Inferring Tabular Analysis Metadata by Infusing Distribution and Knowledge Information

Xinyi He∗
Xi’an Jiaotong University
Xi’an, Shaanxi, China
hxyhxy@stu.xjtu.edu.cn
Mengyu Zhou†
Microsoft Research
Beijing, China
mezho@microsoft.com
Jialiang Xu∗
University of Illinois at Urbana
Champaign
jx17@illinois.edu
Tianle Li∗
Hong Kong University of Science and Technology
tliax@connect.ust.hk
Yijia Shao∗
Peking University
Beijing, China
shaoyj@pku.edu.cn
Shi Han
Microsoft Research
shihan@microsoft.com
Zejian Yuan
Xi’an Jiaotong University
yuan.ze.jian@xjtu.edu.cn
Dongmei Zhang
Microsoft Research
donmeiz@microsoft.com

ABSTRACT
Many data analysis tasks heavily rely on a deep understanding of tables (multi-dimensional data). Across the tasks, there exist commonly used metadata attributes of table fields/columns. In this paper, we identify four such analysis metadata: Measure/dimension dichotomy, common field roles, semantic field type, and default aggregation function. While these metadata face challenges of insufficient supervision signals, utilizing existing knowledge and understanding distribution.

To inference these metadata for a raw table, we propose our multi-tasking Metadata model which fuses field distribution and knowledge graph information into pre-trained tabular models. For model training and evaluation, we collect a large corpus (~582k tables from private spreadsheet and public tabular datasets) of analysis metadata by using diverse smart supervisions from downstream tasks. Our best model has accuracy = 98%, hit rate at top-1 > 67%, accuracy > 80%, and accuracy = 88% for the four analysis metadata inference tasks, respectively. It outperforms a series of baselines that are based on rules, traditional machine learning methods, and pre-trained tabular models. Analysis metadata models are deployed in a popular data analysis product, helping downstream intelligent features such as insights mining, chart/pivot table recommendation, and natural language QA, etc.

Table 1: Example Table of Student Data. At the bottom of the table are the labels of analysis metadata tasks.

| Student ID | Name  | Department | Class | Midterm Exam | Final Exam |
|------------|-------|------------|-------|--------------|------------|
| 201301     | Alice | CS         | 1     | 90           | 95         |
| 201312     | Bob   | EE         | 1     | 80           | 96         |
| 201331     | Carol | CS         | 2     | 86           | 85         |
| 201256     | Betty | EE         | 3     | 78           | 63         |
| 201125     | David | ART        | 1     | 85           | 86         |

Figure 1: Example Chart Created from Table 1. During chart composing, dimension fields are usually labels while measure fields are value series.

1 INTRODUCTION
Multi-dimensional data (i.e., tabular) analysis is an everyday activity widely performed by individuals and organizations. To make data analysis easier, lots of intelligent assistants are built to automate and accelerate the process: e.g., translating natural language queries into analysis queries [11, 18], exploratory data analysis by automatic insights mining [10, 20], learning and recommending common data visualizations [32, 40], etc. However, these data analysis systems share a group of common pain points:

(1) Robustness and Interpretability. To our knowledge, because of robustness and interpretability, end2end models are hard to be applied to real-world scenarios. One general solution is to symbolize the task into multi-steps and improve accuracy one by one. Thus, finding and improving those steps are important, which is the focus of this paper.

(2) Universality. Most of data analysis systems have tasks depending on what roles its fields (or columns) can perform. And there exist commonly used field attributes shared across various...
Table 2: Example Table of Sales Data. At the bottom of the table are the labels of analysis metadata tasks.

| Date   | Region | SalesRep | #Products | Sales |
|--------|--------|----------|-----------|-------|
| 10/27/18 | West   | Chin     | 2         | $859  |
| 10/27/18 | Southwest | Freddy   | 4         | $1,786|
| 10/28/18 | Northwest | Gigi     | 4         | $2,134|
| 10/28/18 | Southwest | Gigi     | 3         | $1,736|
| 10/28/18 | Northwest | June     | 5         | $1,401|

| SUM(Sales) | Region   |
|------------|----------|
| SalesRep   | Northwest | Southwest | West | Grand Total |
| June       | 8606     | 9217       | 1786 | 3401        |
| Chin       | 6729     | 5130       | 6729 | 11859       |
| Freddy     | 15335    | 6916       | 10832| 33083       |

Figure 2: Example Pivot Table Created from Table 2. When creating pivot tables, dimension fields perform the role of breaking down records into categories, while corresponding numbers in a measure field are aggregated.

downstream analysis tasks. However, there is a lack of research on universal attributes and their corresponding taxonomy.

(3) Missing metadata. The original raw tables do not have those important universal attributes. Thus, we formulate those attributes into metadata tasks and propose models to resolve them.

Thus, we propose commonly used field attributes, namely, analysis metadata, which are shared across various downstream analysis tasks, and define those taxonomies. In this paper, we propose four types of analysis metadata including eight tasks.

Dimension / Measure dichotomy is a widely used metadata borrowed by us from dimensional modeling in databases [12, 19]. A measure field contains numerical measurement results on which calculations (e.g., sum, count, average, minimum, maximum) can be made. Meanwhile, a dimension field contains categorical information and provides functions of filtering, grouping, and labeling. Many downstream analysis tasks benefit from the understanding of whether a field is a measure or a dimension.

Example 1 (Measure and Dimension Fields). In Table 1 and Table 2, all the measure fields are marked by (MSR) and all the dimension fields are marked by (DIM).

During chart composing, dimension fields are usually labels for the x-axis while measure fields are value series for the y-axis. For example, when recommending line and bar charts for Table 1, one should select a measure field (e.g. “Final Exam”) for the y-axis and a dimension field (usually with unique values, e.g. “Name”) for the x-axis. This leads to proper charts such as the one shown in Figure 1.

Similarly, when creating pivot tables, dimension fields perform the role of breaking down records into categories (corresponding to GROUP BY operation in SQL), while corresponding numbers in a measure field are aggregated. For Table 2, the example in Figure 2 is one such pivot table with “Region” and “SalesRep” as group-by dimensions and “Sales” aggregated by SUM function.

More types of metadata can be identified based on the concept of measure and dimension. In §3, we describe four common ones and establish corresponding taxonomy:

§3.1 Measure/dimension dichotomy: For each field in a given table, is it a measure or a dimension?

§3.2 Common field roles: Which measures in a table are commonly used and more important? Which dimension is the nature key? Which dimensions are commonly used for group-by operation?

§3.3 Semantic field type: Measure type taxonomy is established with 19 types in 5 categories, and which type can be assigned to measure? (e.g., “Score”, “Count”, “Money” in Table 1 and Table 2) Which semantic type can assign to dimension? (e.g., “person.person”, “location.region” in Table 1 and Table 2) Are two measures from a table comparable with each other?

§3.4 Default aggregation function: There are 9 popular aggregation functions, and which is the most suitable default one to apply to a specific measure field? (e.g., “AVG”, “SUM” in Table 1 and Table 2)

The last three types of metadata try to unleash more table understanding power to downstream tasks. In §3, examples demonstrate more on their use cases.

Unfortunately, such metadata cannot be directly obtained from raw tables prevalent in daily usage - these raw tables usually only contain header names and data values, and lack additional information for each field. For real-world spreadsheets files, web pages, and databases, the above four types of metadata usually require extra steps of inference. When trying to infer analysis metadata in a data-driven way via machine learning approaches, we immediately face the following challenges:

(1) Tasks and labels: How to formulate metadata inference as machine learning tasks? Where to find supervision signals for each task? Can we leverage existing datasets?
(2) Utilizing existing knowledge: There is a wealth of existing information in the knowledge graph (KG). How do utilize existing knowledge to enhance table representation and analysis metadata?
(3) Understanding distribution: All the four types of metadata involve measures. How to design a model that understands what a series of values means?

In order to get supervision signals for metadata inference (challenge 1), we collect a large corpus and use smart supervision from downstream tasks, the manual labels through crowdsourcing, the existing information attached to tables (§4). This corpus is collected from spreadsheet datasets, public web table datasets, and public synthetic datasets.

Recently, the pre-training strategy has achieved significant success in natural language processing, and some works [13, 34] try to use it on tabular data. However, existing tabular pre-training works seldom consider the existing knowledge (challenge 2) and column distribution (challenge 3). To fill this gap and better solve the metadata problem, we propose a multi-encoder Metadata framework (§5), which can be applied to almost all the transformer-based pre-training tabular models. To represent column distribution, the
framework extracts data features and proposes **Distribution Fusion** to infuse them with the tabular model. To utilize extra knowledge graph information, the framework extracts KG information (entity linking, column type, and property) for each table and proposes **Knowledge Fusion** to infuse them with the tabular model.

We conduct several experiments (§6) to illustrate the importance of Metadata tasks and the effectiveness of our Metadata model. Our model has accuracy = 98%, hit rate at top-1 > 67%, accuracy > 80%, and accuracy = 88% for the four analysis metadata inference tasks, respectively, and outperforms baselines varying from rule-based, traditional machine learning methods to pre-trained tabular models. And in especial, Distribution fusion improves one of the common field roles tasks by 3.64% and Knowledge fusion improves one of the semantic field types tasks by 3.54%. To illustrate the importance of Metadata tasks, we further show its improvement on down-stream task (§6.5, in the case of chart), and the real-world deployment (§7).

In summary, our main contributions are:

- The novel problem of inferring analysis metadata is proposed by us to help downstream data analysis tasks. 4 types of analysis metadata are on basis of data profiling and knowledge matching, containing 8 tasks and corresponding taxonomy.
- A large corpus of analysis metadata on tables is collected by us using smart supervisions from down stream tasks, public dataset, and our manual labels.
- We propose a multi-encoder Metadata framework that further understands the numerical distribution and incorporates the knowledge graph information. Besides, the framework can be applied to almost all the transformer-based pre-training tabular models.

## 2 RELATED WORK

### 2.1 Dimensional Modeling and Metrology

The terms “measure” and “dimension” have their roots in dimensional modeling from data warehousing and business intelligence [12]. In relational and multidimensional databases, dimensional models are implemented as star schemas and online analytical processing (OLAP) cubes [19]. Our definition of “dimension” extends the concept in dimensional modeling. As we will discuss in §3.1, it contains primary keys and natural keys in addition to the dimension attributes. Most of our analysis metadata involves measures. As the scientific study of measurements, Metrology includes the definition of quantities and units of measurement. In §3.3, we will define our measure types with common units from the International System of Units (SI) [1, 3], which is a widely accepted metric system.

### 2.2 Data Profiling

[2] has the similar term “metadata”. These metadata, such as statistics about the data or dependencies among columns, can help understand and manage new datasets. Analysis metadata is proposed for further analysis tasks based on part of data profiling metadata. The details about their relationship for each task as followings:

1. Unique column combination and primary key. There is plenty of work to explore them, while key with semantics is not their focus. [4] proposed natural key based on primary key, and they use engineered features and Random Forest to solve the problem.

(2) Identifying semantic domain of a column. [36] first propose semantic domain labeling by clustering columns. [27] matches columns to pre-defined semantics with specific features and Naive Bayes. This track of works evolves towards column type identification in table interpretation, which we discuss in §2.3.

(3) Quantity name recognition: This activates detect quantity name for a column, which is highly correlated with measure type in analysis metadata. [25] point out that unit extraction is a significant step for queries on web tables, and design unit extractors for units in column names by developing a unit catalog tree. [33] extends to inferring unknown units with extracted feature and Random Forest. However, those existing works heavily depend on units appearing in the table, so we propose measure types that can also identify measure fields without units and property.

State-Of-Art of those related work often extracts specific features and uses traditional machine learning to solve the problem, which lacks further semantic representation with the pre-training model.

### 2.3 Table Interpretation

There is a long line of work trying to understand tables symbolically, especially for entity or content tables where we can conduct entity linking, column type annotation, and relation extraction [6, 14, 17, 29, 30]. Some domain ontology or knowledge graph, such as DBPedia [21] and Wikidata [28], is often provided for alignment. Column type annotation and relation extraction are related to our measure / dimension (§3.1) and field type (§3.3) classification. However, most previous work focus on the entity type of the columns [7, 14], and few public datasets provide real-world tables with rich labels of measurements [6, 24].

### 2.4 Neural Representation Learning of Tables

Pre-trained language models [5, 8] are widely used in NLP tasks. Recently also emerge several pre-trained tabular models with transformer as the primary backbone [7, 13, 15, 31, 34]. TaBERT[34] and TAPAS [13] serialize the table and jointly pre-train the text-table pairs with MLM objective to resolve the table QA problem. However, their representation capability is limited by the expensive computation due to table serialization. TABBIE [15] remedies the issue by leveraging two transformers to encode rows and columns independently to reduce the length of inputs.TURL[7] proposes a structure-aware Transformer with Masked Entity Recovery (MER) objective and injects entity knowledge in relational Web Tables [7].

### 2.5 Downstream Analysis Tasks

Lots of intelligent data analysis features could benefit from analysis metadata. Typical examples include automatic insights discovery [10, 20], chart and pivot table recommendations [32, 40, 41], Text2SQL and query recommendations [11, 18, 35], table expansion [37, 38], etc. Most of these tasks involve searching, enumerating, and comparing in a large space. Analysis metadata could help narrow down possible candidates (prioritized searching order) and provide good ranking references.
3 PROBLEM FORMULATION

Following our brief introduction to analysis metadata in §1, in this section, we further describe their details and formulate them as machine learning tasks with the table as input.

3.1 Measure / Dimension Dichotomy

Measure and dimension fields play different roles in data analysis. The measure/dimension dichotomy metadata can tell downstream tasks the legal analysis operations for each field, thus could help greatly instruct their search spaces. They are simply defined in [10].

Definition 1 (Measure). A measure (MSR) field contains numerical measurement values on which calculations can be made.

Definition 2 (Dimension). A dimension (DIM) field contains categorical values. It provides functions of filtering, grouping, and labeling. A dimension is called a group-by (or breakdown) dimension when its values have duplication. Otherwise, a dimension with unique values is called a key dimension.

The machine learning formulation of the metadata is Binary classification of measure/dimension for each field of a given table. However, this definition requires a non-trivial understanding of tables. First, not all numerical fields are measure fields. “Student ID”, “Class” fields consist of categorical numbers, thus are dimensions. Second, there exists a weak dependency between field positions and roles. Starting from the left, usually, key dimensions come before group-by dimensions, and measures come after dimensions. An ML model should take vague hints among fields into account.

3.2 Common Field Roles

Fields have been identified with measure and dimension, while not all of them are highly regarded. In daily analysis activities, measures and dimensions with some semantic meanings are more frequently selected. In other words, there are common patterns about which fields are more preferred than others within a table. As shown in Table 1 and Table 2, we mark common preferences of measure, group-by dimensions, and key dimensions by shade (darker means more preferred).

Definition 3 (Natural Key). Natural Key is a dimension field with all unique data values and using them to represent each record in semantic terms.

For example, when composing charts from Table 1, because the “Name” field is more human-comprehensible, it is more commonly chosen than “Student ID” as a key dimension mapping to the x-axis.

It’s worth noting that there are several existing works on “primary key”, while “natural key” is different from “primary key” as described in §2.2. Both of them are chosen to represent records, while “primary key” focuses on unique (e.g., ID) and “natural key” focuses on semantic terms (e.g., name). Sometimes it is also called key / core / subject / name / entity column in relational tables [24, 39]. As discussed in §2.3, natural key can be used for entity linking and other table understanding tasks.

Definition 4 (Common Breakdown). Common Breakdown is the dimension field(s) that are the most commonly used for breakdown among a given table in data analysis.

For example, when selecting group-by (breakdown) dimensions during insights mining in Table 1, “Department” has richer semantics and better readability compared to “Class”. In Table 2, “Region” and “SalesRep” are commonly grouped by because their low cardinality and time-invariant qualities, leading to high-quality insights.

Definition 5 (Common Measure). Common Measure is the measure field(s) that are the most commonly used for further analysis (e.g., applying aggregation function, composing chart) among a given table in data analysis.

For example, when choosing measure from Table 2, most people would analyze “Sales” first as it’s a more important KPI that a store would care about prior to the number of products sold.

To help downstream tasks on prioritizing search order and re-ranking results, we formulate them as three machine learning tasks providing 0−1 commonness score (higher means more preferred in general) for each field of a given table and recommend fields list for each table in order of commonness score.

3.3 Semantic Field Type

Semantic types for common fields are important to data cleaning, table interpretation, data analysis, and so on. There are several existing works focusing on identifying semantic types for columns as described in §2.2, e.g., column type identification.

However, most works focus on dimension fields, especially the subject columns. Less than 5% of column types are for measure fields, which are based on statistics of existing column type datasets (Sherlock [14], TURL [7], and Semtab [6, 16]). Thus, we propose semantic field type, including measure and dimension type.

Table 3: The Most Frequency Dimension Types.

| Dimension type                  |
|---------------------------------|
| people.person                   | government.political_party |
| location.location               | location.administrative_division |
| organization.organization       | sports.sports_league_season |
| sports.sports_team              | soccer.football_player     |
| sports.pro_athlete              | sports.sports_league       |
| soccer.football_team            | government.politician      |
| time.event                      | film.films                 |
| location.country                | business.business_operation|
| location.citytown               | ...                        |

Definition 6 (Dimension type). The common semantic types for dimension fields, which are based on knowledge graph type of entities. We adopt TURL [7] column types as dimension type. It contains 255 types and the most frequency dimension types are shown in Table 3.

Definition 7 (Measure Type). The common mutually exclusive types for measure fields, which are based on unit, magnitudes, and entity property. Each type represents a magnitude, and each type is mutually exclusive. Each measure type corresponds to a set of convertible units (see “Common Units” in Table 4), and highly correlated concepts (see “Common Examples” in Table 4). “Dimensionless” category with 6 types is summarized in taxonomy. Different
Table 4: Measure Type Taxonomy and Details.

| Category       | Type                  | Common Examples                      | Common Units |
|----------------|-----------------------|--------------------------------------|--------------|
| Dimensionless  | Count (Amount)        | Population, daily passenger flow, ... |              |
|                | Ratio                 | Percentage, change rate, proportion, ... |              |
|                | Angle                 | Angle, longitude, latitude ...        |              |
|                | Factor / Coefficient  | Coefficient of thermal expansion, drag coefficient, ... |              |
|                | Score                 | Rating, exam score, indicator (index), ... |              |
|                | Rank                  | University ranking, projected GDP ranking, ... |              |
| Money          | Sales                 | Sales, asset, income, revenue, cost, GDP, ... | $, €, £, ... |
|                | Data/file size        | Memory size, disk size, ...           | GB, kb, ...  |
| Time           | Duration              | Age, runtime, time length, ...        | s, min, hr, d, yr, ... |
|                | Frequency             | Audio frequency, rotational speed, ... | Hz, RPM, ... |
|                | Length                | Length, width, elevation, depth, height, ... | m, cm, yard, feet, ... |
| Scientific     | Area                  | Surface area, gross floor area, ...   | m², acre, ... |
|                | Volume (Capacity)     | Vital capacity, water capacity, ...   | m³, L, ... |
|                | Mass (Weight)         | Body weight, salt consumption, ...    | kg, lbs, ... |
|                | Power                 | Source power, rated power, ...        | kW, ... |
|                | Energy                | Calories, energy consumption, ...     | J, kcal, ... |
|                | Pressure              | Atmospheric pressure, blood pressure, ... | Pa, mmHg, ... |
|                | Speed                 | Velocity, average speed, ...          | m/s, km/h, ... |
|                | Temperature           | Effective temperature, melting point, boiling point, ... | °C, K, ... |

from the existing magnitude or units taxonomy, there is plenty of measure without units and corresponding measure types are important in analysis. Thus we summarize the most common mutually exclusive 6 dimensionless measure types.

In Table 4 you can find the commonly seen measure types in daily data analysis scenarios. These 19 types (except “Others”) can be further grouped into 5 categories. The taxonomy is based on the units in Wikidata\(^1\), DBPedia properties in T2D Golden Standard\(^2\), and units in the International System of Units\(^3\).

As we will discuss in §4.2, we only keep the measure types with a number of appearances above the threshold.

Semantic field types can be helpful in writing rules, constraints, and templates for data analysis systems. For example, in Table 1 and Table 2, measure types are shown in round brackets. When knowing “Products” belongs to Count and “Sales” belongs to Money, for Table 2, a natural language QA system could suggest default queries by easily filling the templates “The total (Count) by [a group-by dimension]” and “Which record has the highest (Money)?”.

The field still has a measure type, when values of the measure field are not entity or property in KG. Measure type is important to all common measure fields. But the existing column type identification and relation extraction task need table values to be entities or properties in the knowledge graph. And more than 90% of numerical fields can not match entity or property in KG (we perform statistics on non-WikiTable (Web table and spreadsheet) mentioned in §4 and match with MTab[23]). For example, “Final Exam” in Table 1 can not match KG, while it has “Score” measure type.

Many tables contain multiple same-typed measures, among which further analysis can be made, implying multi-measure analysis.

Definition 8 (Measure Pair). Within a table, a **measure pair** is a pair of comparable measures – They should have the same type of unit (including convertible ones), related semantic meanings, and similar numerical value range.

Measure pair can assist in the measure type identification. Besides, when measure type in the table is difficult to identify, the measure pair identification is particularly important. For example, in Table 1, a measure pair is marked by an arrow. “Midterm Exam” and “Final Exam” in Table 1 are a pair of scores (in the Dimensionless category). Such measure pairs are frequently analyzed together. E.g., the chart in Figure 1.

The machine learning task for measure pair identification is a binary classification of any pair of numerical fields within a given table. For measure type and dimension type, it is a 19-way and 255-way classification problem.

3.4 Default Aggregation Function

Definition 9 (Default Aggregation Function). **Default aggregation function** is an aggregation function that is the most commonly used for applying to the measure field. Popular aggregation (AGG) functions and its statistics are shown in Figure 3.

---

\(^1\)https://www.wikidata.org/wiki/Wikidata:Units
\(^2\)http://webdatacommons.org/webtables/goldstandard.html#toc2
\(^3\)https://www.bipm.org/en/measurement-units

---

![Figure 3: Common Aggregation Functions.](image-url)
Table 5: Supervision Labels from Each Dataset. Each row represents one task, and its labels are obtained in different ways from different datasets.

| Supervision | Chart dataset | Pivot dataset | Vendor dataset | T2D dataset | TURL dataset | SemTab dataset |
|-------------|---------------|---------------|---------------|-------------|-------------|----------------|
| Measure     | Y-axis field  | Value field   | Manual label  | Msr. type field | -           | -              |
| Dimension   | X-axis field  | Rows and columns field | Manual label | Primary key field | -           | -              |
| Com. measure | Chart msr.    | Pivot msr.    | -             | -           | -           | -              |
| Com. breakdown | -            | Non-unique dim. | -             | -           | -           | -              |
| Natural key | Single unique dim. | -             | -             | Primary key | -           | -              |
| Dim. type   | -             | -             | -             | Column type (cell hyperlinks) | -           | -              |
| Msr. type   | -             | -             | Manual label  | Msr. property (manual label) | -           | Msr. property (from synthetic info.) |
| Msr. pair   | Same chart axis msr. | -             | -             | -           | -           | -              |
| Agg. function | -             | Agg. for value field | Manual label | -           | -           | -              |

An essential operation in data analysis is to aggregate multiple measurements from a field, usually grouped by another breakdown dimension. For each measure, there are some AGG functions more widely applied to it. The most suitable AGG function could be adopted as default calculation by downstream analysis tasks. For most measures, AVG can be applied directly, but often SUM is a better choice if possible. For some downstream scenarios, AVG/SUM can cover most usage cases.

**Example 2** (Default Aggregation Function). The default AGG function for “Midterm Exam” and “Final Exam” in Table 1 is average (AVG), and summation (SUM) for “#Products” and “Sales” in Table 2.

The Default AGG function can be directly used to avoid further searching efforts in data analysis. E.g., there is no need to try AVG and other functions before SUM for the pivot table in Figure 2.

The machine learning task for the default aggregation function is to provide 0~1 ranking scores for popular AGG functions.

# 4 CORPUS AND SUPERVISIONS

As discussed in §1, a major challenge for building machine learning models for analysis metadata inference is where to find supervision labels. In this section, we will bring several datasets together to prepare labels for metadata inference tasks.

## 4.1 Table Datasets

### 4.1.1 Spreadsheet Dataset

From the public Web, we crawled millions of Excel spreadsheet files with English-language tables in them. Lots of these files contain analysis artifacts created by users. From this spreadsheet dataset, we extract the following dataset:

1. Chart dataset: There are 254,666 charts (e.g., the one in Figure 1) created from 158,529 tables (including 1,103,587 fields). Line, bar, scatter and pie charts are the most dominant chart types, covering more than 98.91% of charts. The x-axes of bar and pie charts directly display the data values of their reference field one by one, which play the role of natural key. Y-axes displays data size or trend, which plays the role of measure. And in some charts, the y-axis is plotted with multi-series, which plays the role of measure pair.

2. Pivot dataset: There are 54,928 pivot tables (e.g., the one in Figure 2) created from 32,761 tables (including 477,956 fields). Pivot table has “rows”, “columns” and “values”. Both the rows and columns hierarchically break down records into groups. Its values are for applying aggregation to each group.

3. Vendor dataset: We randomly sample 882 tables with 6,715 fields, and manually label measure/dimension dichotomy, measure type, and aggregation function.

### 4.1.2 Web Table Datasets

Several web table (HTML table) datasets have been extracted in the existing work, and the following datasets are used in this work:

1. T2D dataset: T2D Gold Standard [24] is a dataset for evaluating matching systems on the task of matching Web tables to the DBpedia knowledge base. It contains schema-level correspondences between 1,724 Web tables (consisting of 7,705 fields) from the English-language subset of the Web Data Commons Web Tables Corpus^4 and DBpedia^5. In total, more than 2,000 fields are mapped to ~290 DBpedia properties.

2. TURL dataset: TURL[7] constructs a dataset based on the WikiTable corpus, and utilizes each cell hyperlink (link to Wikipedia pages) to get entity linking, column type, and relation extraction supervision labels. This work utilizes TURL tables with column types that contain 654,670 columns from 406,706 tables.

### 4.1.3 Synthetic Dataset

To utilize richer data and supervision labels, synthetic datasets are also taken into consideration. SemTab challenge [16] aims at benchmarking systems dealing with the tabular data to knowledge graph matching problem, including CEA task (matching a cell to a KG entity), CTA task (assigning a semantic type to a column), and CPA task (assigning a property to the relationship between two columns). This challenge provides large datasets automatically generated from the knowledge graph. SemTab 2020 dataset contains 131,289 tables and 68,001 tables with CPA task labels. To avoid data leakage, 17,995 schemata are extracted from 68,001 tables, and randomly keep one table in each schema.

## 4.2 Supervision Labels

Based on the above datasets, supervision labels are prepared in three ways: First, from the analysis artifacts created in downstream

---

^4 [http://webdatacommons.org/webtables/](http://webdatacommons.org/webtables/)

^5 [https://www.dbpedia.org/](https://www.dbpedia.org/)
tasks including chart and pivot table; Second, from manual labels in the public dataset (i.e., T2D) and by ourselves; Third, from the information attached to the table (i.e., SemTab, TURL). Table 5 shows supervision labels for each task from each dataset.

In rows 1-2 of Table 5, for the measure / dimension binary classification in §3.1, the positive samples (measures) are the fields referenced by y-axis in charts and “values” in pivot table, the negative samples (dimensions) are x-axis (except scatter and line chart) in charts and “rows” and “columns” in pivot table. In total, we have 385,425 measures and 118,552 dimensions from 179,245 tables.

For the commonness scores in §3.2, the labels are slightly modified on measure / dimension dichotomy labels. In row 3 of Table 5, for common measures, the positive samples (with score = 1) are all the 382,022 measures from charts and pivot tables. In row 4 of Table 5, for common breakdowns, 50,790 positive samples come from the non-unique dimensions in pivot tables (fields of pivot rows and columns). In row 5 of Table 5, for natural key, 58,513 positive samples come from dimensions in charts, and satisfy that 1) data values are unique 2) don’t have multi-dimensions in the table. The negative samples (with score = 0) are all other fields besides the positive samples in a given table. In total, there are 1,159,258, 427,189, ≥1,003,697 negative samples for common measures, common breakdowns, and natural key, respectively.

In row 8 of Table 5, for measure pair binary classification in §3.3, its 118,236 positive labels come from pairs of measure fields referenced by the same chart axis. For each table, we randomly sample the same number of negative samples (numerical field pairs) as its positive samples. In row 7 of Table 5, for measure type labels, we merge three label sources together: DBPedia property labels in T2D, Wikidata property labels in T2D, and our manual labels on 882 tables (consisting of 6,715 fields) randomly sampled from our spreadsheet dataset. We start with a longer list of measure types as discussed in Definition 7, map DBPedia properties to the list for T2D, map Wikidata properties for SemTab and mark all 3,139 measures with types in the 882 sampled tables. As mentioned in §3.3, we only keep measure types with ≥10 labels (in T2D and sampled tables) or ≥100, resulting in 36859 fields fall in our measure taxonomy in Table 4. Although TURL dataset also has property (relation extraction in its paper) labels, there are less than 1% labeled as measure property, so they are not used in this task. In row 6, for dimension type labels, we utilize column types in TURL[7], which contains 255 types, 654,670 columns from 406,706 tables.

In row 9 of Table 5, for default aggregation ranking scores in §3.4, we focus on the 9 most frequently used aggregation functions in pivot tables as shown in Figure 3. For a field, the actual aggregation applied by users has score = 1, otherwise unused functions are assigned with 0 score.

5 METADATA MODEL

As discussed in §1, a major challenge for analysis metadata inference is how to represent distribution and utilize extra knowledge graph information. Thus, we propose Metadata model to infuse information. In this section, we provide a detailed description of Metadata model architecture as depicted in Figure 4a.

5.1 Overall Model Architecture

Our model is based on the pre-trained tabular model and utilizes a transformer encoder on top. Because distribution and knowledge graph information describes both cells and columns, we design two encoders for Metadata model. The first encoder is sub-token or cell level (depending on the pre-trained tabular model), and the second one is column level.

(1) Pre-trained tabular models are used as preliminary encoders generating initial representation for table elements. After the preliminary encoding phase (“Pre-trained tabular model” in Figure 4a), we get a sub-token level or cell level Transformer[26] encoder (“Sub-token level encoder”).

(2) We fuse knowledge of cell entity and tabular model embedding with “Knowledge fusion”, and pass it into sub-token (or cell) level Transformer encoder (“Column level encoder”) and linear output heads for each metadata task.

The pre-trained tabular model could be almost all transformer-based pre-trained tabular models. In this work, we use TAPAS[13] and TABBBIE[15] to illustrate effectiveness.

5.2 Knowledge Fusion Module

Knowledge graph information provides extra knowledge about tables and helps the model interpret tables. In this subsection, we describe how to get those information and how to use them.

There are three kinds of knowledge information linked with knowledge graph according to table interpretation tasks – cell entity, column type, and property. For a table with knowledge linking (e.g., TURL dataset, Semtab dataset), we utilize original knowledge linking. Otherwise, we adopt MTab[7] to link the knowledge graph. MTab is a tool to annotate tables with knowledge graphs, and they get first place in Semtab2019[16] and Semtab2020[6].

For both entity and property, existing knowledge representation covers comprehensively and performs well. However, if their embedding is trained with table corpus, it can only cover limited entity and property. To increase the extensibility and performance of the model, we directly use knowledge representation in OpenKE[9].

Fusion module is illustrated in Figure 4b. Model respectively fuses cell entity in cell level, column type, and property type in column level. Because one tabular token needs to pay attention to several entities, e.g., a cell token needs to pay attention to a specific entity in the same column or row, knowledge fusion module applies attention to fuse tabular embedding and knowledge graph embedding, as shown in the following equation.

\[
H = W_5 \text{ (Attention(TOK, ENT, ENT)) + ENT}
\]

\[
Attention(Q, K, V) = W_2 \text{ softmax}((QW_1K^T) \log(M)) V
\]

where, \(TOK \in \mathbb{R}^{n \times d_{tok}}\) is the sequence of tabular model embedding, \(ENT \in \mathbb{R}^{n \times d_{ent}}\) is the sequence of knowledge graph embedding.

\[\text{https://github.com/phucty/mtab_tool}\]

\[\text{http://openke.thunlp.org/}\]
(a) Overall metadata model. ① represents knowledge fusion module in Figure 4b, and ② represents distribution fusion module in Figure 4c.

(b) Knowledge Fusion Module. The module represents each sequence.

(c) Distribution Fusion Module. The module represents each token.

Figure 4: Metadata Model Architecture

\[ W_1 \in \mathbb{R}^{d_{tok} \times d_{ent}}, W_2 \in \mathbb{R}^{d_{ent} \times d_{ent}} \text{ and } W_3 \in \mathbb{R}^{d_{ent} \times d_h} \text{ is learnable weight. } M \in \mathbb{R}^{n \times n} \text{ is visibility matrix.} \]

The visibility matrix is to control whether an entity is visible to a tabular model token, as shown in Equation (1). Besides, it only works in sub-token (or cell) level fusion.

\[ M_{i,j} = \begin{cases} 
1 & i \text{ is in the same column and row with } j \\
0.5 & i \text{ is only in the same column or row with } j \\
0 & \text{others} 
\end{cases} \quad (1) \]

where, \( m \in (0, 1) \) is half visible hyper parameter.

After getting attentive entity embedding \( (H) \), we concatenate tabular model embedding \( (TOK) \) with it as the module’s output.

5.3 Distribution Fusion Module

It is hard for a pre-trained tabular model to capture the entire column distribution, which is important to analyze metadata tasks. To learn numerical distribution, we extract 31 data statistics features (including progression features, string features, number range features, and distribution features) and 6 categories for each field\[40\]:

- Progression features: ChangeRate, PartialOrdered, Ordered-Confidence, ArithmeticProgressionConfidence, GeometricProgressionConfidence.
- String features: AggrPercentFormatted, medianLen, Length-StdDev, AvgLogLength, CommonPrefix, CommonSuffix, Cardinality, AbsoluteCardinality.
- Number range features: Aggr01Ranged, Aggr0100Ranged, AggrInteger, AggrNegative, SumIn01, SumIn0100.
- Distribution features: Benford, Range, NumRows, KeyEntropy, CharEntropy, Variance, Cov, Spread, Major, Skewness, Kurtosis, Gini.

(2) Field categories:

Including FieldType (Unknown, String, Year, DateTime, Decimal), IsPercent, IsCurrency, HasYear, HasMonth and HasDay.

The module architecture is illustrated in Figure 4c. Tabular model embedding and field categories are added together after linear layer and embedding lookup respectively. The output of the module is the concatenation of the embedding and statistics features.
5.4 Multi-task Learning

We use 19-class Cross Entropy loss for measure type, and standard binary Cross Entropy loss for measure/dimension dichotomy, common measures and dimensions, measure pair tasks. In common measures and dimensions task, we use scores of “true” class as their commonness scores. For aggregation function and dimension type task, we respectively use 7- and 255-class Cross Entropy loss and average loss of each field by the number of its ground truth, because there may be more than one ground truth for one field, which will lead to an unbalanced loss without averaging the fields.

6 EXPERIMENTS

Experiments are conducted on all metadata tasks. We respectively use TAPAS [13] and TABBIE [15] as a pre-trained tabular model to illustrate the effectiveness of Metadata model. To compare with Metadata model performance, we do experiments on rule-based, traditional machine learning, and pre-trained tabular model baseline.

To illustrate the effectiveness of Distribution fusion and Knowledge fusion, we ablate both of them respectively and together. To illustrate the importance of analysis metadata, we apply metadata to down-stream tasks. Besides, according to different tasks, we conduct additional baseline and case studies.

All the experiments are run on Linux machines with 24 CPUs, 448 GB memory, and 4 NVIDIA Tesla V100 16G-memory GPUs. We use hit rate@k ($HR@k = \frac{\#hits@k}{\#samples}$) to evaluate recommendation tasks on each table and accuracy to evaluate classification tasks on each field.

6.1 Experiment Details

6.1.1 Dataset preprocessing. To avoid data leakage and imbalance, we carry out the following steps. Note that for a fair comparison, we adopt TURL dataset train/valid/test split, so do not perform the following steps.

1. **Table Deduplication**. To avoid the “data leakage” problem that duplicated tables are allocated into both training and testing sets, tables are grouped according to their schemas$^3$.

2. **Down Sampling**. After deduplication, the number of tables within each schema is very imbalanced – 0.23% schemas cover 20% of these tables. To mitigate this problem, we randomly sample unique tables under the threshold (10 for the Chart dataset, 2 for the Pivot dataset, 1 for Vendor & T2D & Semtab dataset).

The schemas are randomly allocated for training, validation and testing in the ratio of 7:1:2.

For spreadsheet datasets, we follow the steps in [40] and [41], including extraction charts and pivot tables.

For all six datasets, we extract data features (§5.3) and map knowledge graph (§5.2) as the input of our models.

6.1.2 Metadata Model details. In detail, we have the following model size:

- **Sub-token level**: transformer encoder: 2 layers, 8 heads, dimension of embedding: $d_{tok} = 192$, $d_{ent} = 100$, $d_{h} = 64$

- **Column level**: transformer encoder: 2 layers, 8 heads, dimension of embedding: $d_{tok} = 128$, $d_{ent} = 100$, $d_{h} = 64$

We train Metadata on 10 epochs, 64 batch sizes, AdamW optimizer with $1 \times 10^{-4}$ learning rate and 0.001 weight decay. We choose half visible hyper parameter $m = 0.5$. It’s worth noting that, due to insufficient supervision, aggregation function tasks use the result of epoch 5.

6.2 Baselines

To compare the performance of Metadata model on all tasks, we generally use three kinds of baselines – the rule-based baseline, the traditional machine learning baselines, and the pre-trained tabular model baselines. Note that, there is no direct existing experiment on metadata tasks, so we slightly change the above models to adapt to our tasks.

6.2.1 Rule-based Baseline. The rule-based model predicts an output for every field based on the field surface attributes since it is off-the-shelf ready to use and does not involve additional training. The specific rules for each metadata task are as follows:

1. **Measure/dimension dichotomy**: The model predicts Measure when the input field is numerical (i.e., the field consists purely of numbers); otherwise, the model predicts Dimension.

2. **Nature key**: The model predicts a field to be Natural Key if and only if the field is the leftmost field among fields with cardinality of 1 (i.e., the field contains all unique data values).

3. **Common breakdown**: The model predicts a field to be Common Breakdown if and only if the field is the leftmost field among dimension fields whose cardinality is less than 0.4.

4. **Common measures**: The model predicts a field to be Common Measure if and only if the field is the rightmost among numerical fields.

5. **Dimension type**: The model predicts every field to be of the type with most samples (sports.sports_team in our case).

6. **Measure type**: The model predicts every field to be of the type with most samples (Count in our case).

7. **Measure pair**: The model predicts two fields to be a Measure Pair if and only if they are contiguous Measure fields.

8. **Default Aggregation Function**: The model predicts the default aggregation function of every field to be the function with most samples (SUM in our case).

6.2.2 Traditional Machine Learning Baseline. Traditional machine learning is widely used due to both the effectiveness and the efficiency. In data profiling[2], there exist several state-of-art works that apply traditional learning with specifically extracted features. To compare with them, we design a traditional machine learning baseline by adopting a series of manually designed field data features (elaborated in §5.3).

The implementation details of each metadata task are as follows:

1. **In binary classification tasks (i.e., measure/dimension dichotomy, measure pair)**, a single classification model is trained, and the model hyperparameters are determined based on validation set metrics.

2. **In ranking tasks (i.e., common measures, common breakdown, and primary key)**, we train a binary classifier to make predictions for each field and rank the predictions with the raw probabilities given by the model outputs.

$^3$Two tables are defined to have the same schema if they have the same number of fields, and each field’s data type and header name are correspondingly equal.
Table 6: Main Results on Metadata Tasks. Experiments are repeated 3 times and the metrics are averaged. All numbers in %. Blod font is the result that achieves the best performance on all models. Δ means “Metadata(TABBIE)” metric minus corresponding evaluation metric, and color formatting reacts value size.

| Model               | Msr Type | Msr Pair | Com. Type | Com. Measure | Dim Type | Msr Type | Msr Pair | Aggregation |
|---------------------|----------|----------|-----------|--------------|----------|----------|----------|-------------|
|                     | Acc. Δ   | HR@1 Δ   | HR@1 Δ   | HR@1 Δ       | Acc. Δ   | Acc. Δ   | Acc. Δ   | Acc. Δ      |
| Rule based          | 94.65 Δ  | 81.58 Δ  | 12.91 Δ  | 51.85 Δ      | 11.11 Δ  | 65.27 Δ  | 7.53 Δ   | 12.99 Δ     |
| GBDT                | 96.30 Δ  | 93.41 Δ  | 1.07 Δ   | 59.49 Δ      | 8.47 Δ   | 67.51 Δ  | 5.29 Δ   | 24.44 Δ     |
| Random Forest       | 94.47 Δ  | 94.46 Δ  | 0.02 Δ   | 61.59 Δ      | 6.37 Δ   | 68.34 Δ  | 4.46 Δ   | 45.56 Δ     |
| TURL                | 97.31 Δ  | 92.01 Δ  | 2.47 Δ   | 60.68 Δ      | 7.28 Δ   | 69.73 Δ  | 3.07 Δ   | 96.55 Δ     |
| Metadata(Tapas)     | 98.11 Δ  | 94.30 Δ  | 0.18 Δ   | 65.91 Δ      | 2.06 Δ   | 71.27 Δ  | 1.53 Δ   | 97.97 Δ     |
| Metadata(Tabbie)    | 98.48 Δ  | 94.48 Δ  | 0.00 Δ   | 67.96 Δ      | 0.00 Δ   | 72.80 Δ  | 0.00 Δ   | 97.33 Δ     |

(3) In multi-class classification tasks (i.e. dimension type, measure type and default aggregation function), we directly train a multi-class classifier using traditional machine learning algorithms.

In the experiment, we choose GBDT, Random Forest, Adaboost, and Naïve Bayes to evaluation on each task, and display the performance of the best two baselines (GBDT and Random Forest) to compare with Metadata model.

6.2.3 Pre-trained Tabular Model Baseline. TURL is a structure-aware Transformer encoder to model the structural information about tables on table interpretation tasks and achieves State-Of-Art results. They also use Masked Entity Recovery pre-training objective to learn knowledge information about entities in relation tables. Since this model consider entity information, we select it as a strong baseline to show the effectiveness of Metadata model architecture in fusing the knowledge graph information. To evaluate the performance of TURL in metadata tasks and have a fair comparison, we take the following steps:

1. Map knowledge entity. Because entity linking in Semtab2020 dataset only have Wikidata id but TURL use DBPedia id in embedding, we use WikiMapper to map Wikidata id to DBPedia id. It’s worth noting that dimension type task is one of TURL’s original tasks, so WikiMapper is not used in this task. Thus, this task can prove that Metadata still exceed TURL without the influence of WikiMapper (details are shown in §6.3).

2. Interpret the original table. We use TURL pre-trained encoder to interpret the original table and get the embedding representation of flat tables.

3. Train and evaluate metadata tasks. To do classification on metadata tasks, we use the same sub-token level encoder, column level encoder and loss function as our Metadata model while adopting the same training strategy with TURL.

6.2.4 More baselines for measure type. As mentioned in §2.2, there are several existing works focusing on the similar or same task. To compare dimension type, we apply the State-Of-Art table interpretation model – TURL as a baseline as discussed in §6.2.3. To compare measure type, we apply the most relevant unit detection work [33] (RQN) as a baseline. It proposes a feature-based method to automatically determine the quantity names for column values. It uses hand-designed rules to extract features from raw tables in both column values and name. The extracted features are used to train a random forest classifier. The results are shown in Table 7.

It’s worth noting that to avoid label leaking, we do not use Knowledge fusion for column type and property in dimension type task. For the measure type task, we focus on the columns that can not be directly linked with properties from the knowledge graph for these columns are hard for existing table interpretation methods. To imitate those tables, we mask knowledge graph information for the field to train and evaluate in measure type task.

6.3 Main results

As described in Table 6, we find the following insights across tasks by comparing with baseline.

Semantic information captured by the pre-trained tabular models brings great gain to metadata tasks. Metadata model outperforms rule-based and traditional machine learning, especially on common field roles and semantic field types (e.g. dimension type outperforms 70% on average). Data features used in traditional machine learning are good at representing field distribution, while they lack semantic information and entire table understanding. Pre-trained tabular model fills the gap.

In Table 6, Metadata (TABBIE) achieve the top one on all common field roles tasks, and outperforms traditional machine learning and TURL by 1.19%, 7.38% and 4.27% (HR@1) on average. It outperforms the rule-based baseline by more than 10% on each task. Because common tasks can not be directly identified by field type or positional information, they are not easy for rules.

In Table 7, Metadata models outperform RQN more than 10%. RQN has a limit to identifying quantities from the existing unit in the column, while Metadata model learns more.

Successful distribution and knowledge fusion bring a better representation of fields. Metadata models exceed the other pre-trained tabular model, especially on common field roles, measure type and measure pair tasks. On all common field roles tasks,

Table 7: Measure Type Additional Results.

| Model               | Measure Type Acc. |
|---------------------|-------------------|
| RQN                 | 65.55%            |
| RF                  | 70.09%            |
| Metadata(TAPAS) w/o DF | 80.39%          |
| Metadata(TABBIE) w/o DF | 78.62%          |
Metadata (TABBIE) outperforms TURL by 4.27% (HR@1) on average. On measure type task, Metadata (TAPAS) outperforms TURL by 6.84%. On measure pair task, Metadata (TAPAS) outperforms TURL by 8.29%. Different from the other pre-trained tabular model, our Metadata model explicitly fuses distribution and knowledge information. Thus, it can better represent the whole field and integrate useful external information. More analyses about distribution and knowledge fusion are described in ablation experiments (§6.4.1).

On dimension type task, Metadata (TAPAS) achieves the top result and exceeds TURL by 1.42%. It’s worth noting that this task is one of TURL’s original tasks, while our Metadata model still has advantages. Benefit from better knowledge fusion and pre-training model representation.

The replaceable pre-trained tabular model brings a good opportunity to get better performance. Metadata (TABBIE) outperforms Metadata (TAPAS) on most tasks. Besides, Metadata (TAPAS) and Metadata (TABBIE) outperform TURL by 2.55% (2.98%) on average. In addition to distribution and knowledge infusing, the replaceable pre-trained tabular model is another key point. More and more pre-trained tabular models are emerging with better performance, and Metadata model can be based on almost all transformer-based pre-training models. Thus, it’s convenient to replace them with the models in Metadata, which can further improve performance and choose the best one.

Aggregation Function. In Table 6, Random Forest achieves the top one. The traditional model with a professional data feature is a better choice for this task. On the one hand, in this task, semantic information does not play a dominant role and can even interfere with identification. On the other hand, the high dimensionality of semantic information brings a large number of parameters to Metadata model, and the inadequately supervised signal leads to underfitting.

6.4 Ablation results

6.4.1 Ablating DF and KG. We ablate Distribution Fusion and Knowledge Fusion respectively and together, and the results are shown in Table 8.

Successful distribution and knowledge fusion gain the performance of models together. Compared with ablation models, for both TAPAS and TABBIE, the top-1 scores come most frequently from the non-ablation Metadata model. Especially, KG and DF boost 4.34% (3.92%) for common breakdown task with Metadata TAPAS
Especially on dimension type and measure type tasks, knowledge graph information is of great help to understand tables. For measure type task, KG improves performance by 2.75% (5.96%) on Metadata TAPAS (TABBIE) (w/o DF - w/o DF KG). The model could get additional information from the knowledge graph to enhance the understanding of the table. And it learns the useful pattern between measure type and cell entity/column type.

6.4.2 More analysis on ablating DF. Why can DF improve performance? And can this improvement be replaced by more pre-training? To answer those important questions, we find that the gain from DF is strongly correlated with the table length, as shown in Figure 5 and Figure 6. The values of those charts come from “Metadata(TAPAS) w/o DF” in blue and “Metadata(TAPAS) w/o DF KG” in orange. The improvement of DF is more pronounced as the length of the table grows. There are 3 reasons: (1) More rows involve more distribution information, which can be discovered by data features and distribution fusion module. (2) Long table distribution information is hard to be captured by a semantic model. Pre-trained language models are good at capturing context information. While the distribution of long tables needs structure and complex calculation ability, which is hard to a semantic model. (3) Sequence length of some long tables exceeds the upper length limit. Truncating sequence is the most common way, which will lead to information loss. While distribution fusion is a good way to fill the gap. It’s worth noting that long tables are always a limitation of existing pre-training tabular models (such as TAPAS, TABBIE and so on), and distribution fusion is an effective way. The change of hit rate as the number of rows grows is also an interesting phenomenon. As shown in Figure 5, tables with too few or many rows are hard to be inferred metadata. While, as shown in Figure 6, hit rate decreases with the increase of token number. Because few rows are not equal to a few tokens. Few rows mean each field includes less information, which leads to hard inference. While few tokens mean that the table is small, in another word, few columns, few rows and short cells. Those tables are easier for pre-trained language models because it’s easy to attend to important information from the table.

6.5 Downstream analysis task results

To illustrate the importance of analysis metadata, it is applied to downstream task (in the case of chart). Chart generation can be divided into 2 steps – field selection and design choice[40]. Metadata is compared with DeepEye[22] and Data2Vis[9] on the field selection step, which evaluates whether the selected fields for the composition chart are right. Field selection of metadata consists of the following ways: (1) For bar and pie chart, we adopt the first common measure field and the first common dimension field as the selected fields. (2) For scatter and line charts, we adopt the first common measure field as the selected field.

The results are shown in Figure 7. Metadata can yield very promising performance, which is almost double of the results of DeepEye and Data2Vis. It indicates that analysis metadata prompts performance of down stream task. Meanwhile, it splits the chart generation into multi-steps, which increases the robustness and interpretability of the data analysis system.

6.6 Case Study

Measure / dimension dichotomy is the cornerstone of all tasks, and its accuracy affects all the tasks. Thus, we focus on its case study. The key point of this task is how to identify the dimension field from numerical fields. Table 9 shows some cases that Metadata successfully identifies dimension from numerical fields. “ID” and “QP Code” show that when numerical values are hard to identify the field, Metadata learns semantic representation from the pre-training model and successfully identifies the field. “Style” shows that when the semantic of the header is hard to identify, Metadata learns the numerical distribution to identify the group-by field.

7 REAL-WORLD DEPLOYMENT

By automatic recognition of table metadata, automatic analysis can be conducted. With large corpus and smart supervision mentioned in §4, We have worked with a leading IT company and deployed a simplified version of the tree-based metadata model as a fundamental spreadsheet table understanding APIs. The model is integrated with intelligent features, such as one-click data analysis (insights mining[10]), natural language QA, pivot table, and charts recommendation in its product.

8 CONCLUSION

In this paper, we propose the novel analysis metadata for tabular data analysis and collected a large corpus with supervision by using smart supervisions from downstream tasks, public datasets, and our manual labels. Then we propose Metadata model to understand field distribution and utilize knowledge graph information. We conduct several experiments to illustrate the importance of metadata tasks and the effectiveness of Metadata model. Analysis metadata and our experiences with tabular model design could benefit a wide variety of downstream tasks.

REFERENCES

[1] [n.d.]. ISO 80000-1:2009 Quantities and units. https://www.iso.org/standard/30669.html. Accessed: 2021-09-15.

[2] Ziauasch Abedjan, Lukasz Golab, Felix Naumann, and Thorsten Papenbrock. 2018. Data profiling. Synthesis Lectures on Data Management 10, 4 (2018), 1–154.

[3] Mesures BIdPe. 2019. SI Brochure: The International System of Units (SI). ed. https://www.bipm.org 2019 (2019). https://www.bipm.org/en/publications/si-brochure
