# Experimental Evidence Extraction System in Data Science with Hybrid Table Features and Ensemble Learning

Wenhao Yu[1], Wei Peng[1,2], Yu Shu[1,3], Qingkai Zeng[1], Meng Jiang[1]
[1]Department of Computer Science and Engineering, University of Notre Dame, Notre Dame, IN, USA
[2]College of Computer Science and Technology, Zhejiang University, Hangzhou, China
[3]School of Computer Science, Sichuan University, Chengdu, China
{wyu1,wpeng}@nd.edu,stellashu98@gmail.com,{qzeng,mjiang2}@nd.edu

## ABSTRACT

Data Science has been one of the most popular fields in higher education and research activities. It takes tons of time to read the experimental section of thousands of papers and figure out the performance of the data science techniques. In this work, we build an experimental evidence extraction system to automate the integration of tables (in the paper PDFs) into a database of experimental results. First, it crops the tables and recognizes the templates. Second, it classifies the column names and row names into "method", "dataset", or "evaluation metric", and then unified all the table cells into (method, dataset, metric, score)-quadruples. We propose hybrid features including structural and semantic table features as well as an ensemble learning approach for column/row name classification and table unification. SQL statements can be used to answer questions such as whether a method is the state-of-the-art or whether the reported numbers are conflicting.

## KEYWORDS

Data Science Education, PDF Tables, Information Extraction

**ACM Reference Format:**
Wenhao Yu, Wei Peng, Yu Shu, Qingkai Zeng, Meng Jiang. 2020. Experimental Evidence Extraction System in Data Science with Hybrid Table Features and Ensemble Learning. In *Proceedings of The Web Conference 2020 (WWW '20), April 20-24, 2020, Taipei, Taiwan.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3366423.3380174

## 1 INTRODUCTION

Data scientist was selected as the sexiest job of the 21st century[1], and thus many higher education institutions have opened the new programs for data science training and research. Though data scientists are highly educated – 88% have at least a Master's degree and 46% have PhDs, it is not easy to get into the field at all.

One of the data science projects we did was developing algorithmic tools for *multilabel classification* to predict the labels of objects where multiple labels may be assigned to each object. We started from literature study in this topic. It cost us as long as 23 days to collect, read, and digest hundreds of related works. We found two papers under this topic that were accepted to ACM SIGKDD 2017 Research Track: PPDSparse [42] and AnneXML [36]. Each of them proposed a new multilabel classification model and compared with baseline methods. They both reproduced and tested existing methods (such as SLEEC, FastXML, PfastreXML, and PDSparse) on publicly available data sets (such as AmazonCat-13K, Delicious-200K, and WikiLSHTC-325K) using standard evaluation metrics (such as Precision@1, P@3, and P@5). Table 1 summarizes and compares the numbers given by the two papers, [42] on the left and [36] on the right. We find out that almost half of the pairs have bigger than 3% difference on the scores, which has been able to be claimed as significant improvement on the well-accepted benchmarks! This may be due to the random initialization, parameter settings, or computational environments. We have no idea about the exact reason, but we argue that it is worthwhile to investigate the *experimental evidences* in data science literature.

| Dataset | (%) | SLEEC | FastXML | PfastreXML | PDSparse |
|---|---|---|---|---|---|
| AmazonCat | P@1 | 90.56/89.19 | 94.02/93.10 | 86.06/89.94 | 87.43/89.31 |
| -13K | P@3 | 76.96/75.17 | 79.93/78.18 | 86.06/77.24 | 87.43/74.03 |
| | P@5 | 62.63/61.09 | 64.90/63.38 | 63.65/63.53 | 56.70/60.11 |
| Delicious | P@1 | 47.78/47.03 | 48.85/43.20 | 26.66/37.62 | 37.69/34.37 |
| -200K | P@3 | 42.05/41.67 | 42.84/38.68 | 23.56/35.62 | 30.16/29.48 |
| | P@5 | 39.29/38.88 | 39.83/36.21 | 23.21/34.03 | 27.01/27.04 |
| WikiLSHTC | P@1 | 58.34/55.57 | 50.01/49.75 | 57.17/58.10 | 60.70/61.26 |
| -325K | P@3 | 36.70/33.06 | 32.83/33.10 | 37.03/37.61 | 39.62/39.48 |
| | P@5 | 26.45/24.07 | 24.13/24.45 | 27.19/27.69 | 29.20/28.79 |

**Table 1: Our system found inconsistent precision scores reported by two papers [42] (left numbers) and [36] (right numbers) in ACM SIGKDD 2017 Research Track for multilabel classification. Precision differences of bigger than 3% are underlined, which has been able to be claimed as significant improvement on the well-accepted benchmarks.**

Since that, we started building a system using *data science techniques* to extract and structure experimental results in the *data science literature.* We hope that researchers and practitioners in the fields of data science and artificial intelligence will use it to satisfy their needs of exploring and analyzing the experimental evidences.

The key challenges lie in automating the "reading" of tables in the experimental section of paper PDFs. *First*, there was no well-defined structure of experimental evidence. The tables are embedded in the PDF format. It takes careful engineering efforts on cropping, parsing, and cleaning the tables. *Second*, the tables have different

[1] https://hbr.org/2012/10/data-scientist-the-sexiest-job-of-the-21st-century

*WWW '20, April 20–24, 2020, Taipei, Taiwan*

ACM ISBN 978-1-4503-7023-3/20/04.
https://doi.org/10.1145/3366423.3380174

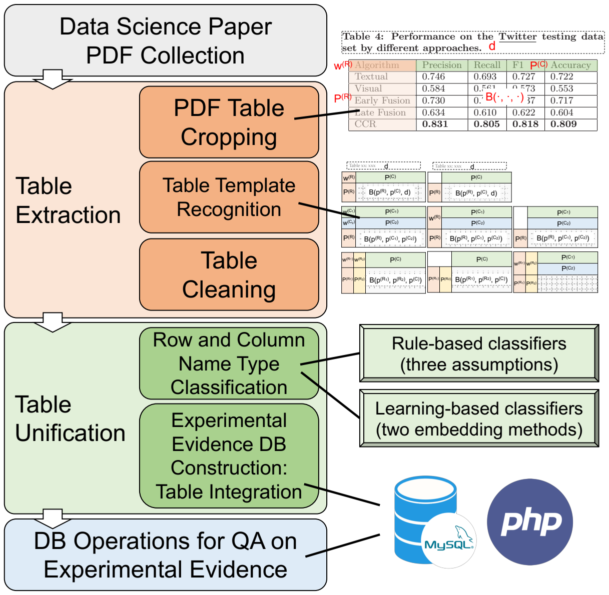

Data Science Paper PDF Collection
Table Extraction
PDF Table Cropping
Table Template Recognition
Table Cleaning
Table 4: Performance on the Twitter testing data set by different approaches.
d
$w^{(R)}$
$P^{(R)}$
$P^{(C)}$
$B(\cdot,\cdot,\cdot)$
Algorithm
Precision
Recall
F1
Accuracy
Textual
Visual
Early Fusion
Late Fusion
CCR
Table xx: xxx
$B(p^{(R)}, p^{(C)}, d)$
$B(p^{(R)}, p^{(C_1)}, p^{(C_2)})$
$B(p^{(R_1)}, p^{(R_2)}, p^{(C)})$
Table Unification
Row and Column Name Type Classification
Experimental Evidence DB Construction: Table Integration
Rule-based classifiers (three assumptions)
Learning-based classifiers (two embedding methods)
MySQL
php
DB Operations for QA on Experimental Evidence

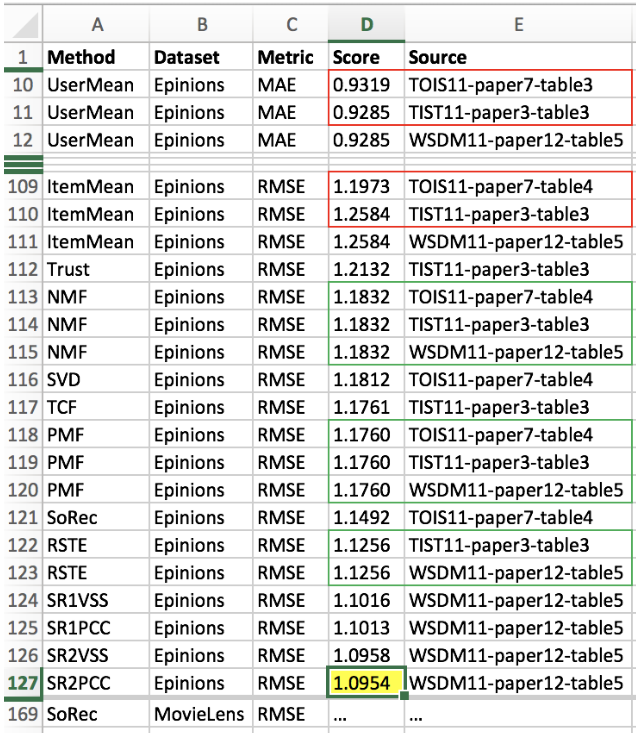
**Figure 1: Workflow of the proposed system: from PDF collection, to table extraction, to experimental evidence database construction, to database operations and visualization.**

kinds of templates, so there was no standard of interpreting the cells. *Third*, the roles of row and column names (such as SLEEC and P@1), say, datasets or methods or metrics, are unknown. The gap between PDF table and queryable database is huge.

**Proposed approach.** This paper presents a novel system that transforms data science paper PDFs into a structured database of experimental evidences, and support multiple exploratory and analytic functions over the constructed database for knowledge discovery. It has three modules. The first module *table extraction* crops the tables from PDFs and recognize their templates. The second module *table unification* classifies the column names and row names into the three types of labels (method, dataset, and metric) and then unifies each cell into a **(method, dataset, metric, score, source)-tuple**. The score is the cell’s value and the source is the PDF file name, page, and number of the table. We propose hybrid features (including structural and semantic table features) and an ensemble learning approach for column/row name classification and table unification. This module constructs a five-column database of the tuples for every table that contains experimental results. The third module *database operation for QA* uses SQL statements (i.e., *select* and *join*) for question-answering on the experimental result database.

**Contributions.** The contributions and features of the proposed system are summarized as follows.

- A novel system that extracts experimental evidences from data science literature in PDF format. This builds up the first experimental database for related research.
- An effort-light framework that leverages both rule-based and learning-based methods to unify the tables of experimental results into (method, dataset, metric, score, source)-tuples.
- Capabilities for exploration and analysis over the structured knowledge to facilitate research and practice.



| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | Method | Dataset | Metric | Score | Source |
| 10 | UserMean | Epinions | MAE | 0.9319 | TOIS11-paper7-table3 |
| 11 | UserMean | Epinions | MAE | 0.9285 | TIST11-paper3-table3 |
| 12 | UserMean | Epinions | MAE | 0.9285 | WSDM11-paper12-table5 |
| 109 | ItemMean | Epinions | RMSE | 1.1973 | TOIS11-paper7-table4 |
| 110 | ItemMean | Epinions | RMSE | 1.2584 | TIST11-paper3-table3 |
| 111 | ItemMean | Epinions | RMSE | 1.2584 | WSDM11-paper12-table5 |
| 112 | Trust | Epinions | RMSE | 1.2132 | TIST11-paper3-table3 |
| 113 | NMF | Epinions | RMSE | 1.1832 | TOIS11-paper7-table4 |
| 114 | NMF | Epinions | RMSE | 1.1832 | TIST11-paper3-table3 |
| 115 | NMF | Epinions | RMSE | 1.1832 | WSDM11-paper12-table5 |
| 116 | SVD | Epinions | RMSE | 1.1812 | TOIS11-paper7-table4 |
| 117 | TCF | Epinions | RMSE | 1.1761 | TIST11-paper3-table3 |
| 118 | PMF | Epinions | RMSE | 1.1760 | TOIS11-paper7-table4 |
| 119 | PMF | Epinions | RMSE | 1.1760 | TIST11-paper3-table3 |
| 120 | PMF | Epinions | RMSE | 1.1760 | WSDM11-paper12-table5 |
| 121 | SoRec | Epinions | RMSE | 1.1492 | TOIS11-paper7-table4 |
| 122 | RSTE | Epinions | RMSE | 1.1256 | TIST11-paper3-table3 |
| 123 | RSTE | Epinions | RMSE | 1.1256 | WSDM11-paper12-table5 |
| 124 | SR1VSS | Epinions | RMSE | 1.1016 | WSDM11-paper12-table5 |
| 125 | SR1PCC | Epinions | RMSE | 1.1013 | WSDM11-paper12-table5 |
| 126 | SR2VSS | Epinions | RMSE | 1.0958 | WSDM11-paper12-table5 |
| 127 | SR2PCC | Epinions | RMSE | 1.0954 | WSDM11-paper12-table5 |
| 169 | SoRec | MovieLens | RMSE | ... | ... |

**Figure 2: The proposed system generates this experimental evidence database from data science paper PDFs. For a dataset and an evaluation metric, one can use the database to check what the state-of-the-art (highlighted in yellow) is and whether the reported numbers in existing research are consistent (green box) or conflicting (red box).**

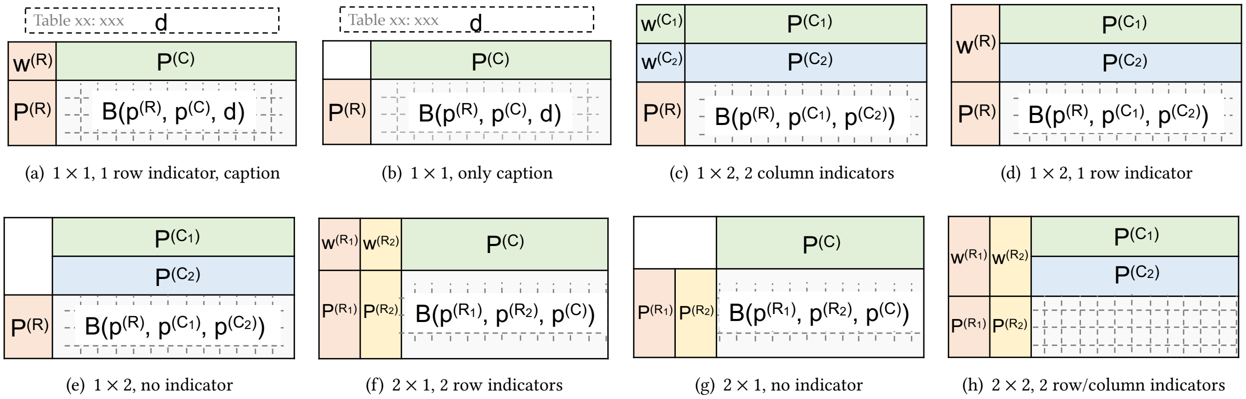
## 2 THE PROPOSED SYSTEM

In this section, we first introduce the workflow of our proposed system and explain the experimental evidence database it generates. Then we introduce details of the three modules of the system.

**Overview.** Figure 1 shows the overflow. The system collects a set of data science paper PDFs.[2] It has three modules to process the PDF data. It first crops the tables from PDFs, recognizes the table templates, and cleans the table data. Second, it classifies the row and column names of each table into three categories (method, dataset, metric). The experimental evidence database is constructed through the integration of table cells. Lastly, it designs database operations for knowledge exploration in the structured database.

**Expected output and impact.** Figure 2 shows a snapshot of the experimental evidence database. It has several examples of data records. They are experimental facts that can be found in tables of conference and journal papers on building recommender systems that were published in the same year: TOIS’11 [27], TIST’11 [25], and WSDM’11 [26]. The tables share popular method names such as “User Mean”, “NMF”, and “PMF”. Which method performs the best? Are the reported numbers of their performances consistent in these tables? When the tables were well structured into such a

[2] This is the research work of new method design, implementation, deployment, and evaluation, compared to a preliminary proof-of-concept demonstration [45].



(a) 1 × 1, 1 row indicator, caption (b) 1 × 1, only caption (c) 1 × 2, 2 column indicators (d) 1 × 2, 1 row indicator

(e) 1 × 2, no indicator (f) 2 × 1, 2 row indicators (g) 2 × 1, no indicator (h) 2 × 2, 2 row/column indicators

**Figure 3: Eight major table templates: We will use the first seven templates which cover more than 95% of the tables in our dataset. The cells in the table's body are triplets based on rows/columns/caption. (Best viewed in color)**

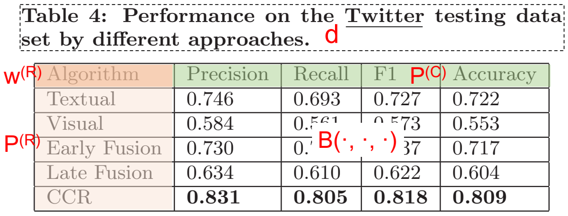


Table 4: Performance on the Twitter testing data set by different approaches.

| Algorithm | Precision | Recall | F1 | Accuracy |
|---|---|---|---|---|
| Textual | 0.746 | 0.693 | 0.727 | 0.722 |
| Visual | 0.584 | 0.561 | 0.573 | 0.553 |
| Early Fusion | 0.730 | [illegible] | [illegible]37 | 0.717 |
| Late Fusion | 0.634 | 0.610 | 0.622 | 0.604 |
| CCR | **0.831** | **0.805** | **0.818** | **0.809** |

**Figure 4: Table example in [44]: illustration of template (a).**

database, the above questions could be easily answered. The number of publications in the field of data science has been tremendously increasing because of the great use of data mining and machine learning in real applications. Practitioners are curious about what method will generate good performance on a specific task and dataset. Researchers are wondering whether the baseline methods are the state-of-the-art and whether the reported numbers on the baselines are correct when they review papers.

## 2.1 Table Extraction

We use Tabula to extract tabular content from PDF [14, 34]. Tabula was created by Manuel Aristaran *et al.* with the first release made available early 2013 as an open source project. The developers stated that they were inspired by academic papers [13, 43] about analysis and extraction of tabular content. Tabula is available as a Java library [3]. Unfortunately, it does not work for scanned documents, so we filter those files out.

**Table representation and templates.** A table $T = \{\mathcal{R}, C, d, \mathcal{B}\}$ has four components: (1) a model of horizontal *Rows* (identifiable by name) $\mathcal{R}$, (2) a model of horizontal *Columns* $C$, (3) *Caption* and the set of words in the caption $d$, and (4) cells (data elements) in the table's *Body* $\mathcal{B}$. We observe that the tables in our dataset can be categorized into eight major templates (with very few exceptions). Figure 3 visualizes the components of each templates and Table 2 presents symbolized definitions of the table's components. We use red/yellow colors to represent the *Rows* $\mathcal{R}$, green/blue colors to represent the *Columns* $C$, dashed block to represent the *Caption* $d$, and the grey area for the *Body* $\mathcal{B}$. Note that $\mathcal{R}$ or $C$ may have one or *two* rows ($R_1$, $R_2$) or columns ($C_1$, $C_2$). So the template "scale" could be 1 × 1 (templates a–b), 1 × 2 (c–e), 2 × 1 (f–g), and 2 × 2 (h).

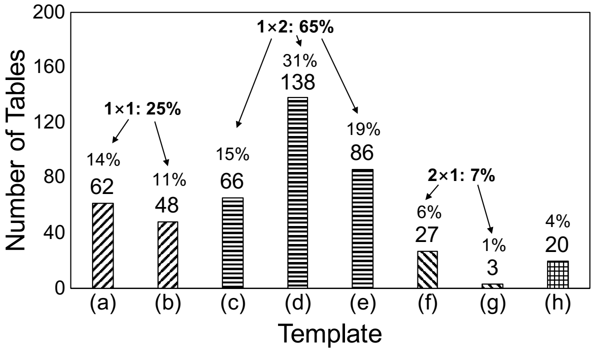


**Figure 5: The distribution of table templates.**

Let's use one of the templates, template (a), to explain the visualization and symbolized definition. Figure 4 presents the components of an example – Table 4 in [44]. On the rows, we have $\mathcal{R} = [w^{(R)}, P^{(R)}]$, where $P^{(R)}$ is the set of row names (e.g., "Early Fusion", "CCR") and $w^{(R)}$ is *an indicating word for row name's type*, simply called the "row indicator" (e.g., "Algorithm") that indicates the type of row-name concepts in $P^{(R)}$. On the columns, we have $C = [-, P^{(C)}]$ where $P^{(C)}$ is the set of column names. There is no "column indicator" word in template (a) though the column names are evaluation metrics, so $w^{(C)} =$ " – ". $d$ is the set of words in the caption. The "body function" $B(p^{(R)}, p^{(C)}, d) : P^{(R)} \times P^{(C)} \times d \to \mathbb{R}$ is the value in the cell as the intersect of row name $p^{(R)}$ and column name $p^{(C)}$, when the table has a caption $d$.

Here we present a few specific settings of table templates. First, templates (a–b) of the scale 1 × 1 have caption, while others have no caption. Second, templates (c–e) have two column models $C_1$ and $C_2$ and no caption, so the body function is $B(p^{(R)}, p^{(C_1)}, p^{(C_2)})$. Similarly, templates (f–g) have two column models $R_1$ and $R_2$ and no caption, so the body function is $B(p^{(R_1)}, p^{(R_2)}, p^{(C_1)})$. Third, template (c) has one column indicator for $C_1$ and $C_2$ and template (e) has no column indicator. Similarly, template (f) has one row indicator for $R_1$ and $R_2$ and template (g) has no row indicator. Lastly, the body functions should have *three variables/concepts*, and we expect their

[3] The Code of Tabula-Java could be found at https://github.com/tabulapdf/tabula-java

| | Template (a) | Template (b) | Template (c) | Template (d) |
|---|---|---|---|---|
| Rows $\mathcal{R}$ | $[w^{(R)}, P^{(R)}]$ | $[\text{-}, P^{(R)}]$ | $[\text{-}, P^{(R)}]$ | $[w^{(R)}, P^{(R)}]$ |
| Columns $C$ | $[\text{-}, P^{(C)}]$ | $[\text{-}, P^{(C)}]$ | $[w^{(C_1)}, P^{(C_1)}], [w^{(C_2)}, P^{(C_2)}]$ | $[\text{-}, P^{(C_1)}], [\text{-}, P^{(C_2)}]$ |
| Caption $d$ | $d$ | $d$ | - | - |
| Body $\mathcal{B}$ | $B(p^{(R)}, p^{(C)}, d)$ | $B(p^{(R)}, p^{(C)}, d)$ | $B(p^{(R)}, p^{(C_1)}, p^{(C_2)})$ | $B(p^{(R)}, p^{(C_1)}, p^{(C_2)})$ |

| | Template (e) | Template (f) | Template (g) | Template (h) |
|---|---|---|---|---|
| Rows $\mathcal{R}$ | $[\text{-}, P^{(R)}]$ | $[w^{(R_1)}, P^{(R_1)}], [w^{(R_2)}, P^{(R_2)}]$ | $[\text{-}, P^{(R_1)}], [\text{-}, P^{(R_2)}]$ | $[w^{(R_1)}, P^{(R_1)}], [w^{(R_2)}, P^{(R_2)}]$ |
| Columns $C$ | $[\text{-}, P^{(C_1)}], [\text{-}, P^{(C_2)}]$ | $[\text{-}, P^{(C)}]$ | $[\text{-}, P^{(C)}]$ | $[\text{-}, P^{(C_1)}], [\text{-}, P^{(C_2)}]$ |
| Caption $d$ | - | - | - | - |
| Body $\mathcal{B}$ | $B(p^{(R)}, p^{(C_1)}, p^{(C_2)})$ | $B(p^{(R_1)}, p^{(R_2)}, p^{(C)})$ | $B(p^{(R_1)}, p^{(R_2)}, p^{(C)})$ | N/A |

**Table 2: Symbolized definitions of a table’s four components (rows, columns, caption, and body) for each template in Figure 3.**

types include all the three labels “method”, “dataset”, and “metric” to position the cell value. So template (a–g) have valid body functions but (h) does not have it.

**Distribution of table templates.** Among the 456 tables we have, **450** tables can be matched to the eight templates (at a high rate of **98.7%**). Figure 5 presents the number of tables in our dataset (450 tables) for each template. Templates (a–g) take as many as 96% of the tables, so there will not be a serious problem if we drop the tables of template (h) that has no valid body function. Templates of scale 1 × 1 (a–b) take 25%; templates of scale 1 × 2 (c–e) take 65%, and templates of scale 2 × 1 (f–g) take 7%. We will unify the tables of templates (a–g) for cell value integration and experimental result database construction.

## 2.2 Table Unification

Based on the template representation, we define the set of concept items that can be found as row names or column names:

$$\mathcal{P} = \cup_{T=[\mathcal{R},C,d,\mathcal{B}]} P^{(R_{(:)})} \cup P^{(C_{(:)})}, \tag{1}$$

where $T$ is a table, $P^{(R_{(:)})}$ is the set of row names (no matter single row or double rows), and $P^{(C_{(:)})}$ is the set of column names. We denote by $\mathcal{L}$ by the set of three labels for the concept items:

$$\mathcal{L} = \{\text{“}method\text{”}, \text{“}dataset\text{”}, \text{“}metric\text{”}\}. \tag{2}$$

Then we define table unification as a two-step problem.

Problem (Table unification). ***Given*** *a set of tables* $\{T\}$ *and each table has been well defined based on its template (as shown in Table 2), (1)* ***classify*** *the concepts into three categories, or say,* ***find*** *a classification function* $f : \mathcal{P} \rightarrow \mathcal{L}$*; (2)* ***unify*** *the cells into (method, dataset, metric, score)-quadruples, or say,* ***find*** *a function of three variables* $g$*:* $P^{(\text{“}method\text{”})} \times P^{(\text{“}dataset\text{”})} \times P^{(\text{“}metric\text{”})} \rightarrow \mathbb{R}$*, where the target value is a score (a real number) in the Table’s body function* $B$*.*

We assume that for data science papers, each concept in $\mathcal{P}$ has a label in $\mathcal{L}$. So $\mathcal{P}$ is the union of three *exclusive* sets of concepts:

$$\mathcal{P} = P^{(\text{“}method\text{”})} \cup P^{(\text{“}dataset\text{”})} \cup P^{(\text{“}metric\text{”})}, \tag{3}$$

where $P^{(l)} = \{p | f(p) = l\}, \forall l \in \mathcal{L})$.

Take Figure 4 as an example. The body function is $B(p^{(R)}, p^{(C)}, d)$, so for the first cell in the table body “0.746" is the output of $B$ when $p^{(R)} = \text{“}Textual\text{”}$, $p^{(C)} = \text{“}Precision\text{”}$, and $d = \text{“}Twitter\text{”}$. (In our method design, we assume the word “Twitter" can be found in other tables and predicted to be a “dataset", so we can match it in the caption to have this concept in the body function.) The first-step of this problem is to predict the label of the three concepts. We expect the output to be $f(\text{“}Textual\text{”}) = \text{“}method\text{”}$, $f(\text{“}Precision\text{”}) = \text{“}metric\text{”}$, and $f(\text{“}Twitter\text{”}) = \text{“}dataset\text{”}$. Then the second step will align the concepts in $B$ with the function $g$. We will have a value in function $g$: $g(\text{“}Textual\text{”}, \text{“}Twitter\text{”}, \text{“}Precision\text{”}) = 0.746$, which can be easily transformed to one row in the experimental result database (as shown in Figure 2) and added with the source of this information as an additional column.

We will address this problem in Section 3.

## 2.3 Database Operations for QA

Once the experimental result database was constructed by module 1 (extraction) and 2 (unification), we would be able to use SQL statements to answer interesting questions from researchers and practitioners in the data science field. There could be many questions and corresponding SQL queries. Here are three examples.

Question 1. *How many methods were used/proposed on the Epinions dataset? And how many metrics were used?*

Question 2. *What are the top three methods on the Epinions dataset if the evaluation metric is RMSE?*

Question 3. *Are there conflicting reported numbers in the database? What are they?*

SQL consists of many types of expressions, predicates and statements such as *select*, *join*, and *distinct*, based upon *relational algebra* and *tuple relational calculus*. Suppose the experimental result data table is constructed and named as “ERD". Here are the SQL queries that find answers to the above questions.

SQL Queries 1.
select count(distinct Method) from ERD where Dataset=“Epinions”;
select count(distinct Metric) from ERD where Dataset=“Epinions”;

SQL Query 2. select * from ERD where Dataset = “Epinions” and Metric = “RMSE” order by Score desc limit 3;

SQL Query 3. select distinct d1.Method, d1.Dataset, d1.Metric, d1.Score, d1.Source from ERD as d1, ERD as d2 where d1.Method = d2.Method and d1.Dataset = d2.Dataset and d1.Metric = d2.Metric and d1.Score <> d2.Score order by d1.Method, d1.Dataset, d1.Metric;

The term count is used for question “how many”; order by is used for ranking/finding “top three”; and the third query uses *self-join* to compare values in a column ("Score") with other values in the same column in the same table ("ERD").

We developed user-friendly functions to answer the questions. For example, users can fill the values in the questions, the SQL queries will be updated, and then answers will be returned.

# 3 THE PROPOSED METHOD

In this section, we address the problem defined in Section 2. We first give an overview of our approach and then present its details.

## 3.1 Overview

Our proposed approach has two parts. The first part *concept classification* is to classify the concepts (i.e., row/column names) into the three aforementioned categories. The second part *tuple extraction* is to extract the quadruples from table cells after the row/column names are well categorized. When the first part is done, the second part is not hard to do. Details will be given in Section 3.4.

For the first part, we propose an ensemble method that iteratively use two different methodologies (and two different classifiers) to predict the concept's label: one is a rule-based method and the other is learning-based. It has been widely observed and accepted that these two methodologies are complementary in solving real application problems. For example, a successful auto-driving car combines rule-based control system based on driving policies and deep learning techniques with big data for end-to-end decision making on the road [3, 22].

The first methodology is to predict the probability of a row/column name $p$ belonging to the class-specific set $P^{(l)}$ ($l \in \mathcal{L}$), which is denoted as $\phi(p \in P^{(l)}) : \mathcal{P} \times \mathcal{L} \rightarrow \mathbb{R}$, based on the three assumptions (see the introduction section). Given a set of concepts that have been labelled, we predict the unknown $\phi$ scores for the remaining concepts. Only when $\phi(p \in P^{(l^*)})$ is ranked at the top $\alpha$ among all the remaining concepts, where $l^* = argmax_l \phi(p \in P^{(l)})$, we set $f(p) = l^*$, i.e., add $p$ into the set of labelled concepts.

The second methodology is to predict $\phi(p \in P^{(l)})$ by training a supervised learning model with the set of labelled concepts. The idea is to extract useful features of the concepts from the data including both paper text and tabular structure. We also use the parameter $\alpha$ to control the amount of newly labelled concepts for high precision.

The iterative process needs to be initialized with a set of seed labelled concepts though the set could be very small. We apply Assumption 1 with just one header indicator word for each label: "Methods" for label "method", "Datasets" for label "dataset", and "Metrics" for label "metric". We use the top five frequent concepts (as row/column names) per label type as seeds, where the frequency is defined as the number of tables that have the concept whose header is the corresponding indicator word. Here are the seed concepts (and their frequencies in the brackets) for each label type:
"method": SVM (100), LR (72), RF (64), KNN (56), DT (42);
"dataset": Amazon (34), Wiki (30), DBLP (30), Iris (18), Google (16);
"metric": Precision (120), Recall (104), F1 (32), MAP (32), MAE (20).

In our ensemble learning approach, we adopt the *boosting* strategy to iteratively learn the above two classifiers that are relatively weak as individuals and add them to a final strong classifier.

## 3.2 Rule-based Classifier with Tabular Structure Assumptions

We will give the objective function that optimizes the predictor $\phi(p \in P^{(l)})$ for each of the assumptions.

Assumption 1 (Row/column header indication). *If the upper-leftmost cell of the table has a specific word (e.g., "Methods", "Dataset"), the names on the corresponding columns/rows are more likely to have the label as the word indicates.*

**Objective 1.** The first assumption utilizes the indicator words on the row/column headers to predict the label of row/column names. Suppose we have the set of all the indicator words as below:

$$\mathcal{W} = \cup_{T=[\mathcal{R},\mathcal{C},d,\mathcal{B}]}\{w^{(R_{(:)})} \in \mathcal{R}, w^{(C_{(:)})} \in \mathcal{C}\}. \tag{4}$$

Similarly as Eq.(1) and (3), each word in $\mathcal{W}$ indicates one label in $\mathcal{L}$. So $\mathcal{W}$ is the union of three *exclusive* sets of words:

$$\mathcal{W} = W^{(\text{“}method\text{”})} \cup W^{(\text{“}dataset\text{”})} \cup W^{(\text{“}metric\text{”})}. \tag{5}$$

Based on Assumption 1 ("Row/column header indication"), we have the following objective, which is the least square of error between the probabilities of label prediction $\phi$ and word indication $\psi$:

$$\begin{aligned}\min_{\phi,\psi} J_1(\phi,\psi) \quad = \quad & \sum_{T=[\mathcal{R},\mathcal{C},\dots]} \sum_{(w,P)\in\mathcal{R}\cup\mathcal{C}} \sum_{l\in\mathcal{L}} \\ & \left(\sum_{p\in P} \phi(p \in P^{(l)}) - |P| \cdot \psi(w \in W^{(l)})\right)^2, \end{aligned} \tag{6}$$

where $\psi(w \in W^{(l)}) : \mathcal{W} \times \mathcal{L} \rightarrow \mathbb{R}$ is the probability of the word $w$ indicating the label $l$. For each table $T$, $(w, P)$ can be found in rows $\mathcal{R}$ and columns $\mathcal{C}$ in Table 2.

Assumption 2 (Row/column type consistency). *The concept items on the same column/row are likely to have the same type of label. For example in Figure 4, if we know "Precision" is a "metric", then "Recall" is likely to be a "metric".*

**Objective 2.** The second assumption suggests that for a specific table, the names on the rows/columns should have the same label type. For the set of names on the rows/columns of table $T$, we first find the most frequent label type:

$$l^*(P) = argmax_l \sum_{p\in P} \phi(p \in P^{(l)}), \tag{7}$$

where $P$ is in either $\mathcal{R}$ or $\mathcal{C}$ for table $T$. Based on Assumption 2 ("Row/column type consistency"), we measure the type consistency as below and we will maximize the consistency by optimizing $\phi$:

$$\max_{\phi} J_2(\phi) = \sum_{T=[\mathcal{R},\mathcal{C},\dots]} \sum_{P\in\mathcal{R}\cup\mathcal{C}} \sum_{p\in P} \phi(p \in P^{(l^*(P))}), \tag{8}$$

where $\phi(p \in P^{(l^*(P))})$ is the probability that the label of concept $p$ is consistent with the label of the majority of the concepts on the same row/column $l^*(P)$. If the consistency is higher, the rows/columns are more likely to have good purity in terms of the label types.

Assumption 3 (Cell context completeness). *A table often covers all the three types of labels on its columns, rows, and caption, in order to provide complete contexts to explain the values in the cells. For example, if the caption has a metric name (i.e., "MAE") and the row names are methods, then the column names are likely to be datasets.*

**Objective 3.** In Table 2, for each table template, we represent the cell values as a function of the table body $\mathcal{B}$. This function has three variables. For example, the table body function of template (a) is $\mathcal{B}(p^{(R)}, p^{(C)}, d)$: the three variables are a row name, a column name, and the table's caption. The function for template (c) is $\mathcal{B}(p^{(R)}, p^{(C_1)}, p^{(C_2)})$: the variables are a row name and a column name for each of the two column headers. The idea of Assumption 3 is that when researchers carefully presented the tables, each table cell can explain a fact with *full contexts* including method, dataset, and metric. Therefore, if we have known the label types of two of the variables, we can use the completeness assumption to infer the label type of the third. For example, in Figure 4, if we know the label of the row names is *method* (because of the indicator word "Algorithm") and we find a *dataset* name "Twitter" in the table's caption, we can infer that the label type of the column names such as "Precision", "Recall", or "F1" is *metric*.

Now we denote $B_k$ $(k = 1, 2, 3)$ as the three variables of table body function $\mathcal{B}$. Each of them is actually a set of concept names (in the rows, columns, or caption). We first find the most frequent label type $l^*_k$ for each variable $B_k$:

$$l^*_k = argmax_l \sum_{p \in B_k} \phi(p \in P^{(l)}). \tag{9}$$

Based on the above assumption, we have the following objective function to optimize $\phi$:

$$\max_{\phi} J_3(\phi) = \sum_{T=[\ldots,\mathcal{B}(B_1,B_2,B_3)]} |\cup_{k\in\{1,2,3\}}\ l^*_k|. \tag{10}$$

Because $|\mathcal{L}| = 3$, maximizing $J_3$ is equivalent to making $l^*_k$ $(k = 1, 2, 3)$ different from each other.

*Optimization.* Joint optimization of the three objectives has a high computational complexity. Therefore, we adopt the strategy of greedy algorithms to iteratively optimize each objective, find locally optimal choice at each stage, and terminate in a reasonable number of steps. The idea is to build a classifier that utilizes the *information inside the tabular structures* for label prediction. Its individual performance could be a bit weak but the final prediction will be significantly improved by the ensemble framework (including the complementary, learning-based classifier in Section 3.3).

## 3.3 Learning-based Classifier with Semantic and Structural Concept Embeddings

The idea of the learning-based classifier is to consider the task as a standard classification problem and solve it in two steps: feature extraction and supervised learning. For each iteration in the proposed ensemble framework, the output of the former classifier (i.e., the rule-based classifier) provides training labels as supervision. So we will focus on how to generate features of the concepts and what classification models we use for training and prediction.

We use two kinds of low-dimensional representations of the concepts and concatenate them into a long feature vector.

**Structural concept embeddings.** We construct the table body function $\mathcal{B}$ into a hyper-edge heterogeneous network. This network has two types of nodes: one is concept on the rows, columns or in the caption; the other is cell value. Each cell value is connecting to three concepts that are the three objects in the function $\mathcal{B}$. We use HyperEdge-Based Embedding (HEBE) [15] to learn object (concept) embeddings with events (quadruples) in heterogeneous information network that models proximity among objects in each event. The insight is that structurally related objects are more likely to participate in the same event. For instance, in Data Science papers, it is more frequently to observe performance score with "CNN" method and "F-score" or "Accuracy" metrics in "MINST" dataset.

HEBE learns a function $\mathcal{M}$ that projects each object to a low dimension space $\mathbb{R}^d$ that preserves structural roles of each concepts participating in a quadruple of generating the cell value in the experimental result table, where $d \ll |\mathcal{P}|$, i.e, $\mathcal{M} : \mathcal{P} \to \mathbb{R}^d$. HEBE aims to predict a target object out of all alternative objects given the other participating objects on the same hyperedge as context. We denote the target object $u$, context concepts set as $C$. Obviously, $|C| = |\mathcal{L}| - 1$ for quadruples and $u \notin C$. The conditional probability of predicting the target object $u$ is defined as:

$$P(u|C) = \frac{exp\big(S(u, C)\big)}{\sum_{p\in\mathcal{P}} exp\big(S(p, C)\big)}, \tag{11}$$

$$S(u, C) = cosine\Big(\mathbf{w}_u, \big((|\mathcal{L}| - 1)^{-1} \sum_{l=1}^{\mathcal{L}} \mathbf{w}_{c_l}\big)\Big), \tag{12}$$

where $\mathbf{w}_u \in \mathbb{R}^d$ is the embeddings of $u$, $S(\cdot)$ is a scoring function reflecting the similarity between target object $u$ and contextual concepts $C$. Suppose $C = \{c_1, ..., c_{|\mathcal{L}|-1}\}$. Intuitively, Equation (11) could be understood as given contextual concepts $C$ selecting $q$ from the pool of concept candidates.

Then, we take all unlabelled concepts $D = \{d_1, d_2, ..., d_m\}$ as candidates, where $d_m$ is $m$-th concept in the set of unlabelled concepts. Then we retrieve $m$ concepts in $D$ to generate a ranking $R$, s.t. higher probabilities appear at the top of the list. In the experiment, we choose a simple method *Top-$\alpha$* to select similar concepts, where $\alpha$ could control the amount of new labelled concepts with highest probabilities during each iteration in order to reduce errors propagation.

**Semantic concept embeddings.** Our dataset was collected from a specific domain, i.e., Data Science, which is why we observed that pre-trained embeddings (on general corpora like Wiki and news) could not give satisfactory performance. We fine-tuned word embeddings with the *paper's full text* data by the state-of-art language model [11]. Note that the concepts on the table's rows/columns, no matter they are words or phrases, were regarded as units in training. These embeddings carry the semantic information of the concepts in the paper text through the introduction, methodology, to experiment sections. The advantage of semantics compared with other information/assumption we use is that the concepts of similar meanings have similar semantic embeddings. For example, the classifier will be aware of the potential acronyms/abbreviations such as "Prec." and "Precision", "F-score" and "F-measure".

For each concept $p \in \mathcal{P}$, we learn a low dimensional vector $\mathbf{v}_p$. Therefore, we predict $\phi(p \in \mathcal{P}^{(l)})$ by training a classification model $g$ (e.g, logistic regression, random forest) with all labelled concepts obtained after each rule-based iteration, then take all labelled concepts as candidates $D = \{\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_n\}$, where $\mathbf{v}_n$ is the vector of $n$-th concept in the set of labelled concepts. For each unlabelled concept $p_m$ where $m = n + 1, ..., |\mathcal{P}|$, we retrieve $n$

| Method | Micro F1 | Avg. Precision | Avg. Recall | Macro F1 | Micro AUC | Macro AUC |
|---|---|---|---|---|---|---|
| TableUni-R | 0.6908 (0.0040) | 0.6479 (0.0044) | 0.6807 (0.0058) | 0.6542 (0.0047) | 0.8879 (0.0023) | 0.8601 (0.0029) |
| TableUni-L | 0.6333 (0.0024) | 0.5921 (0.0021) | 0.6187 (0.0023) | 0.6072 (0.0021) | 0.7611 (0.0033) | 0.7264 (0.0035) |
| TableUni-(R+E1) | 0.7505 (0.0039) | 0.7007 (0.0049) | 0.7443 (0.0018) | 0.7115 (0.0053) | 0.8901 (0.0018) | 0.8705 (0.0027) |
| TableUni-(R+E2) | 0.8175 (0.0021) | 0.7821 (0.0025) | 0.7777 (0.0035) | 0.7798 (0.0029) | 0.9087 (0.0017) | 0.8920 (0.0018) |
| TableUni-(A1+L) | 0.6980 (0.0024) | 0.6531 (0.0025) | 0.6756 (0.0029) | 0.6612 (0.0026) | 0.8316 (0.0027) | 0.8123 (0.0028) |
| TableUni-(A2+L) | 0.7567 (0.0037) | 0.7123 (0.0046) | 0.7250 (0.0047) | 0.7179 (0.0046) | 0.8788 (0.0023) | 0.8633 (0.0024) |
| TableUni-(A3+L) | 0.6474 (0.0032) | 0.6052 (0.0035) | 0.6306 (0.0037) | 0.6129 (0.0038) | 0.7766 (0.0039) | 0.7443 (0.0052) |
| **TableUni-(R+L)** | **0.8307** (0.0022) | **0.8195** (0.0025) | **0.8053** (0.0024) | **0.8104** (0.0023) | **0.9112** (0.0011) | **0.9000** (0.0013) |

**Table 3: Performances on classifying the concepts into three categories (dataset, method, and metric): We compare the proposed method with its multiple variants. We report the *mean* as well as the *standard deviation* in the brackets.**

concepts in $D$, use above neutral network classifier $g$ to generate an optimal ranking $R$, s.t. similar concepts appear at the top of the list. Similar with structural concept embeddings, we choose a simple method *Top-$\alpha$* to select similar concepts, where $\alpha$ could control the amount of new labelled concepts with highest probabilities during each iteration in order to reduce errors propagation. A detailed parameter insensitivity discussion of $\alpha$ is in Section 4.1.2.

## 4 EXPERIMENTS

In this section, we first present experimental results to demonstrate the effectiveness of the proposed solution (called **TableUni**) to the problem of **table unification** (in Section 2.3). We also study a few cases of discovering quantitative knowledge in the literature.

### 4.1 Experiments on Annotated Data before Deployment

We first introduce a carefully-annotated dataset and the setting of experiments including ground truth, evaluation methods, competitive methods, and parameter settings. Then we present method performances and give observations and analysis.

*4.1.1 Experimental Settings.*

**Data description.** We downloaded from web portals such as ACM Digital Libraries a PDF file collection of four data science conference proceedings (WWW, SIGKDD, ICDM, and WSDM) and three ACM transactions (TOIS, TIST, and TKDD) in the decade (2008–2017). After careful PDF converting, cropping, and cleaning, we have **450** tables on experimental results for the task of database building.

We carefully label the concepts in the 450 tables. The total number of concepts is 3,992 and the total count of concepts is 10,944. So the average number of concepts per table is 24, and the average number of unique concepts per table is 9.77.

We recruited three volunteers to manually label the concepts into the three classes, {dataset, method, metric}, and used the strategy of majority voting to find the suitable label. If there was a tie of votes, we had another volunteer to make the decision. With heavy human efforts, we have the final set of labelled concepts as ground truth: 1,728 datasets, 1,803 methods, and 461 evaluation metrics.

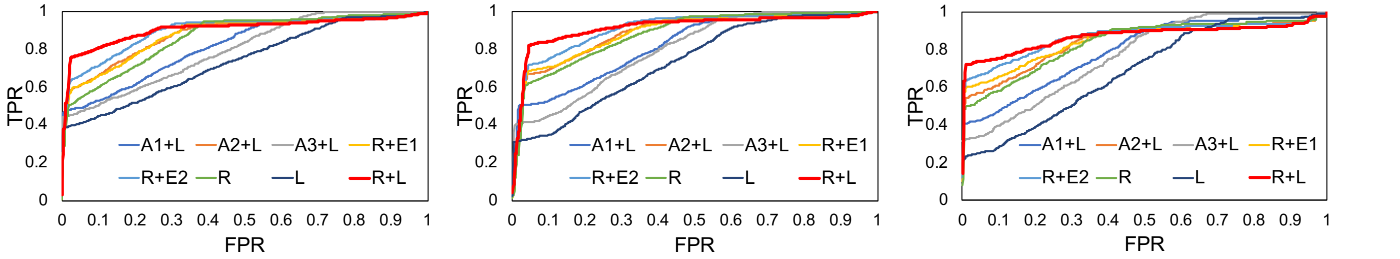
**Evaluation metrics.** As it is a standard multi-class classification task, firstly, for each type of classes $l \in \mathcal{L}$, we calculate Precision and Recall, and report *Avg. Precision* and *Avg. Recall*. Secondly, we calculate the F1 score which is the harmonic average of the precision and recall. We use *Micro F1* which globally counts the TPs, FNs, FPs, and TNs. In our case, because all the concepts were assigned to exactly one class in the ground truth, the *Micro F1* is the same as *Accuracy*. We also use *Macro F1* which is the unweighted mean of the F1 scores per type of classes. Moreover, we plot the *Precision-recall curve* per type of classes as well as the Receiver Operating Characteristic Curve (ROC). We calculate the Area under the ROC (AUC) for evaluation including *Micro AUC* and *Macro AUC*. For all the metrics above, bigger score means better performance.

**Competitive methods.** We will compare our proposed method TableUni with different settings of the components. The ensembled method includes a **R**ule-based classifier and a **L**earning-based classifier. We have three series of method variants: (1) TableUni-R/L is a non-ensembled method. (2) TableUni-(R+E1/E2) is an ensembled method that has a full rule-based classifier and a learning-based classifier using one of the embeddings. (3) TableUni-(A1/A2/A3+L) is an ensembled method that has a full learning-based classifier and a rule-based classifier using one of the assumptions. Lastly, TableUni-(R+L) is the proposed method of full settings.

*4.1.2 Experimental Results.*

**Rule-based vs Learning-based vs Ensembled.** The ensembled method is TableUni-(R+L), and the rule/learning-based only method is TableUni-R/L. First, we observe that TableUni-R performs better than TableUni-L in terms of all the metrics: relatively **+5.75%** on Micro F1 and **+4.70%** on Macro F1. So, the rule-based method (using three assumptions) is more effective than the learning-based method. Table structures are important for predicting the concept types. Second, we observe that the ensembled method significantly outperforms the rule/learning-based only method: relatively **+13.99%** on Micro F1, **+15.62%** on Macro F1, **+17.16%** on Avg. Precision, **+12.46%** on Avg. Recall than TableUni-R and relatively **+19.74%** on Micro F1, **+20.32%** on Macro F1, **+22.74%** on Avg. Precision, **+18.66%** on Avg. Recall than TableUni-L, respectively. We reasonably come to the conclusion that both classifiers are important: by combining the rule-based and learning-based methods, we can have a much more satisfactory performance.

**Rule-based: One vs all assumptions.** Here we compare the TableUni methods (TableUni-A1/A2/A3/R+L) that use one of the assumptions or all of them in the rule-based classifier. We still adopt the ensembled strategy and use the standard learning-based part



(a) Dataset

(b) Method

(c) Metric

**Figure 6: ROC curves comparing the variants of our proposed TableUni methods with respect to the type of classes.**

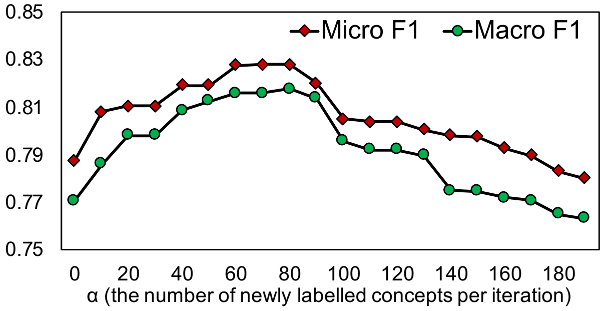


**Figure 7: Our proposed TableUni-(R+L) method is insensitive to the parameter $\alpha$. F1 scores are higher than any of the variants or baseline methods when $\alpha \in [10, 200]$.**

(E1+E2). Compared with TableUni-L that does not use any assumption, TableUni-(A1+L) improves Micro F1 and Macro F1 relatively by **+6.74%** and **+5.69%**, respectively; TableUni-(A2+L) improves the two metrics relatively by **+12.34%** and **+11.07%**, respectively; and TableUni-(A3+L) improves them by **+1.41%** and **+0.57%**. We conclude that (1) A2 (Type consistency) plays the most significant role in predicting the concept type; (2) though the improvement brought by A3 (Completeness) is not significant, all the assumptions have positive impact on the prediction. The reason of the first point is that A2 can infer the type of concepts on an entire row/column if one of the concepts has been labelled by A1/A3, while the coverage of A1 and/or A3 is limited. Only a small portion of tables have indicator words. For the second point, A3 mainly contributes to table template (a): If we only look at the Micro F1 on template (a), it is improved relatively by as much as **+10.1%** over A1+A2 only.

**Learning-based: Semantic embedding vs structural embedding vs concatenated.** Here we compare TableUni-(R+E1/E2/L) methods: they all adopt the ensembled framework and rule-based classifier but use one of the embeddings (E1/E2) or both (L). We observe that TableUni-(R+E2) achieves relatively **+5.10%** Micro F1 and **+4.48%** Macro F1 over TableUni-(R+E1). So structural embeddings play a more significant role than semantic embeddings in this task. Concept co-occurrences may reflect more nature of the common type than similar meanings in the text. We also observe that TableUni-(R+L) improves significantly higher than either of them: relatively **+4.04%** Micro F1 and **+6.79%** Macro F1. This demonstrates that concatenated embeddings perform the best for its combination of the semantic and structural information.

**Performances on each class type.** Figure 6 presents the ROC curves of the variants of our proposed TableUni with respect to the three concept types, respectively. TableUni-(R+L) performs the best for its full ensemble learning from rule-based and embedding-based methods. The difficulty levels of the types (from highest to lowest) are method, dataset, and metric.

**Performances on parameter insensitivity.** Figure 7 presents the Micro F1 and Macro F1 scores of TableUni-(R+L) when $\alpha$ varies from 10 to 200. We observe that our method performs better than any of the variants or baselines (with higher-than-0.80 Micro F1 and Macro F1). When $\alpha \in [60, 80]$, the scores are the best.

## 4.2 Experiments on User Evaluation after Deployment

**Deployment.** Users can upload their paper PDFs in the field of data science to the system. The system will extract tables, recognize the templates, classify row/column names, and return the list of (method, dataset, metric, score)-quadruples to the users. With the tuple list, the system will do two things. One is data collection – it integrates the list of new tuples with those existing in the system so that the database becomes bigger and would have more information and become more accurate on table extraction and unification. The other is to support QA – users can ask questions related to the uploaded document. The system will translate the questions into database queries and return the answers through operations.

**User evaluation design.** Forty (40) students in the Data Science class were invited to test our system. They uploaded data science papers they collected (which were related to the topics of their course projects such as recommender systems, fake news detection, and game result prediction) to test the system's performance. After they submitted a paper PDF and got returned with extracted tables and answers to their queries (as given in Section 2.3), we asked them a few questions:

***Q1-1:*** *Is the extracted and (5-column) unified table correct?*

The options are (1) *Correct* and (2) *Incorrect*. If the answer was *Incorrect*, we ask a following question: ***Q1-2:*** *What are the reason(s) of incorrectness?* The options are (1) *Failure* if no result could be given; (2) *Missing Cell Value*; (3) *Misplaced Cell Value*, i.e., the row/column positions of cell value was false recognized; and (4) *Wrong Cell Value*, for example, text values were filled in the cells that were supposed to be numbers.

| Method | Poor | Fair | Good | Excellent |
|---|---|---|---|---|
| TableUni-R | 11 | 212 | 321 | 136 |
| | 1.6% | 31.2% | 47.2% | 20.0% |
| TableUni-L | 27 | 244 | 297 | 112 |
| | 4.0% | 35.9% | 43.7% | 16.5% |
| **TableUni-(R+L)** | 6 | 106 | **391** | **177** |
| | 0.9% | 15.6% | **57.5%** | **26.0%** |

**Table 4: TableUni-(R+L) performs the better than TableUni-R and TableUni-L because it uses both rule-based and learning-based classifiers. 40 users submitted 680 queries in total; over 83% were rated as "Good" or "Excellent".**

*Q2: How is your search & QA experience?*

The options are (1) *Poor* if no answer was returned, (2) *Fair* if answers were given but sometimes incorrect, (3) *Good* if answers were correct but not complete, and (4) *Excellent* if answers were correct and complete. Moreover, uses could optionally submit some subjective comments. In order to verify our experimental results, we present our three different results of TableUni-L, TableUni-R, TableUni-(R+L) randomly, and invite users to evaluate the qualities.

**Evaluation results on correctness (feedback on Q1-1).** It is not easy to perfectly extract the tuples from PDF because of the complicated process including table cropping, template recognition, and column/row name classification. 40 users submitted **728** papers, and **1003** tables in total. **68%** were labelled as correct unified tables. We analyze the reasons of the 32% not-completely-correct case.

**Evaluation results on reason of incorrectness (feedback on Q1-2).** Among the 1003 tables from 728 user submitted papers, 321 tables were reported as incorrect. Among the 321 tables, 35 were reported as *Failure* – no results were returned (11% among incorrect tables / 3.5% among all the tables). For the other three reasons, 164 tables have *Misplaced Cell Values* problem (51.2% / 16.4%); 96 tables have *Missing Cell Values* problem (29.9% / 9.6%); and 26 tables have *Wrong Cell Values* problem (8.1% / 2.6%). *Merged cell* is the main reason that a table was not correctly cropped or extracted. In the future, we will develop more table templates to improve the extraction.

**Evaluation results on search & QA experience (feedback on Q2).** We observed that rule-based classifier (TableUni-R) has good precision but the recall is low; learning-based classifier (TableUni-L) has good coverage but the errors are more frequent than rule-based results. Overall, the rule-based classifier has better performance than the learning-based classifier because users are more sensitive to incorrect results than missing information. The ensemble method TableUni-(R+L) takes the advantages of both methods and generates good precision and good recall. So the evaluation is much better than the other two classifiers.

Suppose we count Poor as 1 point, Fair as 2 points, Good as 3 points, and Excellent as 4 points. TableUni-R makes an average of 2.86. TableUni-L makes an average of 2.73. And TableUni-(R+L) makes an average score of **3.09**. Fortunately, all the methods achieve an average score bigger than 2.50 – users are satisfactory with the performances. Only the proposed TableUni-(R+L) has a bigger-than-3 average score, showing that it is generally better than Good.

## 4.3 Case Studies on Usage

We will give data statistics of the database we build using the proposed framework. When all of three modules are implemented, we can use the SQL queries to answer the questions we listed in Section 2.3. We present the answers given by the database we have.

**Statistics of the resulting database.** After table unification, our method has categorized the 3,992 concepts into three classes, {dataset, method, metric}. We transform each of the cells in the tables into a value of the function $f$, or say, a data record in the final experimental result database (ERD): The data record must have one dataset name, one method name, one metric name, and a corresponding score; otherwise, it is invalid and not included in the database.

Currently, the resulting database has as many as **35,137** data records (or called experimental result facts) from 450 tables in PDF files. The database includes (a) **1,728** unique datasets in the tables, (b) **1,803** unique methods in the tables, and (c) **461** unique metric names in the tables. The count of metric looks incredibly big because we do not merge similar metrics without prior knowledge, such as "p@1", "prec@5", and "precison". Each dataset, method, and metric has **18.9**, **17.3**, and **64.6** related data records in average, respectively. Associations among the concepts are rich.

We will use the database to answer the following questions. This is just to show the power of exploring quantitative knowledge in the experimental result database and the usefulness of our approach. Because the database was constructed with only 450 tables, we are **NOT** claiming that the answers to these questions are the truths all over the tons of literature.

***Question 1:* Find related methods, metrics, and datasets.**

**Q-1(a)** How many methods were used for the Epinions dataset?
select count(distinct Method) from ERD where Dataset="Epinions";
**A-1(a)** 36. If one uses more SQL queries to look for the detail, one will see the method names such as "UserMean", "ItemMean", "Trust", "NMF", "SVD", "TCF", "PMF", "SoRec", and "RSTE".

**Q1(b)** How many metrics were used to evaluate on Epinions?
select count(distinct Metric) from ERD where Dataset="Epinions";
**A-1(b)** 7. More queries will find the concrete metric names such as "F1 score", "Precision", "Recall", "MAE", and "RMSE".

**Q1(c)** How many datasets used with Epinions in the same table?
select count(distinct Dataset) from ERD where Source=(select (distinct Source) from ERD where Dataset= "Epinions");
**A-1(c)** 17. The data names are "Amazon", "Ciao", "Douban", and so on. They are popular datasets for evaluating recommender systems.

**Q-1(d)** How many methods were used for the Amazon dataset?
select count(distinct Method) from ERD where Dataset="Amazon";
**A-1(d)** 70. The method names include LDA (Linear Discriminant Analysis), LR (Logistic Regression), and so on.

**Q1(e)** How many metrics were used to evaluate on Amazon?
select count(distinct Metric) from ERD where Dataset="Amazon";
**A-1(e)** 15. ("Precision", "Recall", "F1", "Accuracy", etc.)

**Q1(f)** How many datasets used with Amazon in the same table?
select count(distinct Dataset) from ERD where Source=(select (distinct Source) from ERD where Dataset= "Amazon");
**A-1(f)** 53. ("DBLP", "Wiki", "Delicious", "Epinions", etc.)

***Question 2:* Find top-performing methods on a dataset.**

**Q2(a)** What are the top 3 methods on Epinions in terms of RMSE?
select Method, Score from ERD where Dataset = "Epinions" and Metric = "RMSE" order by Score desc limit 3; // desc is for the fact that a smaller RMSE means a better performance.
**A-2(a)** "SR2pcc" (1.0954), "SR2vss" (1.0958), "SR1pcc" (1.1013).

**Q2(b)** What are the top 3 methods on Amazon in terms of F1?
select Method, Score from ERD where Dataset = "Amazon" and Metric = "F1" order by Score limit 3; // Compared to Q2(a), desc was deleted because a bigger F1 means a better performance.
**A-2(b)** "LEMON" (0.953), "LEMON-auto" (0.91), "LC" (0.815).

***Question 3:* Find conflicting reported numbers.**
The query has been given in Section 2.3. Surprisingly, we found a large set of conflicting records in the database. A number of them are worthy of investigation: Firstly, as the example we have given in the introduction, if the dataset is Epinions, *plus* the metric is MAE, then we have three pairs of conflicting numbers reported by [27] and [25]: (1) UserMean: 0.9319 vs 0.9285, (2) ItemMean: 0.9115 vs 0.9913, (3) Trust: 0.9044 vs 0.9215. Conflicting numbers for the metric RMSE can be observed as well: (1) UserMean: 1.1968 vs 1.1817, (2) ItemMean: 1.1973 vs 1.2584, (3) Trust: 1.1761 vs 1.2132.

Secondly, as presented in Table 1, the two KDD 2017 papers on multi-label classification, [42] and [36], gave different numbers for the same set of methods, the same datasets, and the same metrics, respectively. Though variance could happen when reproducing the results, we found many of the precision differences are bigger than 3%, which is often a sufficient margin to claim a new achievement! Finally, we also find a number of conflicting pairs that were not correctly aligned because of the missing contexts in the extraction such as the ratio of training data and the number of dimensions. In this paper, while claiming the importance of integrating PDF tables, we are aware of tons of challenging and interesting future works.

## 5 RELATED WORK

In this section, we review the literature on three relevant topics.

**Web Table Mining.** Mining knowledge from web tables has been studied for long [10, 12, 31, 39, 41]. Yang *et al.* analyzed the structural aspects of web tables, within which rules are devised to process and extract attribute-value pairs from the table [41]. Gatterbauer *et al.* proposed a method to find tabular structures without HTML table tags through cues such as onscreen data placement [12]. Later, researchers started considering table search. Cafarella *et al.* approached it as a modification of document search [6, 7]. Banko *et al.* applied open-domain information extraction to automatically discover possible relations of interest [2]. Venetis *et al.* [37] and Limaye *et al.* [24] annotated tables on the Web with column labels and relation labels. Compared with our work, first of all, the above works focused on Web tables that were designed following a specific language (i.e., HTML/XML) and semi-structured with tags and texts [4, 9, 35]. These are not available in our case of working on experimental result tables in paper PDF files. Mining tables in PDFs has its special challenges. Second, the Web tables are usually descriptive tables. In our research, we work on experimental result tables that are full of digital numbers. We have to find the context for each number/cell, or say, we need to classify the row/column names and even words in the captions. Third, the text environment was ignored when mining web table information. We propose to use the text around the tables to better understand the contexts.

**Bootstrapping for Information Extraction.** Bootstrapping methods have been widely used for information extraction (i.e., extracting relational tuples from text) since the never-ending learning came out [1, 5, 20, 21]. The success of this methodology lies in its ability to learn sufficient patterns and instances simply by iterations starting from a small number of seeds. Its central assumption is the pattern-relation duality principle that good seed samples lead to good patterns, while good patterns help to extract good instances. Here, good patterns are usually referred to patterns that have high coverage (high recall) and low error rate (high precision), and good instances are instances that are realized by good patterns [2, 8, 18, 29, 46]. Gupta *et al.* [16] and Halevy *et al.* [17] proposed Entity-Attribute patterns to apply to users' fact-seeking queries. Yahya *et al.* proposed Subject-Attribute-Object patterns for human-annotated corpus [40]. Jiang *et al.* proposed a general textual pattern, called meta pattern, using semantic type information [19]. All these methods were successful for finding information from text, however, capturing relations from tabular data, especially for experimental result tables in paper PDFs, brings new challenges to the bootstrapping framework, which requires carefully method design based on the unique data structures.

**Semantic Embedding Learning.** With the success of deep learning techniques, representation learning becomes popular starting from practices on text data [28] to other applications [15, 33, 38]. Mikolov *et al.* proposed the *word2vec* to learn distributed vector representations that capture precise syntactic and semantic word relationships [28]. Pennington *et al.* proposed GloVe to leverage statistical information by training only on the nonzero elements in a word-word co-occurrence matrix, rather than on the entire sparse matrix or on individual context windows in a large corpus [32]. Le *et al.* extended the embedded objects from words or phrases to paragraphs [23]. Recently, Nichel *et al.* proposed Poincare embedding based on a non-Euclidean space to preserve hierarchical semantic structures [30]. Devlin *et al.* proposed BERT for training deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. We used semantic embeddings for the task of table unification in our work.

## 6 CONCLUSIONS

In this work, we proposed a system for building a scientific database from experimental result tables in data science paper PDFs. Our framework has three modules. First, it cropped the tables and recognized table templates. Second, it classified column/row names into "method", "dataset", or "metric", and then combined with each score cell into a quadruple. We proposed hybrid features and an ensemble learning approach for column/row name classification and table unification. Third, it used SQL statements to make inference on the database. The informative database can facilitate researchers and practitioners who are interested in the field of data science, answering questions such as whether the baseline methods are the state-of-the-art or whether the reported numbers are conflicting.

## ACKNOWLEDGEMENTS

This work was supported in part by NSF Grant IIS-1849816.

## REFERENCES

[1] Eugene Agichtein and Luis Gravano. 2000. Snowball: Extracting relations from large plain-text collections. In *Proceedings of the fifth ACM conference on Digital libraries*. ACM, 85–94.

[2] Michele Banko, Michael J Cafarella, Stephen Soderland, Matt Broadhead, and Oren Etzioni. 2007. Open information extraction from the web. In *Proceedings of the 20th international joint conference on Artifical intelligence (IJCAI)*. Morgan Kaufmann Publishers Inc., 2670–2676.

[3] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. 2016. End to end learning for self-driving cars. *arXiv preprint* (2016).

[4] Katarina Boland, Dominique Ritze, Kai Eckert, and Brigitte Mathiak. 2012. Identifying references to datasets in publications. In *International Conference on Theory and Practice of Digital Libraries (TPDL)*. 150–161.

[5] Sergey Brin. 1998. Extracting patterns and relations from the world wide web. In *International Workshop on The World Wide Web and Databases*. 172–183.

[6] Michael Cafarella, Alon Halevy, Hongrae Lee, Jayant Madhavan, Cong Yu, Daisy Zhe Wang, and Eugene Wu. 2018. Ten years of webtables. *Proceedings of the VLDB Endowment* 11, 12 (2018), 2140–2149.

[7] Michael J Cafarella, Alon Halevy, Daisy Zhe Wang, Eugene Wu, and Yang Zhang. 2008. Webtables: exploring the power of tables on the web. *Proceedings of the VLDB Endowment* 1, 1 (2008), 538–549.

[8] Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R Hruschka, and Tom M Mitchell. 2010. Toward an architecture for never-ending language learning. In *Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI)*.

[9] Chia-Hui Chang, Mohammed Kayed, Moheb R Girgis, and Khaled F Shaalan. 2006. A survey of web information extraction systems. *IEEE transactions on knowledge and data engineering (TKDE)* 18, 10 (2006), 1411–1428.

[10] Hsin-Hsi Chen, Shih-Chung Tsai, and Jin-He Tsai. 2000. Mining tables from large scale HTML texts. In *Proceedings of the 18th conference on Computational linguistics (ACL)*. Association for Computational Linguistics, 166–172.

[11] 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 (ACL)*. 4171–4186.

[12] Wolfgang Gatterbauer, Paul Bohunsky, Marcus Herzog, Bernhard Krüpl, and Bernhard Pollak. 2007. Towards domain-independent information extraction from web tables. In *Proceedings of the 16th international conference on World Wide Web (WWW)*. ACM, 71–80.

[13] Max Göbel, Tamir Hassan, Ermelinda Oro, and Giorgio Orsi. 2012. A methodology for evaluating algorithms for table understanding in PDF documents. In *Proceedings of the 2012 ACM symposium on Document engineering*. ACM, 45–48.

[14] Max Göbel, Tamir Hassan, Ermelinda Oro, and Giorgio Orsi. 2013. ICDAR 2013 table competition. In *2013 12th International Conference on Document Analysis and Recognition (ICDR)*. IEEE, 1449–1453.

[15] Huan Gui, Jialu Liu, Fangbo Tao, Meng Jiang, Brandon Norick, Lance Kaplan, and Jiawei Han. 2017. Embedding learning with events in heterogeneous information networks. *IEEE transactions on knowledge and data engineering (TKDE)* 29, 11 (2017), 2428–2441.

[16] Rahul Gupta, Alon Halevy, Xuezhi Wang, Steven Euijong Whang, and Fei Wu. 2014. Biperpedia: An ontology for search applications. *Proceedings of the VLDB Endowment* 7, 7 (2014), 505–516.

[17] Alon Halevy, Natalya Noy, Sunita Sarawagi, Steven Euijong Whang, and Xiao Yu. 2016. Discovering structure in the universe of attribute names. In *Proceedings of the 25th International Conference on World Wide Web (WWW)*. International World Wide Web Conferences Steering Committee, 939–949.

[18] Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL)*. Association for Computational Linguistics, 541–550.

[19] Meng Jiang, Jingbo Shang, Taylor Cassidy, Xiang Ren, Lance M Kaplan, Timothy P Hanratty, and Jiawei Han. 2017. Metapad: Meta pattern discovery from massive text corpora. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 877–886.

[20] Tianwen Jiang, Tong Zhao, Bing Qin, Ting Liu, Nitesh Chawla, and Meng Jiang. 2019. Multi-Input Multi-Output Sequence Labeling for Joint Extraction of Fact and Condition Tuples from Scientific Text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. 302–312.

[21] Tianwen Jiang, Tong Zhao, Bing Qin, Ting Liu, Nitesh V Chawla, and Meng Jiang. 2019. The Role of" Condition" A Novel Scientific Knowledge Graph Representation and Construction Model. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*.

[22] Jinkyu Kim and John Canny. 2017. Interpretable learning for self-driving cars by visualizing causal attention. In *Proceedings of the IEEE international conference on computer vision (ICCV)*. 2942–2950.

[23] Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *International conference on machine learning (ICML)*. 1188–1196.

[24] Girija Limaye, Sunita Sarawagi, and Soumen Chakrabarti. 2010. Annotating and searching web tables using entities, types and relationships. *Proceedings of the VLDB Endowment* 3, 1-2 (2010), 1338–1347.

[25] Hao Ma, Irwin King, and Michael R Lyu. 2011. Learning to recommend with explicit and implicit social relations. *ACM Transactions on Intelligent Systems and Technology (TIST)* 2, 3 (2011), 29.

[26] Hao Ma, Dengyong Zhou, Chao Liu, Michael R Lyu, and Irwin King. 2011. Recommender systems with social regularization. In *Proceedings of the fourth ACM international conference on Web search and data mining (WSDM)*. ACM, 287–296.

[27] Hao Ma, Tom Chao Zhou, Michael R Lyu, and Irwin King. 2011. Improving recommender systems by incorporating social contextual information. *ACM Transactions on Information Systems (TOIS)* 29, 2 (2011), 9.

[28] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems (NIPS)*. 3111–3119.

[29] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing (ACL-IJCNLP)*. Association for Computational Linguistics, 1003–1011.

[30] Maximillian Nickel and Douwe Kiela. 2017. Poincaré embeddings for learning hierarchical representations. In *InAdvances in neural information processing systems (NIPS)*. 6338–6347.

[31] Gerald Penn, Jianying Hu, Hengbin Luo, and Ryan McDonald. 2001. Flexible web document analysis for delivery to narrow-bandwidth devices. In *Proceedings of Sixth International Conference on Document Analysis and Recognition (ICDAR)*. IEEE, 1074–1078.

[32] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[33] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD)*. ACM, 701–710.

[34] AC Silva. 2010. Parts that add up to a whole: a framework for the analysis of tables. *Edinburgh University, UK* (2010).

[35] Hassan A Sleiman and Rafael Corchuelo. 2013. A survey on region extractors from web documents. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* 25, 9 (2013), 1960–1981.

[36] Yukihiro Tagami. 2017. AnnexML: Approximate nearest neighbor search for extreme multi-label classification. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 455–464.

[37] Petros Venetis, Alon Halevy, Jayant Madhavan, Marius Paşca, Warren Shen, Fei Wu, Gengxin Miao, and Chung Wu. 2011. Recovering semantics of tables on the web. *Proceedings of the VLDB Endowment* 4, 9 (2011), 528–538.

[38] Daheng Wang, Tianwen Jiang, Nitesh V Chawla, and Meng Jiang. 2019. TUBE: Embedding Behavior Outcomes for Predicting Success. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*. 1682–1690.

[39] Yalin Wang and Jianying Hu. 2002. A machine learning based approach for table detection on the web. In *Proceedings of the 11th international conference on World Wide Web (WWW)*. ACM, 242–250.

[40] Mohamed Yahya, Steven Whang, Rahul Gupta, and Alon Halevy. 2014. Renoun: Fact extraction for nominal attributes. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 325–335.

[41] Yingchen Yang and Wo-Shun Luk. 2002. A framework for web table mining. In *Proceedings of the 4th international workshop on Web information and data management (WIDM)*. 36–42.

[42] Ian EH Yen, Xiangru Huang, Wei Dai, Pradeep Ravikumar, Inderjit Dhillon, and Eric Xing. 2017. Ppdsparse: A parallel primal-dual sparse method for extreme classification. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 545–553.

[43] Burcu Yildiz, Katharina Kaiser, and Silvia Miksch. 2005. pdf2table: A method to extract table information from pdf files. In *Proceedings of the 2nd Indian International Conference on Artificial Intelligence (IICAI)*. 1773–1785.

[44] Quanzeng You, Jiebo Luo, Hailin Jin, and Jianchao Yang. 2016. Cross-modality consistent regression for joint visual-textual sentiment analysis of social multimedia. In *Proceedings of the Ninth ACM international conference on Web search and data mining (WSDM)*. ACM, 13–22.

[45] Wenhao Yu, Zongze Li, Qingkai Zeng, and Meng Jiang. 2019. Tablepedia: Automating PDF Table Reading in an Experimental Evidence Exploration and Analytic System. In *The World Wide Web Conference (WWW)*. ACM, 3615–3619.

[46] Qingkai Zeng, Mengxia Yu, Wenhao Yu, Jinjun Xiong, Yiyu Shi, and Meng Jiang. 2019. Faceted hierarchy: A new graph type to organize scientific concepts and a construction method. In *Proceedings of the Thirteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-13)*. 140–150.