CD-003

センテンス埋め込みを用いたデータ中身の類似度による関連データセット検索ツール
Query by Dataset Based on Instance Similarities Generated by Sentence Embeddings

姜　逸越†　　坂巻　慶行†　　野間　唯†
Yiyue Jiang†　Yoshiyuki Sakamaki†　Yui Noma†

Abstract

Before analysis from data for various purposes, such as usage related to artificial intelligence (AI), it often takes three to four times longer at the beginning stage for discovering and preparing relevant datasets from a heterogeneous database, as well as systematically organizing their structures, such as aggregating similar attributes, for later processing. The large data volume, and sometimes lack of dataset's domain knowledge or their access limitations make it even harder to cut down the related workloads and achieve it efficiently. Like a web search engine, we propose a navigating way of finding out the similarities and relevance both semantically and syntactically, to improve the effectiveness and efficiency for data discovery and preparation. Our proposed method mainly takes advantage of vectors generated from *sentence embeddings*, to combine the syntactic and semantic features from each attribute's instances of both the attribute names and values under them, which are altogether regarded as a *virtual document*. The similarities between such features of attributes, are utilized to further generate a dataset relevance score. Thus relevant dataset pairs or clusters can be automatically found and recommended for usage at later stages for analysis. We quantitatively evaluate the relevance-finding performance of this model with conventional automatic methods, and provide examples of results from open datasets, while a use case scenario is also introduced at the same time.

1. Introduction

Nowadays, the analysis of heterogeneous data from dilated data lakes and databases becomes more common. Usually the analysis requires target data to be firstly selected as relevant ones of interest during data preparation stage, such that people can obtain desired datasets and establish proper disciplines for later analysis. However, in practice, there is no fully integrated schema or global schema, which may probably require impractical cost to be built, to consistently describe the data [1], and sometimes people are not allowed to access the content [2]. Hence, much more time has been spent on discovering relevant relationships among the datasets rather than analysis itself. According to surveys over several industries, data preparation including relevant data retrieval as well as pre-processing until it meets with the standards for analysis that follows, usually takes around 70 to 80 percent of the whole time [3]. Hence, it is indispensable and has been expected to cut down the manual workloads involved in the preparation stage, such as by providing potential automation solutions.

1.1 Data Relevance Discovery by Schema Matching

As one of the key workflows in relevant data discovery during data preparation, data integration which is aimed to integrate same or relevant data, has been developed with some navigating tools that can help finding out relevant datasets. More specifically, the integration of schemata belonging to different datasets, called *schema matching*, has been necessary to get the data systematically organized, and has potentials to achieve the above goals of discovery for data relevance, with investigation of the schemata trying to obtain relevance among columns that belong to each schema.

1.2 Problems

To achieve relevant data retrieval by schema matching, there have been existing tools that are automated and commonly used for schema matching. However, as for relevant data retrieval purposes, people have to resolve some inherent problems of such tools for being unable to retrieve enough relevance, or not feasible for some common use cases for those purposes.

Specifically, many currently existing tools for schema matching investigate the instances from attribute names as well as the values inside the columns under them, by extracting the syntactic significance for almost all words or phrases as the counted units [4, 5]. But such tools are sometimes not appropriate enough for data discovery, since they only recognize completely identical sub-sequences as same units, and hence have no identification metrics for term definitions or semantic relevance. Thus for discovery on different terms or expressions aimed for a same meaning or entity such as synonyms, they will probably result in low recall as poor performance. On the other hand, some other tools achieve the inner semantic relevance by referring to some definitive or semantic dictionaries and ontology databases that usually need to be constructed in advance [5, 6, 7]. Although those methods are able to indirectly obtain semantic information by metadata reference, the referral and comparative procedures bring about heavy computation complexity or manual intervention. While there is usually big data with which people need to find purpose-relevant data, that restricts them to getting similarities from the attribute names only, and this problem has been also notified in the work of Sato, A. et al. [8]. So such tools are also not appropriate enough since they cannot take into account the similarities of instances from actual values. For example, this will get incorrect results misled by attribute names that have no actual meanings, or simply cannot be applied to handle with data that has no systematically organized attributes (column names).

†株式会社富士通研究所　Fujitsu Laboratories Limited

1.3 Contributions

In this study, we proposed another instance-based schema matching method based on similarities among the attribute names as well as their actual values. Nevertheless, we mainly employ it for relevant data retrieval. The proposed method can characterize data with a syntactic metric as well as semantic features. Still like some of the existing tools [4, 5, 6, 7], it is aimed to achieve automated recommendation for relevant dataset candidates during the data discovery process. Besides, human intervention is also necessary to make final decisions for selecting the desired ones from a converged range with recommended candidates. However, by considering the combined feature relevance at instance level, it provides an improved recommendation with more likelihood for users to reach the desired results. Users who want to get relevant datasets for specific purposes, are usually aware of what the desired relevant ones should be like but not familiar with the others. Hence, a high-quality recommendation can more appropriately converge the candidate investigation range for users, to obtain desired datasets much faster and more accurately over relevant candidates with most likelihood.

To reach the goal stated above, the most important issue to address is to solve the bottleneck problems owned by conventional methods that no instance-level semantic metric has been investigated. The solution is to be explained and validated with typical example results as well as a quantitative evaluation comparing with existing ones.

In addition, after applied with the model mainly developed for generating a ranked list of most relevant datasets, which uses similarities obtained from schema matching, a search-engine-like system will be introduced and shown with expected results. Such results are obtained from open data in a tabular format. After that, introduction about a use case scenario will be given to show how the model or system is able to help data discovery for later data analysis while given heterogeneous data.

2. Model

In this section, related concepts and definitions for the proposed model are introduced, including the detailed matching models, top-*k* matches, and feature extraction algorithms as well as similarities between features, selected as the relevance scores.

2.1 Table Relevance and Schema Matching Model

For a pair of tables consisting of datasets *D, D'* which contain two schemata *S, S'*, respectively, the process of finding out dataset relevance results from schema matching, that refers to alignment between the two sets of schema attributes $A=\{a_1, a_2, ..., a_m\}$ and $B=\{b_1, b_2, ..., b_n\}$ (including their respective contents or values). Algorithms that are sometimes called matchers for the matching process are used to generate similarities from comparison between attributes of pairs based on extracted features which are to be explained in the next part. As a result, a similarity matrix, *M(D,D')* for the investigated pair of datasets is usually formed with all the pairwise attribute similarities as its entries $M_{i,j} = Sim(a_i, b_j)$. The similarity value $M_{i,j}$ which is a real number indicates the degree of relevance between a_i and b_j, it is calculated in this study as cosine similarity between two feature vectors that are extracted from two compared columns. As for the relevance of dataset level, the overall sum of every top similarity $TopSim(a_i)$ for every attribute a_i in one dataset (one set of attributes), generated with any attribute in the other set ($B=\{b_1, b_2, ..., b_n\}$), is regarded as a relevance value (Rel_{AB}) between datasets inside the investigated pair:

$$Rel_{AB} = \sum_{i=1}^{m} TopSim(a_i) = \sum_{i=1}^{m} Max(Sim(a_i, b_1), Sim(a_i, b_2), \dots, Sim(a_i, b_n)).$$

In this way, it is not difficult to notice that two relevance values would be generated for respective attribute sets inside each dataset pair, and thus such relevance is a directed measurement parameter.

2.2 Algorithms for Feature Extraction and Similarities

We introduce the feature extraction method based on instance level. For matching problems using data like open datasets, since the quality of metadata is not ensured, and sometimes even the attributes are named with meaningless strings or code IDs, similarities are necessary to be generated from instance level to be able to achieve accurate and fruitful matching. Given structured data format such as that of tabular datasets, each schema consists of a set of attributes that are usually located on the top of columns, namely as the header row. In a similar way with the existing typical instance-level approaches such as the one employed in COMA++, all contents inside a column is regarded as a virtual document, and thus column similarity calculation is done by investigating the similarity between features extracted from the two virtual documents.

All virtual documents containing words are firstly preprocessed as will be explained in details later. After that, methods like COMA++ usually focus on *tf-idf* features which can be extracted by vectorization upon all the virtual documents under investigation, to generate a list of m index words: $w_1, w_2, ..., w_m$. The *tf-idf* index words are syntactically significant ones among the investigated virtual documents and each document (or column) is also assigned with a set of *tf-idf* values corresponding to the list index words. That is, for a given column as a virtual document D_i, it is generated with a set of *tf-idf* coordinates $t_{D_i}(1), t_{D_i}(2), ..., t_{D_i}(m)$ corresponding to the index words as the coordinate names. As some typical instance-level approaches such as those in COMA++, they stop here right after the syntactic vectorization and just use similarities between the pairwise coordinate sets of two investigated columns as the final similarities.

In our proposed approach, however, because the syntactically significant index words may also have relevant or similar meanings in natural language explanation with one another, and hence they should not be strictly separated as totally irrelevant indices, we continue to extract semantic features as well and obtain word vectors for them based on Word2Vec model [9] as a typical word embedding method for vectorization. We employed a 200-dimensional Japanese Word2Vec model and a 300-dimensional English model, trained with Japanese and English Wikipedia

articles, respectively. Thus, a *tf-idf* index word w_j (j ranges from 1 to m) was able to be converted into a vectorized representation as $\boldsymbol{v}_j = (v_j(1), \dots, v_j(dim))$, where dim equals 200 or 300 for processing Japanese datasets or English datasets, respectively. After that, for each virtual document, or column, the word vectors of the *tf-idf* index words were summed up with their corresponding *tf-idf* values, out of the coordinates for the virtual document, as the boosting weights to form a syntactically-and-semantically combined feature vector as;

$$\boldsymbol{FV}_{D_i} = \sum_{j=1}^{m} \left[t_{D_i}(m) \cdot \boldsymbol{v}_j \right].$$

Such model combining word embeddings by assigning weights that are proportional to the words' respective frequency, has been given an experimental proof in [10] to represent overall semantic features of long text out of a sentence (or a virtual document here). It is called sentence embeddings and we have applied this model to reach a combination of syntactic and semantic features from the columns to reveal their inner connections. As mentioned in the previous part, based on the proposed feature vectors, the similarity we chose to be implemented and compared for evaluation is cosine similarity.

3. Experiment and Evaluation

3.1 Dataset

Enterprise Data Tables. For preliminary implementation with Japanese datasets and simple example demonstration purpose for schema matching only, we used two facility ledger datasets provided by FUJITSU FACILITIES Limited (FFL), specifically, tabular data files from Nasu and Kumagaya factories of FUJITSU Limited. They contain facility management information recorded in Japanese including few English terms and abbreviations to handle with certain domain terminology such as "kVA" representing kilovolt-ampere. Since they only contain tens of columns and we can easily obtain domain requirements for the column or attribute mapping as the reference ground truth, we carried out the experiment with these two datasets as an early-stage trial.

Public Open Data Tables. Two datasets mentioned above are accompanied with ground truth for schema matching which is desired in practical usage, but are obviously far from enough for evaluation and benchmark purposes. Although there is no publicly known data with available ground truth for searching relevant data, we still perform search on open local government data, using data tables from another source of such data, as search queries. In this report, data tables from Kanagawa prefecture open data [11] with various domain information are used to try finding tables in Tokyo metropolitan data source [12] in respectively relevant domains.

3.2 Preprocessing

All datasets in the tabular data format were loaded into Python Pandas DataFrames as string type and cleaned out 'nan' values resulted from blank cells to avoid unnecessary noises. Within the scope of the study in this report, since the benchmark datasets as well as our two enterprise datasets have no purposes for dealing with columns that contain numeric values only, we ignore the matching for numeric data and focus on processing string or text data both in Japanese and English.

Firstly, as similar problems to those faced with by people in natural language fields, the differences due to composing or wording preferences among people cannot be simply regarded as typos and ignored. For the case of English, lowercase and punctuation removal need to be firstly done which is relatively simpler. For Japanese, since there are much more various input formats, a Japanese text converter for unification in Python, Jaconv module [13] was utilized to convert half-width Japanese characters, environment-dependent characters etc. into unified full-width Japanese characters, with necessary equivalent punctuation marks. To parse the units with semantic meanings such as single words or phrases in a vocabulary, MeCab [14] was used with reference to an International Phonetic Alphabet (IPA) system which originally showing word pronunciations. The parsed units were later tokenized and regarded as the smallest units for processing. Such parsing process can be also called morphological analysis for generation of meaningful partitions from word aggregations. Also, the parsed contents in cells within one column were also joined altogether with spaces to form a virtual document out of the column.

3.3 Implementation

As previously mentioned in Section 2, after preprocessing for format unification and morphological analysis, contents out of the values from columns of text type only, were processed with vectorization by *tf-idf* analyzer from Python scikit-learn to be characterized into a syntactic feature vector for each column, and the vector's corresponding index words that are syntactically significant enough which usually appear highly frequently.

After syntactic vectorization with scikit-learn *tf-idf* functional module, we did semantic word vectorization, for each *tf-idf* index word. The word vectors were trained and generated with Japanese and English Wikipedia article dump data after different language-specific morphological analysis and with different vector dimensions. The training process was set the window size as five words and employed negative sampling for higher training efficiency. Then as shown in previous formula, the word vectors for *tf-idf* index words were weighted with their respective *tf-idf* values with respect to each column or virtual document. Finally, based on these combined feature vectors which are often implemented for sentence embeddings, cosine similarities were selected, calculated pairwisely, and regarded as scores for generating a ranked list as the result of matching each column.

3.4 Evaluation

As for heterogeneous data retrieval and integration problems, top-*k* matching [15] which provides a ranked list until the *k*-th position, has been proposed to give a focused search space for easier and better decisions by human, while faced with the

complex and heterogeneous settings. Since the ultimate goal in this study is to obtain relevant datasets by human, with tabular datasets as the target units, we should investigate the tables as the objects from the ones with highest relevance to lowest. However, since we reach it with an improved schema matching model and top-*k* matching are often utilized for schema matching, in the comparative evaluation with existing models, top-*k* of schema matching should also be considered. Besides, as far as we know, since there are no ground truth for matching and retrieving objects like tabular datasets, the matching feasibility is indirectly validated from schema matching. Thus, a quantitative performance comparison between the proposed model and conventional ones is firstly given based on top-*k* list results as for the purpose of schema matching. Within the top-*k* range of matching, the performance metrics are compared by using receiver operating characteristic (ROC) curves.

Firstly, we used the enterprise tabular datasets to simply investigate and intuitively show whether the revealed relevance had been successfully extended, or specifically, with respect to conventional syntactic methods, whether results retrieved with semantic features had been correctly added by our proposed model.

Since the numbers of columns inside the two datasets are within range of calculation and evaluation for column-level (attribute-level) mapping, we measured metrics for schema matching performance to preliminarily achieve evaluation upon our proposed model, using these two datasets. The performance is evaluated by a ROC curve which is made from a sensitivity vs. 1 – specificity graph. The sensitivity value, namely recall rate is calculated with the ground truth relevant column number for a query column, dividing the number of true positives inside the obtained search result. Additionally, since the specificity value is calculated as the irrelevant column number as ground truth (true negatives and false positives) dividing the number of relevant one (true negatives), 1 – specificity equals the ratio of result columns that are falsely selected as relevant (false positives). Hence, a ROC curve is used to characterize how well the ability of a relevance data matcher is, to overcome the trade-off between its true and false positives, and in an ideal situation, the point (0,1) on the upper left side in a ROC graph, should be achieved to get rid of all false positive search inside the selected ranges, and only ground truth positives are included.

For a more intuitive but meaningful trial purpose focusing on a practical use case, we perform search using tabular dataset to try to obtain relevant ones from a different database. By investigating results from the simulated search scenario, we can roughly get an image about whether such tool is useful for retrieving ideal relevant tabular datasets in a practical situation.

4. Results and Discussion

4.1 Additional Semantically Similar Result

For a simple instance of results from matching the two enterprise datasets provided by our company, there is a column containing a Japanese term “警報” which means “alert” among its values. When *tf-idf* characterization, which represents typical models of existing methods, is utilized to match this column as the query over the columns in the other dataset. Since none of the searched columns has exactly the same term (even it is just two characters in a row), the matching ended up with no candidate column of none-zero similarity. On the other hand, while being matched with our proposed method, candidate columns containing values such as “警備” (meaning “security”, “defense”), “非常” (meaning “emergency”), and “放送” (meaning “broadcast”) which are semantically relevant to the query column, were found and they are among the reasonable ground truth matching candidates, which seems reasonable as well. By looking into the word vector relationship inside the Word2Vec model, the cosine similarities with respect to the term “警報” were found to be 0.301, 0.207, and 0.157 for the three above terms respectively, which result in non-zero relevance scores for their corresponding columns, with the query one. This indicates that our proposed method has successfully added the semantically related scores, which is able to find with more false negatives to help improving the recall for schema matching.

4.2 Quantitative Evaluation by ROC Curves

Still, for the schema matching (or attribute mapping) carried out between the two enterprise datasets from FFL, a wider range of investigation within the result candidate list for matching from top-1 until the *k*-th position was performed with the analysis on sensitivity (namely the recall) and specificity (the true negative percentage out of all negatives), along with various *k* values, which were further used to generate a receiver operating characteristic (ROC) curves (sensitivity vs 1 - Specificity). The result candidate list is regarded as after sorted in descending order of matching scores. The curves are the ones generated with score types or similarities as cosine similarity. Then in combination with the features between which the similarities were generated, as extracted by three models, syntactic *tf-idf* only, semantic Word2Vec only, and our proposed syntax-semantics (combining *tf-idf* and Word2Vec) method, based on sentence embeddings, in this report, three ROC curves are shown in Figure 1 [16].

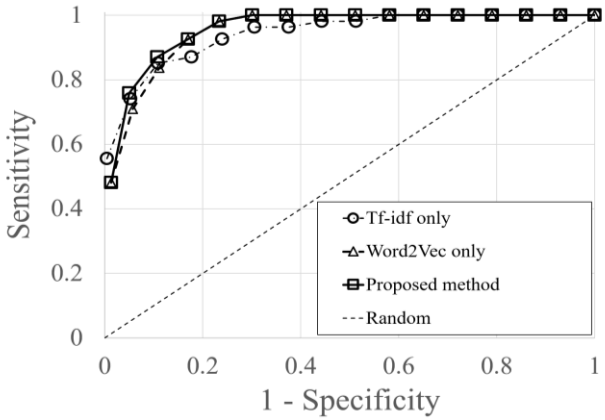

Figure 1. Receiver operating characteristic (ROC) curves of the results obtained with cosine similarity upon *tf-idf*-only features, Word2Vec features, and our proposed syntax-semantics (sentence embedding by combination of *tf-idf* and Word2Vec) features, respectively [16].

In a ROC illustration, the model of a matcher with a curve closer to the upper left corner is able to achieve a better tradeoff between the sensitivity (the true positive ratio out of all positives) and 1-specificity (the false negative ratio out of all negatives), being indicated as a better one for solving such matching problems. It is not difficult to notice that while provided comparison by drawing such ROC curves, the overall performance of our proposed feature model with the cosine similarity between such feature vectors (the curve with squares in the graph) is better over the other eight combinations. Nevertheless, it is slightly outperformed within the narrow range on the left side (when *k* ranges near 1 to 2). However, for the most cases including our enterprise one shown here, users usually prefer higher recall rates or sensitivities for their own working effectiveness, while provided with such converged candidate list (usually with sizes up to around ten), by which the efficiency has already been ensured to some extent. Thus, they can afford the cost for a wider investigating range over more candidate columns or attributes until top-*k* while *k* is no smaller than 1 or 2 for most use cases, indicating that our proposed method is more feasible for most general situations.

5. Proposed System Configuration and Usage

5.1 System Configuration

Figure 2 shows an illustration of system configuration for the proposed relevant data search engine, which takes a search query tabular dataset and a search candidate database as the input, and provides with the most relevant datasets extracted from the database as the output. It mainly includes an interface for the input and output file storing and transaction, a schema matching functional API in Python to generate both the column-level similarities as well as the table-level relevance scores, and a table data ranking part.

The whole processing of the system is completed in a parallel manner which is essential especially for searching in huge databases such as heterogeneous open databases or enterprise scale data. For a more intuitive image about how the system is able to retrieve real-world data for a given tabular dataset as the original master one, we explain in detail in the next part.

5.2 Introduction to a Scenario of Open Data Use Case

Provided the open data obtained from two local governments' open data catalog websites as described in Section 3.1, we use datasets from Kanagawa Prefecture website with each as a query tabular dataset, and try to find most relevant ones from the other website of Tokyo Metropolis. Such cases are common in practical situations especially for people such as data scientists, who often have large databases at hand but only need a very small relevant portion of them when there is a specific analysis purpose. Just like the two public open databases we are investigating, according to a 2017 survey by CrowdFlower [17], 41 percent of use cases for data scientists are based on such published open databases. However, since open data are usually stored in heterogeneous structures and with huge volume, people have to rely on relatively short corresponding metadata describing each tabular dataset, such as

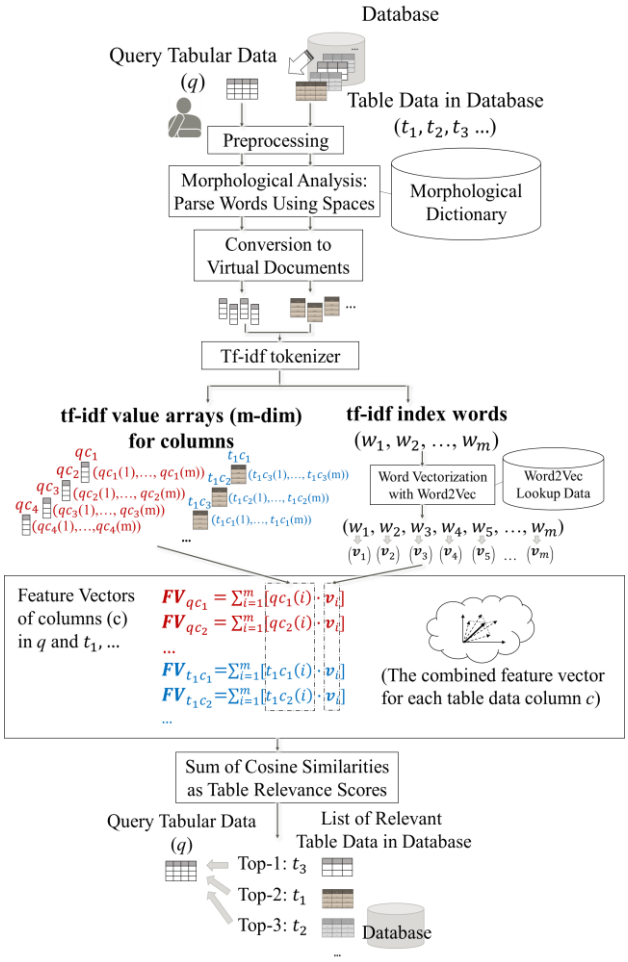

Figure 2. Illustration of system configuration for the proposed relevant data search engine.

table names, short content description, schema information, etc. Such metadata usually appears to be too short to contain enough information for relevance decision. So, manually extracting the useful data for the specific analysis purposes usually results in laborious work in checking the data content, and thus prevents people from efficient analytical work which may be actually the most important step.

Here with the proposed system aimed for efficient and highly accurate search on relevant data, we show several examples here using well-organized tabular datasets from Kanagawa prefecture as the master table data.

One example table dataset contains information about locations of public available automated external defibrillators (AEDs). Such information is necessary if there is a software development group, who wants do provide their users with applications, such like those used to find useful information for an emergency situation, or simply a map application on mobile phones. Obviously, only one such dataset from a local organization is not enough, and people making such products need to obtain much more data, which is relevant in AED locations from sources other than Kanagawa Prefecture. Searching by hand across different databases for them is laborious and ineffective work. However, with the proposed method, we finished searching Tokyo metropolitan open data that contains 10,367 datasets in comma-separated values (csv) format,

using this dataset for Kanagawa prefecture. The search was operated in a parallel manner with 24 CPU cores and completed in less than one hour with each relevance score generation in several seconds. The top 5 candidate tabular datasets as the search result are shown in Figure 3.

name of query master dataset: AED設置施設一覧.csv
search result:
→Top-1: AED設置箇所一覧.csv
→Top-2: 公共施設一覧.csv
→Top-3: 誰でもトイレのバリアフリー情報.csv
→Top-4: 観光施設一覧.csv
→Top-5: 港区の公共施設情報.csv

Figure 3. Illustration of system configuration for the proposed relevant data search engine.

By checking the content in the dataset as top-1 search result “AED 設置箇所一覧.csv”, the attributes are “都道府県コード又は市区町村コード”, “NO”, “都道府県名”, “市区町村名”, “名称”, “住所”, “緯度”, “経度”, “設置位置”, “開館時間”, “閉館日”, “利用可能日時特記事項”, “利用不可能日”, “設置台数” and “併設施設”, and the attributes of query master dataset are “設置自治体名”, “施設名称”, “所在地”, “設置台数”, “設置場所”, “使用可能時間帯”, “通年休業日”, “使用可能日・時間帯の補足”, “その他特記事項” and “最終更新日”. We also notice that among the values inside the top-1 result dataset, there is no strings like “AED” or “除細動器” for exact string match, and hence it is expected as difficult to be found with high similarity by conventional syntactic tools. It would also be left out by keyword searching with the specifically purposed strings in this use case like “AED”. Thus this scenario has provided an intuitive example for possible improvements and feasibility on a real-world use case.

5.3 Related Work on Open Data Benchmark

The trial example of searching open data using master datasets are intuitively useful for relevant information retrieval purposes, but is not able to provide with a quantitative validation on the feasibility for searching. According to literature research, F. Nargesian et al. has carried out a similar research for table union problems using open data [18]. The difficulty in seeking or setting up a general benchmark for open data has also been met with by them, and they also mentioned about the quantitative benchmark synthesized by themselves with projection and selection from several base tables. We are now using the same benchmark data from their published source and hope to get quantitative comparison results in the near future.

6. Conclusions

We propose an improved model for schema matching, in a similar manner with sentence embeddings that is mainly used to process natural language, and we employ this model to retrieve relevant data for data discovery purpose. It can extract both syntactic and semantic features to better achieve a tradeoff between the sensitivity and specificity. We provide a quantitative evaluation of the model with our enterprise datasets. Besides, after synthesizing similarities between such features from schema matching to generate a table relevance score, we carried out a search experiment on local government data between two different open sources, to intuitively validate its feasibility in practical situations. Thus, the proposed method has been proved to be able to shed light on a more efficient and effective solution for relevant data discovery, with a master tabular dataset as the query, based on the above features. In such manner, the manual expense is supposed to be cut down, by only investigating a converged range of candidate relevant datasets, while retaining high confidence to obtain target datasets for usage of desired specific purposes.

References

[1] R. C. Fernandez, et al., "Seeping semantics: Linking datasets using word embeddings for data discovery," In Proc. IEEE International Conference on Data Engineering, 2018, pp. 989-1000.
[2] M. Banek, B. Vrdoljak, A. M. Tjoa, and Z. Skočir, "Automating the schema matching process for heterogeneous data warehouses," In Proc. International Conference on Data Warehousing and Knowledge Discovery, 2007, pp. 45-54.
[3] S. Lohr, “For big-data scientists, janitor work is key hurdle to insights,” *The New York Times*, 17, 2014.
[4] M. Stonebraker, et al., "Data Curation at Scale: The Data Tamer System," presented at 6th Biennial Conference on Innovative Data Systems Research, California, U.S., 2013.
[5] D. Aumueller, H. H. Do, S. Massmann, and E. Rahm, "Schema and ontology matching with COMA++," In Proc. the 2005 ACM SIGMOD International Conference on Management of Data, ACM, 2005, pp. 906-908.
[6] Y. Zhang, et al., "System and method for fuzzy ontology matching and search across ontologies," U.S. Patent Application No. 14/678,943, 2015.
[7] H. H. Do, and E. Rahm, "COMA: a system for flexible combination of schema matching approaches," In Proc. the 28th International Conference on Very Large Data Bases, VLDB Endowment, 2002, pp. 610-621.
[8] A. Sato, et al., “Estimation of Relationships using Schemata and Primary Key Constraints,” In Proc. the 28th Annual Conference of the Japanese Society for Artificial Intelligence, 2014, pp. 1A21-1A21.
[9] T. Mikolov, et al., "Distributed representations of words and phrases and their compositionality," In Proc. Advances in neural information processing systems, 2013, pp. 3111-3119.
[10] S. Arora, Y. Liang, and T. Ma, “A Simple but Tough-to-Beat Baseline for Sentence Embeddings,” In Proc. International Conference on Learning Representations, 2017.
[11] Kanagawa Prefectural Government, “Kanagawa Prefecture Open Data Site,” 1995. [Online]. Available: http://www.pref.kanagawa.jp/docs/b8k/cnt/f534212/. [Accessed Apr. 23, 2019].
[12] Tokyo Metropolitan Government, “Tokyo Open Data Catalog Site,” 2017. [Online]. Available: http://opendata-portal.metro.tokyo.jp/www/index.html. [Accessed May 15, 2019].
[13] Y. Ikegami, “jaconv: Pure-Python Japanese character interconverter for Hiragana, Katakana, Hankaku and Zenkaku,” 2019. [Online]. Available: https://pypi.org/project/jaconv/. [Accessed May 15, 2019]
[14] T. Kudo, “Mecab: Yet Another Part-of-Speech and Morphological Analyzer,” 2013. [Online]. Available: http://taku910.github.io/mecab/. [Accessed May 15, 2019]
[15] A. Gal, "Uncertain schema matching," *Synthesis Lectures on Data Management*, 3.1, 2011.
[16] Y. Jiang, “A Syntax-Semantics Approach for Schema Attribute Identification,” In Proc. The 81st National Convention of IPSJ, 2019, pp. 2C02-2C02.
[17] CrowdFlower, *Data Scientist Report 2017*. [E-book] Available: https://visit.crowdflower.com/WC-2017-Data-Science-Report_LP.html. [Accessed: June 14, 2019]
[18] F. Nargesian, et al., "Table union search on open data," In Proc. the 44th International Conference on Very Large Data Bases, VLDB Endowment, 2018, pp. 813-825.