Table Union Search with Preferences

Hamed Mirzaei¹,∗,†, Davood Rafiei¹,‡

¹University of Alberta, 2-32 Athabasca Hall, Edmonton, Alberta, Canada

Abstract
We study the problem of Table Union Search (TUS) in the presence of preferences. The notion of unionability, as studied in the literature, is too coarse to be effective in down-stream tasks. We introduce preferences for table unionability, as a way to reduce the search space and focus on rows and columns that are important for the follow-up operations. We show how these preferences can be efficiently implemented and how they can improve the performance of some down-stream tasks.

Keywords
Table Union Search, Preference Queries, Information Retrieval, Table Augmentation, Table Unionability, Column Unionability

1. Introduction
The vast corpus of relational tables on the web is a valuable resource for various applications such as table augmentation, knowledge base population, and question answering. Table Union Search (TUS) is an operation that aims to find relational tables on the web, a.k.a. webtables, that can be unioned with a query table. Two tables are considered unionable if their column values are drawn from the same domains. However, in the context of the web, the domains of columns are not known or fixed and existing approaches often rely on measures such as value overlap to find columns with the same domains, a.k.a unionable columns. But using this generic notion of unionability in downstream tasks can be challenging.

Consider a query table \( Q \) as shown in Table 1, and suppose we want to expand \( Q \) vertically or horizontally by adding more rows or columns. TUS returns the same list of webtables regardless of the follow-up operations. Also TUS fails to identify and leverage complex relationships between tables and columns, often returning near-duplicate webtables.

To address these issues, we introduce preferences to TUS, allowing for more efficient and effective retrieval of unionable webtables based on user-defined criteria. Preferences help in tailoring the results to different follow-up operations and mitigating the limitations of TUS. By incorporating these preferences, we aim to improve the usefulness and relevance of the TUS output for various applications. The contributions of this research are as follows:

• We introduce four major preferences to the TUS operation: skyline, novelty, diversity and dependent set.
• We introduce two benchmark datasets for unionable webtables, constructed from Web Data Commons 2015 [1] and WikiTables datasets [2].
• We propose efficient approaches for evaluating each preference.

2. Related Works
Tabular data may be searched using table-based queries [3, 4, 5, 6] and keyword-based approaches [7, 8, 9, 10]. Table-based approaches assume that the search query is a table while keyword-based assume it is a set of keywords. Our work falls under table-based approaches focusing on enriching the results of TUS.

Preferences have long been studied in databases (e.g., see the general survey [11], the foundational work [12] and preferences in SQL [13]) but we are not aware of them being applied to TUS. Khatiwada et al. [14] investigate the semantics and relationships of columns to identify unionable webtables more accurately. This may seem similar to our work on dependent set preference (to be discussed next), but there are some subtle differences.

3. Table Union Search
In this work, we define column unionability in terms of value overlap, and other metrics such as semantic
We may find alignments of different sizes. We define a (Definition 3.1) as a bijective mapping between all columns of \( C \) and \( Q \) and is shown as \( a(C, Q) \).

We define the unionability score of an alignment \( a(C, Q) \) (AUScore) as the product of CU Score of all unionability pairs in \( a(C, Q) \).

\[
AUScore(a(C, Q)) = \prod_{(C, Q) \in a(C, Q)} Jaccard(V_c, V_q). \quad (1)
\]

Two tables \( C \) and \( Q \) can be of different degrees. If we apply projections of the same size over both tables, we may find alignments of different sizes. We define the set of all these possible alignments that result from projections over \( C \) and \( Q \) as the Alignment Set of two tables (Definition 3.3).

An alignment set of two tables \( C(c_1, c_2, ..., c_n) \) and \( Q(q_1, q_2, ..., q_n) \) is the set of alignments between all possible projections of the same degree over \( C \) and \( Q \) and is shown as \( A(C, Q) \).

For an integer \( c \) and a subset \( S \) of query columns, \( A^c_S(C, Q) \) denotes the subset of alignments of size \( c \) in \( A(C, Q) \) with only mappings over query columns in \( S \).

Alignments of different sizes may be compared based on their probability distributions, as estimated using a sample [6], and an Alignment Goodness Score (AGScore) may be defined (Definition 3.4).

The goodness score of alignment \( a(C, Q) \) is its updated AUScore using the corresponding probability distribution function.

\[
AGScore(a(C, Q)) = CDF_{a(C, Q)}(AUScore(a(C, Q))) \quad (2)
\]

where \( CDF_{a(C, Q)} \) is CDF for the probability distribution corresponding to the size of alignment \( a(C, Q) \).

Finally, we define Table Unionability Score (TUScore) of two tables \( C \) and \( Q \) as the maximum AGScore of alignments in \( A(C, Q) \) (Definition 3.5).

### 4. Preferences

**4.1. Skyline**

Skyline preference has been extensively studied in the literature [16, 17]. It is commonly defined over a corpus of data points, each represented as a numeric vector, where one looks for those data points which are not dominated by others. A data point \( D_1 \) represented by vector \( V_1 \) dominates a data point \( D_2 \) represented by vector \( V_2 \) and shown as \( D_1 > D_2 \) if it has as good or better values over all dimensions and a better value over at least one dimension [17]. Our approach is to represent each alignment \( a(C, Q) \) as a vector \( V_a \) of size equal to the number of query columns with each element showing the ununionability score of a pair of columns. Following this idea, each candidate webtable with multiple alignments can be represented as a set of numeric vectors. An observation is that for one candidate webtable, some of the alignments are a subset of the others which we prune them early in the process. Finally, by utilizing existing algorithms on
skyline such as SaLSa [18] or BBS [19], we can return the candidate webtables with the best alignment over each subset of query columns (Definition 4.1).

**Definition 4.1 (Top-k Skyline).** With each alignment between a candidate webtable C and a query table Q represented as a numeric vector (as discussed), the Top-k Skyline of Q is the top-k candidate webtables with the most number of skyline vectors.

### 4.2. Diversity

Diversity is a well-studied preference in the literature with the purpose of resolving ambiguity by returning diverse results [20, 21, 22]. The proposed approaches usually involve a scoring function [22] that balances the diversity of the results while promoting their relevance to the search query.

**Definition 4.2 (Top-k Diversity).** The Top-k Diversity of query table Q over subset S of query columns is the list R of webtables from dataset 𝒟 which maximizes the following objective function and are sorted based on their TUScore.

\[
\text{Diversity}(Q, S, k) = \arg\max_{R \subseteq \mathcal{D}, |R| = k} (k - 1)(1 - \lambda) \cdot \text{Rel}(Q, S, R) + \lambda \cdot \text{Dif}(Q, R),
\]

where the parameters \( \lambda \) and \( k - 1 \) in this formulation control the scores’ contribution and to bring them to the same scale respectively,

\[
\text{Rel}(Q, S, R) = \sum_{i=1}^{|R|} \text{TUScore}(S_i, Q),
\]

and \( \text{Dif}(Q, R) \) is defined as the harmonic mean of the differences of tables in R from each other and from the query table.

As finding such a list R is a computationally hard task [22], we utilize a greedy approach based on Yu et al.’s Swap algorithm [23] which starts with a list of k webtables and iterates over the rest of webtables to see if swapping outsiders with insiders improves the diversity score. We modified the Swap algorithm to choose the initial list more carefully and iterate over webtables in a specific order.

### 4.3. Novelty

Novelty is a well-known preference which has been studied in the literature with the purpose of avoiding redundancy in the returned result [24, 25, 26, 22]. In the context of our work, we want to avoid redundant webtables and return those offering as much unique new values as possible over a subset of query columns. It can be considered as a special case of diversity preference which aims at finding webtables with the most diverse values over a subset of query columns. Hence, we can use the same framework discussed for the diversity preference with some modifications in the proposed scores. The objective function for novelty consists of two scores, \( \text{Rel} \) and \( \text{New} \). The first score, \( \text{Rel} \), is the same as the one we defined for diversity. But, the \( \text{New} \) score computes the amount of new information that the selected candidate webtables bring to the query table as a whole. We define top-k novelty over this framework as the list \( R \) of k candidate webtables which maximizes this objective function.

Similar to diversity, finding the exact list \( R \) which maximizes the objective function is computationally hard [22], hence greedy approaches have been in use. Again, we followed the Swap algorithm [23] after adapting it to our context for novelty.

### 4.4. Dependent Set

A problem with the notions of unionability used in TUS is that each query column is treated as an independent entity. Consequently, important information about the combination of values across query columns will be lost and two candidate webtables with the same set of values over their columns will be considered duplicate even though they offer different combination of them. In order to resolve this issue, we propose dependent set preference, which differentiates between candidate webtables that have the same values but different combinations. Dependent set introduces sets of dependent query columns.

Users can put query columns in different dependent sets to demonstrate the importance of their values together. Note that some query columns may not be part of any dependent set or treated as a dependent set of size one. With this preference, each dependent set \( S_i \) of query columns is considered as a new query column with values generated by concatenating the values of columns in \( S_i \). Following this idea, we define top-k dependent set as the top k candidate webtables with the most table unionability score in the set of newly created query columns (Definition 4.3).

**Definition 4.3 (Top-k Dependent Set).** Having sets \( S_1, ..., S_m \) of dependent query columns, the Top-k Dependent Set is defined as top-k candidate webtables of applying top-k TUS over the new set of query columns \( \{q'_1, ..., q'_m\} \cup \{q_i | q_i \notin S_j \text{ for } j = 1, ..., m\} \) where \( q'_i \)’s values are the concatenation of values of query columns in \( S_i \).

To efficiently find the top-k candidates, we first find webtables with at least one alignment over a dependent set of query columns using the WAND approach over query columns’ posting lists. Then, for each alignment of the filtered webtables, if it covers dependent set \( S_i \), we concatenate candidate columns in the same order we did
for \( q'_i \) to get to a new candidate column \( c'_i \). The last step is to compute the table unionability score over the new set of candidate and query columns and to return the top-k ones to the user.

5. Evaluations

To assess the effectiveness of the proposed preferences, we evaluate their performance in two down-stream tasks: i) expanding query table by adding new rows and ii) extending it by adding new columns. As a measure of performance, we utilize Average F1-Measure@k, calculated by averaging the f1-measure@k scores over query tables.

5.1. Datasets

Existing datasets [6] only provide a limited selection of unionable candidates, and they fail to adequately showcase the impact of preferences. We introduce two datasets for our evaluation: (1) Web Data Commons 2015 (WDC) dataset [1], and (2) WikiTables dataset [2]. We built our dataset of query and candidate webtables from these base tables by randomly selecting rows and random projection of columns. Applying these operations on a base table is expected to generate new tables that are unionable with the base table by the definition of unionability [27].

WDC 2015. We selected 50 webtable with at least 9 columns and 900 records from the WDC 2015 [1] dataset. We checked them manually and pruned those with less than 5 text columns or with non-English content and ended up with 14 webtables as our base tables. We sampled 101 tables from each base table, as discussed above, and designated one table as our query table and the remaining 100 tables as candidate webtables. This gave us a total of 14 query tables and 1400 candidate webtables.

WikiTables. We did a similar process as the one for the WDC dataset. We first selected top 100 webtables with at least 9 columns and 300 rows from the WikiTables dataset [2]. Then, we pruned those with non-English content. In addition, since we wanted to guarantee different levels of unionability among query and candidate columns, we pruned those tables with many columns that had only ‘yes/no’ values in their content. Finally we ended up with 12 base tables which we used to generate query and candidate webtables from. We ended up with 12 query tables, one per each base table, and 1200 candidate webtables, 100 candidates per each base table.

5.2. Task 1: Query Table Expansion

TUS struggles in expanding the query table with additional rows. Figure 1 shows that novelty and diversity outperform other approaches such as TUS and TUS+.

Novelty returns webtables with more new rows for the query table. For \( k > 5 \), dependent sets also outperform TUS and TUS+ in terms of performance. Interestingly, some of the webtables have both new rows and new columns for the query table, which explains why diversity and dependent sets excel in this task.

5.3. Task 2: Query Table Extension

TUS may include webtables unsuitable for extending the query table with new columns. When all columns of a webtable is unionable with at least one query column, no new columns may be added to the query table. Figure 2 shows that diversity and novelty outperform TUS and TUS+. Diversity focuses on returning webtables with more new columns. Dependent sets also outperform TUS and TUS+ for \( k > 5 \).

6. Conclusions

Existing approaches for finding unionable tables primarily focus on efficiently providing an approximate ranked list of candidate tables to the user, which may limit access to suitable tables for follow-up operations. In this paper, we explore the power of preferences, showing that users can retrieve tables that are not only unionable with the query table but are also suitable for various follow-up operations. A possible future direction is exploring more efficient approaches to support preferences with TUS.
References


