What is Your Article Based On? Inferring Fine-grained Provenance

Yi Zhang, Zachary G. Ives, Dan Roth
Department of Computer and Information Science
University of Pennsylvania, Philadelphia, PA, USA
{yizhang5, zives, danroth}@cis.upenn.edu

Abstract

When evaluating an article and the claims it makes, a critical reader must be able to assess where the information presented comes from, and whether the various claims are mutually consistent and support the conclusion. This motivates the study of *claim provenance*, which seeks to trace and explain the origins of claims. In this paper, we introduce new techniques to model and reason about the provenance of *multiple* interacting claims, including how to capture *fine-grained* information about the context. Our solution hinges on first identifying the sentences that potentially contain important external information. We then develop a query generator with our novel *rank-aware cross attention* mechanism, which aims at generating metadata for the source article, based on the context and signals collected from a search engine. This establishes relevant search queries, and it allows us to obtain source article candidates for each identified sentence and propose an ILP based algorithm to infer the best sources. We experiment with a newly created evaluation dataset [1], Politi-Prov, based on fact-checking articles from `www.politifact.com`; our experimental results show that our solution leads to a significant improvement over baselines.

1 Introduction

Misinformation is on the rise, and people are fighting it with fact checking. However, most of the work in the current literature (Thorne et al., 2018; Zhang et al., 2019; Barrón-Cedeno et al., 2020; Hidey et al., 2020) focuses on automating fact-checking for a single claim. In reality, a claim can be complex, and proposed as a conclusion of an article. Therefore, understanding *what information supports the article*, especially information

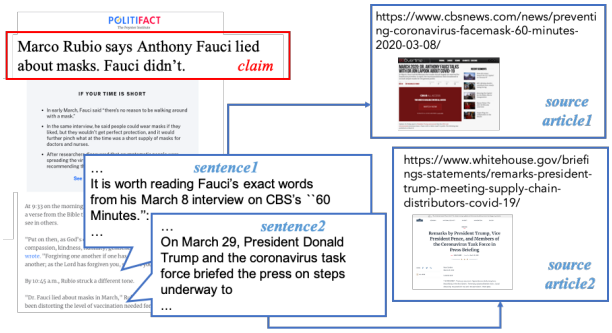

Figure 1: An example of a claim (in the red box) with its article. Sentence 1 and sentence 2 (blue boxes) show examples from the article. Each sentence refers to external information: source article 1 and 2, respectively, with accompanying urls.

that was not originated within the same article, and *where it originates from*, are very important for readers who want to determine whether they can believe the claim.

Figure 1 shows an example of such a claim, *"Marco Rubio says Anthony Fauci lies about masks. Fauci didn't."*[2] with its article from `politifact.com`. A critical reader of the content will find that several major sources support the author's claim: Source article 1 in the figure is *CBS News,"60 Minutes" interview with Anthony Fauci, on March 8, 2020*, which reveals that Dr. Fauci's main point was to preserve masks for those who were already ill and people providing care. If readers can validate all sources used in the article, they will be able to determine whether the article is trustworthy. In this paper, our goal is to automatically find these sources for a given article. This is a different problem from fact-checking: Fact-checking seeks evidence for a claim, while here we only care about the information sources the authors

[1] The data and the code will be available at `http://cogcomp.org/page/publication_view/944`

[2] `https://www.politifact.com/factchecks/2020/dec/28/marco-rubio/marco-rubio-says-anthony-fauci-lied-about-masks-fa/`

Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, pages 5894–5903
August 1–6, 2021.

used when they were writing. Furthermore, the problem we address is critical also to authors who want to give credit to those who have contributed to their article, and it enables a recursive analysis that can trace back to the starting points of an article.

This motivates the study of *provenance* for natural language claims, which describes where a specific claim may have come from and how it has spread. Early work (Zhang et al., 2020) proposed a formulation to model, and a solution to infer, the provenance graph for the given claim. However, that model is insufficient to capture the provenance of an *article*, because (1) an article consists of multiple claims, and it leverages information from other sources, therefore the provenance of all claims should be included in the article's provenance; (2) the inference solution they proposed can only extract domain-level provenance information, e.g., *cbsnews.com*, while it can not directly link the claim to its *source article*, e.g., *https://www.cbsnews.com/news/preventing-coronavirus-facemask-60-minutes-2020-03-08/*. Such fine-grained provenance information is important because it can help people understand the original context that influenced the information they read. Therefore, in this work, we argue that the notion of a provenance graph should be extended to incorporate provenance for articles, and that we need a more comprehensive solution that can identify important external information used in the article and infer its corresponding source article: namely, its fine-grained provenance information.

Technically, capturing fine-grained provenance for an article is challenging because (1) there may be large numbers of sentences in an article, and not all are from external sources nor important (thus, their provenance may not be worth considering); (2) a sentence in an article is usually just a textual fragment of its source article, and simply looking for other articles with related content may result in low precision with regards to finding the correct original article. In our running example, *sentence2* in Figure 1 is *"On March 29, President Donald Trump and the coronavirus task force briefed the press on steps underway to increase ..."*, whose source is *White House's coronavirus task force press briefing on March 29, 2020*. If we directly search for the sentence on the web, it is hard to find this among popular articles from the news. Instead, we need a model that can generate better keywords for a more focused search.

The key contributions of this paper are (1) we introduce and formalize the problem of inferring fine-grained provenance for an article; (2) we propose a general framework to infer the source articles that have provided important information for the given article, including (a) a ranking module that can identify sentences that contain important external information based on the main topic and the main entities in the article; (b) a query generator that can generate possible metadata for the source article, e.g., the title, the published date, the source website, based on the context of the selected sentences; (c) an integer linear program (ILP) based algorithm to jointly identify the source articles from all of the candidates. (3) to evaluate our solutions, we collect a new dataset Politi-Prov from `politifact.com`, and our experimental results show that the solution we proposed can lead to a significant improvement compared with baselines.

2 Problem Statement

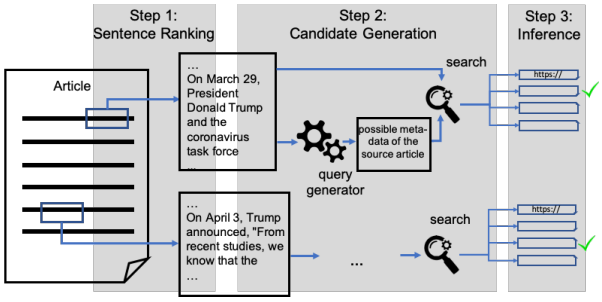

Figure 2: The pipeline of inferring fine-grained provenance for an article.

Given an article d, we are to capture its fine-grained provenance, by inferring k source articles $SA^k(d)$ that provide the most important information for d. We adopt the notion of provenance from (Zhang et al., 2020), while in this paper, we focus on inferring provenance for a claim based on the information from the given article. To find $SA^k(d)$, there are three subproblems we need to solve.

First, we need to locate the important external information in d, which means we need a *sentence ranking* module that can estimate a score σ_i for each sentence in $d = \{s_i\}_{i=1}^n$, based on how likely s_i contains external information. Then we will choose top-k sentences based on their score, and try to find source articles for those sentences.

Second, for each selected sentence, we need to generate a list of candidate links, which can be

its source articles. To achieve this goal, we take advantage of a search engine, based on which we can access all of the articles on the web. As we have discussed in Section 1, directly searching the identified sentence on a search engine may result in a low precision of finding the correct source article. Therefore, we propose to develop a *query generator* to generate the possible metadata of the target source article as new search keywords, so that the search engine is more likely to recall source articles. We then collect all of the search results as the candidates for a selected sentence.

Finally, we need to *infer* the correct source article from the candidates, for each identified sentence. Figure 2 depicts the three steps we need to conduct to infer the fine-grained provenance, which correspond to the three subproblems listed above. We will elaborate the details of each step in Section 4.

3 Politi-Prov Dataset

To the best of our knowledge, there is no existing dataset that can support inferring fine-grained provenance for an article, therefore we create a new dataset based on the fact-checks from `politifact.com` to support the training and the evaluation of this problem.

Specifically, we crawled all of the fact-check questions from `politifact.com` on 4 different issues: *Coronavirus, Health Care, Immigration, Taxes* in September, 2020. For each question, we further crawled its webpage to obtain (1) the title, which is actually the fact-check question itself, (2) the sections of the main text and (3) the "Our Sources" section listing all of the articles (including urls) that provide important information mentioned in the fact-check article. Figure 3 shows an example of such a section.

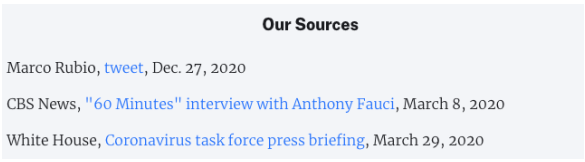

Figure 3: An example of "Our Sources" section of an article from politifact.com, where we obtain the gold fine-grained provenance of the article.

Furthermore, we extract all of the hyperlinks in the webpage, which can tell us where the source articles are mentioned in the main text.

To sum up, we use the main text of each webpage as the given article, and the source articles listed in the section of "Our Sources" as the ground truth our system wants to return. We want to note it is possible that there may be some sources missing in the ground truth we can obtain, therefore, we focus more on the recall in the evaluation.

Overall, we collected data from 1765 articles, where we use 883 of them for training, and 441 and 441 for validation and testing respectively. On average, each article has 9.8 source articles.

4 Inferring Fine-grained Provenance

In this section, we will elaborate how we solve the problems proposed in Section 2.

4.1 Sentence Ranking

Given an article, the first step is to identify the sentences that are most likely to contain important external information. To develop a general data-driven solution, rather than design a ranking function by domain-specific feature engineering, we take advantage of the hyperlinks inserted in the article, so that we can find where the source articles are mentioned. The hyperlink is helpful here because it is standard for the author to provide external information on related topics to the reader. If the hyperlink refers one at the listed source articles, it means the sentence is the one that we are looking for. Then our problem is to learn a model that can distinguish those sentences from the regular ones in the article.

Specifically, we first extract all of the hyperlinks with their corresponding sentences in the given article d, and denote the output as $Hp(d) = \{(l, s)|s \in d\}$, where l represents the link of the article and s represents the sentence. Then, we create a list of positive sentences for d denoted as $P(d)$ by finding the intersection between the articles in $Hp(d)$ and those in $SA^k(d)$, i.e., $P(d) = \{s|s \in d, \exists(l, s) \in Hp(d), s.t., l \in SA^k(d)\}$. Meanwhile, we create a list of negative sentences for d by randomly sampling from the rest of its sentences, denoted as $N(d)$. When a new article is given, the job of the model turns out to estimate a score σ_i of how likely each sentence s_i in d refers to important external information.

Since the sentences referring to important external information are always either directly related to the main topic or about the main entities mentioned in the article, we will leverage them to build our model. Denote the title of d as t_d, and the most

important entities mentioned in the article as E_d. Here, we simply use tf-idf to determine the importance of an entity to an article. We build our model by leveraging Roberta (Liu et al., 2019). Using the same notation in the paper, we concatenate t_d and each $e \in E_d$, feeding it to the model as sentence A, and $s \in P(d)$ or $N(d)$ as sentence B, as the input of Roberta. We then use Roberta as a binary classification model, that is, we use its [CLS] vector as input to a two layer neural network to obtain the probability of s referring to important external information. Instead of learning the features independently for each example, we want to help the model better capture the discriminative feature between the positive and negative examples. Therefore, we add a margin ranking loss to the learning objective, so that it can enforce the model to distinguish the representations between positive and negative examples. We start training from a pre-trained Roberta model and fine-tune it to our ranking task using the following loss, given $s_i \in P(d)$ and $s_j \in N(d)$:

$$\begin{aligned} L_{i,j} = &- \log \sigma_i - \log (1 - \sigma_j) \\ &+ \max \left(0, \tau(s_j) - \tau(s_i) + \epsilon\right) \end{aligned} \tag{1}$$

where $\tau(s_i)$ and $\tau(s_j)$ are the representations, obtained by the output of a single layer neural network τ on top of the [CLS] vector of Roberta.

4.2 Candidate Generation

Identifying the sentences that are describing external information provides us with a clue to finding the source articles. The next step is to find candidate articles that can be the source articles based on the identified sentences. However, as we have described in Section 1, it is hard to find the source article by directly searching the sentence on the web, since so many articles may be talking about the related information. Therefore, we argue that besides using the sentence as the query, we need a query generator that can generate a better query for searching, so that it can increase the possibility that we can recall the correct source article.

4.2.1 Generating Metadata As Query

To generate a query that can improve the recall, the question here is what search keywords are good for finding the source articles besides the identified sentences themselves? In this work, we argue that the *metadata* of the target article, including its *source domain, title* and *published date* is a good choice. Since most of those information may be revealed in the sentence or its context, it is possible that we train a model where we can feed the context of the sentence, and generate a combination of the possible source domain, title and published date of the article it refers to.

In our running example in Figure 1, the sentence identified (sentence 2 in the figure) is *"... On March 29, President ... "*. The source domain of the article it refers to (source article 2 in the figure) is *white house*, the title of the article is *coronavirus task force press briefing*, and the published date is *March 29, 2020*. It is obvious that most of those information has been somehow mentioned in the context or at least can be very easily associated with. Therefore, we treat this problem as a text generation problem, where we feed the identified sentence with its context, and try to generate its metadata. As a baseline, we train this model via fine-tuning BART (Lewis et al., 2020), a pre-trained text generation model.

4.2.2 Integrating Search Engine Signals

Besides the metadata to generate, the content of the identified sentence itself should be useful for searching, when there is an overlap between the sentence and the content of the target article. In this case, if we search for the identified sentence on a search engine, the results returned can be related articles, and their metadata may provide additional useful information that can tell the model what should be included in the target output.

In our running example mentioned in the last section, if we search that sentence on Google, one result it returned is `cspan`'s article *"President Trump with Coronavirus Task Press Briefing"*, which has been very close to the title of the target article. Therefore, our generation model should leverage those signals, which consist of metadata of related articles to the target article.

To incorporate the signals, we first issue the identified sentence as a query to the search engine and collect its top-5 returned urls. Then, as what we do to the identified sentence, we crawl its metadata, i.e., *the source domain*, *title*, and *published date*, and put them together as one document. Then, our problem becomes to generating the metadata of the source article, when we are given the identified sentence, its context, and a concatenation of possible metadata outputs.

In this case, we actually have two types of inputs for the model. One is the identified sentence with its context, where we are to infer the metadata

from, and the other one is the concatenation of possible outputs, where we want to extract the correct metadata components directly from. To solve this problem, we extend the BART baseline to incorporate two sources of inputs, by first feeding the text inputs independently to the BART's encoders, then concatenating the outputs of the encoders together, and finally feeding the unified representations to the BART's decoder.

4.2.3 Rank-Aware Generation

We collect multiple possible metadata for each source article, so that the integration can help us generate better keywords for the search. However, treating the multiple possible metadata as a single document neglects the rank of the urls returned, which reflects the different possibility for each candidate to be the right metadata. Therefore, we propose a *rank-aware multi-head cross-attention* to relieve this problem. The basic idea is when BART's decoders are performing cross-attention over the text input of the sentences and the possible metadata, we require that each set of attention heads (Vaswani et al., 2017) derives different attention scores based on different metadata. Concretely, each set of attention heads will explicitly pay attention to different parts of the input corresponding to different pieces of metadata, and neglect the others. Therefore, after training, each set of attention heads can be used to project the input embeddings into different representation subspaces but focusing on a specific set of candidate metadata. For example, we will have a set of attention heads do cross-attention only over the positions of the sentences and the meta-data from the first url, another set do it only over the positions of the sentences and the meta-data from the first and the second urls, and so on. Note that the candidate metadata from the urls ranked higher will always receive more attention than the others in this case.

Figure 4 summarizes our final design of the generation model.

4.3 Joint Inference

Given the identified sentence and the query keywords generated, we can search for them on a search engine and collect a set of links that are the candidates of the source articles. The next problem is to infer the correct ones from them.

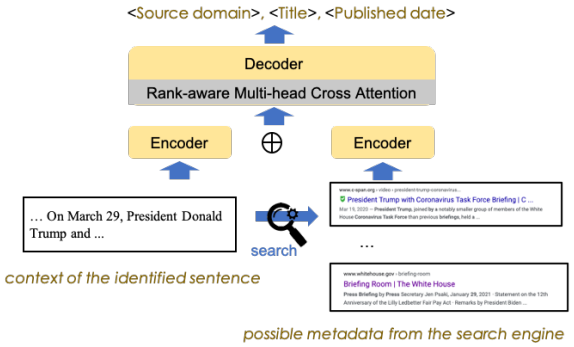

Figure 4: The architecture of the query generator. The model extends (1) BART's encoders to incorporate two types of input, one is the context of the selected sentence, and the other one is possible metadata collected from a search engine, (2) BART's decoders with a rank-aware multi-head cross attention to generate the gold metadata.

4.3.1 Intuitions

Based on our observations, the author is very likely to leverage the external information coming from the same source websites. In our running example introduced in Section 1, the author cited 8 articles in total, and among those articles, two of them come from `whitehouse.gov` and another two come from `politicfact.com`, which are actually two claims they have done fact-check before. Besides the sources, the titles of the articles are also very likely to be related. In the same example, some of them are all talking about the interviews done by *Anthony Fauci* at different time, and some of them are talking about the white house's *Coronavirus Task Force in Press Briefing*. Therefore, we propose an algorithmic inference framework that can take advantage of those relations between the source articles to determine the correct source articles of identified sentences jointly.

4.3.2 ILP-based Inference

We formulate the inference as an Integer Linear Program (ILP) (Roth and tau Yih, 2004; Cheng and Roth, 2013), that allows us to jointly determine the best candidate for each identified sentence.

Formally, we introduce two types of Boolean variables: x_i^k, which represents if the k^{th} candidate is the source article of the i^{th} sentence, and z_{ij}^{kl}, which represents if the source article of the i^{th} sentence and the source article of the j^{th} sentence are related, which means either they come from related source websites or provide related content.

To infer the value of the Boolean variables, our

objective is to assign the best candidate to each identified sentence that can (1) maximize the overall relatedness of the source articles to the query document, and (2) maximize the relatedness between the source articles. To compute the relatedness, we introduce w_i^k, which represents the relatedness score of the candidate article to the identified sentence, γ_{ij}^{kl}, which represents the similarity score between the representations of the source domain of the i^{th} article's k^{th} candidate and the source domain of the j^{th} article's l^{th} candidate, and τ_{ij}^{kl}, which represents the similarity score between the representations of the title of the i^{th} article's k^{th} candidate and the source domain of the j^{th} article's l^{th} candidate. Then, the optimization goal to find the best assignments Γ_d of candidates for the identified sentences is as follows:

$$\Gamma_d = \text{argmax}_\Gamma \sum_i \sum_k \omega_i^k x_i^k + \sum_{i,j} \sum_{k,l} (\tau_{ij}^{kl} + \gamma_{ij}^{kl}) z_{ij}^{kl} \quad (2)$$

s.t.

$$\begin{aligned} x_i^k \in \{0,1\}, z_{ij}^{kl} \in \{0,1\} \\ \forall i, \sum_k x_i^k = 1 \\ 2z_{ij}^{kl} \leq x_i^k + x_j^l \end{aligned} \quad (3)$$

Here, $\sum_k x_i^k = 1$ means only one candidate will finally be chosen as the source article of the i^{th} sentence, and $2z_{ij}^{kl} \leq x_i^k + x_j^l$ means only if the k^{th} candidate of the i^{th} sentence and the l^{th} candidate of the j^{th} sentence have been chosen, we need to consider the relations between them.

In our experiments, we use the last hidden layer of BERT-large (Devlin et al., 2019) as the representation for titles and source domains, and use cosine similarity to compute the similarity score. The ILP problem is solved using an off-the-shelf high-performance package [3].

5 Experimental Evaluation

In this section we aim to answer the following research questions:

RQ1 Can we correctly identify the sentences that refer to important external information in the given article?

RQ2 Given the identified sentences, can we generate the metadata of the target articles from the context?

RQ3 Given a list of candidates for each identified sentence in the article, can we assign the correct candidate to each identified sentence?

RQ4 Given the identified sentences, can we use the query we generated to find candidates, and successfully use them to improve the inference of source articles?

Among those questions, RQ1-RQ3 are to evaluate a specific component of our solution, and RQ4 is to evaluate the joint performance of candidate generation and source article inference. In the following part, we will elaborate the answers to those questions, and for each question, we will start with describing its experimental setting, baselines and the metrics.

5.1 Sentence Ranking (RQ1)

Setup We use Politi-Prov dataset introduced in Section 3. Concretely, we train and validate our models on the articles in the training and validation set, and try to predict the score of a sentence referring to a source article from the article belonging to the test set. To compare the performance, we implement our solution (SR-TE) as described in Section 4.1, and compare it with (1) a retrieval baseline that simply computes the cosine similarity between the embedding vectors (using Roberta) of the title and the sentence in the article (SR). This retrieval baseline only captures the relatedness between the sentence and the main topic of the article; (2) a retrieval baseline similar to SR, but computing the cosine similarity between the embedding vectors of the concatenation of the title and the most important entities (top-50) and the sentence in the article (SR-E), where we want to show the effect of considering important entities; (3) our learning solution without considering entities (SR-T). We report the mean precision and recall of the top-k results respectively.

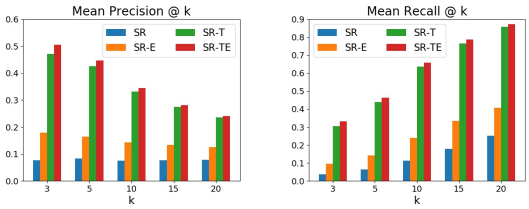

Figure 5: The Performance of Sentence Ranking

Results The results are reported in Figure 5. The gaps between SR, SR-E, and SR-T, SR-TE show that considering important entities always results in an improvement on both precision and recall, which reveals that the sentences can not be identified based on their relatedness to the title (the

[3] https://www.python-mip.com/

main topic) only, but also requires other important information in the article. Furthermore, the figure also shows that the learning method is significantly better than the retrieval baseline without a learning objective.

5.2 Candidate Generation (RQ2)

Setup We collect all of the sentences that correspond to the source articles in training, validation and test set of Politi-Prov serving as training, validation and testing respectively. Overall, there are 5279 cases for training, 1847 for validation, and 1538 for testing. For each case, the source input is the identified sentence with its context (two sentences which are before and after the sentence respectively), and the target output to generate is the metadata of the corresponding source article in a form of a concatenation of its source domain, title and published date. To evaluate the performance, we report Rouge 1, Rouge 2 and Rouge L score of the text generated, and compare with the performance produced by (1) the original BART, (2) our solution integrating signals from Google (BART-S), and (3) our solution integrating signals from Google with our rank-aware multi-head cross attention (BART-SR).

Results We report the results in Table 1. As shown in the table, we can observe that integrating the signals from a search engine can significantly improve the performance of generating the metadata, and considering the ranking of the search results can further lead to an improvement.

	Rouge-1	Rouge-2	Rouge-L
BART	30.102	14.237	28.136
BART-S	34.363	18.398	32.660
BART-SR	36.679	19.017	34.682

Table 1: The performance of generating the metadata for identified sentences.

5.3 ILP Inference (RQ3)

Setup To conduct an isolated evaluation of the ILP based inference, in this experiment, we generate the candidates for each identified sentence based on its metadata from the ground truth. Concretely, we assume there is an oracle that can generate the metadata based on the context for each identified sentence, and we directly search the metadata on Google, and fetch its top-5 results returned as candidates for each identified sentence. Then, our inference algorithm is to find the correct source article for each sentence from those candidates.

To evaluate the performance, we report the mean recall of source articles for each article, and compare it with results provided by the baselines, including (1) simply choosing the top-1 article from the results returned by directly searching the identified sentence on Google (SS1), (2) choosing the top-1 article from the results returned by searching the metadata on Google (MS1), (3) our proposed solution, which conducts ILP inference to find the source article from the search results returned by searching the metadata on Google (MS-ILP). To have a better understanding of the performance, we also report two upper bounds. The first one is the upper bound of the mean recall of the results by directly searching the identified sentence on Google (SS-UB), and the second one is the upper bound of the mean recall of the results by directly searching the meta-data on Google (MS-UB). To compute the upper bounds, if one of the articles returned by Google is correct, then we consider the sentence is correctly assigned. Actually, they are equivalent to the mean recall of the top-5 results, since we only request Google for its top-5 search results.

Results We report the performance in Figure 6. In the figure, we can observe that the mean recall of SS1 is only 0.067, and even its upper bound SS-UB can only achieve 0.15, which reveals that directly searching the identified sentence on a search engine to find the source article is not feasible. Using the metadata of the source article to search can improve the mean recall to around 0.3, and considering the relatedness between the source articles by ILP can further improve it to around 0.37. It demonstrates that the ILP inference is useful for capturing the relatedness between the source articles, and the result has been very close to the mean recall of its top-5 results (MS-UB), which is the upper bound of the performance that the inference can achieve with searching by metadata.

5.4 Source Article Inference (RQ4)

Setup In this experiment, we issue the queries generated by the query generation module to Google, and fetched the top-5 results returned. We combine these results with the top-5 links returned by searching the identified sentence directly, as the candidate pool for each identified sentence. Then, we conduct ILP inference to assign the candidate to each sentence. We report the mean recall of

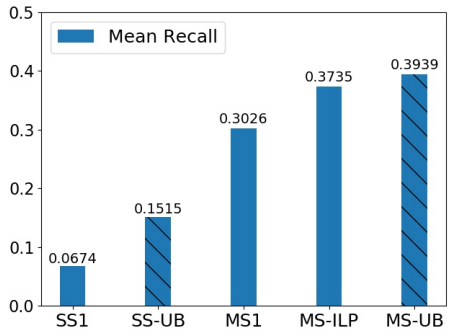

Figure 6: The performance of inferring source articles for each article, MS-ILP is our ILP based solution, and MS-UB is the best possible performance that can be achieved when the candidates are the top-5 results returned by searching for metadata on Google.

the source articles, varying k, which represents the number of the links we returned for each identified sentence. Note that finding the top-k assignments in ILP is actually relaxing the unique solution constraint in Eq 3 to be $\forall i, \sum_j x_i^j = k$, which makes the problem require an additional significant amount of time to solve. Therefore, here we greedily select the best assignment for each variable as an approximate top-k solution.

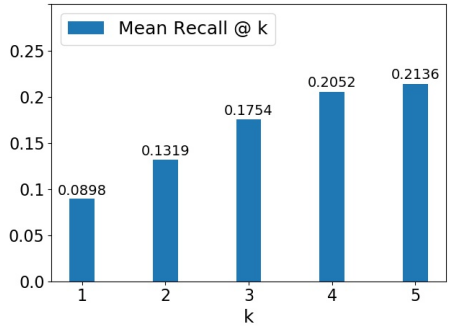

Figure 7: The performance of inferring source articles varying k.

Results As shown in Figure 7, we can observe when $k = 3$, it has already beaten the performance of SS-UB reported in Figure 6, which reveals that the candidates found by the queries generated by our query generator are helpful. When $k = 5$, the mean recall can achieve around 0.21, which is much better than 0.15, the best performance achieved by searching the identified sentence directly. However, as what we can observe in the figure, there is still a gap to the performance of MS-UB in Figure 6. This may result from the insufficiency of the query generation, which implies that a better text generation model may be necessary to further improve the performance, which we think is an interesting topic for future work.

6 Related Work

Our work builds on earlier work on Claim Provenance (see Section 2 for a discussion). Beyond that, we discuss below additional related work.

Fact-checking Fact-checking is related to our problem, since there is usually a document retrieval step to find articles that may provide evidence in most of the solutions (Wang et al., 2018; Thorne et al., 2018; Nadeem et al., 2019). Typically, the input of fact-checking is a single claim instead of an article, therefore it is hard to directly extend their solutions to our problem. Even though fact-checking may find various evidentiary articles for the claim, the source articles we are looking for are those that have been used by the author, which is actually a specific subset of the articles that fact-checking targets to, and the size is also much smaller. Furthermore, we try to extract the metadata of the source articles from the text to support a better search, which is not considered in the document retrieval step of fact-checking.

Recommending Citations Recommending citations for scholarly articles has similarities to our work. The source articles we are looking for can be considered as the citations of the given news article that should be recommended. However, the meaning of the "reference" is different in these two problems. When recommending citations for a paper, the system is to look for previous works that are related to the arguments in the given paper. The argument was created by the author, and the criteria of the recommendation is the relatedness. While inferring provenance is to do reverse engineering to the given article, so that we can find the articles whose information or claims were actually used when the author was writing. Technically, there are two types of citation recommendation systems (Bhagavatula et al., 2018). One is called local (Huang et al., 2012, 2015), that is, a system takes a few sentences (and an optional placeholder for the candidate citation) as input and recommends citations based on the context of the input sentences. Another one is called global (Kataria et al., 2010; Ren et al., 2014; Bhagavatula et al., 2018), that is, a system takes the entire article (and its meta-data which is optional) as input and recommends cita-

tions for the paper. Our solution is more related to local recommendation systems, while we do not assume we can access all of the articles that can be cited and have a way to represent them to be vectors. Therefore, we propose to learn a query generator, which is different with previous works. Furthermore, we do joint inference for all of the identified sentences in the article, which is actually a global inference.

7 Conclusion and Future Work

We propose new techniques to infer fine-grained provenance for an article that contains multiple claims; this is important for a critical reader to understand what information supports the article he/she is reading and what its origins are. The inference consists of models that can identify the sentences that refer to important external information, generate the metadata that can make it more likely to recall the source articles using a search engine, and do an ILP inference to jointly determine the correct source articles from the candidates. We create a new dataset, Politi-Prov, for this task, and our evaluation on it demonstrates the effectiveness of each component, and shows a big improvement compared with the baselines of finding source articles.

However, the problem has not been solved yet. As shown in the analysis, a better text generation model would further improve the performance. Furthermore, it has also been revealed in the experiments that the gold metadata can only recall only around 40% of the source articles, which actually becomes a bottleneck. Therefore, it would be an interesting future work direction to explore what other information should be added to the query, besides the target metadata, so that we can recall more source articles.

Ethical Considerations

Our dataset Politi-Prov is collected from www.politifact.com. The executive director of PolitiFact, based at the Poynter Institute for Media Studies, granted us permission to use their data for this research and to make the new dataset available. The collection process is automatic without additional manual work.

Our collection involves fact-check articles with sources in 4 topics, i.e., coronavirus, health care, immigration and taxes, which were written by the website's journalists. The website seeks to present the true facts, unaffected by agenda or biases, but journalists set their own opinions aside as they work to uphold principles of independence and fairness. Furthermore, the website emphasizes primary sources and original documentation when listing sources, for example direct access to government reports, academic studies and other data, rather than second-hand sources.

Acknowledgements

The authors would like to thank Aaron Sharockman, the executive director of PolitiFact, for kindly granting access to data from the website for academic research. This work is supported in part by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via IARPA Contract No. 2019-19051600006 under the BETTER Program and by a Google Focused Award.

References

Alberto Barrón-Cedeno, Tamer Elsayed, Preslav Nakov, Giovanni Da San Martino, Maram Hasanain, Reem Suwaileh, Fatima Haouari, Nikolay Babulkov, Bayan Hamdan, Alex Nikolov, et al. 2020. Overview of checkthat! 2020: Automatic identification and verification of claims in social media. In *International Conference of the Cross-Language Evaluation Forum for European Languages*, pages 215–236. Springer.

Chandra Bhagavatula, Sergey Feldman, Russell Power, and Waleed Ammar. 2018. Content-based citation recommendation. *arXiv preprint arXiv:1802.08301*.

Xiao Cheng and Dan Roth. 2013. Relational inference for wikification. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1787–1796.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Christopher Hidey, Tuhin Chakrabarty, Tariq Alhindi, Siddharth Varia, Kriste Krstovski, Mona Diab, and Smaranda Muresan. 2020. Deseption: Dual sequence prediction and adversarial examples for improved fact-checking. *arXiv preprint arXiv:2004.12864*.

Wenyi Huang, Saurabh Kataria, Cornelia Caragea, Prasenjit Mitra, C Lee Giles, and Lior Rokach. 2012. Recommending citations: translating papers into references. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 1910–1914.

Wenyi Huang, Zhaohui Wu, Chen Liang, Prasenjit Mitra, and C Giles. 2015. A neural probabilistic model for context based citation recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29.

Saurabh Kataria, Prasenjit Mitra, and Sumit Bhatia. 2010. Utilizing context in generative bayesian models for linked corpus. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 24.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Moin Nadeem, Wei Fang, Brian Xu, Mitra Mohtarami, and James Glass. 2019. FAKTA: An automatic end-to-end fact checking system. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 78–83, Minneapolis, Minnesota. Association for Computational Linguistics.

Xiang Ren, Jialu Liu, Xiao Yu, Urvashi Khandelwal, Quanquan Gu, Lidan Wang, and Jiawei Han. 2014. Cluscite: Effective citation recommendation by information network-based clustering. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 821–830.

Dan Roth and Wen tau Yih. 2004. A Linear Programming Formulation for Global Inference in Natural Language Tasks. In *Proc. of the Conference on Computational Natural Language Learning (CoNLL)*, pages 1–8. Association for Computational Linguistics.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. Fever: a large-scale dataset for fact extraction and verification. *arXiv preprint arXiv:1803.05355*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762*.

Xuezhi Wang, Cong Yu, Simon Baumgartner, and Flip Korn. 2018. Relevant document discovery for fact-checking articles. In *Companion Proceedings of the The Web Conference 2018*, pages 525–533.

Yi Zhang, Zachary Ives, and Dan Roth. 2019. Evidence-based trustworthiness. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 413–423, Florence, Italy. Association for Computational Linguistics.

Yi Zhang, Zachary G. Ives, and Dan Roth. 2020. "Who said it, and Why?" Provenance for Natural Language Claims. In *Proc. of the Annual Meeting of the Association for Computational Linguistics (ACL)*.