

Information Extraction on the Web with Credibility Guarantee

Nguyen Thanh Tam

*Distribute Information Systems Laboratory
École Polytechnique Fédérale de Lausanne
tam.nguyenthanh@epfl.ch*

Abstract

The Web became the central medium for valuable sources of information extraction applications. However, such user-generated resources are often plagued by inaccuracies and misinformation due to the inherent openness and uncertainty of the Web. In this work we study the problem of extracting structured information out of Web data with a credibility guarantee. The ultimate goal is that not only the structured information should be extracted as much as possible but also its credibility is high. To achieve this goal, we propose a learning process to optimize the parameters of a probabilistic model that captures the relationships between users, their unstructured contents, and the underlying structured information. Our evaluations on real-world datasets show that our approach outperforms the baseline up to 6 times.

1 Introduction

The Web became the central medium for valuable sources of information such as articles, blogs, and wikis, where people constantly share knowledge, report scientific studies, upload comments, and write reviews. As a consequence, the Web emerged as the prime source for extracting information. Applications that benefit from information extraction on top of the Web include systems for knowledge base [6, 25], decision-support [10], recommendations [14], and data integration [11, 12, 23, 24].

However, the Web is often plagued by unreliable and untrustworthy information as Web users can express post their contents freely. The openness of the Web have enabled users to share their information and relay data without proper attribution. Moreover, the users also have wide-ranging levels of expertise and availability. This leads to potential erroneous, inconsistent, and out-of-date information when extracting data from the Web. While the quality of Web data will be still questionable in future, it provides a unique opportunity to exploit the wisdom of the crowd.

As a result, there is a need of identifying ‘credible’ information out of structured data. That is, we should extract the data of high correctness and filter out uncertain and erroneous information. This is motivated by the fact that nowadays there is a lot of information extraction applications that need high quality such as one in healthcare domain [20], where the extracted information is very sensitive and affects many lives.

However, existing works on information extraction only focus on extracting structured data from unstructured contents on the Web [3, 4]. Structured information is hindered by the sheer amount of available Web data and its unstructured, free-text representation (i.e. do not follow formal guidelines, and may even lack proper syntax or spelling). The literature concerns how to parse these unstructured data effectively, meaning the structured data should be correctly identified as-is by avoiding parsing errors. As a result, the extracted data comes with high recall of information coverage, but still questionable in terms of quality due to the inherent uncertainty of the Web. Even though there are a few body of work that study the quality assessment of data as a post-processing step [15–18, 20–22], the output of their quality classifiers can be unpredictable when more training data is provided.

Going beyond the literature, we propose an information extraction framework that comes with a credibility guarantee. Our model allows end-user to specify a credibility threshold as input. Then, it tries to extract the structured data as much as possible while ensuring the overall credibility of extracted data is larger than the pre-defined threshold. In summary, the contributions of this paper is as follows:

- We formally define the problem of extracting structured data from the Web with a credibility guarantee.
- We propose a probabilistic model to capture the relationship between the structured data, the unstructured Web contents and their authors.
- We propose a learning algorithm that allows the credibility guarantee to be met while achieving largest output size.

The remainder of this paper is organized as follows. In the second section, we formally define the problem of credible information extraction. We then introduce how to model various elements in this problem using the factor graph model in the third section. In the fourth section, we discuss an algorithm to learn the model that satisfies the credibility constraint. In the experiment section, we provide empirical analysis of our approach using real-world datasets. The experimental results show that we are able to achieve the precision guarantee with the output size is up to 6 times larger than the baseline. Finally, we provide the related works and conclude the paper.

2 Problem Definition

Our credible information extraction problem is put in the context of social media case study. In that, each user provides one or many *posts*, which are unstructured contents. The structured data we are trying to extract are *statements*, which are facts in real life.

Our overall goal is to learning a credibility assessment classifier with a precision guarantee. In particular, our algorithm allows a user to specify a precision threshold as input.

Problem 1 Given a set of users $U = \{u_1, \dots, u_k\}$, their posts $P = \{p_1, \dots, p_m\}$, the distinct statements $S = \{s_1, \dots, s_n\}$ and a precision threshold τ , return a set of statements considered to be credible with precision at least τ that maximizes output size. We have access to an oracle, who can labeled a statement s as credible or non-credible.

Solving this problem needs to take into account the labeling cost – the number of labels that we need from the oracle – that we seek to minimize. Here the computation time – the time it takes to produce an output – is neglected since our problem concerns the quality of the output. Moreover, thanks to efficient self-configuration of our proposed probabilistic model using factor graph below (i.e. using message passing and sampling), the computation is scalable.

3 Modeling Credibility in the Web

Our approach leverages the intuition that there is an important interaction between statement credibility, linguistic characteristics, and user trustworthiness. We therefore model these elements jointly through a probabilistic graphical model, more specifically a factor graph, where each statement, post and user is associated with a binary random variable. Figure 1 provides an overview of our model. For a given statement, the corresponding variable should have value 1 if the statement is credible, and 0 otherwise. Likewise, the values of post and user variables reflect the objectivity and trustworthiness of posts and users.

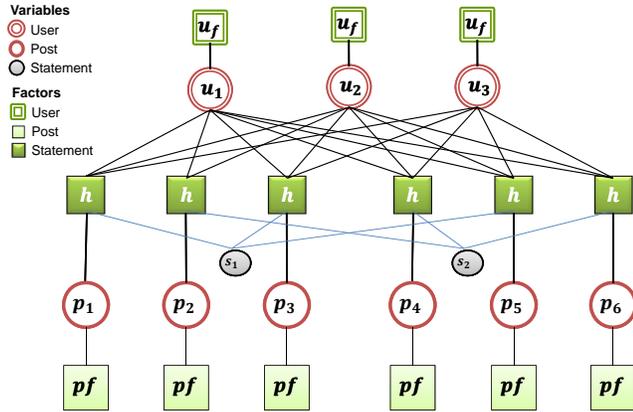


Figure 1: Credibility Evaluation

3.1 Creation of the factor graph

The primary goal of the proposed factor graph is to retrieve the credibility label of statements and the observed features

by leveraging the mutual influence between the model’s variables. The factor graph will capture the following relationships: (i) each user is connected to all hist posts, (ii) each statement is connected to all posts from which it can be extracted (by state of the art information extraction methods), (iii) each user is connected to statements that appear in at least one of his posts.

Formally, a factor graph is a bipartite graph $M = \langle V, F, E \rangle$ where V is a set of random variables, F is a set of functions (factors), and $E \subseteq \{\{v, f\} \mid v \in V, f \in F\}$ are undirected edges. A set of random variables V and a set of factors F fully characterises a factor graph. The definition of the edges relates each factor $f(v_1, \dots, v_d) \in F$ to the random variables over which it is defined, i.e., $\{f, v_i\} \in E$ for $v_i \in V, 1 \leq i \leq d$.

In our context, there are three types of random variables representing users, posts, and statements. We overload notation and use U, P , and S to refer to the actual users, posts, and statements, as well as to the associated random variables, i.e., $V = U \cup P \cup S$. Further, our model includes user factors f_U , post factors f_P , and statement factors f_S to represent the relations between these variables, i.e., $F = f_U \cup f_P \cup f_S$.

Variables Formally we have $u_i \in \{0, 1\}$ is the random variable that represents the user trustworthiness. $u_i = 0$ means that the user u_i must not be trusted while $u_i = 1$ means user u_i is trusted. Second, $s_i \in \{0, 1\}$ is the random variable that represents the statement credibility. $s_i = 0$ means that the statement is not credible and $s_i = 1$ means that the statement is credible. Third, $p_i \in \{0, 1\}$ is the random variable that represents the post objectivity. $p_i = 0$ means that the post is subjective whereas $p_i = 1$ means that the post is objective.

Factors Nodes associated with users and posts have observable features, which can be extracted from the online community.

User factors/features. For users, we derive engagement features (number of questions and answers posted), interaction features (e.g. replies, giving thanks), and demographic information (e.g. age, gender). The features are presented in details in [20]. Formally, each user u is associated with a normalized, multi-dimensional feature vector $\langle f_1^U(u), \dots, f_s^U(u) \rangle$. Against this background, the user factor u_f is defined as:

$$u_f(u) = \exp\left(\sum_{i=1}^s w_i^U f_i^U(u)\right) \quad (1)$$

Here, $f_i^U(u) \in [0, 1]$ is the feature score of the user for the i -th feature, which is normalized into the unit interval by dividing it by the maximum feature value observed among all users. And w_i^U is a weighting parameter indicating the significance of the individual features. Note that every user factor uses the same weight vector $\langle w_1^U, \dots, w_s^U \rangle$.

Post factors. For posts, we extract linguistic features in the form of discourse markers and affective phrases. The features are presented in detail in [20]. Formally, each post p is associated with a normalized, multi-dimensional feature

vector $\langle f_1^P, \dots, f_t^P \rangle$. As a result, a post factor pf is defined as:

$$pf(p) = \exp\left(\sum_{i=1}^t w_i^P f_i^P(p)\right) \quad (2)$$

Here, $f_i^P(p) \in [0, 1]$ is the feature score of the post for the i -th feature, which is normalized into the unit interval by dividing it by the maximum feature value observed among all posts. And w_i^P is a weighting parameter indicating the significance of the individual features. Note that every post factor uses the same weight vector $\langle w_1^P, \dots, w_t^P \rangle$.

Statement factors. Each factor $h(\cdot)$ depicts the mutual reinforcing relationship between user, post, and statement.

$$h_i(u, p, s) = \exp(w_i^H \cdot C(u, p, s)) \quad (3)$$

where w_i^H is the weight parameter of a given factor h_i in the factor graph. Here, $C(u, p, s)$ is an indicator function capturing the consistency between user value, post value, and statement value as follows.

$$C(u, p, s) = \begin{cases} 1 & u = 1 \wedge p = 1 \wedge s = 1 \\ & \text{or } u = 0 \wedge p = 0 \wedge s = 0 \\ 0 & u = 1 \wedge (p \oplus s = 0) \\ & \text{or } u = 0 \wedge (p \oplus s = 1) \\ 0.5 & \text{otherwise} \end{cases} \quad (4)$$

This indicator captures two observations. First, a trustworthy user is likely to post credible statements and a non-trustworthy user is likely to post non-credible statements. Second, a non-trustworthy user is unlikely to post credible statements and a trustworthy user is unlikely to post non-credible statements. For other cases, we assign a value of 0.5 to equally distribute the possibilities as default.

3.2 Probability Computation

The factor graph model enables us to compute the credibility of a statement that is provided by a user. This computation exploits the (marginal) probabilities of the random variables representing the trustworthiness of users, the credibility of a statement, and the objectivity of the posts. Since the trustworthiness of a user $u \in U$ and the credibility of a statement $s \in S$ are binary, $Pr(s = 1)$ (or short $Pr(s)$) or $Pr(u)$ are the probability that the statement is credible and the user is trustworthy, respectively. Probability computation is based on the correlations defined by the factor functions that relate the random variables to each other.

Model Parameter. More precisely, we can define $W = \{w^U\} \cup \{w^P\} \cup \{w^H\}$ as the set of all weights. With all weights are normalized to $[0, 1]$, we define a parameter instance of the factor graph as $I \in [0, 1]^{|W|}$ and $I = \langle w_1, \dots, w_{|W|} \rangle$. As a result, given a parameter instance I , we can compute the representative probability distribution of the factor graph as the normalized product over all factors and variables:

$$\begin{aligned} Pr(S, U, P|I) &= \frac{1}{Z} \prod_u \prod_p \prod_s uf(u) \cdot pf(p) \cdot h(u, p, s) \\ &= \frac{1}{Z} \prod_u \prod_p \prod_s \exp\left(\sum_{w \in I} w \cdot func(\cdot)\right) \end{aligned} \quad (5)$$

where $func(\cdot)$ is a feature or indicator function as aforementioned.

Marginal probabilities. To compute the marginal probabilities, one can employ Bayesian rules. However, the factor graph model allows a more efficient computation by using *belief propagation* or *sampling*. Belief propagation considers the (un)certainty as information that is propagated throughout the factor graph, e.g., by message-passing algorithms or sum-product algorithms [19]. However, these techniques tend to converge slowly if the graph is large and contains circles [32]. When applying factor graph to credibility assessment, the number of variables grows quickly, resulting in large and dense factor graphs. Therefore, we resort to *sampling* to find the most probable values of random variables, while taking into account the factors connecting them. Specifically, Gibbs sampling proved to be a highly efficient and effective mechanism for factor graphs [32].

3.3 Instantiation

Given a parameter instance I , the probability computation on the factor graph returns the marginal probabilities of statement variables. That is, for each statement s , we have a probability $Pr(s \text{ is credible})$. Based on these probabilities, we can instantiate a set of statements (denoted as R) that we believe to be credible. We follow a simple approach to select statements that have probability larger than 0.5; i.e. $R = \{s \in S \mid Pr(s = \text{credible}) > 0.5\}$. It is worth noting that the set of instantiated statements R depends on the parameter instance I and the factor graph model M in which we denote $R = \text{instantiate}(\langle M, I \rangle)$.

4 Learning the model with precision guarantee

Problem 1 can be reformulated as a learning problem where we need to find a parameter instance I from the multi-dimensional parameter space $[0, 1]^{|W|}$ such that the precision of instantiated statements is guarantee and its output size is maximized. Formally, given a set of users $U = \{u_1, \dots, u_k\}$, their posts $P = \{p_1, \dots, p_m\}$, the distinct statements $S = \{s_1, \dots, s_n\}$, we can construct a factor graph model M as above. Now given a precision threshold τ , we need to find a parameter instance I such that $Prec(R) \geq \tau$ and $|R|$ is maximal, where $R = \text{instantiate}(\langle M, I \rangle)$.

For brevity sake, we denote $Prec(I) = Prec(R)$ as the precision of a given parameter instance, which is the fraction of actual credible statements over the total of instantiated statements. As the ground truth for all statements is unknown before-hand, we provide a precision estimation in Section 4.3. Similarly, we denote $Size(I) = |R|$ as the output size of a given parameter instance.

Algorithm 1: The iterative parameter learning process

input : a set of user U ,
a set of posts P ,
a set of statements S ,
a precision threshold τ .

output: a set of credible statement R .

```
// Initialization
1  $M \leftarrow \text{construct}(U, P, S)$ ; // Construct a factor graph
2  $I = \langle 1, 1, \dots, 1 \rangle$ ;
3 for  $i = 1..d$  do
4    $I = \text{binarySearch}(I, i, \tau)$ ;
5  $T = \Omega = \{I\}$ ;
6 while  $\Omega \neq \emptyset$  do
7    $I = \text{pop}(\Omega)$ ;
8   for  $i = 1..d$  do
9      $I' = I$ ;
10     $I'[i] = I[i] - 1/k$ ;
11    for  $j = 1..d, j \neq i$  do
12       $I' = \text{binarySearch}(I', j, \tau)$ ;
13     $I' = \text{binarySearch}(I', i, \tau)$ ;
14     $\Omega = \Omega \cup \{I'\}$ ;
15     $T = T \cup \{I'\}$ ;
16  $I^* = \text{argmax}_{I \in T} \text{Size}(I)$ ;
17  $R = \text{instantiate}(\langle M, I^* \rangle)$ ;
18 return  $R$ ;
```

Algorithm 2: Binary search

input : a parameter instance I ,
the i -dimension of I ,
a precision threshold τ .

output: parameter instance I with new i -dimension value.

```
1  $hi \leftarrow I[i] * k$ 
2  $lo \leftarrow 0$ 
3 while  $hi - lo \geq 2$  do
4    $mid \leftarrow (hi + lo) / 2$ 
5    $I[i] = mid / k$ 
6   if  $\text{Prec}(I) > \tau$  then
7      $hi = mid$ 
8   end
9   else
10     $lo = mid$ 
11  end
12 end
13 return  $I$ 
```

4.1 Parameter Search

In general, to find the parameter instance that satisfies the constraint, it requires us to search over all the possible instances $[0, 1]^{|W|}$, which is very large [1]. In order to reduce the search space, we make two observations: 1) as there are an infinite number of instances in the parameter space, an approximation is required to make it finite and 2) there is a tradeoff tendency between precision and output size that as we increase the feature weights of a parameter instance I , the precision increases while its output size decreases. The second observation comes from the fact that each weighting parameter reflects the perceived importance of the corre-

sponding feature. Hence, increasing a weight value applies a stricter constraint on the features, which increases the precision while reducing the output size and vice versa.

Regarding the first observation, we can discretize the parameter space by using a granularity parameter k and define a set of $(k + 1)^{|W|}$ points to be a set of points of the form $I = \langle b_1, \dots, b_{|W|} \rangle$, where each b is of the form $\frac{j}{k}$ where $j \in \{0, 1, \dots, k\}$. This allows us to reduce the search space by only focusing on the points inside the “grid”.

Regarding the second observation, it allows us to make the assumption on the monotonicity of precision which is stated as follows:

Definition 1 (Monotonicity) *We say that precision is monotonic w.r.t. the features of the factor graph if the following condition holds. For any two parameter instances I_i and I_j , if $I_i \succeq I_j$ then $\text{Prec}(I_i) \geq \text{Prec}(I_j)$.*

This monotonicity assumption is intuitive but not universally valid. One can construct a set of meaningless features for which the precision is not monotonic. Moreover, a feature might not be necessarily monotonic on its whole value domain (but a range of values is often enough). In the experiments, we show that this assumption holds for the studied features in real-world datasets.

Here we define an ordering on the vector representation of parameter instance. That is, $I_i \succeq I_j$ if $\forall m \in [1, |W|], w_{im} \geq w_{jm}$. If the monotonicity assumption holds, it entails the anti-monotonicity of output size i.e., if $I_i \preceq I_j$ then $\text{Size}(I_i) \geq \text{Size}(I_j)$. This is due to the trade-off between maximizing precision and maximizing output size: one can increase precision at the cost of output size and vice-versa. For example, one can achieve a perfect precision (with high chance) by returning the most credible statements but the output size is only 1.

4.2 Iterative learning process

The monotonicity assumption implies that there is a set of parameter instances $T = \{I \mid \text{Prec}(I) \geq \tau \wedge \forall I' \prec I, \text{Prec}(I') < \tau\}$. On the other hand, based on the anti-monotonicity of output size, the instances in T also satisfy another property that $\forall I \in T, \forall I' \succ I, \text{Rec}(I') < \text{Rec}(I)$. In other words, Problem 1 can be reformulated as finding amongst the parameter instances $I \in T$ the one with the highest recall as T contains the instances that satisfy the precision threshold while being candidates for the one with the largest output size. In the following, we discuss how to enumerate the parameter instances in T as when T is available, finding the one with the maximum output size is straightforward.

The iterative learning process is shown in Algorithm 1. Initially, we construct a factor graph based on the users, their posts and statements (line 1) as discussed in the previous section. Then, the algorithm will search for all the parameter instances in T . It first searches for an initial parameter instance that belongs to T (line 2-4) using the *binarySearch* function. This function does binary search to look for the parameter instance I' by changing value in the i -th dimension of the parameter instance I such that $I'[i]$ is the smallest value in the i -th dimension such that $\text{Prec}(I')$ is still larger than

τ . From the initial parameter instance that belongs to T , it generates other instances based on this instance. These new instances, in turn, are used to find other instances (line 5-15). From the current instance I (line 7) that is already in T , it generates another instance I' by changing the i -th value in I , which makes $I \succeq I'$ (line 10). However, this instance I' may not satisfy the precision threshold, which requires us to search through other dimensions $j \neq i \in [1, d]$ using *binarySearch* to make $I' \in T$ (line 11). However, we need to re-adjust the i -th dimension again to avoid oversatisfying the precision threshold (line 13). The algorithm halts when there is no such instance remaining (line 6). Then, we find the parameter instance with the largest output size (line 16) and from this parameter instance, we instantiate the set of credible statements (line 17).

4.3 Precision estimation

We now discuss the implementation of the precision estimation (i.e. $Prec(I)$ call) in Algorithm 2. Since the number of statements is large, it would be costly to obtain ground truth for all statements. Instead, it is practical to ask ground truth for a number of sample statements (i.e. available training data or expert annotations). In addition, asking for labels for many statements goes against the purpose of our setting, which is finding a set of credible statements.

Guided sampling. To this end, we approximate the precision of the instantiated statements R using sampling technique. We apply Monte-Carlo sampling technique [] in which we ask an expert to label a set of randomly selected statements with confidence coefficient of 90%. Especially, the sampler is guided by the probability distribution in Eq. 5. This guidance helps the samples reflect the true distribution of data and make the precision estimation more accurate. The precision of R is then approximated based on the precision of the sample statements.

Reusing samples. Although the above approach only requires a small number of samples per call, the total number of samples across different calls to estimate precision can be high as the selected samples can be different. To this end, we introduce a technique to reuse the samples from different calls to reduce the total cost to obtain the samples. That is, we order the statements in S based on the hashed values of the statements. As a result, in order to sample statements from R , we pick the statements by the increasing order of hash values. This increases the chance of reusing statements in different calls to the oracle, hence, reducing the cost to obtain the labels for the statements.

The hash function is designed in such a way that the probability distribution of data is preserved. To do this, we compute the importance of each statement as the number of users and the number of their posts sharing this statement. Intuitively, the importance distribution of statements aligns with their probability distribution, since credible/non-credible statements are often shared by many trustworthy/untrustworthy users. As a result, the order of hash values is the order of importance degrees of statements.

5 Experimental Evaluation

5.1 Settings

Datasets To validate the effectiveness of our proposed approach, we conducted experiments on a healthcare dataset[]. The dataset contains 2.8 million posts provided by 15,000 users on a healthcare forum. There is a total of 291276 statements about side-effects of drugs extracted from the posts. The dataset also contains the “ground-truth” which is the recognized side-effects of the drugs provided by experts. According to this ground truth, we can derive the labels (credible or non-credible) of 72819 statements; i.e. check whether a given statement agrees or negates against the expert knowledge.

Algorithm In this set of experiments, we compare our algorithm with a baseline method. The baseline method constructs a linear classifier L based on the features discussed in Section 3. The linear classifier is a weighted combination of the features. Each weight described the importance of the corresponding feature. In order to learn the weights, we use a training dataset of 200 samples and leverage the MIRA algorithm[5] to learn the weights of the linear classifier.

However, the linear classifier constructed as above does not guarantee the precision. In order to guarantee the precision, we set a threshold δ for the classifier to select the credible statements from the dataset. For a statement s , if $L(s) > \delta$, statement s is considered credible and vice versa. The threshold δ allows us to control the precision of the returned statements by the baseline method. The higher the threshold, the higher the precision. After constructed the linear classifier, we use binary search to find a smallest possible threshold δ such that the returned statements have a precision higher than the precision constraint τ .

5.2 Monotonicity of dataset

We first carried out an experiment to verify the monotonicity assumption. In the dataset, we have 72819 statements that already had labels from the experts. For each statement, we compute the well-known features in the literature, including affective features and document length. Then, for different values of these features, we report the precision of the statements, by computing the fraction of credible statements among the statements whose feature values fall into the given range.

Figure 2 shows the monotonicity result. The X-axis is the value of the features. The Y-axis presents the “precision” of the respective statements. The key finding is that as the values of the affective features increase, the precision of corresponding statements is non-decreasing. For example, when the values of the affective features increases from [0.6,0.9] to [0.9,1.0], the precision increases from 0.81 to 0.95. We observe the same trend for the post length feature, even though the monotonicity holds from 0 to 0.8. This shows that our monotonicity assumption is valid in real datasets.

5.3 Correctness of learning model

In this experiment, we study the correctness of our learning model is finding credible statements. The idea is that

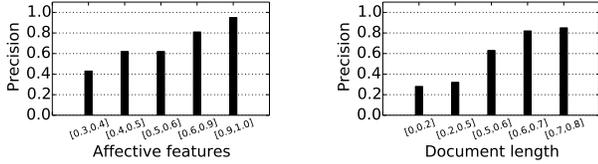


Figure 2: Monotonicity assumption

the more labels are elicited from user, the better the model should be in returning credible statements. Regarding the setting, we set the precision threshold $\tau = 0.9$.

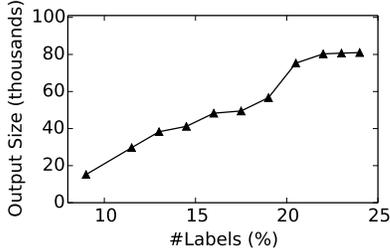


Figure 3: Recall w.r.t. # samples

Figure 3 depicts the results of instantiation during the learning process (i.e. Algorithm 1). The X-axis presents the number of expert labels used by the process (accumulated after each precision estimation call) over the total number of available labels. The Y-axis presents the number of statements that are instantiated as credible. It can be clearly seen that the more labels are provided, the output size of instantiated statements increases. Especially, at 22% labels, the model reaches stable and the increase of labels after this point does not significantly increase the output size. This represents the effectiveness of our learning process in using the labels as we can instantiate most of the credible statements without too much efforts.

Deep understandings of the learning process. We further provide the characteristics of our learning model given different precision requirements. In particular, we vary the precision threshold from $\tau = 0.8$ to $\tau = 0.9$. Then we report the quality metrics of learning process, including the output size, the actual precision, the number of labels, and the number of precision estimations. Regarding the setting, we perform the experiment on the 72819 statements whose labels are known as we want to compute the actual precision by comparing the instantiated statements with the expert labels.

Metrics	τ : precision threshold		
	0.8	0.85	0.9
Output size	91.7K	87.6K	84.2K
Precision	0.82	0.84	0.91
#Labeled statements	21%	28%	24%

Table 1: Characteristics of learning process

Table 1 illustrates the result. An important observation is that as we increase the precision threshold, the output size decreases. This is due to the trade-off between maximizing precision and maximizing the number of credible statements. Another noticeable observation is that the actual precision always satisfies the threshold. Although at $\tau = 0.85$, the actual precision is 0.84 but the difference is negligible. The final finding is that our learning process does not require too many labeled statements in order to achieve the quality requirement.

5.4 Effectiveness of learning model

In this experiment, we compare the effectiveness of our algorithm with the baseline w.r.t. the output size of instantiated statements. The idea is that within the same precision requirement, our model is able to instantiate more statements than the baseline. Regarding the setting, we vary the precision threshold from $\tau = 0.8$ to $\tau = 0.96$.

Figure 4 presents the result. The X-axis is the precision threshold and the Y-axis is the ratio of output size between our learning model and the baseline. We observe that our algorithm outperforms the baseline significantly. For instance, when the precision constraint is 0.82, we are able to return nearly 6 times the amount of the baseline. The above experiments show that our approach is not only able to satisfy the precision guarantee but also with higher output size. This can be explained as follows. When the precision threshold is low, the baseline often over-satisfies the requirement and thus instantiate far less number of statements than our model. When the precision threshold is high, the over-satisfying becomes negligible and thus the baseline performs better than before. However, it is still worse than our model since it does not taking into account the relationships between users, posts, and statements as captured by our factor graph model.

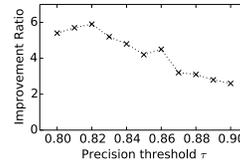


Figure 4: Effectiveness of learning model

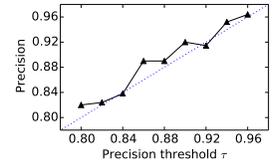


Figure 5: Effectiveness of precision estimation

Effectiveness of precision estimation. The core of our learning model is the precision estimation to guide the search of parameters. In this setting, we would like to verify the accuracy of our precision estimation using sampling techniques as proposed in Section 4.3. To this end, we will compare the precision of our model output with the precision requirement. To able to compute the actual precision, we also have to provide the evaluation only on the 72819 statements that already had labels from the experts.

Figure 5 illustrates the effectiveness of our precision estimation. The experiment is conducted by varying precision threshold from $\tau = 0.80$ to $\tau = 0.96$. Our algorithm is able

to return the statements that satisfy the precision constraint. This can be seen as the precision values of our algorithm are near the identity line. For instance, when the precision constraint is 0.9, the instantiated statements have a precision of 0.92. In some cases, the precision is less than the threshold but the difference is negligible (< 0.02).

6 Related Work

Active learning. Active learning approaches allows the learning algorithm to choose the data it wants to get labels [28], which is suitable in settings where the labels are costly to obtain. It has been applied in various machine learning applications such as speech recognition [34], information extraction [2] and text classification [29]. In credibility assessment, active learning approaches can be classified into two categories: classifier-independent and classifier-specific. Classifier-specific approaches requires constructing a classifier. Notable works in this category are SVMs [29] and decision trees [31] while committee-based approaches [2] belong to classifier-independent category. Although our approach is similar to active learning approaches, there is a fundamental difference. Traditional active learning approaches do not guarantee that the values returned by the learning algorithm satisfy a predefined quality. On the other hand, our approach is able to achieve this constraint while maximizing the output size.

Truth Finding. Given a set of data items claimed by multiple sources, the truth finding (a.k.a. truth discovery) problem is to determine the true values of each claimed item, with various usages in information corroboration [13], and data fusion [7]. Existing work on truth finding also models the mutual reinforcing relationship between data items and sources, e.g., by a Bayesian model [33], maximum likelihood estimation [30], and latent credibility analysis [27]. In addition, these techniques often incorporate prior knowledge about various aspects of the source and the data, such as the dependence between sources [8] and the temporal dimension in evolving data [9]. Although these techniques are able to estimate the truthfulness of the posts, they do not have a precision guarantee on the truth estimation.

Credibility Features. A lot of prior research have been conducted on identifying credible statements from the Web [26]. In these works, they capture the credibility of a statement as a combination of individual features/indicators. The first type of features is content-based, such as semantic features (e.g. category, entities, keywords), sentiments features (e.g. subjectivity), and syntactic features (part-of-speech tag, punctuation marks, spelling errors), advertisements, and page layout. The second type of features is network-based, such as the overall ratings of users sharing the same statements. However, most of the existing works only compute the credibility as an aggregation function of these features. On top of these works, we reuse these features and additionally take into account the mutual relationships between users, their posts, and statements by the factor graph model.

7 Conclusions and Future Work

This paper proposed techniques to find credible statements in online user-generated contents with a precision guarantee. The approach comprises of two components: a probabilistic model and a learning process. The former is responsible for capturing the mutual reinforcing relationships between users, posts, and statements and enforce the consistency observations about credibility. For example, a trustworthy user is likely to post credible statements and vice-versa. The instantiation of the probabilistic model is a set of (probably) credible statements, which is the output. The latter searches for the best-suitable parameters of the probabilistic model such that the output precision satisfying a pre-defined threshold and the output size is maximal. Our evaluation showed that our techniques outperform respective baselines significantly, up to 6 times better.

In future work, as online users are often divided by communities with a wide range of characteristics, we aim to extend the probabilistic model to capture the community membership of users. In addition, we also intend to extend our approach to handle continuous/multi-set labels as the current approach only supports discrete labels.

References

- [1] D. T. Anh et al. "Generating complete university course timetables by using local search methods." In: *RIVF*. 2006, pp. 67–74.
- [2] S. Argamon-Engelson et al. "Committee-based sample selection for probabilistic classifiers". In: *J. Artif. Intell. Res.(JAIR)* 11 (1999), pp. 335–360.
- [3] M. Banko et al. "Open information extraction for the web". In: *IJCAI*. 2007, pp. 2670–2676.
- [4] J. Cowie et al. "Information extraction". In: *Communications of the ACM* (1996), pp. 80–91.
- [5] K. Crammer et al. "Online passive-aggressive algorithms". In: *JMLR* (2006), pp. 551–585.
- [6] O. Deshpande et al. "Building, Maintaining, and Using Knowledge Bases: A Report from the Trenches". In: *SIGMOD*. 2013, pp. 1209–1220.
- [7] X. L. Dong et al. "Data fusion: resolving data conflicts for integration". In: *VLDB*. 2009, pp. 1654–1655.
- [8] X. L. Dong et al. "Solomon: Seeking the truth via copying detection". In: *VLDB*. 2010, pp. 1617–1620.
- [9] X. L. Dong et al. "Truth discovery and copying detection in a dynamic world". In: *VLDB*. 2009, pp. 562–573.
- [10] M. T. Dzindolet et al. "The role of trust in automation reliance". In: *Int. J. Human-Computer Studies* (2003), pp. 697–718.
- [11] A. Gal et al. "Completeness and ambiguity of schema cover". In: *CoopIS*. 2013, pp. 241–258.
- [12] A. Gal et al. "Making sense of top-k matchings: A unified match graph for schema matching". In: 2012, p. 6.
- [13] A. Galland et al. "Corroborating information from disagreeing views". In: *WSDM*. 2010, pp. 131–140.
- [14] J. L. Herlocker et al. "Explaining collaborative filtering recommendations". In: *CSCW*. 2000, pp. 241–250.
- [15] N. Q. V. Hung et al. "ERICA: Expert guidance in validating crowd answers". In: *SIGIR*. 2015, pp. 1037–1038.

- [16] N. Q. V. Hung et al. “Minimizing efforts in validating crowd answers”. In: *SIGMOD*. 2015, pp. 999–1014.
- [17] N. Q. V. Hung et al. “SMART: A tool for analyzing and reconciling schema matching networks”. In: *ICDE*. 2015, pp. 1488–1491.
- [18] N. Q. V. Hung et al. “Towards enabling probabilistic databases for participatory sensing”. In: *CollaborateCom*. 2014, pp. 114–123.
- [19] F. R. Kschischang et al. “Factor Graphs and the Sum-Product Algorithm”. In: *IEEE TRANSACTIONS ON INFORMATION THEORY* 47 (1998), pp. 498–519.
- [20] S. Mukherjee et al. “People on Drugs: Credibility of User Statements in Health Communities”. In: *KDD*. 2014, pp. 65–74.
- [21] H. Q. V. Nguyen et al. “Minimizing human effort in reconciling match networks”. In: *ER*. 2013, pp. 212–226.
- [22] Q. V. H. Nguyen et al. “BATC: a benchmark for aggregation techniques in crowdsourcing”. In: *SIGIR*. 2013, pp. 1079–1080.
- [23] Q. V. H. Nguyen et al. “Collaborative Schema Matching Reconciliation”. In: *CoopIS*. 2013, pp. 222–240.
- [24] Q. V. H. Nguyen et al. “Privacy-Preserving Schema Reuse”. In: *DASFAA*. 2014, pp. 234–250.
- [25] T. T. Nguyen et al. “Result selection and summarization for Web Table search”. In: *ICDE*. 2015, pp. 231–242.
- [26] T. G. Papaioannou et al. “A Decentralized Recommender System for Effective Web Credibility Assessment”. In: *CIKM*. 2012, pp. 704–713.
- [27] J. Pasternack et al. “Latent credibility analysis”. In: *Proceedings of the 22nd international conference on World Wide Web*. International World Wide Web Conferences Steering Committee. 2013, pp. 1009–1020.
- [28] B. Settles. *Active Learning Literature Survey*. Computer Sciences Technical Report 1648. University of Wisconsin–Madison, 2009.
- [29] S. Tong et al. “Support vector machine active learning with applications to text classification”. In: *The Journal of Machine Learning Research* 2 (2002), pp. 45–66.
- [30] D. Wang et al. “On truth discovery in social sensing: A maximum likelihood estimation approach”. In: *IPSN*. 2012, pp. 233–244.
- [31] B. Zadrozny et al. “Learning and making decisions when costs and probabilities are both unknown”. In: *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2001, pp. 204–213.
- [32] C. Zhang et al. “Towards High-Throughput Gibbs Sampling at Scale: A Study across Storage Managers”. In: (2013). URL: http://hazy.cs.wisc.edu/hazy/papers/elementary_sigmod.pdf.
- [33] B. Zhao et al. “A bayesian approach to discovering truth from conflicting sources for data integration”. In: *VLDB*. 2012, pp. 550–561.
- [34] X. Zhu et al. *Semi-supervised learning with graphs*. Carnegie Mellon University, Language Technologies Institute, School of Computer Science, 2005.