
<tr>
<td>SRoBERTa</td>
<td>0.896</td>
<td>0.891</td>
</tr>
<tr>
<td>Intent-DQD - Bert</td>
<td>0.887</td>
<td>0.882</td>
</tr>
<tr>
<td>Intent-DQD - Roberta</td>
<td>0.908</td>
<td>0.902</td>
</tr>
</tbody>
</table>

Table 3: The result of ablation study. It shows the overall duplicate question detection performance on the two datasets. A, B and C donate three combinations of intent recognition (ir), intent matching (im) and relation aggregation (ra) illustrated in Formula 1.

<table>
<thead>
<tr>
<th></th>
<th>ir</th>
<th>im</th>
<th>ra</th>
<th>SODup</th>
<th>CQADup</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>0.894</td>
<td>0.896</td>
</tr>
<tr>
<td>B</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>0.903</td>
<td>0.912</td>
</tr>
<tr>
<td>C</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>0.900</td>
<td>0.914</td>
</tr>
<tr>
<td>Intent-DQD</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>0.908</td>
<td>0.921</td>
</tr>
</tbody>
</table>

5 Result and Analysis

5.1 Duplicate Question Detection Result

We compare Intent-DQD with a few baselines, including BiMPM (Wang et al., 2017), and MFAE (Zhang et al., 2020) that are representative duplicate question detection methods, as well as general-purposed SBERT and SRoBERTa models with fine-tuning. BiMPM encodes each question individually and then employs a multi-perspective matching mechanism for question pairs. MFAE utilizes pre-trained language models to encode questions and enhance keywords in question matching. Additionally, to compare with SBERT and SRoBERTa, we implement a BERT-based and a RoBERTa-based Intent-DQD, respectively.

Table 2 demonstrates the F1-score of classification results on fifteen domains. From the results, our Intent-DQD outperforms the baselines in all fifteen domains. The largest improvement is on the Math domain at 4.8%, and the least increase is 1.1% on the Python domain. We observe that even incorporating keyword emphasis on top of BERT, MFAE is not able to surpass general-purposed SBERT in most domains. One possible reason is that MFAE does not differentiate intent, and all emphasized keywords can also appear in non-intent sentences, which may introduce extra noise when detecting duplicates.

5.2 Ablation Study

Since Intent-DQD has different components, it is essential to dissect the impact of each component contributing to performance improvement. Considering that Roberta-based Intent-DQD achieves the highest performance in most forums, we choose it for the following ablation experiments.

As shown in Table 3, the combination A is a SRoBERTa-only model which performs the worst, and the complete Intent-DQD achieves the highest performance on both datasets. With only intent recognition, combination B can only feed the recognized intents into Bi-Encoder and combine encoded intents with encoded questions. Even so, we see an increase in performance, which suggests the significance of utilizing intents for detecting duplicate questions. However, in combination C, only using intent matching can slightly increase performance on CQADupStack dataset and even decrease...
the performance on SODup dataset. This is likely due to the fact that questions in forums tend to have multiple intents, and without effectively using intent-level relations, it is not comprehensive enough to determine question-level relations. As shown in Table 1, SODup has higher average intent per question than CQADupStack, thus more challenging without proper question-level aggregation. Lastly, we observe that aggregating intent-level relation can significantly improve classification performance, which further verifies that the relation aggregation module is crucial and it can only achieve its effectiveness when paired with the intent matching module.

### 5.3 Intent Recognition Verification

Since the intent recognition is the basis of our method, it is essential to verify the performance of our intent recognizer. We randomly sample 5,000 sentences from QA forums and manually annotate whether a sentence is intent or not. After reaching an agreement (kappa > 0.9), authors annotated a test dataset including 1,916 sentences labeled as intent and 3,084 labeled as non-intent. We then verify our intent recognition module on the test dataset. Table 4 shows that either regex or title contributes to duplicate detection in the two cases.

### 5.4 Case Study

To better understand how Intent-DQD works, we have studied two test cases from the CQADupStack and SODup datasets. To analyze the contributions of each component, we further provide intermediate outputs to illustrate how intent recognition, intent matching, and duplicate question detection contribute to duplicate detection in the two cases.

The first case consists of two questions from the English domain. The label of the question pair is "Duplicate" as both questions are asking for the difference between "got" and "gotten". The prediction performance, which further verifies that the combination of R, T and S can produce an effective intent recognizer without human annotation.

| Case 1: Duplicate | Case 2: Neutral | Table 4: As mentioned in Sec 3.1, R uses Regex patterns to find intents for train data. T treats Titles as intents for train data. S integrates sentence Structure features(POS, Sentence Position, and Keyword) in training. Our result in table 4 shows that the combination of R, T and S can produce an effective intent recognizer without human annotation.

<table>
<thead>
<tr>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>0.816</td>
<td>0.621</td>
</tr>
<tr>
<td>T</td>
<td>0.914</td>
<td>0.633</td>
</tr>
<tr>
<td>R+T</td>
<td>0.862</td>
<td>0.809</td>
</tr>
<tr>
<td>R+T+S</td>
<td>0.864</td>
<td>0.831</td>
</tr>
</tbody>
</table>

Figures: Two cases that Intent-DQD correctly makes predictions while SRoBERTa makes wrong predictions.
matching component in Intent-DQD is able to output a high matching score of the intent pairs, thus closing the lexical gap. As seen in Figure 4, QA has quite consistent intents, and the main intent of QB matches the intent of QA.

The second case consists of two questions from the python domain. The label of the question pair is "Neutral". Although both questions are about looping in python, the nuances of the intents make a difference i.e. the second intent of $Q_A$ is to catch errors and loop, while the second intent of $Q_B$ is to loop after condition control such as if-else. Additionally, the non-intents of the two questions are similar, and they both indicate that the question authors have little experience in the python domain. Therefore, it is not surprising that SRoBERTa predicts it as "Duplicate". As for Intent-DQD, the intent matching component can discover the difference between $I_A^2$ and $I_B^2$ because of training from hard negative samples. In the end, the relation aggregation component can comprehensively consider intent-level relations and produce relatively low question-level match scores for the duplication classification.

5.5 Scalability in Real-world

In our method, given a new question, the intermediate results of intent recognition and intent embeddings can be stored and reused. Additionally, a scalable paradigm for detecting duplicates typically involves two stages: (1) a coarse-grained stage that retrieves duplication candidate question pairs and (2) a fine-grained stage that classifies candidates to detect duplication. Intent-DQD contributes directly to the fine-grained stage and thus our evaluation focuses on the classification results. As for the coarse-grained stage, our intent embeddings can be utilized to construct question-level embeddings and extend to duplication candidate retrieval.

6 Related Work

Instruction tuning is commonly regarded as the first step in the RLHF training process for training human-aligned LLMs including GPT-3.5, GPT-4, Claude (Ouyang et al., 2022; Bai et al., 2022). The goal of instruction tuning is to train LLM in following human intents expressed in instructions (OpenAI, 2023; Taori et al., 2023; Chiang et al., 2023). For instance, when a human asks a question to a LLM, their intent is to receive a proper answer. Before instruction tuning, a pre-trained LLM may continue the question or generate similar questions, whereas an aligned LLM trained through instruction tuning is more likely to generate an answer based on the question.

Collecting high-quality training data is fundamental for effective instruction tuning. CQA forums have millions of well-aligned Q&A data, making them ideal sources for instruction tuning (Zhou et al., 2023). However, the issue of duplicate questions arises as the volume of content within CQA continues to grow, posing a threat to content quality. Recent research highlights the importance of detecting and removing duplication to optimize for LLM instruction tuning. Hernandez et al. (2022) find deduplication increases generalization ability and learning stability. It can also save computation and decrease train-test set overlap Lee et al. (2022). Deduplication also reduces privacy risks because LLM is vulnerable to privacy attacks when sensitive information is duplicated in the instruct-tuning dataset (Kandpal et al., 2022).

Existing duplication detection studies mainly build text-pair classifiers based on bi-encoders or cross-encoders. Bi-encoders (Mueller and Thirumurugan, 2016; Conneau et al., 2017; Reimers and Gurevych, 2019; Zhang et al., 2020), encode each question individually while cross-encoders (Devlin et al., 2019; Liang et al., 2019; Liu et al., 2019; He et al., 2020), use cross-attention between questions or self-attention by concatenating multiple questions. Despite the popularity of cross-encoders, they typically have high computational cost (Reimers and Gurevych, 2019) and inconsistent performance (Chen et al., 2020). In this work, we focus on bi-encoders, which are more computationally efficient and have shown consistent performance in duplicate question detection. However, the above-mentioned methods are generally generic text-pair matching and do not consider intents in question, which can lead to sub-optimal results in deduplicating datasets for instruction tuning.

Therefore, we incorporate intent features in Intent-DQD to detect duplicate questions. Unlike existing research on intents (Kim et al., 2016; Coucke et al., 2018; Weld et al., 2021, 2022) which typically rely on human annotation, our proposed method is annotation-free, which leverages characteristics of CQA forums and automatically mines labels for intents.
7 Conclusion
In this work, we propose an intent-based duplicate question detection method Intent-DQD. We design intent recognition to identify question intent without human annotation. Intent matching employs contrastive learning to enhance the embeddings. Duplicate question detection effectively aggregates intent-level relations and obtains question-level relations. Evaluation on fifteen domains from two real-world datasets shows that our method outperforms existing state-of-the-art methods.

8 Limitations
The main assumption of this work is that the user-generated content in CQA forums contains explicit intent. The effectiveness of our carefully designed intent recognition, intent matching and duplicate detection modules may have limited success if the questions do not have clear intent. This can be true in forums that focus on open discussions, where the objective of the discussion is unclear.

9 Ethics Statement
We acknowledge and conform ACL and ACM Code of Ethics. The datasets we used are from the official public data source, and we use the data under the license. In data processing, we do not use user private information. Our method does not create toxic content or bias in different groups.

10 Acknowledgments
We would like to thank the anonymous reviewers and appreciate their insightful comments that helped improve the quality of the paper. This work was supported by the National Natural Science Foundation of China (NSFC) under Grants 61932007 and 62172106. Peng Zhang is a faculty of School of Computer Science, Fudan University. Tun Lu is a faculty of School of Computer Science and Shanghai Key Laboratory of Data Science, Fudan University and Shanghai Institute of Intelligent Electronics & Systems. Peng Zhang and Tun Lu are the corresponding authors.

References


Nikhil Kandpal, Eric Wallace, and Colin Raffel. 2022. Deduplicating training data mitigates privacy risks


### Appendix

#### A.1 Regex Expression Examples

- **Regex expressions for mining intents:**
  
  ```
  does ["\s"]+ [0-3].work
  can ["fwe"] use ["\s"] apply ["\s"] detect
  tell ["\s"] explain ["\s"] why
  does ["\s"] know ["\s"] do ["\s"] logcat ["\s"] stacktrace ["\s"] log ["\s"] gradle ["\s"] error ["\s"] exception
  
  how ["\s"] solve ["\s"] it
  
  does ["\s"] work
  can (I|we) (use|apply|detect)
  (tell|explain) why
  does (logcat|stacktrace|log) (error|exception)
  ```

- **Patterns for removing noise:**
  
  ```
  how ["\s"]+ [0-5].solve ["\s"] does ["\s"] understand ["\s"]+ [1-5].how ["\s"]+ [0-5].does ["\s"]+ (try|try|tried|trying) ["\s"] how ["\s"]+ [0-5].does ["\s"]+ how ["\s"]+ [0-5].does ["\s"]+
  ```
A.2 Detail Statics of Datasets

Table 5 shows the statics of each domain in SODup dataset, and Table 6 shows the CQADupStack dataset.

### Table 5: The statics of SODup dataset.

<table>
<thead>
<tr>
<th>Domain</th>
<th># of Question Pairs</th>
<th># of Intents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Javascript</td>
<td>33334</td>
<td>171061</td>
</tr>
<tr>
<td>Python</td>
<td>33334</td>
<td>153704</td>
</tr>
<tr>
<td>Java</td>
<td>33334</td>
<td>143051</td>
</tr>
</tbody>
</table>

### Table 6: The statics of CQADupStack dataset.

<table>
<thead>
<tr>
<th>Domain</th>
<th># of Question Pairs</th>
<th># of Intents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Android</td>
<td>3426</td>
<td>18529</td>
</tr>
<tr>
<td>English</td>
<td>7782</td>
<td>32412</td>
</tr>
<tr>
<td>Game</td>
<td>4558</td>
<td>23813</td>
</tr>
<tr>
<td>Gis</td>
<td>2232</td>
<td>14270</td>
</tr>
<tr>
<td>Mathematics</td>
<td>2744</td>
<td>15793</td>
</tr>
<tr>
<td>Physics</td>
<td>3936</td>
<td>22075</td>
</tr>
<tr>
<td>Programmers</td>
<td>3476</td>
<td>23813</td>
</tr>
<tr>
<td>Stats</td>
<td>1832</td>
<td>11714</td>
</tr>
<tr>
<td>Tex</td>
<td>10390</td>
<td>54887</td>
</tr>
<tr>
<td>Unix</td>
<td>3428</td>
<td>20164</td>
</tr>
<tr>
<td>Webmasters</td>
<td>2790</td>
<td>14474</td>
</tr>
<tr>
<td>Wordpress</td>
<td>1496</td>
<td>9142</td>
</tr>
</tbody>
</table>

A.3 The Effect of Contrastive Learning

We randomly sample a batch of matched intents and mismatched intents. We visualize the similarity between intents in two situations: (1) without contrastive learning (CL), we encode intents through pre-trained BERT (2) with contrastive learning, we further train the BERT through the contrastive loss function. As shown in Figure 5, after contrastive learning, matched intents (diagonal elements) have a more prominent distinction with mismatched intents (off-diagonal elements).

![Figure 5: Comparison of intent cosine similarity before or after contrastive learning. The diagonal elements are matched intent pairs, and the non-diagonal elements are mismatched intent pairs. The color of the cell indicates the similarity score.](image)

A.4 ChatGPT for Duplicate Question Detection

We have randomly selected 100 question pairs from SODup test data and 100 pairs from CQADupStack test data. Then we ask ChatGPT (GPT-3.5-turbo) to classify these pairs, and the final F1-scores are 0.855 and 0.861 for SODup and CQADupStack, whereas the F1-scores for Intent-DQD are 0.9 and 0.92 respectively. The results imply that duplicate question detection can be more complex than initially anticipated. The complexity arises from the inherent difficulty in precisely defining what constitutes duplication. In CQA forums, duplicate questions are not strictly limited to having exact lexical or semantic equality. Instead, duplication can encompass scenarios where one question can be inferred from or infer another question to some degree. However, it is difficult to have an explicit definition, making it equally challenging for ChatGPT to accurately detect duplication.

A.5 Relationship between the intent recognition and DQD

Table 7 presents an extended experiment examining the relationship between the accuracy of intent recognition and the performance of duplicate question Detection (DQD). The table indicates that a higher F1 score in intent recognition correlates with a higher F1 score in DQD.

<table>
<thead>
<tr>
<th></th>
<th>SODup F1</th>
<th>CQADup F1</th>
<th>Intent F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>0.861</td>
<td>0.879</td>
<td>0.706</td>
</tr>
<tr>
<td>T</td>
<td>0.875</td>
<td>0.892</td>
<td>0.748</td>
</tr>
<tr>
<td>R+T</td>
<td>0.904</td>
<td>0.916</td>
<td>0.834</td>
</tr>
<tr>
<td>R+T+S</td>
<td>0.908</td>
<td>0.921</td>
<td>0.847</td>
</tr>
</tbody>
</table>

The rank of DQD performance across various strategies is consistent with the rank of the F1 score for intent recognition. This correlation can be attributed to the phenomenon of error propagation: inaccuracies in intent recognition can lead to subsequent mistakes in intent matching, which in turn influence the final DQD results.