

# Labeling without Seeing?

## Blind Annotation for Privacy-Preserving Entity Resolution

**Yixiang Yao**

*Department of Computer Science  
University of Southern California*

*yixiangy@usc.edu*

**Weizhao Jin**

*Department of Computer Science  
University of Southern California*

*weizhaoj@usc.edu*

**Srivatsan Ravi**

*Department of Computer Science  
University of Southern California*

*srivatsr@usc.edu*

**Reviewed on OpenReview:** <https://openreview.net/forum?id=bAM8y3Hm0p>

### Abstract

The entity resolution problem requires finding pairs across datasets that belong to different owners but refer to the same entity in the real world. To train and evaluate solutions (either rule-based or machine-learning-based) to the entity resolution problem, generating a ground truth dataset with entity pairs or clusters is needed. However, such a data annotation process involves humans as domain oracles to review the plaintext data for all candidate record pairs from different parties, which inevitably infringes the privacy of data owners, especially in privacy-sensitive cases like medical records. To the best of our knowledge, there is no prior work on privacy-preserving ground truth labeling in the context of entity resolution. We propose a novel blind annotation protocol based on homomorphic encryption that allows domain oracles to collaboratively label ground truth without sharing data in plaintext with other parties. In addition, we design a domain-specific, user-friendly language that conceals the complex underlying homomorphic encryption circuits, making it more accessible and easier for users to adopt this technique. The empirical experiments indicate the feasibility of our privacy-preserving protocol (f-measure on average achieves more than 90% compared with the real ground truth).

## 1 Introduction

*Entity resolution* is the task of linking entity pairs across datasets that refer to the same entity in the real world. It is an essential problem in massive data integration and quality improvement, which has been widely employed in various domains including academia, industries, and government agencies (Christen, 2011; Winkler, 2014). Traditionally, it requires the participation of multiple owners of the data sources, and it is unavoidable that the data is shared among parties. As the example shows in Figure 1, datasets  $D_A$  and  $D_B$ , which belong to two different parties, consist of the IDs and names of the digital camera lens products but in different representations. Each lens is an entity and the goal is to link the same entities between these two datasets. To achieve that, the data owners of  $D_A$  and  $D_B$  share the data with each other and jointly determine that A1 from  $D_A$  and B1 from  $D_B$  could be a match since they are all “Canon 24-70mm f2.8” camera lenses though A1 explicitly notes the lens is with “USM” motor and B2 is the "II" generation. On the contrary, A2 and B1 are not a match since they have different focal ranges (“24-105” vs. “24-70”) and aperture sizes (“4” vs. “2.8”), as well as A3 and B2 since they belong to different brands (“Canon” vs. “Sony”). As real-world entity data usually contains sensitive personal information or is related to intellectual

property, *privacy-preserving entity resolution* (PPER) (Gkoulalas-Divanis et al., 2021; Yao et al., 2021; Ghai et al., 2023; Yao et al., 2024), as a solution, is to link entities with the protection for data privacy. The linking process does not reveal any unnecessary information and the parties will not learn any entities that are not common among all parties. Back to our example of lens products,  $D_A$ 's owner only knows A1 has a potential match B1 in  $D_B$  but does not learn anything about B2. Likewise,  $D_B$ 's owner also learns no other lens products from  $D_A$  except the match with A1.

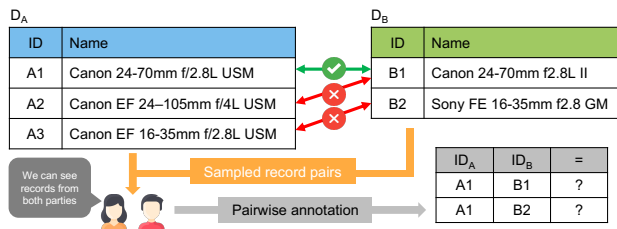


Figure 1: The entity resolution and annotation example.  $D_A$  and  $D_B$  are two lens product datasets owned by two parties separately. Making pairwise annotation requires the **visibility** of both parties’ sampled records, revealing private data.

In the lifecycle of entity resolution and PPER, a ground truth dataset with entity pairs or clusters needs to be prepared to evaluate the quality of linking or train learning-based entity resolution models. Constructing such a ground truth dataset, known as *annotation*, requires the involvement of domain oracles (data owners or trusted delegators)<sup>1</sup>. Normally, the first step is to sample some representative entities from all data sources and convert them into pair-wised forms. The oracles then manually determine if such a candidate “pair” is considered to be legitimate in the real world. For example, in Figure 1, assume that all records are sampled, oracles could recognize (A1,B1) is a match but (A1,B2) is not. Annotation tools (Ipeirotis, 2010; Neves & Leser, 2014) including Amazon Mechanical Turk (Buhrmester et al., 2016; Kang et al., 2014; Lease et al., 2013) are for constructing and curating ground truth datasets.

However, this is a human-in-the-loop process and the oracles need to see all the raw data of entities side-by-side as pairs in plain. It is worth pointing out that even though most of the privacy-aware entity resolution algorithms themselves are considered to be somewhat privacy-preserving, the ground truth labels they used, no matter what strategies are conducted for minimizing the size of the required labels, are still annotated in plaintext (Kurniawan & Mambo, 2022) and can not be identified as *privacy-preserving annotation*. For example, healthcare facilities aim to perform entity resolution on their confidential medical records, yet the process of creating accurate reference annotations still relies on conventional annotation tools that lack robust privacy guarantees.

Recently, several privacy-preserving annotation methods have been proposed. While some methods (Kajino et al., 2014; Zhang et al., 2021) employ perturbation for crowd-sourced annotations, others leverage existing AI-driven ground truth generation in specific tasks (Kavasidis et al., 2012; Wangt et al., 2001), they are limited to certain domains (D’Orazio et al., 2009; Jing et al., 2017) and almost implausible to have such techniques universally generalized for privacy-preserving entity resolution. Additionally, the differential privacy-based methods (Feyisetan et al., 2019) protect the Personal Identifiable Information (PII) of individual records, but the content remains in plaintext. Contrary to the relatively well-researched privacy-preserving entity resolution solutions for the application phase (Schnell et al., 2009; Scannapieco et al., 2007; Inan et al., 2008; 2010), there still exists a privacy protection void in the annotation phase for PPER. Making annotations for entity resolution privacy-preserving is challenging because the domain oracles must compare the data across parties *side-by-side* in order to determine if two records are the same. Moreover, the raw data can certainly be manipulated accordingly or the annotation process can be semi-auto for the purpose of privacy protection, but the *partial content* of candidate pairs is still inevitably revealed for oracles at least need to see some of the plaintext to understand the meaning and then make decisions.

<sup>1</sup>In practice, domain oracles are individuals who possess a deep understanding of datasets, enabling them to make high-quality annotations.

Different from the aforementioned approaches that are not suitable for pairwise privacy annotation or merely hide the individual identifier but not the content, we propose a novel “blind annotation” approach of ground truth labeling, based on homomorphic encryption (HE) (Fontaine & Galand, 2007; Acar et al., 2018), for privacy-preserving entity resolution where oracles from each party can only inspect the data of their own but are still able to work collaboratively. The conceptual model is as shown in Figure 2, where data owners “blindly” annotate the ground truth set, which can later be used in the model training and evaluation phases of privacy-preserving entity resolution. No data in the plain view of each party is shared with any other parties so the privacy of the data is guaranteed without loss of data utility.

It is important to clarify that this paper does **NOT** aim to address PPER itself. Instead, we propose a novel *privacy-preserving method for pairwise ground truth annotation*, which is applicable to any PPER pipeline. The relationship between PPER and the proposed privacy-preserving annotation method is illustrated in Figure 2. We summarize our main contributions as follows:

- We propose a novel homomorphic-encryption-based privacy-preserving protocol that allows multiple parties to collaboratively produce pairwise entity annotations without sharing their data with any other parties in plaintext. To the best of our knowledge, this is the first solution to discover the potential of *privacy-preserving ground truth annotation for pairwise entity annotations* in entity resolution.
- The proposed protocol allows the domain oracles to focus solely on the annotation logic, and the protocol handles the optimizations for the underlying homomorphic circuit. Each party involved in the protocol can perform the computation efficiently and independently. The privacy properties of the protocol have been rigorously proven.
- We implement the protocol and conduct empirical studies on heterogeneous real-world datasets to prove the feasibility of our blind annotation protocol.

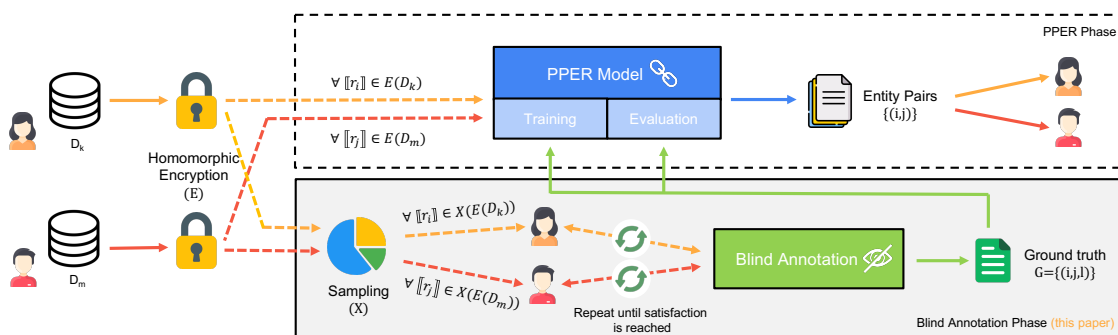


Figure 2: The relation between privacy-preserving entity resolution (upper white dashed box) and blind annotation (lower grey solid box). In this paper, we focus on the annotation phase and utilize “blind annotation” to label ground truth for PPER with *zero-knowledge* shared with other parties. The homomorphically encrypted records from each party are first sampled. Then, the data owners or their delegated oracles annotate sampled records *without* looking at the other party’s data but interacting with the blind annotation protocol. Finally, the ground truth as a set of triplets  $G = \{(i, j, l)\}$  ( $i$  and  $j$  are record ids from two parties and  $l$  is a label indicating if  $r_i$  and  $r_j$  is a match) is formed and can be used for training and evaluation in any PPER task. Note that no record content  $r_i$  or  $r_j$  in plaintext is revealed throughout the annotation process. The solid line denotes the data stream in plaintext and the dashed line denotes the data stream in ciphertext.  $D$  denotes dataset,  $X$  denotes sampling,  $E$  denotes homomorphic encryption, and  $\llbracket r \rrbracket$  denotes record  $r$  in ciphertext.

## 2 Problem Definition

The **privacy-preserving entity resolution** (PPER) problem can be defined as a triple  $T = (D, M, E)$  where  $D = \{D_1 \dots D_n\}$  is a set of  $n$  different datasets consisting of records  $r$ , from  $n$  different data owners  $P = \{P_1 \dots P_n\}$  respectively.  $E$  denotes encoding or encryption algorithm that keeps the  $r$  from each  $D$  in encoded or ciphertext form, that is,  $\llbracket r \rrbracket \in E(D)$ , where  $\llbracket r \rrbracket$  explicitly denotes  $r$  is in ciphertext.  $M$  is the match set containing record pairs between any two datasets amongst the  $n$  parties, so  $M = \{(\llbracket r_i \rrbracket, \llbracket r_j \rrbracket) \mid r_i = r_j; \llbracket r_i \rrbracket \in E(D_k), \llbracket r_j \rrbracket \in E(D_m)\}$ , where  $r_i = r_j$  indicates that  $r_i$  and  $r_j$  refer to the same entity in the real world, despite having different representations.

Finding the complete and precise  $M$  depends on the domain-specific algorithm/model which relies on the high-quality ground truth data for training, fine-tuning or evaluation. The ground truth set  $G$  is a set of triples  $(i, j, l)$  where  $i$  and  $j$  are record ids of a record pair  $r_i$  and  $r_j$ , and  $l$  is a Boolean label which indicates if such record pair is the same entity. The process of constructing  $G$  is called **annotation**. Formally,  $G = \{(i, j, l) \mid r_i \in X(D_k), r_j \in X(D_m)\}$ , where  $X$  is a sampling algorithm.  $l$  is determined by domain oracles with the content of  $r_i$  and  $r_j$ .

Usually,  $G$  is not constructed individually from each party: because the domain oracles have to see  $r_i$  and  $r_j$  in clear in order to determine whether two records are the same real-world entity based on the features the records have. Therefore, the owners of the records need to share sampled records in the plain with other parties which makes the raw content of  $r_i$  and  $r_j$  revealed.

To prevent the potential privacy leakage during this process, this paper focuses on creating ground truth data with  $\llbracket r_i \rrbracket$  and  $\llbracket r_j \rrbracket$  straight so that no plaintext from any parties reveals to any other parties. We name this process **blind annotation**, that is,  $G = \{(i, j, l) \mid \llbracket r_i \rrbracket \in X(E(D_k)), \llbracket r_j \rrbracket \in X(E(D_m))\}$ .

## 3 Preliminaries

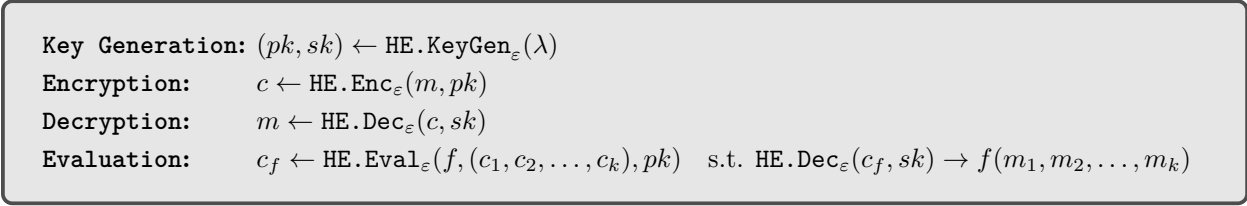
### 3.1 Related Works

**Privacy-preserving annotation** As stated in this privacy-preserving record linkage survey paper (Gkoulalas-Divanis et al., 2021), almost no open method is available for making ground truth annotation for PPER due to privacy concerns regardless of its a time-consuming and erroneous process.

The only similar scenario we found that explored the attempts for privacy-preserving annotation is about active learning using differential privacy (Feyisetan et al., 2019). The general active learning method aims to learn a distribution of the data by selecting less training data that carries the highest information with the active learner. The sampled data from the learner needs to be annotated by domain oracles or crowd workers. The problem is that crowdsourcing the labels is usually from an open call and transmitting non-public data to crowd workers has an inevitable privacy leakage risk, especially for the identities. For the insufficient privacy guarantee of  $k$ -anonymity, the authors adopted a differential privacy algorithm (Li et al., 2012), which prevents a user from sustaining additional damage by including their data in a dataset, on binary classification tasks, this method achieves similar accuracy scores as the non-privacy counterpart with a small performance hit but strong privacy guarantee. Though Personal Identifiable Information (PII) is protected from revealing, the content of the sampled data as plain text is still known by the crowd workers for them to understand and annotate. If the content of the non-public data itself requires to be kept private, such a method is not yet qualified.

Another method to prepare data for machine learning tasks is to learn the distribution of the original data and generate synthetic data based on that (El Emam, 2020). Even though this method might work for training statistical models and conducting analytical works, it is not feasible to be applied to entity resolution tasks. Because the representation of the token/word is an important signal to determine the similarities between records, a tiny modification or substitution of the original representation could cause huge judgment deviations for the annotators.

Yu et al. (2020) provided a method of not considering ground truth but instead using unsupervised heuristic measures based on a greedy matching approach to evaluate and optimize the hyper-parameters for entity

Figure 3: Operations of a typical public-key homomorphic encryption scheme  $\varepsilon$ 

resolution models. This method is based on the assumption that a match of record pairs gets the highest similarity score among a set of heuristic measures against other candidate pairs. Using heuristics to estimate linkage quality is doable in some certain scenarios, however, evaluating heuristics by heuristics, in general, is somewhat a “the chicken or the egg” problem. Additionally, this method only works in some general cases where the same entity looks similar: in some extreme conditions or some hard record linking problems, the representations of the record are different even when they are referring to the same entity.

In short, labeling ground truth in a privacy-preserving manner is tough and no direct work exists. Our approach achieves the goal without compromising privacy protection by employing homomorphic encryption.

**Comparison to PSI and PIR** Private Set Intersection (PSI) (Morales et al., 2023) or weighted PSI are general-purpose solutions for identifying common elements between sets. However, they have limited extensibility, particularly when it comes to supporting complex annotation logic. Specifically, PSI does not natively support conditional expressions or complex predicates, and incorporating such logic typically requires substantial additional effort. In contrast, the blind annotation protocol can express more sophisticated logical operations more naturally.

Moreover, PSI and weighted PSI can still be used within blind annotation protocol for computing set intersections. Although they are not currently available in our DSL, they can be incorporated as functions, provided that the PSI operation can be represented by HE circuits (Chen et al., 2017).

Private Information Retrieval (PIR) (Chor et al., 1998), on the other hand, is designed for retrieving data without disclosing which item is being accessed. While state-of-the-art PIR extensions support more expressive queries, such as keyword-based retrieval (Hao et al., 2025), they share the same limitation as PSI in that they do not easily accommodate complex logical operations. In comparison, the blind annotation protocol provides a more flexible foundation for such logic.

### 3.2 Homomorphic Encryption

*Homomorphic encryption* (HE) allows the computation to be performed over encrypted data while preserving the input/output relationship of the function between the plaintext and ciphertext data (Fontaine & Galand, 2007; Acar et al., 2018). In general, an encryption scheme is said to be homomorphic if for some operator  $\odot$  over plaintext ( $\odot_{\mathcal{M}}$ ) and ciphertext ( $\odot_{\mathcal{C}}$ ), the encryption function  $E$  satisfies:  $\forall m_1, m_2 \in \mathcal{M}, E(m_1 \odot_{\mathcal{M}} m_2) \leftarrow E(m_1) \odot_{\mathcal{C}} E(m_2)$ , where  $\mathcal{M}$  denotes the message in plaintext and  $\mathcal{C}$  denotes the message in ciphertext.  $\leftarrow$  declares the computation is direct without any decryption in the middle of the process. Therefore, if we let  $\odot$  to be  $+$ , computing  $m_1 + m_2$  can be done by a computation unit that receives the original message from the data owner in the encrypted form, that is,  $E(m_1)$  and  $E(m_2)$ , and is able to compute the addition of two encrypted messages without decrypting. Only the data owner with the key can decrypt the message from ciphertext back to plaintext.

Figure 3 demonstrates a typical public key homomorphic encryption scheme  $\varepsilon$  primarily characterized by four operations. *Key generation* takes in a security parameter  $\lambda$  and outputs public key pair  $(pk, sk)$ . *Encryption* encrypts plaintext message  $m$  with  $pk$  and returns ciphertext  $c$ . *Decryption*, as an inverse, decrypts ciphertext  $c$  with  $sk$  into plaintext  $m$ . *Evaluation* evaluates a function (represented as a circuit)  $f$  over a ciphertext tuple  $(c_1, c_2, \dots, c_k)$  with  $pk$  and returns  $c_f$ , which is equivalent to  $f(m_1, m_2, \dots, m_k)$  after decrypting.

The party that executes the homomorphic functions does not know anything about the data and the party that provides the encrypted data is unaware of what functions have been executed. This property, which is informally named *blind evaluation*, is the basis of our protocol.

## 4 Blind Annotation Protocol

We propose a privacy-preserving annotation protocol that allows the ground truth to be annotated among multiple parties “blindly” without inspecting other parties’ records.

**Intuition:** The intuition behind this is that without putting a candidate record pair side-by-side for domain oracles to determine if they are the same entity or not, the oracles on each side could extract the core features by just looking at the record content in plaintext of their own and summarize these features into a series of Boolean questions. If any record from the other parties satisfies all conditions defined by these questions, this record is highly likely to be a match of the record that the questions are derived from. Suppose the record content is encrypted, and these Boolean functions can be blindly evaluated over the ciphertext of the record. In that case, no plaintext of the record content is revealed to other parties.

For example, from  $P_A$ ’s perspective, determining if a  $P_B$ ’s record B1 is similar to its record A1 can be based on A1’s core features. If B1 also has keywords “Canon”, “24-70” and “2.8” as A1, it is highly-likely that it is a potential match to A1. The other keywords like “mm”, “USM” are not necessary for identifying the lens based on the oracle’s domain knowledge. Figuring out these features only needs  $P_A$  access to its own record A1. Moreover, if  $P_A$  encapsulates the feature detection functions for A1 to be homomorphic functions and  $P_B$  encrypts B1,  $P_A$  can label record pair (A1, B1) using homomorphic evaluation without knowing the content of B1.

**Protocol:** The protocol is elaborated in Figure 4. Succinctly, assume the ground truth construction is between party  $P_A$  and  $P_B$ , and party  $P_C$  is a coordinator for key management and result collection. In the *initialization* phase, party  $P_A$  and  $P_B$  first send their dataset size to  $P_C$ , and  $P_C$  randomly samples the record ids. Moreover,  $P_C$  also prepares the public key pair  $(pk, sk)$  for homomorphic encryption and sends the public key  $pk$  to  $P_A$  and  $P_B$ , and keeps the secret  $sk$  for decryption.

The next step is called *feature questions*, which is for data owners or domain oracles to annotate their records using homomorphically executable functions. Concretely,  $P_A$  and  $P_B$  sample records  $r_i$  and  $r_j$  respectively according to the sampled record ids from  $P_C$ , and prepare a set of Boolean logic-style questions  $Q_i$  and  $Q_j$  accordingly to the features of the record content for each record. Each question set  $Q$  is combined into a form that returns one Boolean result with first-order logic and converted to a homomorphically executable function.

Following that is *record encryption*, where the clear record data is homomorphically encrypted with  $pk$  in  $P_A$  and  $P_B$  respectively, and the ciphertext  $\llbracket r_i \rrbracket$  and  $\llbracket r_j \rrbracket$  are shared with another party.

The core step is *blind evaluation*, where the encrypted records from the other party are evaluated with prepared feature questions. Concretely, Once  $P_A$  receives a  $\llbracket r_j \rrbracket$  from  $P_B$ , it evaluates  $Q_i$  over it homomorphically, that is  $\text{HE.Eval}_\varepsilon(Q_i, \llbracket r_j \rrbracket, pk)$ , and gets  $\llbracket A_A^{ij} \rrbracket$ .  $P_B$  does a similar operation and gets  $\llbracket A_B^{ij} \rrbracket$ . Both  $\llbracket A_A^{ij} \rrbracket$  and  $\llbracket A_B^{ij} \rrbracket$  are in ciphertext and sent to  $P_C$ .  $P_C$  then tests if  $\llbracket A_A^{ij} \rrbracket$  equals  $\llbracket A_B^{ij} \rrbracket$  and decrypts the result with the secret key  $sk$ .

The last step is to determine the *end conditions*. If  $P_A$  and  $P_B$  agree on the result, the result will be stored. Otherwise, this pair will be picked out for the next round. In the next round,  $P_A$  and  $P_B$  require to refine their questions and evaluate with the encrypted record data from another party again. After  $t$  rounds, the pairs with no agreement will be discarded.

**Privacy:** From  $P_A$ ’s perspective, throughout the protocol,  $P_A$  has no access to  $P_B$ ’s record content  $r_j$  in plaintext but evaluates questions over ciphertext  $\llbracket r_j \rrbracket$ . The question functions  $Q_i$  are evaluated on  $P_A$ ’s side so  $P_B$  does not know anything about the features tested in  $Q_i$  by  $P_A$ . The same situation applies to  $P_B$ .  $P_C$ , which has the secret key  $sk$ , decrypts the final results which contain no record content in plaintext

from either  $P_A$  or  $P_B$ . None of the parties is able to apply ciphertext collision attack because homomorphic encryption is semantically secure (Section 4.3).

In the following subsections, we walk through the protocol details and dissect each essential component. We also provide the protocol security analysis in Appendix B.2. Note that parties operate independently throughout the process as long as the interactions follow the timeline in the Figure 4. Although Secure Multi-Party Computation (MPC) is a potential option for implementing the blind annotation protocol, *HE is preferred in practice due to its ability to support offline evaluation*. This is especially important because each party may perform annotation at different times, with varying time costs, and HE does not require all parties to remain online concurrently.

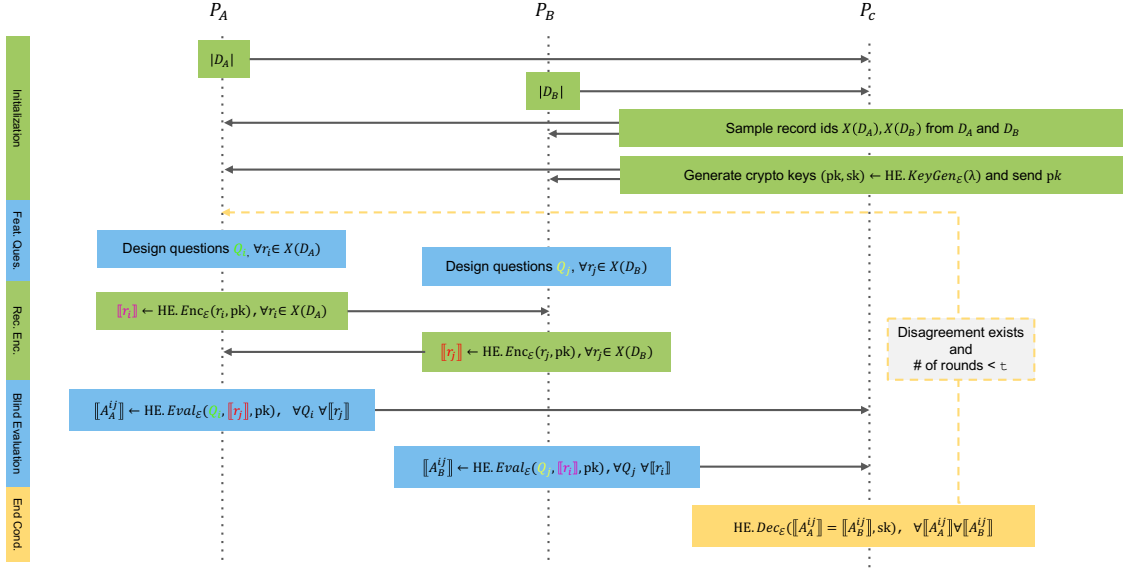


Figure 4: The timing diagram for the blind annotation protocol. The protocol, as well as five main components (initialization, feature questions, record encryption, blind evaluation, and end conditions) are dissected in Section 4.

#### 4.1 Initialization

Prior to getting involved in the annotation, a few initialization steps should be conducted first. The dataset owners ( $P_A$  and  $P_B$ ) need to first consent on the **annotation criteria** including how similar can two records be identified as the match and what kind of difference is tolerable. For example, consider the records “Canon 24-70mm f/2.8L USM” and “Canon 24-70mm f2.8L USM II” (A1 and B1 in Figure 1). Should these two lenses, which differ in generation, be treated as the same? In some contexts, lenses with generation I and II should be distinguished, while in others they may be categorized as the same item. The data used for deciding such criteria in this step can be synthetic or be derived from the record content so it does not leak any (sensitive) information about the original records.

A **data preprocessing** step is necessary for some of the datasets. If the dataset owners believe the format of their data has a noticeable distinction, they could apply a series of data cleaning and standardization operations, for example, removing the dataset-specific characters, lemmatization or stemming for natural language content, on their data individually.

$P_C$  is responsible for **sampling records** from  $P_A$  and  $P_B$ ’s dataset. After the dataset size  $|D_A|$  and  $|D_B|$  are sent to  $P_C$ ,  $P_C$  randomly samples some amount of record from  $D_A$  and  $D_B$  and sends back the selected record ids as a list back to the original data owners respectively.  $P_C$  then **generates the key pairs**  $(pk, sk) \leftarrow \text{HE.KeyGen}_\varepsilon(\lambda)$  with selected homomorphic encryption scheme  $\varepsilon$  and parameters  $\lambda$ , then

distributes the public key  $pk$  to both  $P_A$  and  $P_B$  for encrypting their data and questions. Note that  $P_C$  is the *only* party that can decrypt any ciphertext back to plaintext with access to the secret key  $sk$ .

Note that all steps, except for HE key generation, are also required in a standard *non-private annotation setup* and are essential for the final annotation quality. Our proposed method preserves these steps unchanged. Only after the annotation criteria have been defined and consistent data preprocessing has been applied can appropriate feature questions be formulated for the records.

## 4.2 Feature Questions

Feature questions are designed to test whether essential features of a record are satisfied and are annotated by domain oracles. This is based on the assumption that if all the core features of two records are the same, then they are highly likely to be the same entity. Specifically, the features of the records could be the *sub-strings* that are necessary for representing the record. For example, in a lens name “Canon 24-70mm f/2.8L USM II”, one feature can be the focal length “24-70” which can be used to construct a question that tests if “24-70” is in the compared lens. Therefore, using this question to evaluate a “15-85” lens returns false.

Under this assumption, the data owner can design a set of Boolean questions for all the features of a certain record. Utilizing first-order logic chains a set of questions together and the evaluation of a record turns it into a single Boolean value as the final result. For example, assume the question set is  $Q = \{Q_1, Q_2, \dots, Q_n\}$ , and it could be  $Q = Q_1 \vee \neg Q_2 \dots \wedge Q_n$  after conversion with first-order logic. Therefore,  $Q(r) = Q_1(r) \vee \neg Q_2(r) \dots \wedge Q_n(r)$  returns a Boolean value that implies whether the record  $r$  has all the desired features.

In our protocol, the feature question set  $Q$  is as  $f$  in homomorphic encryption that uses to evaluate the encrypted record  $\llbracket r \rrbracket$  with  $\text{HE.Eval}$ , that is,  $\text{HE.Eval}_\varepsilon(Q, \llbracket r \rrbracket, pk)$ , or simply  $Q(\llbracket r \rrbracket)$ . Constructing  $Q$  that is homomorphically executable is *non-trivial* due to the operator limitation and the hardness of the efficient encryption circuit construction (see details in Section 4.2.2).

Hence, on a high level, we design a simple annotation language for the ease of use in Section 4.2.1 and pre-defined some primitive functions that encapsulate the details of underlying homomorphic circuits in Section 4.2.2. With this layered design, *the domain oracles focus solely on the annotation logic*, and the program handles the tricks and optimizations for the sophisticated underlying homomorphic circuit construction.

### 4.2.1 Domain-specific Language

The annotation is designed to be written in a domain-specific language (DSL). This language provides general methods for data manipulation and logic computation, which are extensible for defining functions. The syntax of the DSL is in Appendix B.1.1.

With this DSL, it is efficient and sufficient to “ask questions”. Take the lens example, the record content from  $P_A$  is “Canon 24-70mm f/2.8L USM II”, the annotation from the oracle could be:

```

1 $r = lower($r)
2 $c1 = is_in("canon", $r) # condition 1
3 $c2 = is_in("24-70", $r) | is_in("2470", $r) # condition 2
4 $c3 = !is_in("24-105", $r) # condition 3
5 ret $c1 & $c2 & $c3

```

where target record  $\$r$  is first being lower-cased, and then requires the brand to be “Canon”, focal range to be “24-70” or simply “2470” as the common abbreviation but can not be “24-105”. Therefore, both “Canon 24-70 f2.8” and “Canon 2470” are considered matches, but “Canon 24-105mm USM” isn’t. Notice not all features but the important ones, that are consensus in the annotation criteria, are tested, e.g., the motor type “USM” or the version “II” are not being used for making the decision.



### 4.2.2 Homomorphic Functions

Writing homomorphic functions from scratch is not as straightforward as writing some normally "obvious" functions because of all the cumbersomeness and restrictions that homomorphic encryption schemes bring: (1) The homomorphic encryption scheme only supports a collection of limited operators. More complicated functions need to be built using these basic building blocks. (2) Simply porting the logic of a normal function to a homomorphic encryption function would suffer from privacy and/or efficiency issues. The function needs to revamp or refactor for it to work properly. (3) To achieve desired privacy protection (Appendix B.2), the encryption is end-to-end, so no decryption is allowed in the middle of the evaluation, which confines the use of control flow in programming.

To address these challenges and for the ease of writing effective and efficient functions along with the proper privacy protection, we pre-define several functions that are essential components for constructing Boolean questions  $Q$  and can be evaluated homomorphically without exposing the details of the underlying circuits. For the encryption schemes that only support logical operators, the arithmetic operators could be enriched by building upon the lower-level gate circuits, for example, constructing an 8-bit adder with AND/OR/NOT/XOR gates (Mano, 2017). On the other hand, the arithmetic schemes such as BGV can also be extended to support logical comparisons (Iliashenko & Zucca, 2021) (e.g., “<” and “=”) and that also retracts the benefit of efficient SIMD operations (Smart & Vercauteren, 2014) naturally come with these schemes.

It is worth mentioning that, among all the predefined functions, one function is `is_in`( $\llbracket a \rrbracket$ ,  $\llbracket b \rrbracket$ ) that tests if the encrypted string  $\llbracket a \rrbracket$  is a sub-string of the encrypted string  $\llbracket b \rrbracket$ . This is the core to measure if a certain feature  $\llbracket a \rrbracket$  exists in a record  $\llbracket b \rrbracket$ . In terms of extensibility, as long as a function needed can be encapsulated in a way that is able to compute homomorphically, it is safe to be added to the framework.

## 4.3 Record Encryption

To make the blind evaluation functions work, the records from both data owners require to be encoded with the same data encoding method. For the current case, since we only work on English characters, we simply apply ASCII encoding over the original records. Each character in a record is then represented as an 8-bit integer. All the sampled records  $r \in X(D)$  are encrypted with the public key  $pk$  that  $P_C$  generated in the initialization step.

The output ciphertext  $\llbracket r \rrbracket \leftarrow \text{HE.Enc}_\varepsilon(r, pk)$  is different even if the content of  $r$  is exactly the same due to the probabilistic encryption property, also known as semantic security, that homomorphic encryption carries (Yi et al., 2014). For instance, encrypting “Canon 24-70mm”  $n$  times results in  $n$  different ciphertexts. Therefore, no party is even able to test the exact match of the records by comparing the encrypted record ciphertext from other parties.

Finally, the encrypted  $\llbracket r \rrbracket \in X(E(D))$  is sent to another data owner for blind evaluation.

## 4.4 Blind Evaluation

From  $P_A$ 's perspective (the perspective for  $P_B$  is interchangeable), once it receives the encrypted  $X(E(D_B))$  from  $P_B$ ,  $\llbracket r_j \rrbracket \in X(E(D_B))$  is fed into each prepared function  $Q_i$  for blind evaluation. Computing  $Q_i$  with  $\llbracket r_j \rrbracket$  is a typical homomorphic evaluation process that  $P_A$  does not know anything about what the content is in  $\llbracket r_j \rrbracket$  but can still evaluate it as the plaintext version  $r_j$  with all the questions defined in  $Q_i$ . Because  $P_A$  does not know which  $\llbracket r_j \rrbracket$  potentially matches its own record  $r_i \in X(D_A)$ , this process needs to test all  $\llbracket r_j \rrbracket$ s against all  $Q_i$ s. Therefore, the total number of such process executes  $|D_A| \times |D_B|$  times for  $|Q_i| = |D_A|$ .

The output  $\llbracket A_A^{ij} \rrbracket \leftarrow \text{HE.Eval}_\varepsilon(Q_i, \llbracket r_j \rrbracket, pk)$  for each  $\llbracket r_j \rrbracket$  is also in ciphertext and since the secret key  $sk$  is only accessible by  $P_C$ ,  $P_A$  is not possible to get any information from  $X(E(D_B))$ . Note  $\llbracket A_A^{ij} \rrbracket$  associates with the record id pairs  $(i, j)$  so that  $P_C$  is able to identify the provenance of the result.

## 4.5 End Conditions

The encrypted results from  $P_A$  and  $P_B$  are collected and evaluated by  $P_C$ . On  $P_C$ 's side, it aligns  $\llbracket A_A^{ij} \rrbracket$  and  $\llbracket A_B^{ij} \rrbracket$  according to  $i$  and  $j$ , and tests if  $\llbracket A_A^{ij} \rrbracket = \llbracket A_B^{ij} \rrbracket$  meanwhile decrypts the result with secret key  $sk$ .  $P_C$  then has a mapping  $F$  with record id pairs  $(i, j)$  as keys and Boolean values indicating if two parties have made an agreement on the same record pairs as values. Formally,  $F = \{(i, j) \mapsto \text{HE.Dec}_\varepsilon(\llbracket A_A^{ij} \rrbracket = \llbracket A_B^{ij} \rrbracket, sk) \mid r_i \in X(D_A), r_j \in X(D_B)\}$ .

For each record pair  $(i, j)$  in  $F$ , if the value is true, the agreement has been established between the two data owners, so no additional process is needed. Otherwise, the record pair needs further investigation, and  $P_C$  extracts all  $i$ s and  $j$ s from disagreed pairs and returns them to the data owner respectively. The data owners have to conduct another round of annotation only for these records. The annotations for the later rounds tend to be not as strict as the former rounds so it increases the possibility of making  $\llbracket A_A^{ij} \rrbracket$  and  $\llbracket A_B^{ij} \rrbracket$  the same. Simultaneously,  $P_C$  maintains another list  $G_h$  that stores the annotation result from one of the parties (assuming  $P_A$  here), that is,  $G_h = \{(i, j, l) \mid h \in [1, t], l \leftarrow \text{HE.Dec}_\varepsilon(\llbracket A_A^{ij} \rrbracket, sk)\}$ , where  $l$  is the label,  $h$  is the round number and  $t$  is a parameter denoting the maximum number of rounds that should be decided between the data owners in the initialization step. When  $t$  rounds have finished, the protocol ends: the pairs whose value is true (consensus archived) in  $F$  are added to the final ground truth, and others are discarded. Therefore, the ground truth set  $G$  is constructed as  $G = \{(i, j, l) \mid (i, j, l) \in G_t, F(i, j) = \text{true}\}$ . Note that increasing  $t$  tends to improve performance, but it also raises the labeling cost. An empirical analysis of the effect of  $t$  is presented in Section 5.2.

Obviously, when all the values in  $F$  are true in the  $h$ -th round, it is by no means to recurse into the  $(h+1)$ -th round. The protocol meets the early termination condition and exits immediately.

## 5 Experiments

In this section, we conduct experiments to empirically evaluate the feasibility of using blind annotation to annotate datasets and the incurred overhead of homomorphic encryption. In general, domain oracles are asked to annotate datasets using blind annotation. The quality of the final annotation results is assessed by comparing them to the ground truth.

Dataset	Domain	Attributes	Original		Selected	
			#E	#M	#E	#M
Abt-Buy	E-commerce	<u>name</u> , description, manufacturer, price	1081+1092	1097	50+50	50
Amazon-Google	E-commerce	<u>name</u> , description, manufacturer, price	1363+3226	1300	49+50	50
DBLP-Scholar	Bibliographic	<u>title</u> , <u>authors</u> , <u>venue</u> , <u>year</u>	2616+64263	5347	49+50	50
DBLP-ACM	Bibliographic	<u>title</u> , <u>authors</u> , <u>venue</u> , <u>year</u>	2614+2294	2224	50+50	50
Febrl	Biomedical	<u>given name</u> , <u>surname</u> , <u>sex</u> , <u>age</u> , <u>title</u> , date of birth, <u>address 1</u> , address 2, <u>phone</u> , soc sec id, culture, family role	1500+3500	1074	50+50	50

Table 1: Dataset specifications and basic statistics. The underlined attributes are the attributes selected in the experiments. #E represents the number of entities, and #M denotes the number of matches.

### 5.1 Settings, Datasets and Metrics

#### 5.1.1 Settings.

To empirically study the feasibility of our blind annotation protocol in a practical sense, we implement a web-based user-friendly GUI and its corresponding HE program (see details in Appendix C.1). Specifically, the web GUI is implemented in Python, in which the DSL syntax is written in Extended Backus–Naur Form

(EBNF) and parsed by Lark library <sup>2</sup> with Look-Ahead Left-to-Right (LALR) parser. The syntax definition can easily be extended for new syntax, operators, and functions. The workflow is as follows: each domain oracle <sup>3</sup> annotates the owned dataset individually, and the annotation program merges and calculates the results. If the annotations do not satisfy the exit condition, the unqualified records are returned to the corresponding oracles for the next annotation.

The underlying HE program is implemented in OpenFHE (Al Badawi et al., 2022), an open-source project that efficiently and extensibly implements the post-quantum Fully Homomorphic Encryption schemes. We use the BinFHE module, a concrete implementation of FHEW/TFHE (Ducas & Micciancio, 2015; Chillotti et al., 2020). Specifically, we set it to work in public-key encryption mode with the crypto context to be STD128, which guarantees more than 128 bits of security for classical computer attacks. Additionally, we implement the program in two versions: serial and parallel. The latter employs OpenMP <sup>4</sup>, a multi-platform shared-memory parallel programming library, for accelerating independent gate operations to be executed in parallel. All the experiments are conducted on a Linux machine with an 8-core CPU @ 3.60 GHz and 32 GB RAM.

### 5.1.2 Datasets.

We use the real-world entity resolution benchmark (Köpcke et al., 2010), which includes 4 tasks and lies in both e-commerce and bibliographic domains. Each task consists of two datasets and a ground-truth file, which contains all the true matches. To mimic the labeling process, we extract a subset of records from each dataset, annotate them, and then compare the annotated pairs to the ground truth. Specifically, we first randomly sample 50 labeled matches from each provided ground truth, and this covers at most 5% (50 records) of each dataset because one record could link to multiple records. Note that 50 records from each dataset are around 2.5-5% of the original dataset except for Scholar. Even though the proportion of sampled records as ground truth seems small, it is **sufficient** for ER in most practical scenarios (Kasai et al., 2019). The specifications and the basic statistics of the dataset are listed in Table 1. We do not make any specific data cleaning and normalization except that all the Unicode characters are mapped into the ASCII range.

Generally, PPER are evaluated using ER’s benchmark datasets. However, an important application area is in healthcare, particularly with patient data. Unfortunately, these datasets often require permission for use or are restricted to specific purposes, making them less ideal for evaluating the effectiveness of the privacy-preserving methods. As an alternative, synthetic datasets are commonly used. One such synthetic dataset is Febrl (Freely Extensible Biomedical Record Linkage) (Christen, 2008), which is widely employed for generating census records containing fields such as name, sex, age, and address. These fields align with the structure of many patient datasets. Furthermore, as a synthetic dataset, it allows for customizable levels of noise. For our experiments, we generated the experiment dataset using the settings from Yao et al. (2021). The record content contains typos, ORC errors and phonetic features, and we summarize the statistics of missing value in Table 2.

given name	surname	sex	age	title	address 1	phone
1.8	1.3	21.2	19.6	57.5	5.7	5.6

Table 2: The percentage (%) of missing value for each attributes in the Febrl dataset

### 5.1.3 Evaluation Metrics.

We use precision, recall and F-measure as the standard evaluation metrics to measure the accuracy of blind annotation against the scores computed from ground truth labels. We additionally analyze other observed phenomena, including the relationship between the annotations rounds and the number of labeled pairs that come to an agreement. As for the homomorphic implementation, the communication and computation costs are evaluated.

<sup>2</sup><https://github.com/lark-parser/lark>

<sup>3</sup>Given that these tasks require only general knowledge, the oracles are individuals who understand the datasets.

<sup>4</sup><https://www.openmp.org/>

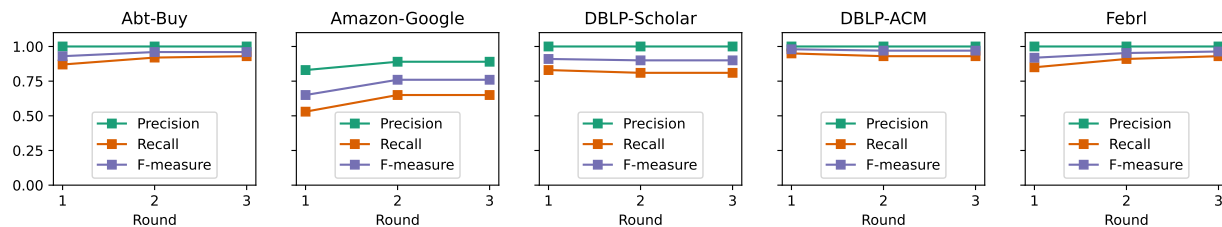


Figure 5: Performance evaluation regarding precision, recall and f-measure.

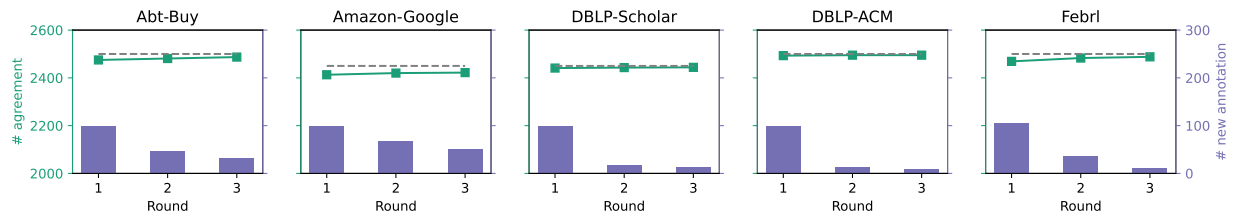


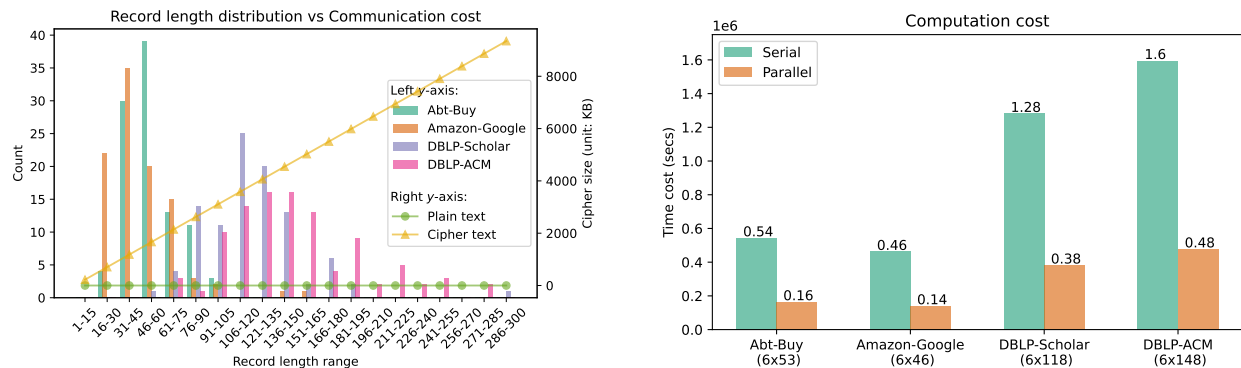
Figure 6: #round to meet agreement. The gray dashed line is the total number of candidate pairs, and the green solid line is the actual number of pairs that meet agreement. These two use the left y-axis as the scale. The purple bars at the bottom are the summation of the number of records to annotate from two parties. It uses the right y-axis as the scale.

## 5.2 Feasibility Verification

We try to validate two hypotheses in the following experiments to verify the feasibility of using blind annotation to construct the ground truth dataset for PPER:

- (H1) The feature extraction from only one side of the dataset without inspecting the other side and comparing candidate pair side-by-side is a feasible approach.
- (H2) More annotation rounds are effective for improving the quality of the PPER ground truth labeling.
- (H3) It is efficient for domain oracles to learn the DSL and perform annotation through the GUI.

**H1** From Figure 5, we observe the F-measure for Abt-Buy, DBLP-Scholar, and DBLP-ACM are all above 0.9, but Amazon-Google is not as good as others. The Amazon-Google dataset used in our experiment can be considered as hard problems for the record representations are full of abbreviations (e.g., software as sw), missing brands or models for products, and a limited amount of information (very short names). If better data pre-processing, especially normalization, is applied beforehand, a visible performance boost should be achieved. With more rounds of records annotated, precision and f-measure increase sharply for Abt-Buy and Amazon-Google, but not that noticeable for DBLP-Scholar and DBLP-ACM since both of them have already achieved high scores at the initial stage. Surprisingly, the recall values for DBLP-Scholar and DBLP-ACM drop slightly with more rounds executed, this is mainly due to the side-effect of the annotation strategy: when annotations from two parties do not come to an agreement, the oracles tend to make the annotations to be more generalized, that is, relaxing on the strictness of the matching criterion for easier capturing similarities, in the next round. Normally, having slack criterion will diminish precision but rise recall for more negatives become positives; however, another variable, the number of agreements between parties, is also increasing in blind annotation, which leads to the increment of both true positives and false negatives and the recall somewhat drops as a consequence. For Febrl, performance has a boost. It finally achieves promising results, largely due to the presence of highly identifiable signals, such as names, phone numbers, and keywords in addresses, in the census data.



(a) Communication cost in the context of the record length range. The left y-axis indicates the number of such records, and the right y-axis indicates the size of the corresponding plaintext and ciphertext.

(b) Computation cost in serial and parallel mode. The x-axis represents the length of the two records as  $\{\text{the average token length} \times \text{the average record length}\}$  in each dataset.

**H2** We further explore other effects that occur with the iteration of annotation rounds. As is shown in Figure 6, the green line, which indicates the number of agreements, approaches the gray line, which is the total number of candidate pairs, as the annotation round increases. On the other hand, the number of records that requires to annotate, shown as the bars, decreases. Specifically, for Abt-Buy and Amazon-Google, the annotation demands drop to 1/4 and 1/2 of the original amount respectively; for DBLP-Scholar and DBLP-ACM, in the second and the third round, this number is even less than 10%, because of the more explicit features the datasets have. For Febrl, the agreement increases notably across rounds, primarily due to the progressive relaxation of the criteria related to missing values.

**H3** The domain oracles became familiar with the DSL and GUI in around 30 minutes. For labeling, the average time per record was under 1 minute using the web-based GUI (implementation details are provided in Appendix C.1). The annotation process was fully offline for each party and executed in parallel.

In conclusion, the first hypothesis holds because all evaluation results lie in the acceptable range. With better cleaned and normalized datasets, the performance in terms of precision, recall and f-measure should be promising. The second hypothesis holds because precision, recall, and f-measure surge at the beginning with the increment of annotation rounds, and tend to be steady after a series of such rounds. The number of disagreements between the parties drops over the rounds before it hits the plateau. The third hypothesis holds as well, as both the learning and annotation costs remain within an acceptable range.

### 5.3 HE Overheads Analyses

Implementing blind annotation based on HE encryption scheme introduces various overheads. We hereby analyze them from the following two perspectives.

**Communication cost** We compare the size difference between plaintext and ciphertext with different lengths of input tokens among four datasets. In Figure 7a, the x-axis indicates the distribution of record length with a step of 15, the left y-axis indicates the number of records, and the right y-axis indicates the average size of serialized record data in a step in KB. It is clear to find out from the left y-axis that the distribution of the record length is in the range of 1 to 300, and the most common lengths are from 16 to 60 for Abt-Buy and Amazon-Google, and 91-160 for DBLP-Scholar and DBLP-ACM. This leads to the average record length for each dataset is 53, 46, 118, and 148. From the right y-axis, the increment of plaintext size, as a baseline, is nuance since each ASCII character only takes 1 byte (0.001KB) and 300 is only 0.3KB. As for the ciphertext, the trend of ciphertext size expands linearly, with an average consumption of 32KB per ASCII character after encryption and serialization. Although the size increment after encryption is significant, the total size of the data is still practical, and such communication is a one-time operation. For example, if the annotation is for 50 records, the average length is 100, then the total size of the encrypted data is 156.25MB.

**Computation cost** We evaluate the computation cost on 2500 comparisons (50 records from each party) with the average length of the token size (1st argument, see Appendix D.1) and the average record length (2nd argument) used in `is_in` function for each dataset: 6x53, 6x46, 6x118 and 6x148. We also compare the cost with ciphertext in serial and parallel. As is shown in Figure 7b, the incurred computation cost for ciphertext is about  $5.4e5$  seconds for 6x53,  $4.6e5$  seconds for 6x46,  $1.28e6$  for 6x118 and  $1.6e6$  seconds for 6x148 in serial mode. The parallel mode executes byte-level equal comparison with independent gate operations in parallel so that the time cost is significantly improved more than 3 times.

Two optimizations we consider doing in the future are: 1) The records from both parties are encrypted due to the limitation of BinFHE. For some other HE schemes (e.g., BGV/BFV) that allow the HE operations between a plaintext and a ciphertext, only the record from the other party needs to be encrypted so that more computation and communication costs can be further saved. 2) The parallelization is only optimized for the byte-level comparison. Since the 2500 pairwise record comparisons are mutually independent and can also be run simultaneously, the fully paralleled program should cost notably less time.

## 6 Conclusion and Future Work

In this work, we propose a blind annotation protocol based on homomorphic encryption for labeling ground truth that is tailored for privacy-preserving entity resolution. Unlike revamping the traditional annotation methods with de-identified records, this protocol explores the possibility of annotating without revealing any record in plaintext to other parties. The domain-specific language lowers the bar of the implementation in the real world and the comprehensive experiment results show the rationality of blind annotation. It can be anticipated that scaling efficiently to more complex datasets and real-world scenarios will become increasingly computationally feasible as HE implementations become more efficient.

In the future, because of the extensibility of DSL, more qualified homomorphic evaluation functions could be invented and integrated. Additionally, supporting the multi-attribute dataset is a practical enhancement that increases information utilization and improves linkage accuracy.

## References

- Abbas Acar, Hidayet Aksu, A Selcuk Uluagac, and Mauro Conti. A survey on homomorphic encryption schemes: Theory and implementation. *ACM Computing Surveys (Csur)*, 51(4):1–35, 2018.
- Ahmad Al Badawi, Jack Bates, Flavio Bergamaschi, David Bruce Cousins, Saroja Erabelli, Nicholas Genise, Shai Halevi, Hamish Hunt, Andrey Kim, Yongwoo Lee, et al. Openfhe: Open-source fully homomorphic encryption library. In *proceedings of the 10th workshop on encrypted computing & applied homomorphic cryptography*, pp. 53–63, 2022.
- Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. Evaluating 2-dnf formulas on ciphertexts. In *TCC*, volume 3378, pp. 325–341. Springer, 2005.
- Zvika Brakerski. Fully homomorphic encryption without modulus switching from classical gapsvp. In *Advances in Cryptology–CRYPTO 2012: 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19–23, 2012. Proceedings*, pp. 868–886. Springer, 2012.
- Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. Fully homomorphic encryption without bootstrapping. Cryptology ePrint Archive, Report 2011/277, 2011. <https://eprint.iacr.org/2011/277>.
- Michael Buhrmester, Tracy Kwang, and Samuel D Gosling. Amazon’s mechanical turk: A new source of inexpensive, yet high-quality data? 2016.
- Hao Chen, Kim Laine, and Peter Rindal. Fast private set intersection from homomorphic encryption. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1243–1255, 2017.
- Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. Tfhe: fast fully homomorphic encryption over the torus. *Journal of Cryptology*, 33(1):34–91, 2020.

- Benny Chor, Eyal Kushilevitz, Oded Goldreich, and Madhu Sudan. Private information retrieval. *Journal of the ACM (JACM)*, 45(6):965–981, 1998.
- Peter Christen. Febrl- an open source data cleaning, deduplication and record linkage system with a graphical user interface. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1065–1068, 2008.
- Peter Christen. A survey of indexing techniques for scalable record linkage and deduplication. *IEEE transactions on knowledge and data engineering*, 24(9):1537–1555, 2011.
- Scott D Constable and Steve Chapin. liboblivious: A c++ library for oblivious data structures and algorithms. 2018.
- Tiziana D’Orazio, Marco Leo, Nicola Mosca, Paolo Spagnolo, and Pier Luigi Mazzeo. A semi-automatic system for ground truth generation of soccer video sequences. In *2009 Sixth IEEE International Conference on Advanced Video and Signal Based Surveillance*, pp. 559–564. IEEE, 2009.
- Léo Ducas and Daniele Micciancio. Fhew: Bootstrapping homomorphic encryption in less than a second. In Elisabeth Oswald and Marc Fischlin (eds.), *Advances in Cryptology – EUROCRYPT 2015*, pp. 617–640, Berlin, Heidelberg, 2015. Springer Berlin Heidelberg. ISBN 978-3-662-46800-5.
- Khaled El Emam. Seven ways to evaluate the utility of synthetic data. *IEEE Security & Privacy*, 18(4): 56–59, 2020.
- Junfeng Fan and Frederik Vercauteren. Somewhat practical fully homomorphic encryption. *Cryptology ePrint Archive*, 2012.
- Oluwaseyi Feyisetan, Thomas Drake, Borja Balle, and Tom Diethe. Privacy-preserving active learning on sensitive data for user intent classification. In *PAL 2019*, 2019. URL <https://www.amazon.science/publications/privacy-preserving-active-learning-on-sensitive-data-for-user-intent-classification>.
- Caroline Fontaine and Fabien Galand. A survey of homomorphic encryption for nonspecialists. *EURASIP Journal on Information Security*, 2007:1–10, 2007.
- Craig Gentry. *A fully homomorphic encryption scheme*. Stanford university, 2009.
- Tanmay Ghai, Yixiang Yao, and Srivatsan Ravi. Lessons learned: Building a privacy-preserving entity resolution adaptation of ppjoin using end-to-end homomorphic encryption. In *2023 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, pp. 117–124. IEEE, 2023.
- Aris Gkoulalas-Divanis, Dinusha Vatsalan, Dimitrios Karapiperis, and Murat Kantarcioglu. Modern privacy-preserving record linkage techniques: an overview. *IEEE Transactions on Information Forensics and Security*, 16:4966–4987, 2021.
- Meng Hao, Weiran Liu, Liqiang Peng, Cong Zhang, Pengfei Wu, Lei Zhang, Hongwei Li, and Robert H. Deng. Practical keyword private information retrieval from key-to-index mappings. *Cryptology ePrint Archive*, Paper 2025/210, 2025. URL <https://eprint.iacr.org/2025/210>.
- Iliia Iliashenko and Vincent Zucca. Faster homomorphic comparison operations for bgv and bfv. *Proceedings on Privacy Enhancing Technologies*, 2021(3):246–264, 2021.
- Ali Inan, Murat Kantarcioglu, Elisa Bertino, and Monica Scannapieco. A hybrid approach to private record linkage. In *2008 IEEE 24th International Conference on Data Engineering*, pp. 496–505. IEEE, 2008.
- Ali Inan, Murat Kantarcioglu, Gabriel Ghinita, and Elisa Bertino. Private record matching using differential privacy. In *Proceedings of the 13th International Conference on Extending Database Technology*, pp. 123–134, 2010.
- Panagiotis G Ipeirotis. Demographics of mechanical turk. 2010.

- Baoyu Jing, Pengtao Xie, and Eric Xing. On the automatic generation of medical imaging reports. *arXiv preprint arXiv:1711.08195*, 2017.
- Yongsoo Song Jung Hee Cheon, Andrey Kim & Miran Kim. Homomorphic encryption for arithmetic of approximate numbers. 2017. URL [https://link.springer.com/chapter/10.1007%2F978-3-319-70694-8\\_15](https://link.springer.com/chapter/10.1007%2F978-3-319-70694-8_15).
- Hiroshi Kajino, Yukino Baba, and Hisashi Kashima. Instance-privacy preserving crowdsourcing. In *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, volume 2, pp. 96–103, 2014.
- Ruogu Kang, Stephanie Brown, Laura Dabbish, and Sara Kiesler. Privacy attitudes of mechanical turk workers and the us public. In *Symposium on Usable Privacy and Security (SOUPS)*, volume 4, pp. 1, 2014.
- Jungo Kasai, Kun Qian, Sairam Gurajada, Yunyao Li, and Lucian Popa. Low-resource deep entity resolution with transfer and active learning. *arXiv preprint arXiv:1906.08042*, 2019.
- Isaak Kavasidis, Simone Palazzo, Roberto Di Salvo, Daniela Giordano, and Concetto Spampinato. A semi-automatic tool for detection and tracking ground truth generation in videos. In *Proceedings of the 1st international workshop on visual interfaces for ground truth collection in computer vision applications*, pp. 1–5, 2012.
- Hanna Köpcke, Andreas Thor, and Erhard Rahm. Evaluation of entity resolution approaches on real-world match problems. *Proceedings of the VLDB Endowment*, 3(1-2):484–493, 2010.
- Hendra Kurniawan and Masahiro Mambo. Homomorphic encryption-based federated privacy preservation for deep active learning. *Entropy*, 24(11):1545, 2022.
- Matthew Lease, Jessica Hullman, Jeffrey Bigham, Michael Bernstein, Juho Kim, Walter Lasecki, Saeideh Bakhshi, Tanushree Mitra, and Robert Miller. Mechanical turk is not anonymous. *Available at SSRN 2228728*, 2013.
- Ninghui Li, Wahbeh Qardaji, and Dong Su. On sampling, anonymization, and differential privacy or, k-anonymization meets differential privacy. In *Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security*, pp. 32–33, 2012.
- M Morris Mano. *Digital logic and computer design*. Pearson Education India, 2017.
- Daniel Morales, Isaac Agudo, and Javier Lopez. Private set intersection: A systematic literature review. *Computer Science Review*, 49:100567, 2023.
- Michael Naehrig, Kristin Lauter, and Vinod Vaikuntanathan. Can homomorphic encryption be practical? In *Proceedings of the 3rd ACM workshop on Cloud computing security workshop*, pp. 113–124, 2011.
- Mariana Neves and Ulf Leser. A survey on annotation tools for the biomedical literature. *Briefings in bioinformatics*, 15(2):327–340, 2014.
- Olga Ohrimenko, Felix Schuster, Cédric Fournet, Aastha Mehta, Sebastian Nowozin, Kapil Vaswani, and Manuel Costa. Oblivious multi-party machine learning on trusted processors. In *25th {USENIX} Security Symposium ({USENIX} Security 16)*, pp. 619–636, 2016.
- Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Advances in Cryptology—EUROCRYPT’99: International Conference on the Theory and Application of Cryptographic Techniques Prague, Czech Republic, May 2–6, 1999 Proceedings 18*, pp. 223–238. Springer, 1999.
- Monica Scannapieco, Ilya Figotin, Elisa Bertino, and Ahmed K Elmagarmid. Privacy preserving schema and data matching. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, pp. 653–664, 2007.



- Rainer Schnell, Tobias Bachteler, and Jörg Reiher. Privacy-preserving record linkage using bloom filters. *BMC medical informatics and decision making*, 9(1):1–11, 2009.
- Nigel P Smart and Frederik Vercauteren. Fully homomorphic simd operations. *Designs, codes and cryptography*, 71:57–81, 2014.
- Yi-Fan Tseng, Chun-I Fan, Ting-Chuan Kung, Jheng-Jia Huang, and Hsin-Nan Kuo. Homomorphic encryption supporting logical operations. In *Proceedings of the 2017 International Conference on Telecommunications and Communication Engineering*, pp. 66–69, 2017.
- Yalin Wangt, Ihsin T Phillipst, and Robert Haralick. Automatic table ground truth generation and a background-analysis-based table structure extraction method. In *Proceedings of Sixth International Conference on Document Analysis and Recognition*, pp. 528–532. IEEE, 2001.
- William E Winkler. Matching and record linkage. *Wiley interdisciplinary reviews: Computational statistics*, 6(5):313–325, 2014.
- Yixiang Yao, Tanmay Ghai, Srivatsan Ravi, and Pedro Szekely. Ampere: A universal abstract machine for privacy-preserving entity resolution evaluation. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pp. 2394–2403, 2021.
- Yixiang Yao, Joseph Cecil, Praveen Angyan, Neil Bahroos, and Srivatsan Ravi. Feasibility of privacy-preserving entity resolution on confidential healthcare datasets using homomorphic encryption, 2024. URL <https://arxiv.org/abs/2405.18430>.
- Xun Yi, Russell Paulet, Elisa Bertino, Xun Yi, Russell Paulet, and Elisa Bertino. *Homomorphic encryption*. Springer, 2014.
- Joyce Yu, Jakub Nabaglo, Dinusha Vatsalan, Wilko Henecka, and Brian Thorne. Hyper-parameter optimization for privacy-preserving record linkage. In *ECML PKDD 2020 Workshops: Workshops of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD 2020): SoGood 2020, PDFL 2020, MLCS 2020, NFMCP 2020, DINA 2020, EDML 2020, XKDD 2020 and INRA 2020, Ghent, Belgium, September 14–18, 2020, Proceedings*, pp. 281–296. Springer, 2020.
- Xiaoyu Zhang, Xiaofeng Chen, Hongyang Yan, and Yang Xiang. Privacy-preserving and verifiable online crowdsourcing with worker updates. *Information Sciences*, 548:212–232, 2021.

## A Preliminaries

### A.1 Homomorphic Encryption

Homomorphic encryptions are widely used for privacy-preserving computation tasks. For example, users outsourcing their data to a cloud computing platform would have privacy concerns; with homomorphic encryption, the platform is able to perform necessary computations on the encrypted data directly (Naehrig et al., 2011).

Homomorphic encryption schemes are normally categorized into three types according to the survey (Acar et al., 2018): (1) *Partially Homomorphic Encryption* (PHE) allows only one type of operation with an unlimited number of times (i.e., no bound on the number of usages), e.g., Paillier (Paillier, 1999). (2) *Somewhat Homomorphic Encryption* (SWHE) allows some types of operations a limited number of times, e.g., Boneh-Goh-Nissim (BGN) (Boneh et al., 2005). (3) *Fully Homomorphic Encryption* (FHE) allows an unlimited number of operations for an unlimited number of times. Since Gentry realized the first FHE scheme (Gentry, 2009) based on the lattice, a lot of follow-up FHE schemes have been proposed to address the issue it has and optimized to be practical in real-world applications. Some schemes including Brakerski-Gentry-Vaikuntanathan (BGV) (Brakerski et al., 2011), Brakerski-Fan-Vercauteren (BFV) (Fan & Vercauteren, 2012; Brakerski, 2012) and Cheon-Kim-Kim-Song (CKKS) (Jung Hee Cheon, 2017) support arithmetic operations while some others (Tseng et al., 2017) including FHEW (Ducas & Micciancio, 2015) is capable for

logical operations. Recently, works are extending the pure arithmetic schemes to run logical comparators without losing advantages such as Single Instruction Multiple Data (SIMD) (Iliashenko & Zucca, 2021).

## B Blind Annotation Protocol

### B.1 Feature Functions

#### B.1.1 Domain-specific Language

Some of the features the DSL equips:

- The primitive data types are `string` which is wrapped in double quotes and `number`.
- Three logic operators are supported. Or operator: `a | b` returns true if either `a` or `b` is true. Otherwise, false. And operator: `a & b` returns true if both `a` and `b` are true. Otherwise, false. Not operator: `!a` returns true if `a` is false, or returns false if `a` is true.
- Variables can be defined by `$v={exp}` where `$` is the variable indicator, `v` is the variable name and `{exp}` is a valid expression.
- A preset variable `$r` is act as the target record for comparison and should be used in the annotation as the argument of the question `Q`.
- At least one return statement `ret` is required. The argument of the return should be in Boolean. If multiple is provided, the code terminates when the first return statement is found.
- Round bracket pair `()` is for prioritizing the execution of specified expressions.
- Comment starts with `#`, e.g., `# this is a comment`.
- Whitespaces and empty lines are ignored.

#### B.1.2 Homomorphic Functions

Besides the challenges mentioned in Section 4.2.2 for constructing pre-defined functions, we still need to tackle one more issue that these functions rely on. As (Yao et al., 2021) pointed out, the control flow including choice (e.g. `if`) and loop (e.g. `for`, `while`) based flows require conditional expressions, which are in Boolean, for execution. These encrypted Boolean values cannot be used in conditional operations unless they are decrypted. However, the decryption of the sensitive encrypted value causes the exposing of the execution path which is not allowed in our privacy-preserving settings. The solution is to import **ternary operation** that converts to logic operation to be arithmetic operation, hence bypassing the decryption of the encrypted Boolean condition. The ternary operator is defined as `cond: a ? b` where `cond` is the encrypted Boolean condition and `a` or `b` are encrypted integer values. To return `a` when `cond` is true or to return `b` when `cond` is false without decrypting `cond`, we exploit oblivious-style ternary operators (Ohrimenko et al., 2016; Constable & Chapin, 2018) and implement `choose([cond], [a], [b])` with the arithmetic operators as `[cond] * ([a] - [b]) + [b]` where `[cond]`, `[a]` and `[b]` are all ciphertexts, and the return of `choose`, which is either `[a]` or `[b]`, is also a ciphertext.

With the ternary operator, we can define more sophisticated functions. Algorithm 1 lists some commonly-used functions and the implementation details. All these functions are evaluated and returned in ciphertexts. `lower([s])` (line 1-6) and `upper([s])` (line 7-12) convert the ASCII character to all lower-case or upper-case for standardization. These are not by default applied because, in some circumstances, letter case is an important signal for identifying record similarity. Both methods take in the encrypted string `s` and loop the characters in it one by one. For each character, it determines if it falls in a certain range of the ASCII table, and uses that as the condition to choose whether to modify the value or not. If the homomorphic scheme supports SIMD operation, calculating `cond` and `choose` can be run as a batch in one operation so the for-loop is saved. We also provide `is_in([a], [b])` function (line 13-22) which detects if `[a]` is a sub-string

**Algorithm 1: Commonly-used Functions**

**Input & return:** The input  $\llbracket s \rrbracket$ ,  $\llbracket a \rrbracket$ ,  $\llbracket b \rrbracket$  are all homomorphically encrypted. The return of all functions is also in ciphertext.  $len$  returns the encrypted string length in plaintext.

```

1 Function lower( $\llbracket s \rrbracket$ : string):
2   for  $i \leftarrow 1$  to  $len(\llbracket s \rrbracket)$  do
3      $\llbracket c \rrbracket \leftarrow \llbracket s[i] \rrbracket$ ;
4      $\llbracket cond \rrbracket \leftarrow (\llbracket c \rrbracket > 0x40) \wedge (\llbracket c \rrbracket < 0x5b)$ ;
5      $\llbracket s[i] \rrbracket \leftarrow \text{choose}(\llbracket cond \rrbracket, \llbracket c \rrbracket + 0x20, \llbracket c \rrbracket)$ ;
6   return  $\llbracket s \rrbracket$ ;

7 Function upper( $\llbracket s \rrbracket$ : string):
8   for  $i \leftarrow 1$  to  $len(\llbracket s \rrbracket)$  do
9      $\llbracket c \rrbracket \leftarrow \llbracket s[i] \rrbracket$ ;
10     $\llbracket cond \rrbracket \leftarrow (\llbracket c \rrbracket > 0x60) \wedge (\llbracket c \rrbracket < 0x7b)$ ;
11     $\llbracket s[i] \rrbracket \leftarrow \text{choose}(\llbracket cond \rrbracket, \llbracket c \rrbracket - 0x20, \llbracket c \rrbracket)$ ;
12  return  $\llbracket s \rrbracket$ ;

13 Function is_in( $\llbracket a \rrbracket$ : string,  $\llbracket b \rrbracket$ : string):
14    $l_a \leftarrow len(\llbracket a \rrbracket)$ ;
15    $l_b \leftarrow len(\llbracket b \rrbracket)$ ;
16    $\llbracket res \rrbracket \leftarrow \llbracket \text{False} \rrbracket$ ;
17   for  $j \leftarrow 1$  to  $l_b - l_a + 1$  do
18      $\llbracket r \rrbracket \leftarrow \llbracket \text{True} \rrbracket$ ;
19     for  $i \leftarrow 1$  to  $l_a$  do
20        $\llbracket r \rrbracket \leftarrow \llbracket r \rrbracket \wedge (\llbracket a[i] \rrbracket = \llbracket b[j+i] \rrbracket)$ ;
21      $\llbracket res \rrbracket \leftarrow \llbracket res \rrbracket \vee \llbracket r \rrbracket$ ;
22  return  $\llbracket res \rrbracket$ ;

```

of  $\llbracket b \rrbracket$  by scanning the existence of  $\llbracket a \rrbracket$  from  $\llbracket b \rrbracket$ 's left to right. Using this  $O(n^2)$  naive method without the early exit, even if a sub-string match exists, is not ideal but unavoidable because of preventing the disclosure of the execution path for privacy, as stated above.

## B.2 Security Analysis

### B.2.1 Security Definitions

We model  $\mathcal{A}$  as a probabilistic polynomial-time machine and the parties as interactive Turing machines.

**Definition 1** (Adversary). *An honest-but-curious adversary  $\mathcal{A}$  is capable of corrupting a subset of parties in the system, both not the server and one of the data owners at the same time. A compromised party will divert to  $\mathcal{A}$  all the received messages and act as  $\mathcal{A}$  requests.*

**Ideal World.** Our ideal-world functionality  $\mathcal{F}$  interacts with annotation parties as follows:

1. Each data owner sends  $\mathcal{F}$  its plaintext data  $r \in D$  and plaintext feature questions  $Q$ .  $\mathcal{F}$  processes and checks Boolean result  $w$  as  $Q_i(r_j) = Q_j(r_i), \forall Q \forall r$ .
2. If  $w$  is true, then add the records from both sides to the ground truth.
3. If  $w$  is false, repeat Step (1) until an agreement is reached or it exceeds a set trial threshold.

**Real World.** In the real world,  $\mathcal{F}$  is replaced and realized by our protocol described in the previous parts of this section.

**Definition 2** (Security). *A blind annotation protocol is simulation secure if for every adversary  $\mathcal{A}$  in the real world, there exists a security simulator  $\mathcal{S}$  in the ideal world that also corrupts the same set of parties and produces an output identically distributed to  $\mathcal{A}$ 's output in the real world.*

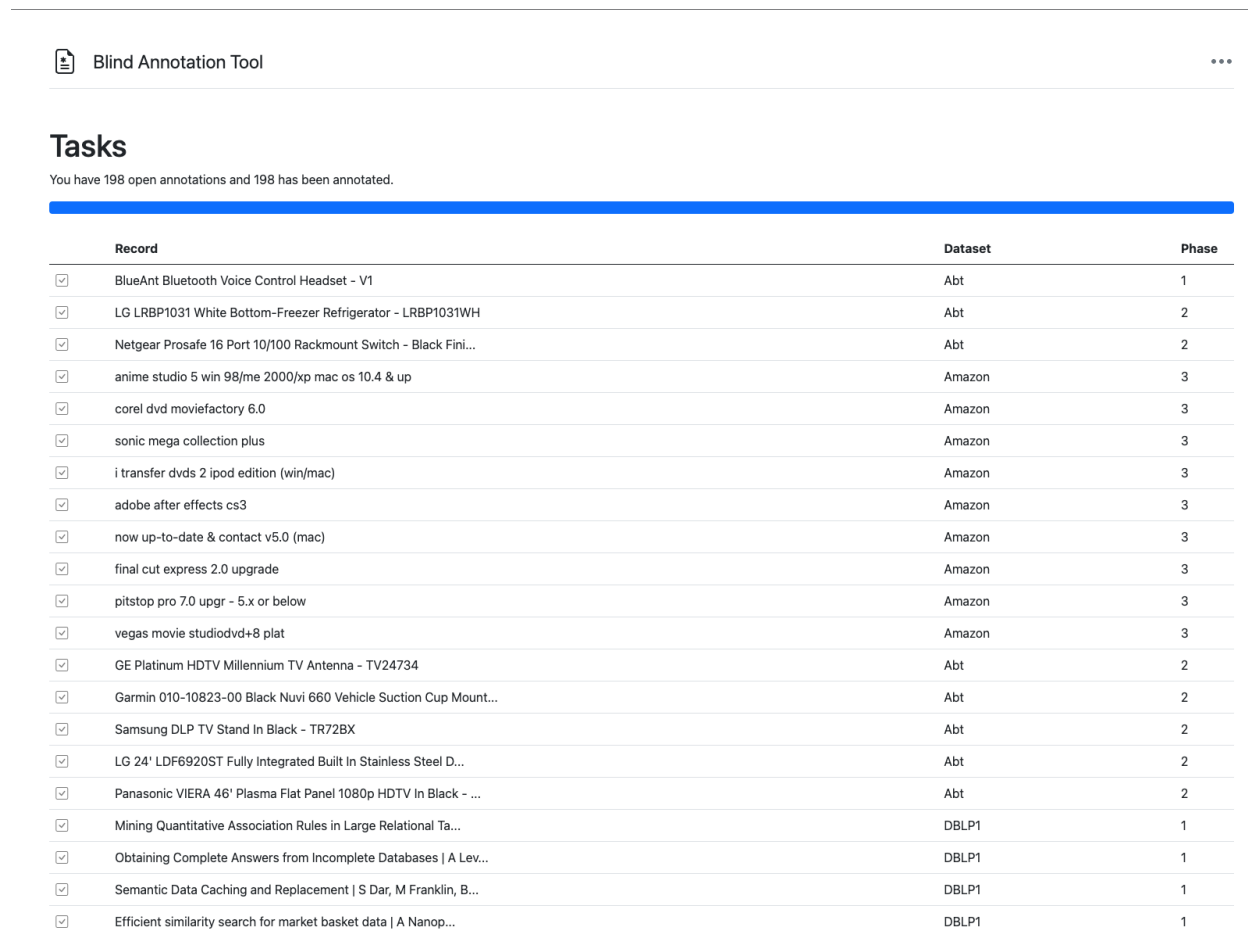
## B.2.2 Security Simulation

We describe a security simulator  $\mathcal{S}$  that simulates the view of the  $\mathcal{A}$  in the real-world execution of our protocol. Our security Definition 2 and  $\mathcal{S}$  ensure both confidentiality and correctness.  $\mathcal{S}$  receives from  $\mathcal{F}$ ,  $\mathcal{F}(C, x)$ , where  $C$  is computing circuits.  $\mathcal{S}$  sends  $\mathcal{F}(C)$  to  $\mathcal{S}$  and obtains fake Homomoprhic Encryption circuits  $HE_{fake}$ .  $\mathcal{S}_{HE}$  generates a random string  $o_{fake}$  of the same length as output.  $\mathcal{S}$  sends  $(HE_{fake}, o_{fake})$  to  $\mathcal{A}$ . As HE circuits distribution is independent,  $HE_{fake}$  is computationally indistinguishable from the real HE circuits  $HE$  in the real execution. The random output  $o_{fake}$  in ideal execution is indistinguishable from  $o$  in the real execution. In the ideal world,  $\mathcal{S}$  creates fake circuits  $HE_{fake}$  and does not use  $x$  for computing. Otherwise,  $\mathcal{A}$  could use  $x$  to evaluate the circuit, which would allow  $\mathcal{A}$  to distinguish between real and ideal executions.

## C Implementation

### C.1 User Interface

Figure 8 shows the main page of the web-based GUI which contains the current progress on the top of the page and the records that need to be annotated in the table. For each row of the table, it has an indicator of if the record is annotated or not, a brief of the record content, dataset name, and the current round of annotation for the record.



Record	Dataset	Phase
<input checked="" type="checkbox"/> BlueAnt Bluetooth Voice Control Headset - V1	Abt	1
<input checked="" type="checkbox"/> LG LRBP1031 White Bottom-Freezer Refrigerator - LRBP1031WH	Abt	2
<input checked="" type="checkbox"/> Netgear Prosafe 16 Port 10/100 Rackmount Switch - Black Fini...	Abt	2
<input checked="" type="checkbox"/> anime studio 5 win 98/me 2000/xp mac os 10.4 & up	Amazon	3
<input checked="" type="checkbox"/> corel dvd moviefactory 6.0	Amazon	3
<input checked="" type="checkbox"/> sonic mega collection plus	Amazon	3
<input checked="" type="checkbox"/> i transfer dvds 2 ipod edition (win/mac)	Amazon	3
<input checked="" type="checkbox"/> adobe after effects cs3	Amazon	3
<input checked="" type="checkbox"/> now up-to-date & contact v5.0 (mac)	Amazon	3
<input checked="" type="checkbox"/> final cut express 2.0 upgrade	Amazon	3
<input checked="" type="checkbox"/> pitstop pro 7.0 upgr - 5.x or below	Amazon	3
<input checked="" type="checkbox"/> vegas movie studiodvd+8 plat	Amazon	3
<input checked="" type="checkbox"/> GE Platinum HDTV Millennium TV Antenna - TV24734	Abt	2
<input checked="" type="checkbox"/> Garmin 010-10823-00 Black Nuvi 660 Vehicle Suction Cup Mount...	Abt	2
<input checked="" type="checkbox"/> Samsung DLP TV Stand In Black - TR72BX	Abt	2
<input checked="" type="checkbox"/> LG 24" LDF6920ST Fully Integrated Built In Stainless Steel D...	Abt	2
<input checked="" type="checkbox"/> Panasonic VIERA 46" Plasma Flat Panel 1080p HDTV In Black - ...	Abt	2
<input checked="" type="checkbox"/> Mining Quantitative Association Rules in Large Relational Ta...	DBLP1	1
<input checked="" type="checkbox"/> Obtaining Complete Answers from Incomplete Databases   A Lev...	DBLP1	1
<input checked="" type="checkbox"/> Semantic Data Caching and Replacement   S Dar, M Franklin, B...	DBLP1	1
<input checked="" type="checkbox"/> Efficient similarity search for market basket data   A Nanop...	DBLP1	1

Figure 8: The main page of the web-based GUI

When clicking on the record, an annotation window pops up (Figure 9): It shows the record content from the dataset that belongs to domain oracles' side, a record placeholder  $\$r$  which represents the record from the

other data owner side, an annotation editor with DSL code highlighting, and three buttons to quickly fill the editor with an auto-generation heuristic, discard the annotation, and save the annotation respectively. The “save” operation also runs the syntax check through the DSL parser to make sure the input is syntactically valid.

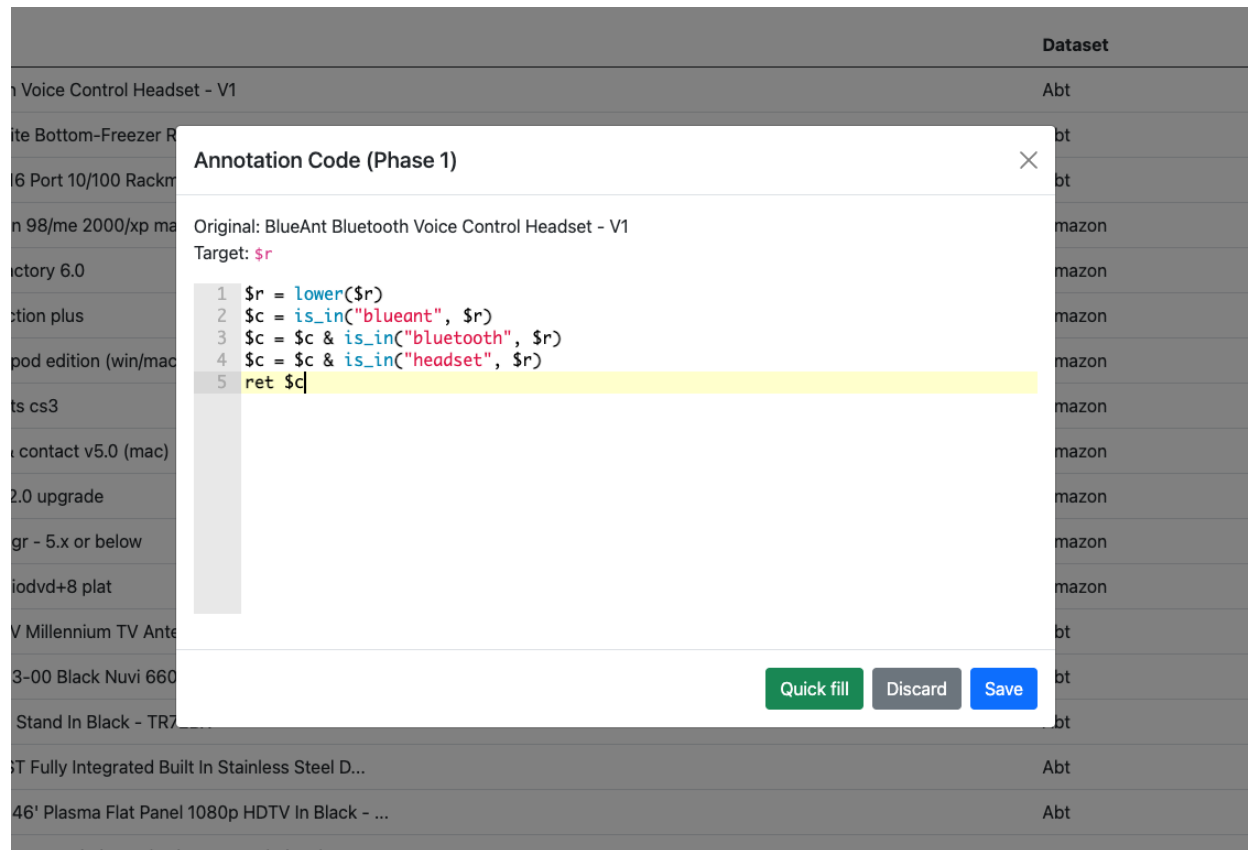


Figure 9: The first round record annotation

When annotating records after the first round, the annotation popup window also shows the previous annotation and allows domain oracles to quickly fill the editor with the previous content (Figure 10).

## C.2 Details of the HE implementation

Each character in a record is ASCII encoded (1 byte), and the Unicode character is converted to the corresponding ASCII character. Therefore, a string is represented by a two-dimensional matrix, where each element in the first dimension represents a character, and each dimension in the second dimension represents 1 bit. For example, a record "Canon" is represented by a 5x8 matrix where the first row is 01000011 as the binary representation for "C".

Naturally, the encryption is on the bit level since the BinFHE module in OpenFHE defines the operations for bits. Hence each item in the matrix is homomorphically encrypted. The logical operators we employed for implementations from BinFHE are AND ( $\wedge$ ), OR ( $\vee$ ), and XOR ( $\oplus$ ). The detail of the implementation is demonstrated in Algorithm 2. Specifically, `byte_equal` is the helper function for character-level comparison, which compares each corresponding bit from two encrypted characters  $[[a]]$  and  $[[b]]$ . `is_in` is the serial version of the function for comparing if the encrypted string  $[[a]]$  is a sub-string of the encrypted string  $[[b]]$ . The parallel version utilizes OpenMP and executes character comparison using `byte_equal` in parallel.

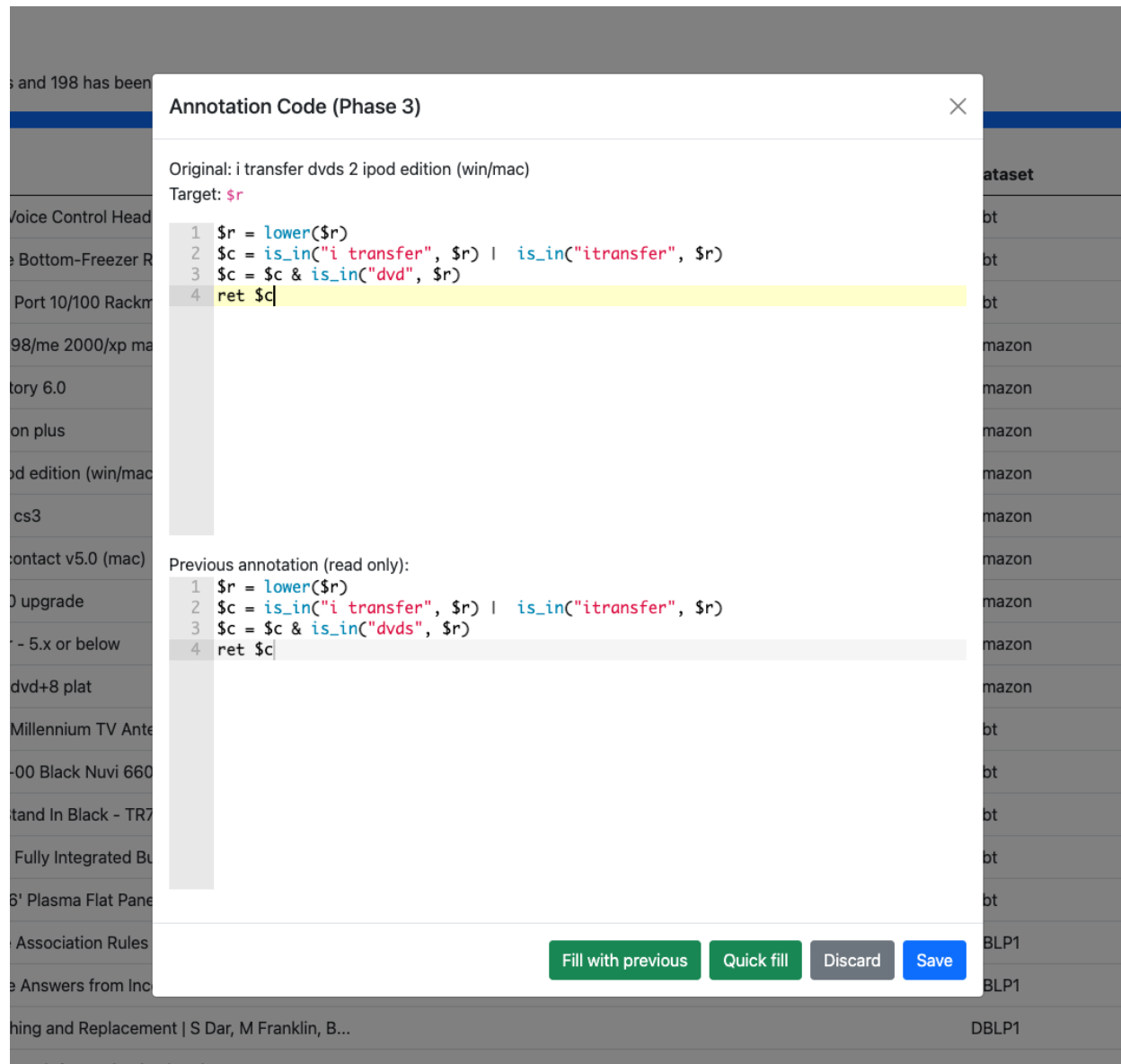


Figure 10: The follow-up round record annotation

## D Experiments

As the core function for feature extractions in annotation, it is essential to understand the first argument, that is, the input token, in the `is_in` function.

### D.1 Feature Length

The length distribution of the tokens is demonstrated in Figure 11. The length ranges from 1 to 16, and most of the tokens have lengths of 2 to 9. The average length of the token is around 6 (6.65, 6.28, 5.60, 6.61 for Abt-Buy, Amazon-Google, DBLP-Scholar and DBLP-ACM, respectively).

**Algorithm 2:** Functions using BinFHE scheme

---

```

1 Function byte_equal( $\llbracket a \rrbracket$ : bit[8],  $\llbracket b \rrbracket$ : bit[8]):
2    $\llbracket res \rrbracket \leftarrow \llbracket \text{True} \rrbracket$ ;
3   for  $j \leftarrow 1$  to 8 do
4      $\llbracket res \rrbracket = \llbracket res \rrbracket \wedge (\llbracket a[j] \rrbracket \oplus \llbracket b[j] \rrbracket)$ ;
5   return  $\llbracket res \rrbracket$ ;
6 Function is_in( $\llbracket a \rrbracket$ : [bit[8],...],  $\llbracket b \rrbracket$ : [bit[8],...]):
7    $l_a \leftarrow \text{len}(\llbracket a \rrbracket)$ ;
8    $l_b \leftarrow \text{len}(\llbracket b \rrbracket)$ ;
9    $\llbracket res \rrbracket \leftarrow \llbracket \text{False} \rrbracket$ ;
10  for  $j \leftarrow 1$  to  $l_b - l_a + 1$  do
11     $\llbracket r \rrbracket \leftarrow \llbracket \text{True} \rrbracket$ ;
12    for  $i \leftarrow 1$  to  $l_a$  do
13       $\llbracket r \rrbracket \leftarrow \llbracket r \rrbracket \wedge \text{byte\_equal}(\llbracket a[i] \rrbracket, \llbracket b[j+i] \rrbracket)$ ;
14     $\llbracket res \rrbracket \leftarrow \llbracket res \rrbracket \vee \llbracket r \rrbracket$ ;
15  return  $\llbracket res \rrbracket$ ;

```

---

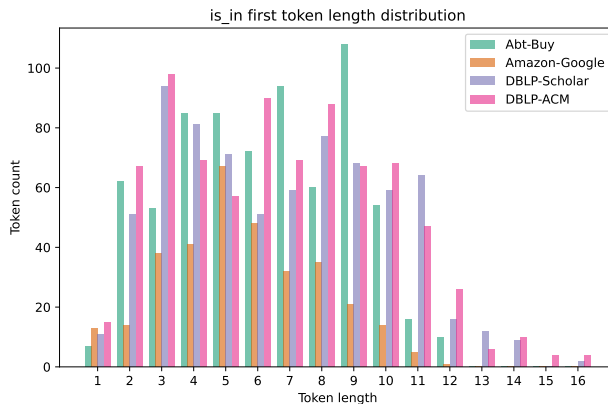


Figure 11: Token length distribution

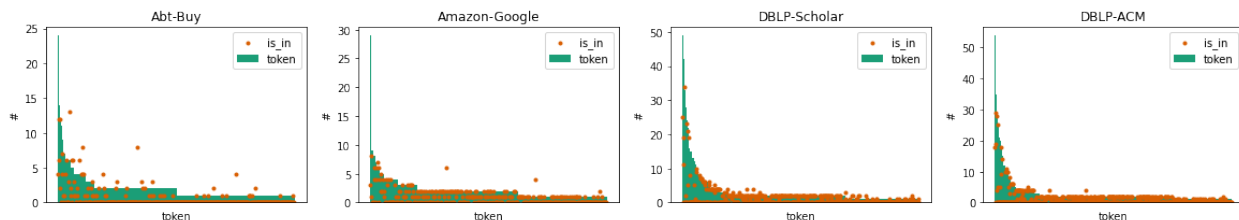


Figure 12: #tokens appears in the dataset and #is\_in uses to extract the corresponding tokens from all annotators. The modified tokens that are not in the original token list are discarded. Additionally, token - is removed from Abt-Buy and Amazon-Google for better visualization.

## D.2 Feature Importance

We are curious about if the important features are more likely to be extracted by the annotators. We tokenize the record based on white space without any additional processing steps. Meanwhile, we count the number of times that function is\_in is being utilized to extract the corresponding tokens. Note that annotators could modify the tokens accordingly, e.g., “photoshop” to be “ps”, and “international” to be “i18n”, we only

use the original token to construct the x-axis, thus none of these non-original tokens are counted if they do not appear in the original token list.

The distribution of tokens and `is_in` usage is summarized in Figure 12. The tokens are present in the long-tail form, whereas the dense invocations of `is_in` function concentrate on its “tail”. This observation shows the common tokens contain relatively less information than the rare ones so that the feature extraction depends more on the latter. Some common tokens also attract a fair amount of tokens, these are usually the common but special tokens for records, for instance, the brand name of a product. Some `is_in` functions are called a significant amount of times more than the token itself, they derive from the modification of the original token.

## E Case Study

We selected two cases from two different domains. For both of them, two parties hold different opinions at first but establish an agreement finally.

The first case in Figure 13 is product records from Amazon-Google dataset. On the Amazon dataset side, the given record content is short. The two attempts of the annotation are identical since not too much useful information can be used. On the Google dataset side, besides the brand and product name, it also has the category “music production software”. The first annotation has relatively strict rules, whereas the second annotation only tests the brand and the product name.

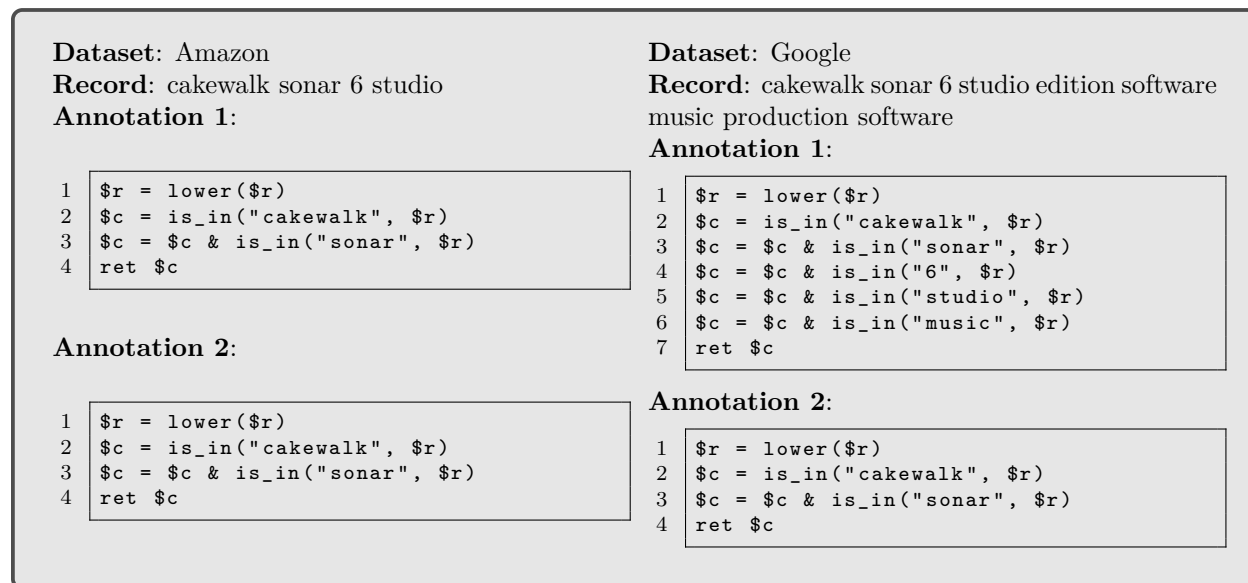


Figure 13: A case in Amazon-Google

The second case is from DBLP-ACM dataset and shown in Figure 14. The record content from ACM mentioned “database”, but in DBLP, it is represented as “DBMS”. One possible solution is that annotators could expand “DBMS” to be “database” with their domain knowledge. However, the record content is about the paper title and it usually remains consistent across different websites, so the annotators do not modify it on the token level. Wisely, the annotation from the DBLP’s side splits the paper title into two parts, the first part before “:” is as the first attempt, and the rest is as the second attempt.



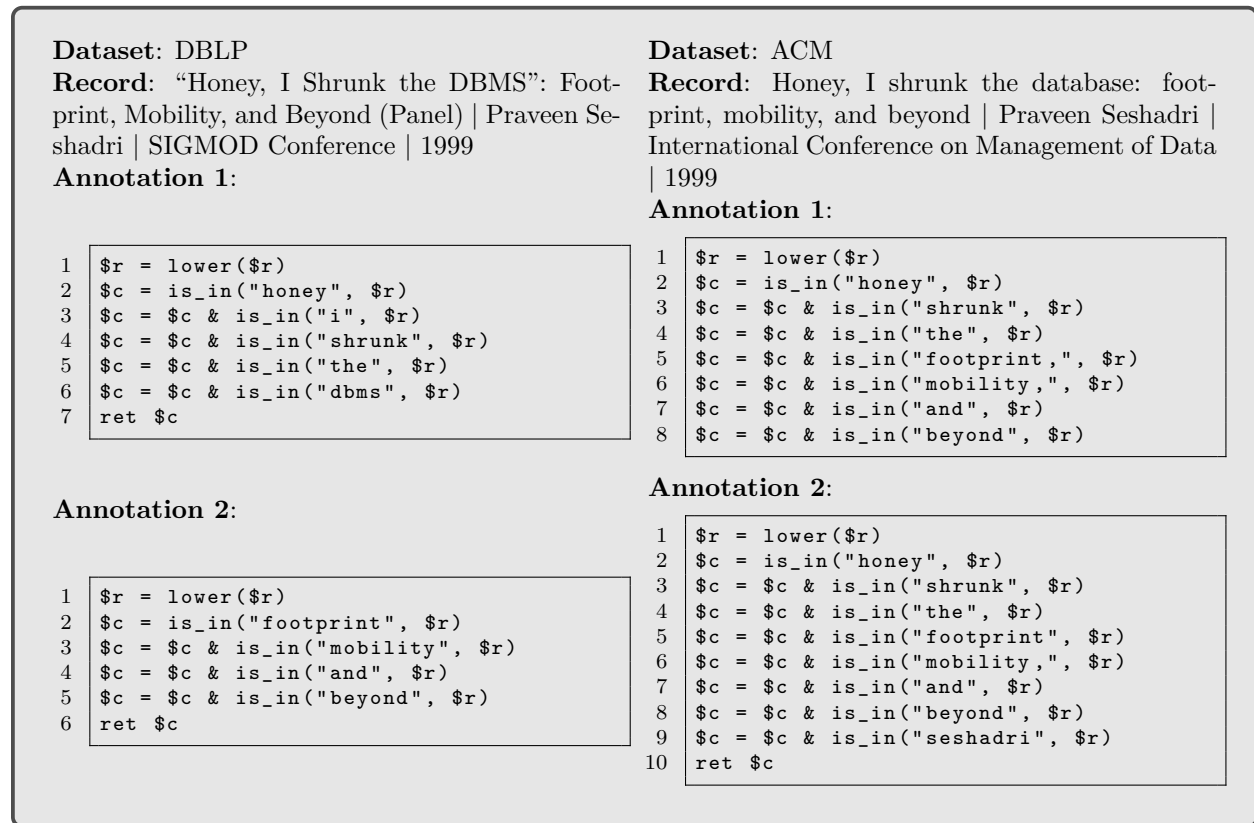


Figure 14: A case in DBLP-ACM