TableNet: An Approach for Determining Fine-grained Relations for Wikipedia Tables

Besnik Fetahu
L3S Research Center, Leibniz University of Hannover, Hannover, Germany
fetahu@L3S.de

Avishek Anand
L3S Research Center, Leibniz University of Hannover, Hannover, Germany
anand@L3S.de

Maria Koutraki
L3S Research Center, Leibniz University of Hannover, Hannover, Germany
FIZ-Karlsruhe Leibniz Institute, Karlsruhe Institute of Technology, Karlsruhe, Germany
koutraki@L3S.de

ABSTRACT

We focus on the problem of interlinking Wikipedia tables with fine-grained table relations: `equivalent` and `subPartOf`. Such relations allow us to harness semantically related information by accessing related tables or facts therein. Determining the type of a relation is not trivial. Relations are dependent on the schemas, the cell-values, and the semantic overlap of the cell values in tables.

We propose *TableNet*, an approach for interlinking tables with `subPartOf` and `equivalent` relations. *TableNet* consists of two main steps: (i) for any *source* table we provide an *efficient* algorithm to find *candidate* related tables with *high coverage*, and (ii) a neural based approach that based on the table schemas and data, determines with *high accuracy* the fine-grained relation.

Based on an extensive evaluation with more than 3.2M tables, we show that TableNet retains more than 88% of relevant tables pairs, and assigns table relations with an accuracy of 90%.

ACM Reference Format:
Besnik Fetahu, Avishek Anand, and Maria Koutraki. 2019. TableNet: An Approach for Determining Fine-grained Relations for Wikipedia Tables. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313629

1 INTRODUCTION

One of the most notable uses of Wikipedia is on knowledge base construction like DBpedia [2] or YAGO [14], built almost exclusively built with information coming from Wikipedia's infoboxes. Infoboxes have several advantages as they adhere to pre-defined templates and contain factual information (e.g. *bornIn* facts). However, they are sparse and the information they cover is very narrow. In most use cases of Wikipedia, availability of factual information is a fundamental requirement.

Wikipedia tables on the other hand are in abundance, and are rich in factual information for a wide range of topics. Wikipedia contains more than 3.23M tables from more than 520k articles. Thus far, their use has been limited, despite them covering a broad domain of information that can be used to answer complex queries, e.g., *"Award winning movies of horror Genre?"* the answer can be found from facts in *multiple tables*. However, question answering systems [1] built upon knowledge base facts, in most cases will provide *incomplete* answers or fail altogether. The sparsity or lack of factual information from infoboxes can be easily remedied by additionally considering facts from tables.

A rough estimate shows that from the 3.23M tables we can generate hundreds of millions of additional facts that can be converted into knowledge base triples [10]. This amount in reality is much higher, when tables are aligned. That is, currently, tables are seen in isolation, and *semantically* related tables are not interlinked. Table alignments allow us to access related tables as *supersets or subsets* (i.e. `subPartOf` relations) or `equivalent` tables, which in turn we can use to infer new knowledge and facts that can be used in QA systems or other use cases.

Determining the fine-grained table relations is not a trivial task. The relations are dependent on the semantics of the columns (e.g. a column with values of type `Country`), the context in which the column appears (e.g. *"Name"* is ambiguous and it can only be disambiguated through other columns in a table schema), cell values etc. Furthermore, not all columns are important for determining the table relation [13]. Finally, to establish relations amongst all relevant table pairs, we need to ensure the efficiency by avoiding exhaustive computations between all table pairs that can be cumbersome given the extent of tables in Wikipedia.

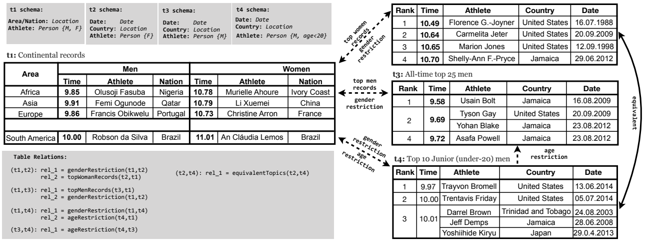
We propose *TableNet*, an approach with the goal of aligning tables with `equivalent` and `subPartOf` fine-grained relations. Our goal is to ensure that for any table, we can find with high coverage candidate tables for alignment, and accurately determine the relation type for a given pair. We distinguish between two main steps: (i) efficient and high coverage table candidate generation for alignment, and (ii) relation type prediction by leveraging table schemas and values therein. An extensive evaluation of *TableNet* over the entire English Wikipedia with more than 3.2 million tables, shows that we are able to retain table pairs with a high coverage of 88%, and predict the fine-grained relation with an accuracy of 90%.

2 RELATED WORK

Google Fusion Tables [5, 6, 13, 15] are the most significant efforts in providing additional semantics over tables, and to the best of our knowledge, only some of the works carried in this project are most related to our work, against which we provide fair comparisons [13].

WWW '19, May 13–17, 2019, San Francisco, CA, USA

ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313629



```
t1 schema:
Area/Nation: Location
Athlete: Person {M, F}

t2 schema:
Date:    Date
Country: Location
Athlete: Person {F}

t3 schema:
Date:    Date
Country: Location
Athlete: Person {M}

t4 schema:
Date: Date
Country: Location
Athlete: Person {M, age<20}
```

t1: Continental records

Area	Men			Women		
	Time	Athlete	Nation	Time	Athlete	Nation
Africa	**9.85**	Olusoji Fasuba	Nigeria	**10.78**	Murielle Ahoure	Ivory Coast
Asia	**9.91**	Femi Ogunode	Qatar	**10.79**	Li Xuemei	China
Europe	**9.86**	Francis Obikwelu	Portugal	**10.73**	Christine Arron	France
South America	**10.00**	Robson da Silva	Brazil	**11.01**	An Cláudia Lemos	Brazil

```
Table Relations:

(t1,t2): rel_1 = genderRestriction(t1,t2)
         rel_2 = topWomanRecords(t2,t1)

(t1,t3): rel_1 = topMenRecords(t3,t1)
         rel_2 = genderRestriction(t1,t2)

(t1,t4): rel_1 = genderRestriction(t1,t4)
         rel_2 = ageRestriction(t4,t1)

(t3,t4): rel_1 = ageRestriction(t4,t3)

(t2,t4): rel_1 = equivalentTopics(t2,t4)
```

t2: All-time top 25 women

Rank	Time	Athlete	Country	Date
1	**10.49**	Florence G.-Joyner	United States	16.07.1988
2	**10.64**	Carmelita Jeter	United States	20.09.2009
3	**10.65**	Marion Jones	United States	12.09.1998
4	**10.70**	Shelly-Ann F.-Pryce	Jamaica	29.06.2012

t3: All-time top 25 men

Rank	Time	Athlete	Country	Date
1	**9.58**	Usain Bolt	Jamaica	16.08.2009
2	**9.69**	Tyson Gay	United States	20.09.2009
		Yohan Blake	Jamaica	23.08.2012
4	**9.72**	Asafa Powell	Jamaica	23.08.2012

t4: Top 10 Junior (under-20) men

Rank	Time	Athlete	Country	Date
1	9.97	Trayvon Bromell	United States	13.06.2014
2	10.00	Trentavis Friday	United States	05.07.2014
3	10.01	Darrel Brown	Trinidad and Tobago	24.08.2003
		Jeff Demps	Jamaica	28.06.2008
		Yoshiihide Kiryu	Japan	29.0.4.2013

Figure 1: Table alignment example with `subPartOf` (dashed line), and `equivalent` relations (full line).

Carafella et al. [5] propose an approach for table extraction from Web pages and also provide a ranking mechanism for table retrieval. An additional aspect they consider is the schema auto-completion, specifically column recommendation for some input column to generate a "complete" schema. Our goals are different, while, we aim at providing more fine-grained representations of columns in a table schema, our goal is to use this information for table alignment.

Das Sarma et al. [13] propose an approach for finding related tables for two scenarios: (i) *entity complement* and (ii) *schema complement*. For (i), the task is to align tables with similar table schemas, however, with complementary instances. In (ii), the columns of a target table are used to complement the schema of a source table, with the precondition that the instances (from *subject* columns) are the same in both tables.

Our work is related to the first case. In the entity complement case [13] the *schema similarity* between two tables is computed in order to decide if a table can be considered for complementing the instances in another table. The similarity of the schemas is considered as a max-weight bipartite matching approach. The edges are weighted and are established between the columns in the disparate schemas. The weights correspond to the *string similarity* between column names and jaccard similarity between the column types (established from the values in a column through the `WebIsA` database). Despite the fact that this approach is unsupervised, we adapt it such that we find the best threshold of the max-weight matching, and consider tables to be either *aligned* or *not-aligned*.

3 PRELIMINARIES AND OVERVIEW

3.1 Terminology

We consider Wikipedia articles $A = \{a_1, \dots, a_n\}$; each article a is associated with a set of Wikipedia categories $\Psi_a = \{\psi_1, \dots, \psi_n\}$. From all the categories we induce the category graph Ψ, which consists of *parent* and *child* relations between categories ψ_i `childOf` ψ^p. The parent/child relations allow us to establish a hierarchical graph in Ψ. The level of a category is denoted by λ_ψ.

Next, we define the tables from an article a as $T_a = \{t_1, \dots, t_n\}$. A table t has a *table schema* consisting of columns $C(t) = \{c_1, \dots c_n\}$, where each column consists of a *textual description* and the set of all values $c_i = \langle desc, \{v_i^1, \dots, v_i^n\}\rangle$ assigned to the corresponding column cells in the table rows $t_i(r) = \{r_i^1, \dots, r_i^n\}$. More specifically, the cell value is indicated by v_i^k, where k is the row r^k and i is the column c_i. Cell values can point to existing articles in Wikipedia, that is $v_i^k = \langle a_k \rangle$, which we will refer to as *instance values*, or *primitive values* in cases of text, numbers, dates etc.

For the tables $T = \{t_1, \dots, t_n\}$ from A, we define two fine-grained types of relations for a table pair $\langle t_i, t_j\rangle$: (i) $t_i \vDash t_j$ where t_j is considered to be *semantically* a `subPartOf` of t_i, and (ii) $t_i \equiv t_j$ where t_i and t_j are *semantically* `equivalent`. We indicate a relation with $r(t_i, t_j) \neq \emptyset$, and define in the next section the table relations.

3.2 Table Alignment Task Definition

Table Alignment. Our task is to determine the *fine-grained relation* for a table pair $r(t_i, t_j)$ from the articles $\langle a_i, a_j\rangle$. The relations can be either `subPartOf`, `equivalent` or `none` (in case $r(t_i, t_j) = \emptyset$).

Definition 1 (subPartOf). *For $r(t_i, t_j) = \{$`subPartOf`$\}$ holds if the schema $C(t_i)$ can subsume either at the* ***data value*** *(i.e.* ***cell-value****) or* ***semantically*** *the columns from $C(t_j)$, i.e., $C(t_i) \supseteq C(t_j)$.*

Definition 2 (equivalent). *For a pair $r(t_i, t_j) = \{$`equivalent`$\}$ holds if both table schemas have* ***semantically similar*** *column representation, that is, $C(t_i) \approx C(t_j)$.*

3.3 TableNet Overview

In TableNet, for any *source* article $a_i \in A$, we first generate candidate pairs $\langle a_i, a_j\rangle$, such that the likelihood of the corresponding tables to have a relation is high. We describe the two main steps: (1) article candidate pair generation, and (2) table alignment.

4 ARTICLE CANDIDATE PAIR GENERATION

Table 1: Article candidate pair features.

	feature	*description*	*group*
f_1	*tfidf*	*tfidf* similarity between abstracts	*abstract*
f_2	*d2v*	*doc2vec* similarity between abstracts	
f_3	*w2v*	*avg.* word2vec abstract vectors similarity	
f_4	$sim(\Psi_{a_i}, \Psi_{a_j})$	similarity in embedding space between Ψ_a and Ψ_a^p categories for the article pair	*Ψ & KB*
f_5	$\bigcap\limits_{a \in \langle a_i, a_j\rangle} \Psi_a$	direct and parent categories overlap	
f_6	$sim(a_i, a_j)$	embedding similarity of the article pair	
f_7	*type*	type overlap	
f_8	$sim(\psi_i, \psi_j)$	column title similarity (f_8^l) and column distance (f_8^d) between the schemas in a table pair	*tables*
f_9	$\left\|\gamma(\psi_i) - \gamma(\psi_j)\right\|$	category representation similarity γ	

In this step, we address the problem of determining article pairs, whose tables will result in a relation. Here, we are interested in two main properties: (1) minimize the amount of *irrelevant article pairs*, whose tables do not have a relation, and (2) filtering out of article pairs should not affect the coverage of relevant pairs (i.e. whose tables in a table relation.

With these properties in mind, we define features and use them in two ways: (i) filter out irrelevant article pairs, and (ii) employ the features in a supervised manner to further filter out such pairs.

4.1 Features

Table 1 shows all the features. We omit the description of features that are obvious from the table.

Article Abstract. Abstracts contain a summary of the most important information of an article. This feature ensures *topical/semantic* similarities between a pair of articles, through contextual similarities like *doc2vec* [9], word based similarity in the embedding space [11], or through *tf-idf* which captures the salience of tokens.

Categories & Knowledge Bases. Wikipedia categories are indicators that two articles are semantically similar. Categories are associated *manually* to articles, and thus are prone to noise. We circumvent this problem by computing *graph embeddings* based on the category graph Ψ, and use node2vec embedding [7] to elevate the category comparisons from the link based structure in Ψ into the embedding space. The features represent the cosine similarity in the embedding space for an article pair, specifically between the *directly associated* and *parent* categories, the articles themselves.

Tables. Article pair features capture only coarse grained similarity. Thus, we compute light-weight features between the columns of a table pair. The features correspond to the maximal similarity w.r.t the *average word embedding* of the column descriptions, and additionally we compute the *positional index difference* between the matching columns in the respective tables.

Column-Representation. For a column c_i with instance-values, we compute a representation based on the attributes associated with the instances $\{v_i\}$ in a target KB, e.g. v_i^1=*"George Lucas' **bornIn** "Modesto, California, U.S."*. Since usually there are multiple instances $|v_i| > 1$ for c_i, we find the *lowest common ancestor* ψ_L category from $\{v_i\}$ in the category graph Ψ, which can be seen as a the common *type* of instances in $\{v_i\}$. The representations allow us to compute the semantic similarity between columns.

The representation of c_i is computed as in Equation 1. We weigh the importance of *attributes* based on how discriminative they are for ψ_L, e.g. an attribute associated with articles directly belonging to category ψ_L are exclusive for ψ_L, and thus are weighted high. For an attribute p, the weight for ψ_L is computed as following:

$$\gamma(p, \psi_L) = \frac{\lambda_{\psi_L}}{\max \lambda_{\psi}} * \left(-\log \frac{|\bigcup o| : \forall \langle a, p, o \rangle \wedge a \in \psi_L}{|o| : \forall \langle a, p, o \rangle \wedge a \in \psi_L} \right) \tag{1}$$

where, the first part of the fraction weighs p by taking into account the level of λ_{ψ_L} and the deepest category where p is present in a target KB. $|\bigcup o|$ represents the number of distinct values assigned to attribute p in ψ_L, whereas $|o|$ is the total number of assignments.

Through $\gamma(\psi_L)$ we capture the most *important* and *descriptive* attributes for a column. For columns whose representations yield a high cosine similarity is an indicator of high semantic similarity.

4.2 Filtering & Classification

We use the computed features in two ways: (i) filter out irrelevant article pairs (i.e. *unlikely* to have a table relation), and (ii) train a supervised model to classify article pairs as *relevant* or *irrelevant*.

Filtering. We consider a *conjunction* of filtering criteria based on empirically evaluated thresholds for the individual features. Our main goal is to retain a high coverage of *relevant article pairs*, and at the same time filter out drastically *irrelevant pairs*. For thresholds we consider the *mean* value of a particular feature. This is an indicator that the pair is unlikely to yield any table relation based on a given feature. Section 7 shows that we are able to drastically reduce the number of pairs by simply applying such thresholds.

Classification. From the remaining pairs we train a classification model and classify pairs as either *relevant* or *irrelevant*. We consider as positive instances all table pairs from the article pair $\langle a_i, a_j \rangle$ which have *at least one* table relation, i.e, $r(t_i, t_j) \neq$ none, where $\exists (t_i \in a_i \wedge t_j \in a_j)$.

We use Random Forests (RF) [4] for classification, as they allow to set *minimal amount of samples* that are allowed for a node in the tree to be split. This directly impacts the accuracy of the classifier, however, it allows us to retain high coverage. Setting this number high makes the leafs of the different trees to be impure containing relevant and irrelevant article pairs. Our classifier will classify such impure leafs as relevant. Section 7 shows that we can maintain high coverage of relevant pairs and at the same drastically reduce the amount of irrelevant pairs.

5 TABLE ALIGNMENT

TableNet is a bidirectional recurrent neural network (RNN) with LSTM cells, which classifies a table pair $r(t_i, t_j)$ into one of the relations `equivalent`, `subPartOf`, or `none`. For a model to accurately align table, the *order* of columns in their schemas needs to be taken into account. Additionally, the matching columns in the two schemas need to fulfill two main criteria: (i) the *context* in which the columns occur needs to be semantically similar, and (ii) the *positions* of the columns needs to be comparably similar [13].

5.1 Table Representation

How we represent columns is key towards an accurate alignment model. A column in a table schema consists of the following information $c_i = \langle desc, \{v_i^1, \ldots v_1^n\} \rangle$ (see Section 3).

Column Description. The column description is a strong indicator of the cell-values v_i. We represent the description based on GloVe pre-trained word embeddings [11]. In the case of multiple tokens, we average the word embeddings. One disadvantage of this representation is that the description can be ambiguous, e.g., *"Title"* can refer to various different value types, e.g. `Movies, Books` etc.

Instance–Values. In case c_i contains *instance values*, we represent c_i through the average embeddings of the values v_i based on pre-computed graph embeddings. We use *node2vec* embeddings [7] trained on the Wikipedia *anchor graph*[1]. The combination of *description* and *instance values* improves the representation of a column and reduces its ambiguity.

Column–Type. Representing the columns based solely on instance values poses a risk of biasing the column representation towards articles that are often linked together in the Wikipedia anchor graph, and thus it may ignore the topic information that is present in such articles based on their category associations.

For columns that contain instance values, we additionally represent it through its *type* or *category*. From the instances v_i of c_i, we extract their LCA category from Ψ (a similar idea was employed by Das Sarma et al. [13]). We represent the LCA category through graph embeddings. In cases we have multiple LCA categories, then we average their corresponding representations.

[1] The anchor graph consists of nodes (Wikipedia articles and categories), while the edges correspond to the anchor text and the category-article associations.

5.2 Table Alignment Architecture

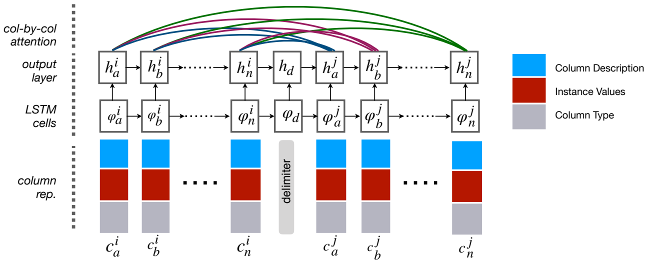


Figure 2: TableNet uses *BiLSTMs* to encode the tables as a sequence of columns. The *column-by-column* captures alignments between columns.

TableNet adopts the model from [12], and for a table pair $r(t_i, t_j)$ it predicts the relation as `equivalent`, `subPartOf`, or `none`. The alignment model is a bidirectional LSTM [8]. We compute a *column-by-column* attention, which helps us generate soft-alignments between columns in the table schemas, and thus further improve the alignment accuracy. Below we describe the encoding of table columns, and the intuition behind the attention mechanism.

Table Encoding. Since we have two separate tables, a precondition for accurate alignment is the encoding of the sequence of columns. Our model provides a *conditional encoding*, in that it first reads the columns from $C(t_i)$, then the cell state c_d (the table separator) is initialized with the last state of t_i (in this case c^i_n) and is used to conditionally encode the sequence of columns in $C(t_j)$.

The advantage of the conditional encoding is that by encoding table t_j with initial cell state that corresponds to the last column cell state of t_i, we bias the model to learn encodings that are optimal for the task of table alignment. That is instead of trying to encode all columns, it will learn to encode the columns of t_j such that it can best predict the relation for the table pair. Since we have a bidirectional LSTM, we encode in a similar fashion the table t_i by conditioning it on the last state of t_j.

Attention Mechanism. For a table pair to be aligned with either `equivalent` or `subPartOf` relation, we expect that the most important columns in each of the tables to have their corresponding matches. This is inline with the intuition that not all columns in a table are equally important [13].

The classification of the table pair $r(t_i, t_j)$ through a standard RNN model, is done by using the last cell state of the encoded table pair. This additionally forces the model to consider columns as being equally important. A common workaround is to use attention mechanism [3] or *global attention*, which is able to capture the importance of certain sequences. However, global attention is geared for language generation tasks, and as such is not suitable for the classification of $r(t_i, t_j)$.

Column-by-Column Attention. In TableNet, we employ the *column-by-column* attention mechanism, which works as following. After having encoded the last column from t_i, we process the columns in t_j individually and generate the attention weights w.r.t the columns in t_i. As a consequence, for each column in t_j we generate soft alignments to highest matching columns in t_j. After having processed all the columns in t_j and computing the corresponding attention weights (the upper part in Figure 2), for classification of the table pair $r(t_i, t_j)$ we will use a non-linear combination of the weighted representation of the last column c^j_n in t_j. We use *softmax* classification function for determining the label for $r(t_i, t_j)$.

6 EXPERIMENTAL SETUP

Here we describe the experimental setup. The entire evaluation setup and code of TableNet are available for download.[2]

6.1 Wikpedida Tables Dataset

We extract tables from the HTML content of Wikipedia articles. From the entire snapshot of Wikipedia, only 529,170 Wikipedia articles contain tables, resulting in a total of 3,238,201 tables. On average there are **6 tables per article** with an average of **6.6 columns**, and with an average of **10 rows** per table.

Additionally, more than 20% of columns in total are instance-values based (see Section 3), and additionally more than 85% of tables contain such columns. This shows that in the vast majority of cases, we can represent tables, specifically the columns with highly rich semantic representations.

6.2 Table Alignment Ground-Truth

Our ground-truth consists of a sample of 50 source Wikipedia articles from which we construct article candidate pairs. Since the *naive* approach would generate 26.5M pairs, we apply a set of *filtering keywords* to filter out irrelevant pairs. We filter articles by checking if a keyword appears *anywhere* in the article's content[3].

We manually inspect a random sample of pairs that are filtered out, and assess if we remove pairs that should be considered relevant, and consequentially refine the keywords. For article pairs that remain after filtering, we check if they can be seen as *false positives* and similarly refine our filtering keywords to remove such cases. We iteratively apply the refine and filtering steps, until we are left with an initial set of article pairs that we deploy for evaluation through crowdsourcing.

For the 50 source articles, we are left with 3.7k pairs after three rounds of filtering refinements, which result in a set of 17k table pairs that we evaluate by means of crowdsourcing.

6.2.1 Ground-Truth Statistics. From **17,047 table pairs**, after labeling our ground-truth consists of 52% table pairs marked with `noalignment` relation, 24% marked with as having `equivalent` alignment, and the remaining 23% with `subPartOf` relation. The 47% portion of table pairs with a relation, result from **876 article pairs**. The average agreement rate amongst crowdworkers is 0.91.

6.3 Baselines and TableNet setup

We compare *TableNet* in two main aspects: (i) efficiency in candidate pair generation, and (ii) table alignment.

6.3.1 ***Candidate Generation Baselines.***
Greedy – G. For each article we consider all other articles as pairs. This has maximal coverage, however, the amount of irrelevant pairs is extremely high.

[2] https://github.com/bfetahu/wiki_tables/

[3] A detailed description on the ground-truth generation is provided in the paper URL.

Direct Categories – C1. We consider as pairs articles that are associated with the same *directly connected* categories. Due to the noisy article-category associations, there is no guarantee that we will have maximal coverage of relevant pairs.

Deepest Category – C2. We consider as pairs all those articles that belong to the same *deepest associated category* in Ψ. Articles are associated with multiple categories across different levels in Ψ.

Parent Categories – PC. To increase the coverage of relevant pairs, we consider as pairs, articles that have the same *parent categories* based on the directly associated categories.

6.3.2 *Table Alignment Baselines*.

Google Fusion. The work in [13] finds related tables for a given table by computing a set relatedness scores against all possible table candidates. Two tables are related if their schemas are related based on max-weight bipartite graph matching score (see Section 2 for a detailed discussion). Google Fusion is unsupervised, thus, we use a threshold τ (we fine tune τ s.t we find the best F1 score) to classify table pairs as either having a relation or not.

BiLSTM. We train a BiLSTMs as a baseline for table alignment and use the different column representations from Section 5.

Setup: BiLSTM & TableNet. We use 100 dimensions for the *hidden layer* for the LSTM cells, and train for 50 epochs. We split the data with 60% for training, 10% for validation, and the remaining 30% for testing.

We represent columns based on the three representations (see Section 5). The simplest representation is the column description, TableNet^d, and then incrementally add the more complex representations, TableNet^{+v} and TableNet^{+t}, respectively.

6.4 Evaluation Metrics

Candidate Generation. The aim is to *minimize* the amount of *irrelevant article pairs* $\langle a_i, a_j \rangle = \emptyset$, and at the same time retain pairs whose tables have an alignment. We compute Δ to measure the amount of reduction we achieve w.r.t the *greedy* approach in generating article pairs:

$$\Delta = 1 - \frac{\langle a_i, a_j \rangle}{k * |A|} \text{ where } a_i \neq a_j \wedge a_i, a_j \in A \tag{2}$$

where, k is the number of source articles.

Coverage we measure through *micro* and *macro* recall indicated with R_μ and R, respectively. R_μ represents the recall in terms of all table pairs from all the source articles, whereas macro recall measures the average recall scores from all source articles.

Table Alignment. We measure the accuracy of the models for determining the table relation based on standard evaluation metrics, such as *precision* (P), *recall* (R), and *F1 score* (F1).

7 EVALUATION RESULTS

7.1 Candidate Generation Results

Table 2 shows the efficiency and coverage results for the baseline strategies. From the baselines we notice that the use of the Wikipedia category graph Ψ reduces the amount of irrelevant pairs drastically. In terms of recall, baseline **PC** maintains high recall with $R = 0.83$, and at the same time reduces by $\Delta = 87\%$ the amount of irrelevant pairs when compared to *greedy* approach. **C2** achieve the highest reduction rate Δ. Despite, the high reduction rates, we still face the issue of either having a highly imbalanced ratio of relevant and irrelevant pairs, or in some cases like **C2** where the reduction rate is the highest, the recall is low $R = 0.49$.

Table 2: Reduction rate for the baseline strategies.

	$\lvert\langle a_i, a_j \rangle\rvert$	Δ	*rel. pairs*	R
G	26,500,000	-	876	1.0
PC	3,486,031	0.87	724	0.83
C1	792,701	0.97	571	0.65
C2	6,738	0.99	440	0.50

***TableNet: Filtering*.** The filtering step uses the features in Table 1 to remove irrelevant article pairs. Figure 3 shows the impact of the different features in reducing the amount of article pairs w.r.t *greedy* approach. For instance, the f_2 feature, which computes the similarity of article abstracts based on their *doc2vec* representation, provides a high reduction with $\Delta = 0.91$. This feature follows our intuition on generating the article pairs for the ground-truth (see Section 6), where the *topic* and other semantic similarities for an article pair can be extracted from the article's content.

Since different features capture different notions of similarity, we apply them in *conjunction*, resulting in very high reduction rate of article pairs with $\Delta > 0.99$, and at the same time retaining a relatively high coverage with $R = 0.68$. The reduction compared to the greedy approach is more than ▼255 times less pairs.

We believe that this high reduction rate and at the same time the relatively high recall of relevant pairs, when compared to the baseline approaches can be attributed to the fact that we consider the similarities of articles, and their corresponding categories and articles' content in the embedding space. This allows us to capture implicit semantics that cannot be captured through the simple link-based structure in the category graph Ψ.

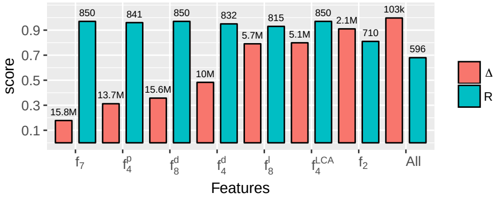


Figure 3: Feature impact in terms of reducing the amount of irrelevant pairs and the coverage of $\langle a_i, a_j \rangle_r$ pairs.

***TableNet: Classification*.** Determining which article pairs, specifically, their tables will have a table relation is a difficult.

Despite the high reduction of irrelevant pairs from 26M pairs to 103k pairs, the amount of irrelevant pairs is still too high for any supervised approach to be able to learn models that predict with great accuracy the table relations. Thus, based on the configured RF model for high coverage (see Section 6), we train it on the feature set in Table 1 to further classify irrelevant pairs and filter them out.

Figure 4 shows the evaluation results for varying thresholds τ for the RF model. The increase of τ is directly proportional with the decrease in the amount of relevant pairs. This is intuitive as from the 103k pairs, only 876 pairs are actually relevant. Based on the configuration of the RF (see Section 6), we retain relevant pairs with high coverage. We choose $\tau = 0.5$, as it shows the best trade-off between the coverage of relevant pairs, and the amount of irrelevant pairs that are passed onto the table alignment step. We achieve a high reduction rate of $\Delta = 0.982$ leaving us with only 1.8k pairs, and with a recall of $R_\mu = 0.81$.

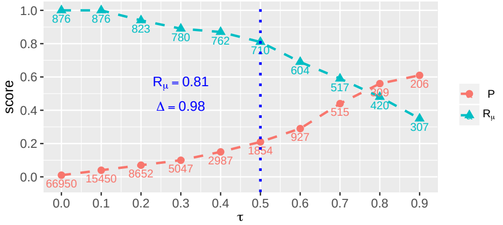


Figure 4: τ (x-axis) shows the scores for R_μ and P, and the corresponding amount of relevant pairs we retain.

Dependent on the scenario, higher τ can be used to have a higher ratio of relevant pairs, making the alignment task more efficient.

7.2 Table Alignment Results

Performance. Table 3 shows the table alignment evaluation results. In the case of *Google Fusion*, we consider a table pair to be *aligned* if their matching score is above some threshold that we determine empirically for which we achieve the highest F1 score.

Google Fusion. This baseline has a reasonably high accuracy in determining whether a $r(t_i, t_j) \neq \emptyset$. Here we cannot distinguish between the different classes as the approach is unsupervised. In terms of recall it has the lowest score. This is due to the fact that the matching is performed by considering only the *column type* and the *column title*similarity. Additionally, the *bipartite* matching algorithm cannot retain the order of the columns, which is highly important for determining the alignment relation.

BiLSTM. One key motivation in this work is the hypothesis that through sequence based models, we can retain the order of columns in their respective schemas, an important aspect in determining the table alignment relation. The BiLSTM approach represent a very competitive baseline. An additional advantage which addresses a deficiency in the standard supervised models, is that we jointly encode the different representations of a column for the classification task. Representing the columns as a combination of their description through word embeddings, and the type and instance values through graph embeddings, we can capture complex relationship between the column description and the underlying cell-values.

For `equivalent` relations, BiLSTM^{+d} achieves a very high F1 score with $F1 = 0.887$, which is close to the best result we achieve with TableNet^{+v}. For `subPartOf` relations, BiLSTM^{+v} the representation based on the column description and instance values achieves the highest F1 scores. The introduction of the column type in BiLSTM^{+t} provides a further boost in the accuracy of determining `subPartOf` relations. One conclusion we draw from the comparison between the two relations and two models, is that for `subPartOf` relations the *column type* provides additional power on determining the table alignment relation, whereas for `equivalent` it does not provide an additional advantage. These findings are inline with [13], where column type can provide important information in finding related tables. Comparing BiLSTM against *Google Fusion* we have a 64% improvement for `equivalent` relation.

TableNet. In our approach, we addressed several deficiencies from the related work. Through our *column-by-column* attention mechanism, we can compute soft alignments between columns in the respective table schemas and thus take into account the position of the matching columns in the corresponding schemas. Additionally, the column representations allow us to capture the similarity between columns and the schema context in which they appear, and additionally the representation context based on their description, type and its instance values.

The evaluation results reflect this intuition. Comparing our best performing setup, TableNet^{+t} achieves an overall $F1 = 0.840$ across all three classes, which presents also the highest F1 score across all competitors. We achieve a relative improvement of 64% when comparing F1 scores for the `equivalent` class against *Google Fusion*, or 56% if we compare the average F1 score for both alignment relations (`equivalent` and `subPartOf`). Furthermore, TableNet^{+t} outperforms the BiLSTM approach on average F1 score across all classes. For the individual classes, we note a variations amongst the different configurations of TableNet that perform best (marked in bold). This confirms the usefulness of the attention mechanism for the alignment task, where we achieve an overall better performance in terms of F1 score.

8 CONCLUSIONS AND FUTURE WORK

We presented TableNet, an approach for table alignment that ensures *high coverage* of retained table relations, and provides highly accurate *fine-grained* table alignments.

We constructed an exhaustive ground truth for a random sample of 50 Wikipedia articles for which we labeled all possible table pairs, providing a dataset against which we can measure the coverage and accuracy of approaches in determining table relations for more than 17k table pairs in our ground-truth.

In terms of *efficiency*, we show that from a naive approach which produces 26.5M pairs, we can guarantee a high coverage of retaining relevant article pairs with more than 68%, while, at the same time reducing the amount of irrelevant pairs by ▼255 times. In terms of *table alignment*, we show that we can improve over strong baselines and provide high improvement over strong baselines and other competitors like *Google Fusion*.

ACKNOWLEDGMENT:

This work is funded by the ERC Advanced Grant ALEXANDRIA (grant no. 339233), and BMBF Simple-ML project (grant no. 01IS18054A).

REFERENCES

[1] Abdalghani Abujabal, Rishiraj Saha Roy, Mohamed Yahya, and Gerhard Weikum. 2018. Never-Ending Learning for Open-Domain Question Answering over Knowledge Bases. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018.* 1053–1062. https://doi.org/10.1145/3178876.3186004

[2] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary G. Ives. 2007. DBpedia: A Nucleus for a Web of Open Data. In *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007.* 722–735. https://doi.org/10.1007/978-3-540-76298-0_52

Table 3: Evaluation results for the tasks of table alignment for the different competitors and TableNet. The evaluation results correspond to our manually constructed ground-truth dataset.

	`equivalent`			`subPartOf`			`noalignment`					
	P	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**Acc**	**R**	**F1**
Google Fusion	0.809	0.405	0.540	-	-	-	-	-	-			
$BiLSTM^{d}$	0.883	0.891	0.887	0.684	0.960	0.799	0.918	0.752	0.827	0.828	0.868	0.838
$BiLSTM^{+v}$	0.877	0.871	0.874	0.684	0.975	0.804	0.915	0.747	0.823	0.826	0.864	0.834
$BiLSTM^{+t}$	0.854	0.908	0.880	0.690	0.957	0.802	0.925	0.741	0.823	0.823	0.869	0.835
$TableNet^{d}$	**0.888**	0.884	0.886	0.686	0.947	0.796	0.909	0.759	0.827	0.828	0.863	0.836
$TableNet^{+v}$	0.856	**0.926**	**0.890**	0.675	**0.993**	0.804	**0.952**	0.719	0.819	0.828	**0.880**	0.838
$TableNet^{+t}$	0.872	0.903	0.887	0.692	0.961	**0.805**	0.925	0.752	**0.829**	**0.830**	0.872	**0.840**

[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).

[4] Leo Breiman. 2001. Random forests. *Machine learning* 45, 1 (2001), 5–32.

[5] Michael J. Cafarella, Alon Y. Halevy, Daisy Zhe Wang, Eugene Wu, and Yang Zhang. 2008. WebTables: exploring the power of tables on the web. *PVLDB* 1, 1 (2008), 538–549. http://www.vldb.org/pvldb/1/1453916.pdf

[6] Hector Gonzalez, Alon Y. Halevy, Christian S. Jensen, Anno Langen, Jayant Madhavan, Rebecca Shapley, Warren Shen, and Jonathan Goldberg-Kidon. 2010. Google fusion tables: web-centered data management and collaboration. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2010, Indianapolis, Indiana, USA, June 6-10, 2010*. 1061–1066. https://doi.org/10.1145/1807167.1807286

[7] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable Feature Learning for Networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*. 855–864. https://doi.org/10.1145/2939672.2939754

[8] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[9] Quoc V. Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*. 1188–1196. http://jmlr.org/proceedings/papers/v32/le14.html

[10] Emir Muñoz, Aidan Hogan, and Alessandra Mileo. 2014. Using linked data to mine RDF from wikipedia's tables. In *Seventh ACM International Conference on Web Search and Data Mining, WSDM 2014, New York, NY, USA, February 24-28, 2014*. 533–542. https://doi.org/10.1145/2556195.2556266

[11] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*. 1532–1543. http://aclweb.org/anthology/D/D14/D14-1162.pdf

[12] Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomas Kocisky, and Phil Blunsom. 2016. Reasoning about Entailment with Neural Attention. In *International Conference on Learning Representations (ICLR)*.

[13] Anish Das Sarma, Lujun Fang, Nitin Gupta, Alon Y. Halevy, Hongrae Lee, Fei Wu, Reynold Xin, and Cong Yu. 2012. Finding related tables. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2012, Scottsdale, AZ, USA, May 20-24, 2012*. 817–828. https://doi.org/10.1145/2213836.2213962

[14] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of the 16th International Conference on World Wide Web, WWW 2007, Banff, Alberta, Canada, May 8-12, 2007*. 697–706. https://doi.org/10.1145/1242572.1242667

[15] Petros Venetis, Alon Y. Halevy, Jayant Madhavan, Marius Pasca, Warren Shen, Fei Wu, Gengxin Miao, and Chung Wu. 2011. Recovering Semantics of Tables on the Web. *PVLDB* 4, 9 (2011), 528–538. http://www.vldb.org/pvldb/vol4/p528-venetis.pdf