NEXUS: On Explaining Confounding Bias

Brit Youngmann
brity@mit.edu
CSAIL, MIT

Michael Cafarella
michjc@csail.mit.edu
CSAIL, MIT

Yuval Moskovitch
yuvalmos@bgu.ac.il
Ben Gurion University of the Negev

Babak Salimi
bsalimi@ucsd.edu
University of California, San Diego

ABSTRACT

When analyzing large datasets, analysts are often interested in the explanations for unexpected results produced by their queries. In this work, we focus on aggregate SQL queries that expose correlations in the data. A major challenge that hinders the interpretation of such queries is *confounding bias*, which can lead to an unexpected association between variables. For example, a SQL query computes the average Covid-19 death rate in each country, may expose a puzzling correlation between the country and the death rate. In this work, we demonstrate NEXUS, a system that generates explanations in terms of a set of *potential confounding variables* that explain the unexpected correlation observed in a query. NEXUS mines candidate confounding variables from external sources since, in many real-life scenarios, the explanations are not solely contained in the input data. For instance, NEXUS might extract data about factors explaining the association between countries and the Covid-19 death rate, such as information about countries' economies and health outcomes. We will demonstrate the utility of NEXUS for investigating unexpected query results by interacting with the SIGMOD'23 participants, who will act as data analysts. Our code, datasets, technical report, and a short video of NEXUS are available at [2].

ACM Reference Format:
Brit Youngmann, Michael Cafarella, Yuval Moskovitch, and Babak Salimi. 2023. NEXUS: On Explaining Confounding Bias. In *Proceedings of the 2023 International Conference on Management of Data (SIGMOD '23), June 18–23, 2023, Seattle, WA, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/XXXXXX.XXXXXX

1 INTRODUCTION

When analyzing large datasets, analysts often query their data to extract insights. Oftentimes, there is a need to elaborate upon the queries' answers with additional information to assist analysts in understanding unexpected results, especially for aggregate queries, which are harder to interpret [12]. Aggregate SQL queries aggregate an *outcome attribute* (O) for some groups of interest indicated by a grouping attribute, referred to as the *exposure* (T).

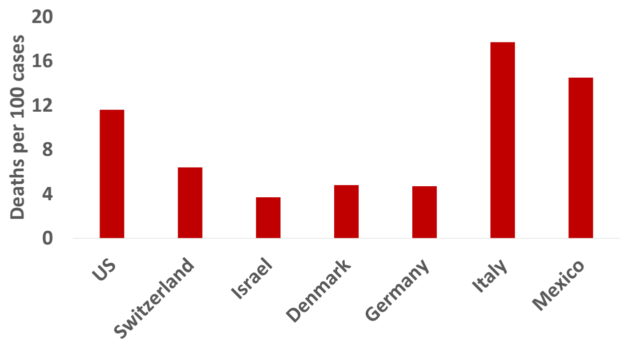


Figure 1: Visualization of the results of the query Q.

For example, consider Ann, an analyst in the WHO organization who aims to understand the coronavirus pandemic for improved policymaking. She examines a dataset containing information describing facts on Covid-19 spread in multiple cities worldwide, such as the number of deaths-/recovered-/active-/new-cases in each city. She evaluates the following query:

```
SELECT Country, avg(Deaths_per_100_cases)
FROM Covid-Data
GROUP BY Country
```

Here, the exposure is COUNTRY and the outcome is DEATHS_PER_100_CASES. A visualization of the query results is given in Figure 1. Ann observes a puzzling correlation between the exposure and outcome.

While aggregate SQL queries expose correlations in the data, the human mind tends to interpret them as causal relationships. Namely, Ann wonders why the choice of country has such a substantial effect on the death rate.

A major challenge that hinders the interpretation of aggregate SQL queries is *confounding bias* [9] that can lead to a spurious association between T and O and hence perplexing conclusions. Confounding bias occurs when analysts try to determine the effect of an exposure on an outcome but unintentionally measure the effect of another factor(s) (i.e., a *confounding variable(s)*) on the outcome. We generate explanations in terms of a set of potential confounding variables that explain unexpected correlations observed in query results. Namely, a set of attributes that can explain to Ann the relationship between COUNTRY and DEATHS_PER_100_CASES.

Our key observation is that, in many real-life scenarios, *critical confounding attributes might be found outside the narrow query results that the analyst observes and the database being used.* In our example, information about critical factors that affect the Covid-19 death rate, such as countries' health outcomes and economies, is missing from the dataset Ann examines.

To this end, we developed NEXUS (for oN EXplaining confoUnding biaS), a system that provides explanations to unexpected correlations observed in queries in terms of a subset of potential confounding attributes that explain the correlation. NEXUS searches for candidate confounding attributes in large external data sources, not only in the analyst's original dataset.

In our example, the explanation generated by NEXUS to Ann is the set of attributes CONFIRMED_CASES (an attribute from the input dataset), HDI[1], and GDP (two attributes that were extracted from external sources). Namely, Ann understands that these attributes are potential confounding attributes that affect the Covid-19 death rate and that the death rate is similar in countries with a similar economy, health outcomes, and number of confirmed cases.

Our framework emphasizes identifying correlations without relying on any background knowledge. However, it is important for the analyst to critically assess the proposed confounding variables and conduct a thorough causal analysis, which may require additional background knowledge, to establish causality.

NEXUS provides an interactive UI that allows analysts to examine the generated explanations (see Figure 3). For each selected attribute, analysts may inspect its source and its individual contribution to the explanation. We provide an easy-to-use query builder, which allows analysts to query their data and view the results. Further, NEXUS enables analysts to automatically identify unexplained data subgroups in which, for them, generated explanations might be insufficient.

NEXUS can extract candidate confounders from any knowledge source (e.g., data lakes, knowledge graphs) as long as it can be (even partially) integrated with the input data. For example, NEXUS might extract candidate attributes from a knowledge graph if some values from the input dataset can be linked to their unique entities in the knowledge graph. The knowledge source might be provided by the analyst. We can potentially extract hundreds of attributes, many of which might be irrelevant for explaining the correlation between T and O and contain many missing values (especially attributes extracted from knowledge graphs where data is sparse). Thus, our system needs an efficient algorithm to search for an explanation (i.e., an attribute set) in this extensive search space and ensure the generated explanation is robust to missing data. We, therefore, developed an efficient algorithm that *avoids iterating over all attribute subsets*, and importantly, *avoids estimating joint probabilities in high dimensions*, which is computationally difficult [10]. It also employs a principled way of handling missing values, ensuring the generated explanations are robust to missing data. This work presents NEXUS usability and its suitability for end-to-end deployment.

Related Work. Previous work provides explanations for trends and anomalies in query results in terms of predicates on attributes that are shared by one (group of) tuple in the results but not by another (group of) tuple [3, 8, 11]. However, those methods do not account for correlations among the exposure and outcome attributes, and are thus inapplicable for explaining the correlation between the outcome O and the exposure T. For example, using such methods, Ann would not be able to discern confounding attributes for the Covid-19 death rate because it requires attribute level (rather than a pattern level) analysis. HypDB [12] aims to identify the direct causes of T to eliminate confounding bias. In this sense, it identifies the most relevant attributes to T and ignores O. This process has several limitations: (1) the causes of T can only be discovered from data under very strong assumptions. It is generally accepted that causal discovery should rely on background knowledge and cannot be fully automated [6]. (2) it only works if all causes of T are observed in the data; (3) the proposed algorithm is computationally prohibitive. In contrast, NEXUS does not claim to discover causal relationships but rather aims to discover potential confounding attributes that can explain the observed correlation, while simultaneously considering both T and O. This could facilitate the process of identifying confounding variables for a more thorough causal analysis. Moreover, our proposed framework is computationally tractable. In our example, since the attributes HDI and GDP are missing from the input data, HypDB would fail to discover them. In case it was given with all extracted attributes, HypDB could not operate on a dataset containing hundreds of attributes as in our setting.

A recent work [5] suggested to augmented an input dataset with openly available knowledge sources to contextualize the data. However, this system seeks for generally relevant attributes to be added to the data rather than to identify confounders that are relevant for specific queries. Nevertheless, this tool can be used as a complementary effort to extract attributes from external sources.

2 TECHNICAL BACKGROUND

We provide an overview of our theoretical foundations of the development of NEXUS. Full details can be found in [2].

2.1 Data Model

The NEXUS system that we demonstrate manages a standard multi-relational dataset $\mathcal{D}$. To simplify the exposition, we assume $\mathcal{D}$ consists of a single relational table. Our framework supports a rich class of SQL queries that involve groping, joins, and different aggregations. The queries we examine compare among subgroups, investigating the association between an aggregated attribute O (the *outcome*) and a grouping attribute T (the *exposure*). We call the condition C (given by the WHERE clause) the context for the query.

NEXUS can extract attributes from any external source, such as data lakes, or Knowledge Graphs (KGs), as long as it can be integrated with the input dataset. In NEXUS the user may decide which external data source NEXUS should use. Given a knowledge source (e.g., domain-specific KG [13], publicly available KG [1]), we extract a set of attributes $\mathcal{E}$ representing additional properties of entities from $\mathcal{D}$. Continuing with our example, $\mathcal{E}$ could be a set of properties of countries, such as their area size, density, and HDI.

2.2 Problem Formulation

Let $\mathcal{A}$ denote the union of extracted attributes and attributes from the input dataset. We aim to find an attribute set $\boldsymbol{E}\subseteq\mathcal{A}$ that controls the correlation between O and T, i.e. when conditioning on $\boldsymbol{E}$, the correlation between O and T is diminished. We call such a set the correlation explanation. Ideally, we look for a minimal-size set of attributes $\boldsymbol{E}$ s.t: $(O \perp\!\!\!\perp T | \boldsymbol{E}, C)$. However, in practice, we may not find such perfect explanations (that entirely explain the correlation).

[1]HDI is a statistic composite index of life expectancy, health outcome, and per capita income indicators.

Hence we search for a set of attributes that *minimizes the partial correlation between T and O*.

In our demonstration, the audience will be able to choose the dataset to investigate (out of four options of commonly used datasets) and specify the aggregate SQL query. NEXUS will use Conditional Mutual Information (CMI), a common measure of partial correlation, to measure the dependence between T and O.

To assist analysts in interpreting the results, we enable them to learn the individual *responsibility* of selected attributes. The responsibility of an attribute is the normalized value of its individual contribution (see Figure 3b).

2.3 Algorithms

Extracting the Attributes: Given a knowledge source, we use existing tools to extract additional attributes that could be joined with $\mathcal{D}$. For example, we may extract attributes from a data lake, leveraging existing methods to join an input table with other tables [4, 14]. For a KG, we may use an off-the-shelf Named Entity Disambiguation algorithm (e.g., [16]) to map non-numerical values that appear in $\mathcal{D}$ to their corresponding unique entities in the KG.

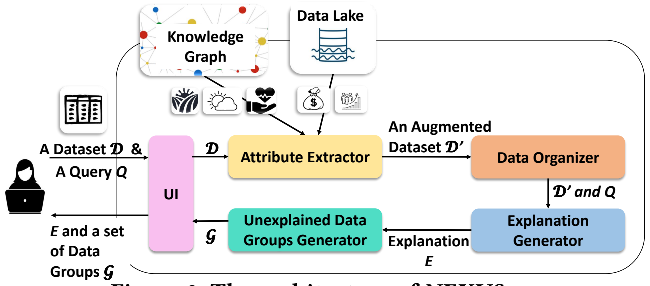
Extracted properties may be associated with multiple values. Because correlation is only defined for sets of paired values, downstream applications typically aggregate the values into a single number [14]. NEXUS supports any user-defined aggregation function (e.g., mean, sum, first). In our demonstration of NEXUS, the audience will be able to choose the aggregation function to be used out of four options.

Handling Missing Values: Attributes extracted from data often have a high number of missing values. To address this issue, imputation is a commonly used approach. However, as reported in [15], imputing data can introduce bias and negatively affect the accuracy of explanations. To mitigate this bias in the context of generating explanations, we developed a method that utilizes Inverse Probability Weighting (IPW). IPW is a widely recognized method for handling missing data, as described in [15].

To ensure the generated explanations are robust to missing data, we define sufficient conditions to assure that probabilities used to compute partial correlation are *recoverable* from data. For cases where the probabilities are not recoverable, we employ a tailored IPW-based solution. In our demonstration, participants will be invited to examine extracted attributes suffering from selection bias

Generating the Explanation: There are potentially hundreds of attributes that could be extracted from external sources. Thus, NEXUS needs an efficient algorithm to search for an attribute set (i.e., explanation) in this extensive search space. The key advantages of our proposed algorithm are that it *avoids iterating over all possible attribute sets*, and it *avoids estimating CMI for high-dimensional conditioning sets*, which is computationally difficult [10].

In a nutshell, our algorithm incrementally selects attributes based on their individual contribution (measured by CMI with T and O), and their redundancy w.r.t. previously selected attributes (measured by their mutual information with previously selected attributes). We then define a stopping criterion, allowing the algorithm to stop when no further improvement is found. Last, we propose multiple pruning optimizations to speed up the computation.


Knowledge Graph
Data Lake
A Dataset $\mathcal{D}$ & A Query Q
UI
$\mathcal{D}$
Attribute Extractor
An Augmented Dataset $\mathcal{D}'$
Data Organizer
$\mathcal{D}'$ and Q
Explanation Generator
Explanation E
Unexplained Data Groups Generator
$\mathcal{G}$
E and a set of Data Groups $\mathcal{G}$
Figure 2: The architecture of NEXUS.

Identifying Unexplained Data Groups: NEXUS finds the explanations for the correlation between T and O. While the generated explanations are optimal considering the whole data, they may be insufficient for some data groups (defined by a set of attribute-value assignments and correspond to refinements of the query Q). We thus propose an algorithm the analyst may use after getting the explanation to identify unexplained data subgroups. This algorithm outputs the top-k (in terms of size) largest data groups for which the generated explanation might be insufficient. Our algorithm exploits the notion of *pattern graph*. Intuitively, the set of all query refinements can be represented as a graph. We traverse this graph in a top-down fashion while generating each node at most once and using a max heap to iterate over the data groups by their size.

3 SYSTEM AND DEMONSTRATION

We implemented NEXUS using Python and Flask. Our code and datasets are available at [2]. In our prototype implementation, we used the DBPedia KG [1] for attribute extraction.

System overview. As shown in Figure 2, NEXUS contains a UI (depicted in Figure 3) and four major components. The analyst provides a dataset $\mathcal{D}$ and specifies a query Q. The *Attribute Extractor* extracts attributes from a KG, yielding an augmented dataset $\mathcal{D}'$. The *Data Organizer* receives $\mathcal{D}$, prunes irrelevant attributes (attributes with low information content), and detects and handles selection bias. The *Explanation Generator* finds the explanation $\boldsymbol{E}$ for Q. It then passes $\boldsymbol{E}$ to the *Unexplained Data Groups Generator*, which identifies the top-k largest unexplained data groups $\mathcal{G}$. Last, the analyst receives t$\boldsymbol{E}$ and $\mathcal{G}$. She may further use NEXUS to generate explanations for groups in $\mathcal{G}$ as well.

3.1 Demonstration Overview

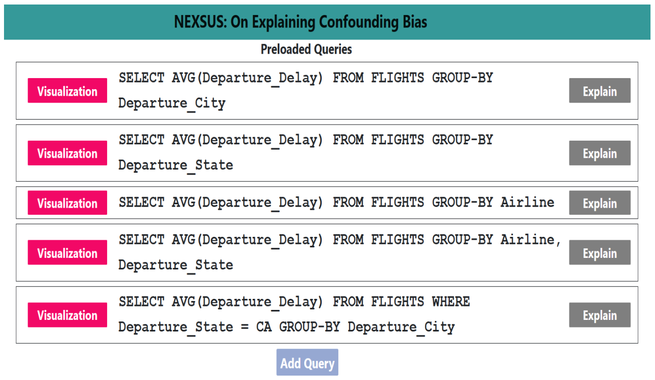
We demonstrate the operation of NEXUS over four datasets, which include attributes that can be linked to entities in DBPedia:

Covid-19 : This dataset includes information such as the number of confirmed, deaths, and new cases across the globe. The countries' names were used for attribute extractions.

SO : Stack Overflow's annual survey contains information about people who code, such as their age, income, and country. The countries' names were used for attribute extractions.

Flights : This dataset contains transportation statistics of domestic flights in the USA. The city, state, and air carrier attributes were used for attribute extractions.

Forbes : This dataset contains the annual earnings and category of celebrities in multiple years. The celebrities' names were used for attribute extraction.


NEXUS: On Explaining Confounding Bias

Preloaded Queries

```
SELECT AVG(Departure_Delay) FROM FLIGHTS GROUP-BY
Departure_City
```

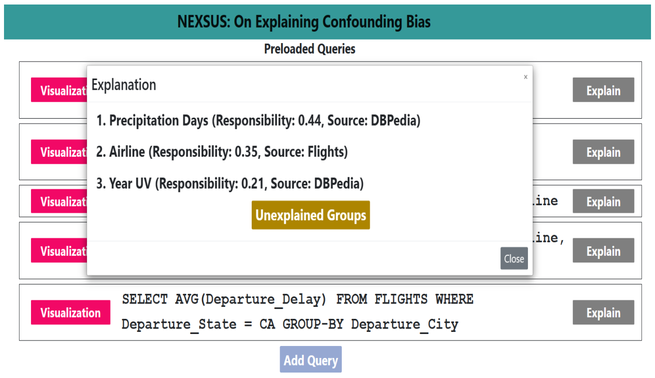
```
SELECT AVG(Departure_Delay) FROM FLIGHTS GROUP-BY
Departure_State
```

```
SELECT AVG(Departure_Delay) FROM FLIGHTS GROUP-BY Airline
```

```
SELECT AVG(Departure_Delay) FROM FLIGHTS GROUP-BY Airline,
Departure_State
```

```
SELECT AVG(Departure_Delay) FROM FLIGHTS WHERE
Departure_State = CA GROUP-BY Departure_City
```

(a) Example queries suffering from confounding bias (Flights).


NEXUS: On Explaining Confounding Bias

Preloaded Queries

Explanation

1. Precipitation Days (Responsibility: 0.44, Source: DBPedia)
2. Airline (Responsibility: 0.35, Source: Flights)
3. Year UV (Responsibility: 0.21, Source: DBPedia)

```
SELECT AVG(Departure_Delay) FROM FLIGHTS WHERE
Departure_State = CA GROUP-BY Departure_City
```

(b) The explanation generated by NEXUS for the first query.

Figure 3: UI of NEXUS.

The participants will play the role of data analysts, attempting to explain unexpected correlations observed in aggregate SQL queries. We start the demonstration by allowing attendees to select the underlying dataset to investigate. The audience then engages with NEXUS via the following scenarios:

Investigating real-life queries: To illustrate the capabilities and usability of NEXUS and for the sake of demonstration, the audience will investigate real-life queries suffering from confounding bias. These queries are inspired by real-life sources, such SO annual reports, media websites (e.g., Vanity Fair, USA Today), and academic papers [7]. For each dataset, NEXUS presents a list of queries with their results visualizations (as depicted in Figure 3a for the Flights dataset). By clicking on the EXPLAIN button to the right of each query, NEXUS displays its explanation (as shown in Figure 3b). For instance, for a query compares the average departure delay among cities in the US, NEXUS generates an explanation consisting of the attributes: PRECIPITATION DAYS, YEAR UV, and AIRLINE, implying that in cities with similar weather and distribution of flights operated by the same airline carriers, the departure delay is similar. The analyst can further inspect the source of each attribute (e.g., DBPedia is the source of YEAR UV). By clicking on the IDENTIFY GROUPS button, NEXUS enables one to examine the top-5 (in terms of size) largest unexplained data groups. For instance, for cities in California, a better explanation for departure delay is the attributes POPULATION DENSITY and SECURITY DELAY.

Manually-defined queries: The participants would be invited to insert their own queries by clicking on the ADD QUERY button (see the bottom part of Figure 3a). The audience specifies their queries using the system Input Builder (omitted from the presentation for space limitations). NEXUS then adds the user-defined query to the repository of queries associated with the selected dataset (to allow other participants to explore them as well). The audience could then inspect the generated explanations and investigate them by exploring the unexplained data groups. This scenario simulates a common real-life task where a data analyst tries to explain the unexpected results of an aggregate SQL query.

Looking under the hood: Interested participants will be invited to examine how system parameters affect quality and performance. For example, the audience will be allowed to vary the percentage of observed values in extracted attributes to learn how the system handles missing values and ensures the explanations are robust to missing data. Further, the audience will be invited to examine the efficiency of our algorithms. For example, we will show that while the explanations generated by our algorithm are similar to those of the naïve approach that iterates over all attribute sets and those generated using HypeDB [12], our algorithm is at least one order of magnitude faster than both of these baselines. We will also show that the difference in running times between the algorithms increases as the number of extracted attributes increases.

Acknowledgments: We appreciate the support of NSF Convergence Accelerator Program award 2132318, the Israeli Science Foundation (grant No. 2121/22), and the Schmidt Futures initiative.

REFERENCES

[1] 2022. DBPedia. https://www.dbpedia.org/.
[2] 2023. Technical Report. https://github.com/ResultsExplanations/ExplanationsFromKG.
[3] Kareem El Gebaly, Parag Agrawal, Lukasz Golab, Flip Korn, and Divesh Srivastava. 2014. Interpretable and informative explanations of outcomes. *PVLDB Endowment* (2014).
[4] Mahdi Esmailoghli, Jorge-Arnulfo Quiané-Ruiz, and Ziawasch Abedjan. 2021. COCOA: COrrelation COefficient-Aware Data Augmentation.. In *EDBT*.
[5] Sainyam Galhotra and Udayan Khurana. 2022. Automated relational data explanation using external semantic knowledge. *PVLDB Endowment* (2022).
[6] Clark Glymour, Kun Zhang, and Peter Spirtes. 2019. Review of causal discovery methods based on graphical models. *Frontiers in genetics* (2019).
[7] A Kaklauskas, V Milevicius, and L Kaklauskiene. 2022. Effects of country success on COVID-19 cumulative cases and excess deaths in 169 countries. *Ecological indicators* (2022).
[8] Chenjie Li, Zhengjie Miao, Qitian Zeng, Boris Glavic, and Sudeepa Roy. 2021. Putting Things into Context: Rich Explanations for Query Answers using Join Graphs. In *SIGMOD 2021*.
[9] Judea Pearl. 2009. *Causality*. Cambridge university press.
[10] Hanchuan Peng, Fuhui Long, and Chris Ding. 2005. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *TPAMI* (2005).
[11] Sudeepa Roy, Laurel Orr, and Dan Suciu. 2015. Explaining query answers with explanation-ready databases. *PVLDB Endowment* (2015).
[12] Babak Salimi, Johannes Gehrke, and Dan Suciu. 2018. Bias in olap queries: Detection, explanation, and removal. In *SIGMOD 2018*.
[13] Alberto Santos et al. 2022. A knowledge graph to interpret clinical proteomics data. *Nature Biotechnology* (2022).
[14] Aécio Santos, Aline Bessa, Fernando Chirigati, Christopher Musco, and Juliana Freire. 2021. Correlation sketches for approximate join-correlation queries. In *SIGMOD 2021*.
[15] Shaun R Seaman and Ian R White. 2013. Review of inverse probability weighting for dealing with missing data. *Statistical methods in medical research* (2013).
[16] Ganggao Zhu and Carlos A Iglesias. 2018. Exploiting semantic similarity for named entity disambiguation in knowledge graphs. *Expert Systems with Applications* (2018).