WebKE: Knowledge Triple Extraction from Semi-structured Web with Pre-trained Markup Language Models

Chenhao Xie[1,2], Wenhao Huang[1], Jiaqing Liang[1,2], Chengsong Huang[1], Yanghua Xiao[1,3]
[1]Shanghai Key Laboratory of Data Science, School of Computer Science, Fudan University,
[2]Shuyan Technology Inc.,
[3]Fudan-Aishu Cognitive Intelligence Joint Research Center
Shanghai, China
{redreamality,l.j.q.light}@gmail.com
{whhuang17,huangcs19,shawyh}@fudan.edu.cn

ABSTRACT

The World Wide Web contains rich up-to-date information for knowledge graph construction. However, most current relation extraction techniques are designed for free text and thus do not handle well semi-structured web content. In this paper, we propose a novel multi-phase machine reading framework that reads the web content with different granularity. It first detect the area of interest at DOM tree node level and then extract the relational triple for each area. To encode the web content, we propose HTMLBERT. It is a pre-trained webpage encoder that fully leverages the visual layout information and DOM-tree structure, without the need of hand engineered features. Experimental results show that the proposed approach outperforms state-of- the-art methods by a considerable gain. The source code is available online.

ACM Reference Format:
Chenhao Xie, Wenhao Huang, Jiaqing Liang, Chengsong Huang and Yanghua Xiao. 2021. WebKE: Knowledge Triple Extraction from Semi-structured Web with Pre-trained Markup Language Models. In *Proceedings of the 30th ACM Int'l Conf. on Information and Knowledge Management (CIKM '21), November 1–5, 2021, Virtual Event, Australia.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/XXXXXX.XXXXXX

1 INTRODUCTION

The task of knowledge extraction (KE) is to harvest relational facts from text sources. These facts are represented in the form of ⟨*subject, relation, object*⟩ triples, in short, $\langle s, r, o\rangle$. For example, ⟨*Ray Allen, Age, 35*⟩ is one of the desired triples to be extracted from the context in Figure 1.

Knowledge extraction is a key step towards knowledge graph construction. In order to keep knowledge graphs up-to-date, previous researches have been devoted to extracting information from various sources, including encyclopedias [19, 32], news articles [32], Among all the sources, semi-structured vertical sites are arguably the most promising source for first-hand and long-tailed information [21, 41]. For example, a startup company in the early stage may not have news reports or an encyclopedia page. However, there is a good chance that it has a homepage that contains its *founder, contact, location*, etc. Therefore, the ability of extraction from semi-structured webpage enables knowledge graph construction for the long-tailed entities.

However, extracting from semi-structured web content is not easy. The first challenge is the representation of the webpage. Unlike normal natural language text, the webpage contains markup language, which has semantic meaning regarding the organization of the contents. Webpage sentences are also usually longer than natural language ones. This hinders the usage of pre-trained language models such as BERT [5], usually having a length limit of 512 tokens. In addition, a webpage can be rendered in a web browser, which provides visual signals that a human can understand. Natural language processing models are preferred, whereas treating the webpage as an image will lose semantic information. However, most of the current natural language processing models only handles one-dimensional inputs and thus lose layout information. The second challenge is how to produce training data. Early approaches such as wrapper induction [4, 11, 17] recruit human annotators to label the DOM-nodes[1]. These methods achieve accurate extraction results (e.g., over 0.95 precision in [11]) but fail to industrial scenarios due to two drawbacks: 1) they usually need annotation for each website, which is too expensive for large-scale extractions; 2) they are not robust to changes even within the same website, which is unbearable for the rapidly updated sites. An alternative path is to apply distant supervision to generate abundant training data [21]. This idea originally appears in relation extraction from natural language text [25], where relational triples $\langle s, r, o\rangle$ from a seed knowledge base (KB) are aligned with a natural language sentence. If the sentence contains both subject end object, then $\langle s, r, o\rangle$ is regarded valid for the sentence. This approach generates abundant training data with small effort. However, due to the incompleteness of KB, generated data often contains *false negatives* [39], which means the unlabeled fields in a webpage contain knowledge triples.

In this paper, we propose a pipelined extraction framework to handle the semi-structured web content. Our solution is based on BERT [5], a pre-trained language model (PLM) that achieve satisfactory results on many natural language processing tasks [29]. However, mainstream BERT implementations have a limit of maximum input length, which is too small for the overwhelmingly

CIKM '21, November 1–5, 2021, Virtual Event, Australia.
© 2021 Association for Computing Machinery.
ACM ISBN 978-1-4503-8446-9/21/11…$15.00
https://doi.org/10.1145/XXXXXX.XXXXXX

[1]DOM is the short hand for Document Object Model, see https://dom.spec.whatwg.org/.

long webpage (with thousands of tokens in a webpage, see Table 1). Therefore, we propose a preliminary step that recursively prunes the irrelevant DOM subtrees so that the remaining DOM subtrees are adequate for subsequent models. We extend its pre-defined vocabulary so that it can work with HTML tags. We extend the positional embedding of BERT to cope with the extreme case (such as the `cast` relation in movie vertical) where the text in a single node is longer than the length limit. We also encode the layout information of a webpage into the PLM, thus providing more visual-semantic clue. We adopt the distant supervision setting. To reduce the impact of false negatives, we propose a novel relation extractor that identifies relation mentions first and then find the objects for each relation. For a false negative training instance, instead of wrongly classifying the $\langle r, o \rangle$ pair as negative [22, 23], our method model the false negatives as "unlabeled positives". Besides, modeling the extraction of relations and objects as two separate steps also ensure that the model is aware of the corresponding relation when extracting objects.

Our contributions are threefold. Firstly, we novelly model the task of extracting knowledge from semi-structured web as machine reading comprehension (MRC) with different granularity and propose a general framework, dubbed WebKE. It works well with distantly supervised training data, which require neither human annotation efforts nor handcrafted features. Secondly, we propose HTMLBERT, a pre-trained webpage encoder based on BERT, to deal with the idiosyncrasies (i.e., exceedingly long content, layout information, vocabulary specially for markup language, etc.) of web content. Thirdly, we conduct comprehensive experiments to verify our solution. The evaluation results show that WebKE consistently outperforms the competitors and behave well even under adverse circumstances where the false negative rate is 70%.

2 RELATED WORKS

Knowledge extraction from semi-structured web connects with a broad body of researches on information extraction.

Relation Extraction from Natural Language Text. Distantly supervised relation extraction receives wide attention [7, 13, 25, 30, 31, 42]. Recent approaches that incorporate PLMs push the performance to a new height [35, 37]. Web contents can be converted to natural language after a simple preprocessing step. By removing the HTML tags (e.g., `<div>`, `<span>`) and keeping only the "inner-text")[2], abundant off-the-shelf relation extraction methods can be applied. Nevertheless, the loss of layout information and the inability to deal with DOM tree structures becomes a fatal drawback.

Visual-based Document Recognition. This line of researches focuses on the extraction from scanned documents [8, 15, 24, 40]. These methods treat documents as image and incorporates coordinates of each textual component into the model together with the text representation. Applying these methods to the semi-structured web will introduce unnecessary OCR[3] step. Even worse, the processing unit of these models are "pages", which is natural in scanned documents, but not applicable for the web context. A webpage usually exceeds the maximum input length of models while division by screens may cause the separation of a $\langle$*relation, object*$\rangle$ pair into multiple pages.

DOM-based Web Extraction. These methods utilize the intrinsic structure of the webpage. Method of this category includes learning wrappers (i.e., a DOM-specific parser that can extract content of interest) [4, 11, 17], visual-pattern-oriented models (or scoring rules) [12, 43] and neural models [6, 20, 23]. All these methods need heavy human efforts such as wrapper annotations, heuristic scoring rules (e.g., visual proximity), handcrafted features that fed into the neural networks or prior knowledge that being used for post-checking. Recent approaches apply distant supervision to automatically generate training instances by aligning triples from existing knowledge bases (KBs) with the source web [21, 22]. Although significantly reducing the annotation effort, it causes inevitable false negatives due to the incompleteness of KBs [39].

3 PROBLEM MODELING

3.1 Problem Definition

We now formally define our task. Given a webpage $w \in \mathcal{W}$ describing a subject entity s (also called topic entity in previous literature), our goal is to extract $\langle r, o \rangle$ pairs to form a relational triple. In w, a *node* n is a part of the DOM tree, where a sub-node (or textual content if the node is leaf node) wrapped by a pair of markup tags (e.g., `<div> text </div>`). Depending on whether the relations are predefined, there exist two settings, i.e., *ClosedIE* and *OpenIE* [22]. In **ClosedIE**, existing neural approaches often model the extraction problem as *leaf node tagging* [20, 23], which aim to learn a classification function $\mathcal{F} : (w, n) \mapsto r$ to indicate whether the current node n is the *object* is expressing the relation r. Here $r \in \mathcal{R}$ is a set of pre-defined relations. However, Lockard et al [22] showed that websites even in the same vertical usually have different relation set. For example, the IMDb ontology can cover only 7% of the total relation set in the `movie` vertical.

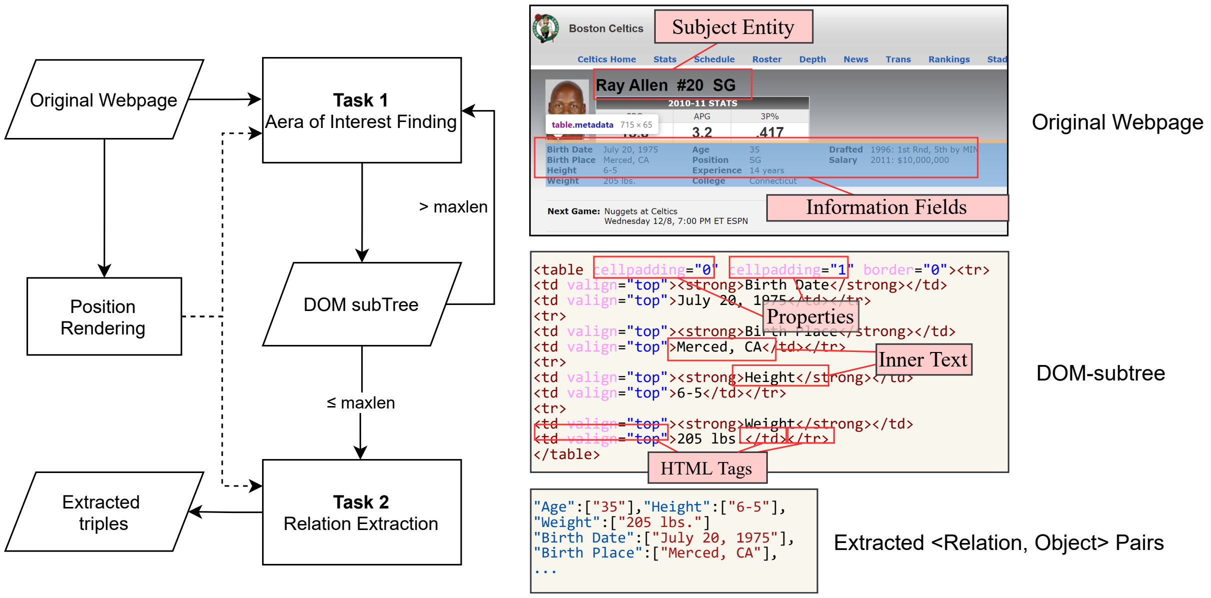
Therefore, we adopt **OpenIE**, where the relation is also expected to be extracted. In previous task formulation [22, 23], the first step is to enumerate all the candidate node pairs in a webpage. Then the task reduces to binary classification on these pairs, indicating whether they form a valid $\langle$*relation, object*$\rangle$ pair. This kind of task definition formulates a hard learning problem for two reasons. First, this setting may generate excessive negative instances since most node pairs do not form valid relational pairs (though filtering heuristic is applied). Second, spurious mentions of objects [21, Figure 1] may result in different labels for the same training instance.

3.2 Solution Framework

The overall framework of our solution is presented in Figure 1. Hereinafter, we consider the webpage w as a sequence of N tokens $\mathbf{t} = [t_1, ..., t_N]$.

Task 1: Area of Interest (AOI) Finding. As is pointed out by previous research [21], a typical webpage content exceeds the maximum length M of neural models. Therefore, we first introduce a task called *area of interest (AOI) finding*. We define an *area* as *a node together with its child node(s)* and the *length of area* as the length of all the tokens in the area. The area is of interest if it contains *information fields* (see Figure 1) for further knowledge extraction.

[2]Refer to https://html.spec.whatwg.org/ for a comprehensive introduction.
[3]https://en.wikipedia.org/wiki/Optical_character_recognition



```
<table cellpadding="0" cellpadding="1" border="0"><tr>
<td valign="top"><strong>Birth Date</strong></td>
<td valign="top">July 20, 1975</td></tr>
<tr>
<td valign="top"><strong>Birth Place</strong></td>
<td valign="top">Merced, CA</td></tr>
<tr>
<td valign="top"><strong>Height</strong></td>
<td valign="top">6-5</td></tr>
<tr>
<td valign="top"><strong>Weight</strong></td>
<td valign="top">205 lbs.</td></tr>
</table>
```

```
"Age":["35"],"Height":["6-5"],
"Weight":["205 lbs."]
"Birth Date":["July 20, 1975"],
"Birth Place":["Merced, CA"],
...
```

Figure 1: The overall framework of WebKE. The example webpage is from the *NBA player* domain.

The HTML content is organized in a DOM tree structure, which make it possible to prune irrelevant page components and hence, limit the input length before extraction. Specifically, we perform topdown traversal on each *level*. At a certain level k, we form a sequence of tokens $\mathbf{n}_k \subset \mathbf{t}$ using only the nodes at k-th layer (their child nodes excluded) and use a machine reading comprehension (MRC) [2] model to decide which node(s) at this level contain(s) information fields for further knowledge extraction. We denote by a the expected area to be extracted. The target of training is to maximize the following likelihood in Eq. (1).

$$\Pr(a|\mathbf{n}_k;\theta) = \prod_{k\in\mathcal{K}_a}\prod_{n=1}^{|\mathbf{n}_k|}(\hat{y}_a^{n,k})^{\mathbb{1}[y_a^{n,k}=1]}(1-\hat{y}_a^{n,k})^{\mathbb{1}[y_a^{n,k}=0]}, \tag{1}$$

where $\hat{y}_a^{n,k}$ refers to the probability of n-th token being the start/end of an area of interest; $\mathcal{K}_a = \{a_{start}, a_{end}\}$ is the pointer identifier; $y_a^{n,k}$ is the ground truth from training data; all the parents nodes become AOI if a child node is AOI; $\mathbb{1}[\text{condition}] = 1$ when the condition happens.

At predicting phases, after the node(s) is (are) recognized as AOI, its (their) child node(s) are expanded to form a new sequence of input (i.e., we proceed to the next level). If there is only one element in a level, we directly proceed to the next level. This process is recursively applied until the length of an area is smaller than maximum length M of subsequent models or none of the area in the current branch is of interest. In practice, the threshold of this step will be rather low, because we are tolerant to the false predictions. Even if a node with no information fields wrongly recognized as AOI (such as the pink node in Figure 2), the final result of AOI finding will be correct as long as none of its child nodes are picked out.

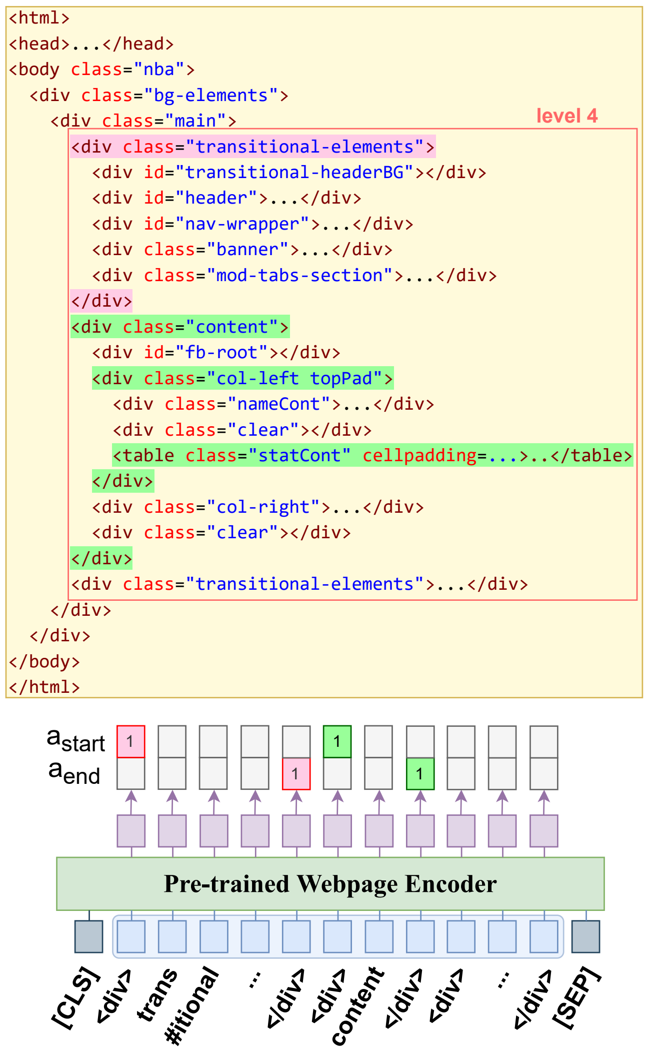
Task 2: Relation Extraction. Next, we formulate the *relation extraction* task. Let $\mathbf{t}_i$ be the area of interest detected by the Task 1. We use $T_i = \langle r, o\rangle$ to denote the set of all triples in $\mathbf{t}_i$ labeled by distant supervision. We assume that the seed knowledge base used by distant supervision is capable of labeling enough samples for training (refer to Section 5.5 for quantification of "enough samples"), though with missing triples (i.e., the false negatives). We omit the subject s in T_i because all the triples of interest have the same s. Let $\mathcal{D}$ denote the whole dataset. Given a training instance $(\mathbf{t}_i, \mathbf{T}_i)$ from $\mathcal{D}$, we aim to maximize the likelihood in Eq. (2). We formulate the task as reading comprehension that first find the relation r and then find its corresponding object o. To implement this idea, Eq. (3) is decomposed into two components using the definition of conditional probability.

$$\prod_{i=1}^{|\mathcal{D}|}\Pr(T|\mathbf{t}_i, s;\theta) \tag{2}$$

$$= \prod_{i=1}^{|\mathcal{D}|}\prod_{r\in T_i}\Pr(r|\mathbf{t}_i, s;\theta)\prod_{\langle o\rangle\in T_i|r}\Pr(o|r,\mathbf{t}_i, s;\theta), \tag{3}$$

where $r \in T_i$ stands for $r \in \{r \mid \langle r, o\rangle \in T_i\}$, r occurs in the triple set w.r.t. $\mathbf{t}_i$; $o \in T_i|r$ stands for $o \in \{o \mid \langle r, o\rangle \in T_i|r\}$; $T_i|r$ denotes a subset of T_i with a common relation r. We denote by θ the model parameters. Under this decomposition, relational triple extraction task is formulated into two subtasks: *relation mention detection* and *object extraction*.

We model both of tasks as machine reading comprehension (MRC) [2]. In relation mention detection, multiple continuous spans from $\mathbf{t}_i$ are extracted representing the expected relations. We formalize it in Eq. (4) as a boundary detection task with two pointers indicating the start (end) positions of the answer spans [34]. For



```
<html>
<head>...</head>
<body class="nba">
  <div class="bg-elements">
    <div class="main">
      <div class="transitional-elements">
        <div id="transitional-headerBG"></div>
        <div id="header">...</div>
        <div id="nav-wrapper">...</div>
        <div class="banner">...</div>
        <div class="mod-tabs-section">...</div>
      </div>
      <div class="content">
        <div id="fb-root"></div>
        <div class="col-left topPad">
          <div class="nameCont">...</div>
          <div class="clear"></div>
          <table class="statCont" cellpadding=...>..</table>
        </div>
        <div class="col-right">...</div>
        <div class="clear"></div>
      </div>
      <div class="transitional-elements">...</div>
    </div>
  </div>
</body>
</html>
```

Figure 2: An example webpage illustrating the level-wise process of finding AOI. All the tokens fed into AOI finder module in the bottom are from nodes at level 4 (without child nodes.). The green nodes are correctly predicted as AOI, while the pink one is a false prediction. We also show the corresponding HTML content on the top. The final AOI (i.e., <table class="statCont" ... </table>) is recognized at level 6.

simplicity, we omit s in the task input because in our assumption, each webpage has only one subject entity. This formulation is extensible to the webpages containing multiple entities by adding the subject entity as a query.

$$\Pr(r|\mathbf{t}_i;\theta) = \prod_{k\in\mathcal{K}_r}\prod_{n=1}^{N}(\hat{y}_r^{n,k})^{\mathbb{1}[y_r^{n,k}=1]}(1-\hat{y}_r^{n,k})^{\mathbb{1}[y_r^{n,k}=0]}, \tag{4}$$

where $\mathcal{K}_r = \{r_{start}, r_{end}\}$ and $\hat{y}_r^{n,k}$ refers to the probability of n-th token being the start/end of the relation mention; $y_r^{n,k}$ is the ground truth from the training data; if $\exists p \in T_i$ appears at position from n to $n+l$, then $y_r^{n,r_{start}} = 1$ and $y_r^{n+l,r_{end}} = 1$, otherwise 0; $\mathbb{1}[\cdot]$ is an indicator function. Similarly, the object extraction task is formulized in Eq. (5).

$$\Pr(o|r,\mathbf{t}_i;\theta) = \prod_{k\in\mathcal{K}_o}\prod_{n=1}^{N}(\hat{y}_o^{n,k})^{\mathbb{1}[y_o^{n,k}=1]}(1-\hat{y}_o^{n,k})^{\mathbb{1}[y_o^{n,k}=0]}, \tag{5}$$

The notations and task formulation are same with the relation mention detection except for two points. 1) The relation mention is taken as a query input of the task; 2) The object, instead of relation mention, become the target of extraction (subscripts are changed, e.g., from y_r to y_o).

4 OUR MODEL

In this section, we describe the model implementation. We first describe *Pre-trained Webpage Encoder*, which is used in all subsequent models. A raw webpage is firstly refined by *AOI finder* to keep those DOM subtrees that contains information fields of interest. Next, the selected nodes are fed into the *relation extractor* to extract the relation and objects.

4.1 HTMLBERT: Pre-trained Webpage Encoder

We regard the input DOM (sub)tree as a sequence of tokens $\mathbf{t}$. Web content differs from the normal textual content in two aspects. It contains additional *markup vocabulary* and semi-structured *layout information*. Therefore, we need to make corresponding upgrades to the original BERT encoder [5].

Extended Vocabulary. HTML tags (e.g., `<a>`, `<table>`, `<tr>`) contribute a lot to the web structure, which is essential for extraction. For example, relation mention and objects are likely to appear in the same `<section>` or `<div>` while tags like `<nav>` `<button>` are less likely to contain information fields. Considering the HTML tag as a single token instead of tokenizing them into subwords will better retain the semantic and structural information.

Extended Positional Embeddings. BERT used an absolute positional embedding, derived from pretraining, which has a length limit (usually 512 tokens). A webpage usually contains overwhelmingly long sequence of tokens. In extreme cases (e.g., the "cast" in IMDb [4]), even within an area of interest, the sequence length exceeds the maximum length limit. Truncating or sliding-window-based methods [36] may cause separation between relation and its objects and fail to capture long-term relations. However, even if the positions are not limited, the memory consumption is unaffordable because the space complexity is $O(N^2)$, which grows very fast with token length N. Therefore, we develop an hierarchical positional embedding that extend the original positional embedding of BERT. We obtain pre-trained position embedding vectors $\mathbf{p} = [p_1, p_2, ..., p_n]$ from BERT. Our aim is to construct a series of new embedding vectors $\mathbf{v} = [v_1, v_2, ..., v_m], m > n$, so that more positions can be represented. Therefore, we introduce a vector basis $\mathbf{u} = [u_1, u_2, ..., u_n]$ for $\mathbf{v}$. Their connection is defined by

$$\mathbf{v}_{(i-1)\times n+j} = \alpha\mathbf{u}_i + (1-\alpha)\mathbf{u}_j, \quad \alpha \in (0.1, 0.5) \cup (0.5, 1) \tag{6}$$

[4]https://www.imdb.com/title/tt5034838/fullcredits

This equation indicates a 2-D representation (i, j) for the $[(i-1) \times n + j]$-th token, which ensures that any position under n^2 has a positional embedding. To be compatible with the originally positional embedding, we hope that the first n positional embeddings are the same with BERT, i.e., $\mathbf{v}_1 = \mathbf{p}_1, \mathbf{v}_2 = \mathbf{p}_2, \cdots, \mathbf{v}_n = \mathbf{p}_n$, which lead us to Eq. (7).

$$\mathbf{u}_i = \frac{\mathbf{p}_i - \alpha \mathbf{p}_1}{1 - \alpha}, \quad i = 1, 2, \cdots, n \tag{7}$$

Layout Embeddings. Relative positions of words in a webpage convey much semantic information. Given that the current node is a relation mention, its corresponding object value is much more likely to appear on its right or below rather than on the left or above. Hence, we render the webpage using headless chrome[5] and record the position of elements by *selenium*[6]. Inspired by [40], we use bounding box to locate the position of an area in the webpage. Specifically, each DOM node is characterized by four position indicators, i.e., $(p_{x_0}, p_{y_0}, p_{x_1}, p_{y_1})$, where (p_{x_0}, p_{y_0}) is the upper-left position of the node and (p_{x_1}, p_{y_1}) corresponds to the bottom-right position. If a node contains child node(s), we select the minimum bounding box that can cover all of them. The position of a token is the same with the node it belongs to.

Hidden Layers and Pre-training. The hidden layers are given by

$$\mathbf{H}_0 = \mathbf{T}\mathbf{W}_t + \mathbf{W}_g + \sum_k \mathbf{W}_p^k \tag{8}$$

$$\mathbf{H} = \text{TRANS}(\mathbf{H}_0) \tag{9}$$

where $\mathbf{T}$ is one-hot tokens indices matrix; $\mathbf{W}_t$ is the token embedding matrix; $\mathbf{W}_g$ is the segmentation embedding matrix where g represents the segmentation index in the input sequence; TRANS$(\cdot)$ is the multi-layer bidirectional Transformer structure [33], the architecture is the same as BERT [5]. $\mathbf{W}_p^k$ is the position embedding matrix where $k \in \{p_0, p_{x_0}, p_{y_0}, p_{x_1}, p_{y_1}\}$; $\mathbf{W}_p^{p_0}$ is the 1-D positional embedding of token sequence and the others are defined above and p is the position index in the input sentence; $\mathbf{H}$ is the last-layer hidden representation, i.e., the output of BERT. We pre-train the webpage encoder via the standard "masked language model" and "next sentence prediction" tasks [5]. The SWDE corpus we used for pre-training is described in Section 5.1 Note that, our webpage encoder do not involve any hand-crafted features.

4.2 Module 1: AOI Finder

The AOI finder module recognize all the areas of interest at level k. As is illustrated in Figure 2, the input of the module is a sequence of tokens $[\texttt{[CLS]}, \mathbf{n}_k, \texttt{[SEP]}]$, where $\mathbf{n}_k$ are the nodes from k-th level of the DOM tree (see Section 3.2). We perform preprocessing by moving out the properties from inside the HTML tags to the outside, which make it easier for BERT tokenizer to recognize the HTML tags as a whole while retaining the signals provided by the properties. For example, the *property* `class="translational-elements"` in Figure 2) are moved out of the HTML tag, resulting in `<div>` to be recognized as a special character. Only the values in `class` and `id` are used, while those property values without a semantic clue such as `cellpadding` are removed. This operation will not affect the inner text because in most cases the nodes other than the leaf nodes do not contain textual content. We use $\hat{\mathbf{y}}_a = [0, 1]^{N \times 2}$ to estimate $\hat{y}_a^{n,k}$ in Eq. (1). $\hat{\mathbf{y}}_a$ consists of two N-dim vectors, representing a *pointer* of start/end of an AOI. They are defined by

$$\hat{\mathbf{y}}_a^k = \sigma(\mathbf{W}_a^k \mathbf{H}_a + \mathbf{b}_a^k), \tag{10}$$

where $k \in \{a_{start}, a_{end}\}$ is in accordance with Eq. (1); $\mathbf{H}_a$ be the hidden representation specific to AOI finder module, whose general form is defined in Eq. (9); $\mathbf{W}_{(\cdot)}^k$ and $\mathbf{b}_{(\cdot)}^k$ are trainable parameters;σ is the sigmoid activation function. The final AOI spans are generated by gathering all the pointers above the threshold then pairing the nearest a_{end} to a given a_{start}.

4.3 Module 2: Relation Extractor

We next extract relational triples from the AOI detected in Section 4.2. The relation extractor consists of two cascaded MRC models. The first model tags the start/end positions of the relation mentions, which are fed one by one into the second model as queries to tag the start/end positions of the objects. Specifically, $\hat{\mathbf{y}}_r = [0, 1]^{N \times 2}$ corresponds to $\hat{y}_r^{n,k}$ in Eq. (4) and $\hat{\mathbf{y}}_o = [0, 1]^{N \times 2}$ corresponds to $\hat{y}_o^{n,k}$ in Eq. (5). The input of the *relation mention tagger* is $\mathbf{x}_r = [\texttt{[CLS]}, \mathbf{t}_i, \texttt{[SEP]}]$, We use $\mathbf{H}_r$ to denote output matrix of BERT. The k-th output pointer of relation mention tagger is defined by

$$\hat{\mathbf{y}}_r^k = \sigma(\mathbf{W}_r^k \mathbf{H}_r + \mathbf{b}_r^k), \tag{11}$$

where $k \in \{r_{start}, r_{end}\}$ is in accordance with Eq. (4). A relation mention is formed by the same nearest-first principle in Section 4.2.

The input of the *object tagger* is slightly different, which is $\mathbf{x}_o = [\texttt{[CLS]}, \mathbf{q}_i, \texttt{[SEP]}, \mathbf{t}_i, \texttt{[SEP]}]$. $\mathbf{q}_i$ is formed by the relation mention extracted above. For a comprehensive study of how the different queries affect the extraction performance, readers may explore many recent works [18, 44]. The object tagger is triggered $|r|$ times, where $|r|$ is the number of relations extracted by the relation mention tagger. The k-th output pointer of object tagger is defined by

$$\hat{\mathbf{y}}_o^k = \sigma(\mathbf{W}_o^k \mathbf{H}_o + \mathbf{b}_o^k), \tag{12}$$

where $k \in \{o_{start}, o_{end}\}$ is in accordance with Eq. (5). Likewise, an object is formed by nearest pairing. Note that if a relation mention occurs at multiple locations in the webpage, all the mentions are leveraged to perform multiple extractions, where the token representation are the same but position and layout embeddings are different. The final results of the same relation are merged.

An obvious advantage of this kind of model design is that the relation mention and objects values are one-to-one mapped. That is, we do not need extra effort to score the candidate $\langle r, o \rangle$ pairs [22] and filter out the spurious mentions (falsely generated from webpage sections such as "you may also like").

4.4 Learning Objective

As is pointed out in previous works [21, 39], false negatives are caused by incomplete KB or imperfect ontology, which are common cases in practice. As a result, we need to apply corresponding learning objective to alleviate their impact. For AOI finder, normal cross entropy is applied because knowledge triples usually share a common information field, which enable us to get a correct supervision of AOI even with incomplete triple labeling. For relation

[5]https://sites.google.com/a/chromium.org/chromedriver/getting-started
[6]https://www.seleniumhq.org/

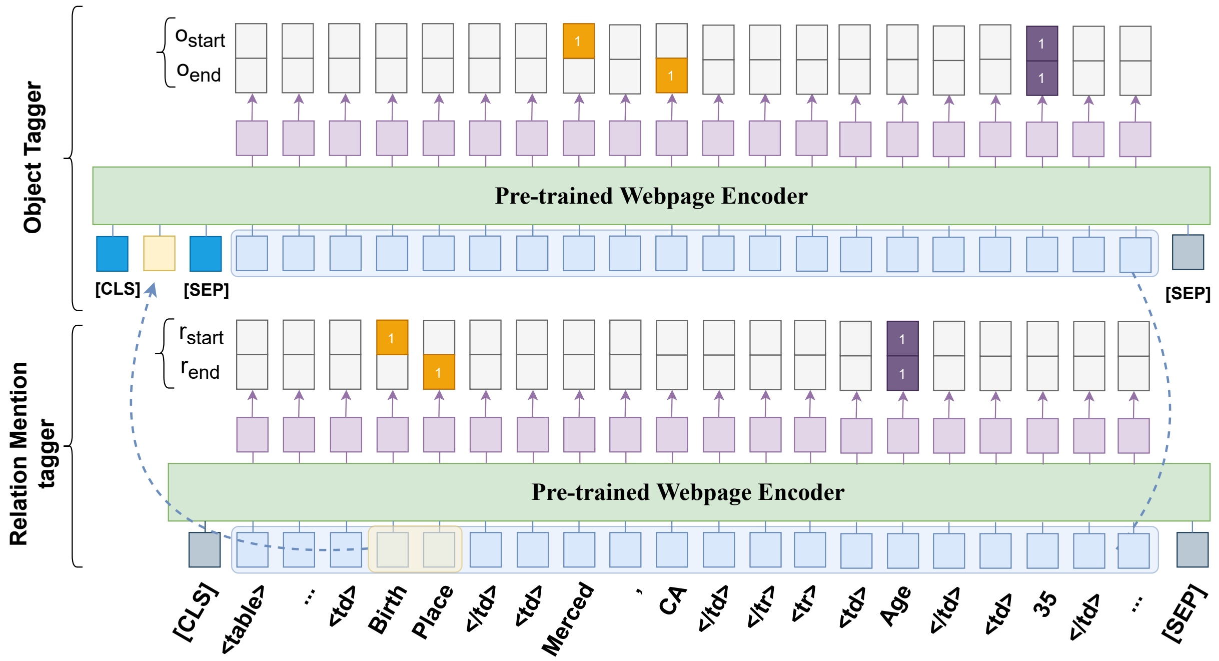


Figure 3: The two-staged relation extractor model. The full DOM subtree is fed into the model. We omit some leaf nodes for clarity. In the example sentence, the relation mention tagger first tag `Birth Place` and Age as relation mentions, then each of them is considered as a query and sent to the object tagger together with the sentence. Here, the object tagger is triggered twice, extracting `Merced, CA` and 35 as object values for the two relations (indicated by the "1" in the blocks, marked with different colors), respectively.

extractor, we adopt collective loss function [38, 39], which is designed to handle the false negatives in data. Its basic idea is to view a small portion of the negative labels as the unlabeled positives. Thus, the excessively imposed penalty on negative labels should be reduced. The loss functions for relation mention tagger and object tagger are detailed in Eq. (13) and Eq. (14), respectively.

$$\ell_r(\hat{\mathbf{y}}_r^k, \mathbf{y}_r^k) = \begin{cases} -\gamma_r \ln(\sum_{n=1}^{N} \hat{y}_r^{n,k}]) & \text{if } y_r^{n,k} = 1, \\ -\ln(1 - |\sum_{n=1}^{N} \hat{y}_r^{n,k} - \mu_r|) & \text{otherwise;} \end{cases} \tag{13}$$

$$\ell_o(\hat{\mathbf{y}}_o^k, \mathbf{y}_o^k) = \begin{cases} -\gamma_o \ln(\sum_{n=1}^{N} \hat{y}_o^{n,k}]) & \text{if } y_o^{n,k} = 1, \\ -\ln(1 - |\sum_{n=1}^{N} \hat{y}_o^{n,k} - \mu_o|) & \text{otherwise,} \end{cases} \tag{14}$$

where $\gamma_{(\cdot)} \in (0, 1)$ is a hyperparameter that downweighs the positive penalty to avoid losing the original class label ratio; $\mu = \pi(\tau+1)$ is set according to [39], where

$$\tau \approx 1 - \frac{\text{\# labeled positive}}{\text{\# all positive}}$$

is the unlabeled positive ratio in positive data and

$$\pi = \frac{\text{\# all positive}}{\text{\# all samples}}$$

is the positive class prior.

5 EXPERIMENTS

With the goal of putting WebKE to practical use, we investigate the following research questions: 1) How is the overall performance of WebKE in comparison to the current state-of-the-art. 2) Are all the components proposed by this method necessary? 3) How many samples are needed to train the model? 4) Will WebKE behave properly under adverse circumstances, such as starting with an incomplete seed knowledge base?

5.1 Datasets

For BERT pretraining, we take the Structured Web Data Extraction (SWDE) dataset [12], which contains webpages and ground truth extractions from 10 websites in 8 domains, with 124,291 webpages in total. For evaluating WebKE, we use the augmented version [22] of the SWDE dataset, which provides more gold labels for OpenIE extractions for 21 sites in 3 vertical domains (i.e., NBA player, Movie and University). The dataset contains between 400 and 2000 pages per sites, with an average of 36 predicates and 41K triples. The statistics of the dataset is presented in Table 1. A relation is *common* if it is covered by half of the sites. The *congruity* is calculated by a Jaccard-like metric to characterize the homogeneity of relations in the vertical, congruity $= \frac{\text{\#common relations}}{\text{\# all relations}}$. A larger congruity means more homogeneous relation are in the vertical. We perform some preprocessing steps: We preprocess the dataset to better suit our problem setting. 1) When encountering hierarchical relations (e.g., cast-director) we only keep the fine-grained relation. 2) We remove the relations whose mentions do not appear in the website. 3) We remove those elements in a webpage that do not contribution to the semantics, such as the contents between `<scripts>` tags.

5.2 Experimental Settings

We implemented our algorithms in Keras and Tensorflow. All experiments were run on a machine with Nvidia RTX3090 GPU and 24GB of graphical memory. We adopt mini-batch mechanism for stochastic optimization. The hyperparameters are as follows. We set with batchsize as 4; the learning rate as 2e-5 and use Adam [16] as the optimizer. We also save the best F1 score model in validation set to prevent the model from over-fitting. The pre-trained BERT model we used is `BERT-Tiny-Uncased` [14], which contains 14.5M parameters. The number of stacked bidirectional Transformer blocks is 2 and the size of hidden state is 128. Besides, we set the threshold to 0.2 for both start and end of the AOI finder to improve the recall.

Table 1: Statistics of the dataset with three common verticals.

	Movie	NBA player	University
# Websites	8	8	5
# Common Relations	8	9	11
# All Relations	54	30	123
Congruity of Relations	14.80%	30%	8.90%
Average Token Length	11764	14540	8662

5.3 Overall Comparison

Compared Methods. Our main results are in comparison with three web extraction baselines in the OpenIE setting.

- **WEIR** [1]. The Web Extraction and Integration of Redundant data (WEIR) method utilizes the websites in the same domain and learn from the overlap in observed entities to extract predicates. Besides, WEIR proposed a method that identify predicate in HTML with the extraction rules it learned through retrieving strings that frequently appear nearby extracted objects.
- **Colon Baseline** [22]. This is a baseline method borrowed from Open-Ceres. It is designed based on the observation that relation mentions and objects in semi-structure webpages often appear in pairs in sibling DOM nodes. A relation mention usually ends with a colon, with corresponding objects locates right or below. This baseline identifies the relation mention that ends with a colon, and extracts the closest text field behind as objects.
- **OpenCeres** [22]. Based on seed facts for relations present on the site, OpenCeres proposed a semi-supervised label propagation technique to generate training data, and use them to learn a site-specific classifier for relation extraction.
- **WebKE**. This implements our model as described in Section 4, using the expanded SWDE dataset to train a webpage information extractor.

Evaluation and Metrics. We use *strict* match for evaluation. An extracted relational triple $\langle s, r, o \rangle$ is considered as correct only if both the relation mention and the object are correct and related to the subject. We use the metrics as in accordance with the baseline methods, which are standard micro Precision (Prec.), Recall (Rec.) and F1 score.

Main Results. Table 2 shows the main result of WebKE in three verticals, namely *Movie*, *NBA player* and *University*. We observe that WebKE outperforms all the compared models by a considerable margin. In specific, WebKE achieves an overwhelming 16%, 40% and 57% improvement over the current state-of-the-art method [22] on three verticals, respectively.

5.4 Comparison to CloseIE Methods

Although our method is designed for openIE, we also compare those methods designed for closeIE. We make the comparison as fair as possible by only focusing on the relations that appear in the closeIE schema. Nevertheless, openIE is still harder due to its problem setting, which selects a span instead of closeIE that performs multiclass classification on the relation set. To fit the CloseIE setting, we select part of triples in expanded SWDE dataset that mention in SWDE ground truth and map the open relations to the pre-defined ones. Considering the fact that some objects in SWDE dataset that does not have an obvious mention of relation[7]. We compared with human annotation methods: HAO *et al.* [12], XTpath [3], BigGrams [26], Vertex++ [10]; unsupervised methods: RR+WADaR [27], RR+WADaR 2 [28], WEIR [1]; and distant supervision methods: LODIE [9], CERES [21], We show the comparison in Table 3. We observe that WebKE performs among top 2 of the competitors, even this setting is unfair to our openIE extraction approach.

5.5 WebKE in Few Shot Setting

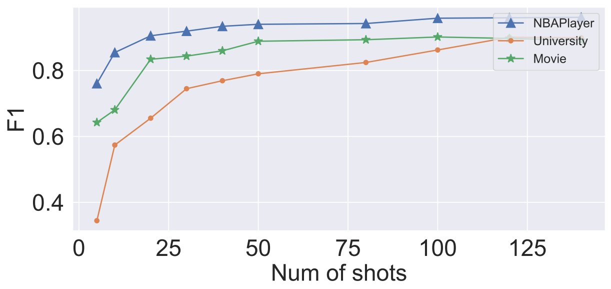


Figure 4: Performance of WebKE using different quantities of training samples.

This experiment answers the practical problem: how many shots are adequate for a new website? We first perform a pre-training step that use a bunch of seed sites in a vertical to train WebKE and then finetune on a new site in the same vertical. If no new

[7]In accordance with previous researches, we only keep the relations pre-defined in their closeIE setting.

Table 2: Overall Comparison for the OpenIE setting. The results of three baselines are excerpted from [23]. Best results are marked Bold. The last four rows are ablation studies where "-" means to remove that component. The abbreviations are explained as follows. LE: Layout Embedding, EV: Extended Vocabulary, PE: Extended positional Embedding

	Movie			NBA player			University		
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
WEIR	0.14	0.1	0.12	0.08	0.17	0.11	0.13	0.18	0.15
Colon Baseline	0.63	0.21	0.32	0.51	0.33	0.40	0.46	0.31	0.37
OpenCeres	0.71	0.84	0.77	0.74	0.48	0.58	0.65	0.29	0.4
WebKE	**0.97**	**0.89**	**0.93**	**0.99**	**0.97**	**0.98**	**0.99**	**0.96**	**0.97**
- AOI	0.86	0.92	0.89	0.99	0.96	0.97	0.98	0.96	0.97
- LE	0.96	0.87	0.91	0.97	0.93	0.95	0.98	0.93	0.95
- EV - LE	0.79	0.75	0.77	0.94	0.94	0.94	0.93	0.94	0.94
- EV - LE - PE	–	–	–	0.94	0.94	0.94	–	–	–

Table 3: Evaluation on CloseIE setting. The abbreviations *unsup.* stands for unsupervised. The results of previous methods are quoted from [21]. Best and second-best results are marked bold **and** <u>underlined</u>.

	Labels	Movie	NBA player	University
Hao *et al.*	human	0.79	0.82	0.83
XTPath	human	0.94	**0.98**	0.98
BigGrams	human	0.74	0.90	0.79
Vertex++	human	0.90	<u>0.97</u>	**1.00**
RR+WADaR	unsup.	0.73	0.80	0.79
RR+WADaR 2	unsup.	0.75	0.91	0.79
WEIR (CloseIE)	unsup.	0.93	0.89	0.97
LODIE	distant	0.86	0.90	0.96
CERES	distant	**0.99**	**0.98**	0.94
WebKE	distant	<u>0.95</u>	**0.98**	<u>0.99</u>

webpages are used for training, it is called a zero-shot setting [23], which is an ideal setting. However, extraction results are far from satisfactory. Therefore, we make few-shots of training data available to the model to enhance the performance. Suppose there are N websites in a vertical, we select $N-1$ site from the same domain as background knowledge and train our model on it. Then we select k webpage from the remaining website for training, namely, k-shot. We finetune our model on it and test on other webpages from this website.

The results are shown in Figure 4. We observe that for *movie* and *NBA player* vertical F1 increases with the number of shots and gradually comes to a standstill at around 50 shots. This indicates that for real applications, 50 shots might be a proper choice. However, *university* vertical requires much more samples to train. This may be caused by the fact that the university ontology is much bigger than the others (see Table 1, "# all relations"). That is, the model needs to learn the representation for more relations, some of which are even not available in the pre-training step.

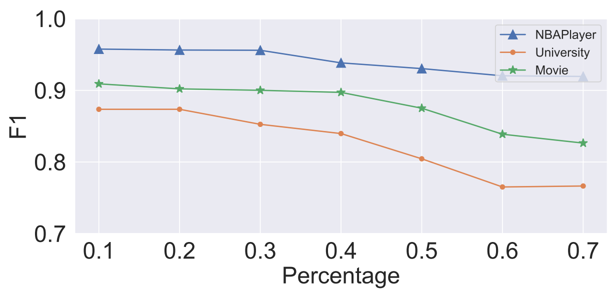


Figure 5: Experiment on synthetic dataset with false negatives. The abscissa represents the percentage of triples are deleted from the dataset and the ordinate is the F1 score.

5.6 Robustness to False Negatives

To further investigate the robustness of WebKE towards false negative labels, we simulate the situation where different percentages of relational triples are missing in the knowledge base. We first generate **seven** synthetic datasets, where we randomly delete 10%-70% of triples from the original training set, representing different false negative rates. We show the F1 score of the three verticals in Figure 5. We observe that there is only a slight drop of performance in every vertical when more triples are deleted. In the *NBA player* vertical, the F1 score remains over 0.9 even when 70% of the triples are missing, which indicates that our method is robust to the false negatives caused by incompleteness of KB. We also find out that the domain with more diverse relations suffers from more performance drop. For example, the university domain drops the most (from 0.88 to 0.76) when more triples are deleted, whose congruity of relations is 8.9% (see Table 1), the least among three verticals.

5.7 Ablation Study

To further justify the effectiveness of each component of WebKE, we perform an ablation study. The results are shown in the last four rows of Table 2. The AOI finder plays an important role when

Table 4: Percentage of remaining content after AOI finder.

Vertical	Remaining
Movie	41%
University	37%
NBA player	61%

the webpages are long. For example, in the *movie* vertical, the precision drops dramatically when AOI finder is removed. We study the error extractions and find out that there usually exists 'movie recommendation' section, which contains $\langle r, o \rangle$ facts of players other than the current subject. It is easily recognized by AOI finder, which leverages more of the DOM structure by design. However, removing it does increase the recall, which means there may exist wrongly pruned DOM subtrees. Besides, from Table 4 we can conclude that AOI finder greatly reduce the content size. Among the components in the webpage encoder module, *Extended vocabulary* contributes the most to the performance. It is reasonable because without those vocabulary the tokenizer will cut HTML tags into sub-words, which impairs the understanding of the web content. We also observe that in NBA player domain, the extended position embedding does not harm the overall performance, which means that it sacrifices nothing but memory space to encode longer sequences. We do not try the other domains without positional embedding due to their exceeding length even after AOI finding.

6 CONCLUSION

In this paper, we introduce a framework called WebKE that extracts knowledge triples from semi-structured web. WebKE novelly extend the power pre-trained language models to markup language and encode the layout semantics of a webpage. Our extraction framework works perfectly with distantly supervised training data, which require neither human annotation efforts nor handcrafted features. Experimental results shows that WebKE achieve a huge improvement over the state-of-the-art baselines on different verticals.

ACKNOWLEDGEMENT

The authors thank the anonymous reviewers for their constructive comments. This work was supported in part by Shanghai Science and Technology Innovation Action Plan (No.19511120400).

REFERENCES

[1] Mirko Bronzi, Valter Crescenzi, Paolo Merialdo, and Paolo Papotti. 2013. Extraction and integration of partially overlapping web sources. *Proceedings of the VLDB Endowment* 6, 10 (2013), 805–816.
[2] Danqi Chen. 2018. *Neural Reading Comprehension and Beyond*. Ph.D. Dissertation. Stanford University.
[3] Joseph Paul Cohen, W. Ding, and A. Bagherjeiran. 2015. Semi-Supervised Web Wrapper Repair via Recursive Tree Matching. *ArXiv* abs/1505.01303 (2015).
[4] Nilesh Dalvi, Ravi Kumar, and Mohamed Soliman. 2010. Automatic Wrappers for Large Scale Web Extraction. *Proceedings of the VLDB Endowment* 4, 4 (2010).
[5] J. Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of NAACL-HLT*.
[6] Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. 2014. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *Proceedings of SIGKDD*. 601–610.
[7] Anthony Fader, S. Soderland, and Oren Etzioni. 2011. Identifying Relations for Open Information Extraction. In *EMNLP*.
[8] Lukasz Garncarek, Rafal Powalski, Tomasz Stanislawek, Bartosz Topolski, Piotr Halama, and Filip Grali'nski. 2020. LAMBERT: Layout-Aware language Modeling using BERT for information extraction. *ArXiv* abs/2002.08087 (2020).
[9] Anna Lisa Gentile, Ziqi Zhang, and Fabio Ciravegna. 2015. Early steps towards web scale information extraction with lodie. *AI Magazine* 36, 1 (2015), 55–64.
[10] Pankaj Gulhane, Amit Madaan, Rupesh Mehta, Jeyashankher Ramamirtham, Rajeev Rastogi, Sandeep Satpal, Srinivasan H Sengamedu, Ashwin Tengli, and Charu Tiwari. 2011. Web-scale information extraction with vertex. In *Proceedings of ICDE*. IEEE, 1209–1220.
[11] P. Gulhane, A. Madaan, Rupesh R. Mehta, J. Ramamirtham, R. Rastogi, S. Satpal, Srinivasan H. Sengamedu, Ashwin Tengli, and Charu Tiwari. 2011. Web-scale information extraction with vertex. *Proceedings of ICDE* (2011).
[12] Qiang Hao, Rui Cai, Yanwei Pang, and Lei Zhang. 2011. From one tree to a forest: a unified solution for structured web data extraction. *Proceedings of SIGIR* (2011).
[13] Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. ACL.
[14] Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. TinyBERT: Distilling BERT for Natural Language Understanding. In *Proceedings of EMNLP: Findings*. 4163–4174.
[15] Anoop R. Katti, C. Reisswig, Cordula Guder, Sebastian Brarda, Steffen Bickel, Johannes Höhne, and J. Faddoul. 2018. Chargrid: Towards Understanding 2D Documents. In *Proceedings of EMNLP*.
[16] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).
[17] N. Kushmerick, Daniel S. Weld, and Robert B. Doorenbos. 1997. Wrapper Induction for Information Extraction. In *Proceedings of IJCAI*.
[18] Xiaoya Li, Fan Yin, Zijun Sun, Xiayu Li, Arianna Yuan, Duo Chai, Mingxin Zhou, and J. Li. 2019. Entity-Relation Extraction as Multi-Turn Question Answering. In *Proceedings of ACL*.
[19] Jiaqing Liang, Sheng Zhang, and Y. Xiao. 2017. How to Keep a Knowledge Base Synchronized with Its Encyclopedia Source. In *IJCAI*.
[20] Bill Yuchen Lin, Ying Sheng, N. Vo, and Sandeep Tata. 2020. FreeDOM: A Transferable Neural Architecture for Structured Information Extraction on Web Documents. *Proceedings of SIGKDD* (2020).
[21] Colin Lockard, Xin Luna Dong, Arash Einolghozati, and Prashant Shiralkar. 2018. CERES: Distantly Supervised Relation Extraction from the Semi-Structured Web. *Proceedings of the VLDB Endowment* (2018).
[22] Colin Lockard, Prashant Shiralkar, and X. Dong. 2019. OpenCeres: When Open Information Extraction Meets the Semi-Structured Web. In *Proceedings of NAACL*.
[23] Colin Lockard, Prashant Shiralkar, X. Dong, and Hannaneh Hajishirzi. 2020. ZeroShotCeres: Zero-Shot Relation Extraction from Semi-Structured Webpages. In *Proceedings of ACL*.
[24] Eric Medvet, Alberto Bartoli, and G. Davanzo. 2010. A probabilistic approach to printed document understanding. *IJDAR* (2010).
[25] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of ACL*.
[26] Marcin Michał Mirończuk. 2018. The BigGrams: the semi-supervised information extraction system from HTML: an improvement in the wrapper induction. *Knowledge and Information Systems* 54, 3 (2018), 711–776.
[27] Stefano Ortona, Giorgio Orsi, Marcello Buoncristiano, and Tim Furche. 2015. Wadar: Joint wrapper and data repair. *Proceedings of the VLDB Endowment* 8, 12 (2015), 1996–1999.
[28] Stefano Ortona, Giorgio Orsi, Tim Furche, and Marcello Buoncristiano. 2016. Joint repairs for web wrappers. In *Proceedings of ICDE*. IEEE, 1146–1157.
[29] Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences* (2020), 1–26.
[30] Xiang Ren, Zeqiu Wu, Wenqi He, Meng Qu, Clare R Voss, Heng Ji, Tarek F Abdelzaher, and Jiawei Han. 2017. Cotype: Joint extraction of typed entities and relations with knowledge bases. In *Proceedings of WWW*.
[31] Ryuichi Takanobu, Tianyang Zhang, Jiexi Liu, and Minlie Huang. 2019. A hierarchical framework for relation extraction with reinforcement learning. In *Proceedings of AAAI*, Vol. 33. 7072–7079.
[32] Jizhi Tang, Yansong Feng, and Dongyan Zhao. 2019. Learning to Update Knowledge Graphs by Reading News. In *Proceedings of EMNLP*.
[33] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of NeuroIPS*. 6000–6010.
[34] Shuohang Wang and Jing Jiang. 2017. Machine comprehension using match-lstm and answer pointer. In *Proceedings of ICLR*.
[35] Yucheng Wang, Bowen Yu, Yueyang Zhang, Tingwen Liu, Hongsong Zhu, and Limin Sun. 2020. TPLinker: Single-stage Joint Extraction of Entities and Relations Through Token Pair Linking. In *Proceedings of COLING*.
[36] Zhiguo Wang, Patrick Ng, Xiaofei Ma, Ramesh Nallapati, and Bing Xiang. 2019. Multi-passage BERT: A Globally Normalized BERT Model for Open-domain

Question Answering. In *Proceedings of EMNLP*. 5881–5885.

[37] Zhepei Wei, Jianlin Su, Yue Wang, Y. Tian, and Yi Chang. 2020. A Novel Cascade Binary Tagging Framework for Relational Triple Extraction. In *Proceedings of ACL*.

[38] Chenhao Xie, Qiao Cheng, Jiaqing Liang, Lihan Chen, and Y. Xiao. 2020. Collective Loss Function for Positive and Unlabeled Learning. *ArXiv* abs/2005.03228 (2020).

[39] Chenhao Xie, Jiaqing Liang, Jingping Liu, Chengsong Huang, Wenhao Huang, and Yanghua Xiao. 2021. Revisiting the Negative Data of Distantly Supervised Relation Extraction. In *Proceedings of ACL*.

[40] Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and M. Zhou. 2020. LayoutLM: Pre-training of Text and Layout for Document Image Understanding. *Proceedings of SIGKDD* (2020).

[41] Alexander Yates, Michele Banko, Matthew Broadhead, Michael J Cafarella, Oren Etzioni, and Stephen Soderland. 2007. Textrunner: open information extraction on the web. In *Proceedings of NAACL)*. 25–26.

[42] Xiangrong Zeng, Daojian Zeng, Shizhu He, Kang Liu, and Jun Zhao. 2018. Extracting Relational Facts by an End-to-End Neural Model with Copy Mechanism. In *Proceedings of ACL*.

[43] Yanhong Zhai and B. Liu. 2007. Extracting Web Data Using Instance-Based Learning. *World Wide Web* 10 (2007), 113–132.

[44] Tianyang Zhao, Zhao Yan, Y. Cao, and Zhoujun Li. 2020. Asking Effective and Diverse Questions: A Machine Reading Comprehension based Framework for Joint Entity-Relation Extraction. In *Proceedings of IJCAI*.