Author's pre-print.
Version of record available at: https://doi.org/10.1109/TVCG.2020.3030443

CAVA: A Visual Analytics System for Exploratory Columnar Data Augmentation Using Knowledge Graphs

Dylan Cashman, Shenyu Xu, Subhajit Das, Florian Heimerl, Cong Liu, Shah Rukh Humayoun, Michael Gleicher, Alex Endert, Remco Chang

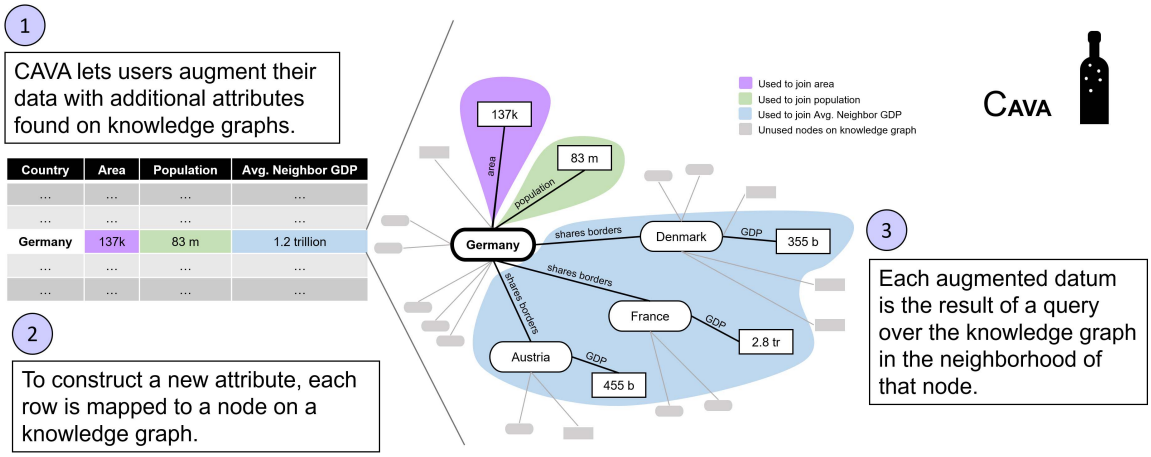

Fig. 1. The CAVA system allows a user to augment a tabular dataset with additional attributes gathered from a knowledge graph. This figure illustrates the process of gathering three additional attributes corresponding for a single row of the data.

Abstract— Most visual analytics systems assume that all foraging for data happens before the analytics process; once analysis begins, the set of data attributes considered is fixed. Such separation of data construction from analysis precludes iteration that can enable foraging informed by the needs that arise in-situ during the analysis. The separation of the foraging loop from the data analysis tasks can limit the pace and scope of analysis. In this paper, we present CAVA, a system that integrates data curation and data augmentation with the traditional data exploration and analysis tasks, enabling information foraging in-situ during analysis. Identifying attributes to add to the dataset is difficult because it requires human knowledge to determine which available attributes will be helpful for the ensuing analytical tasks. CAVA crawls knowledge graphs to provide users with a broad set of attributes drawn from external data to choose from. Users can then specify complex operations on knowledge graphs to construct additional attributes. CAVA shows how visual analytics can help users forage for attributes by letting users visually explore the set of available data, and by serving as an interface for query construction. It also provides visualizations of the knowledge graph itself to help users understand complex joins such as multi-hop aggregations. We assess the ability of our system to enable users to perform complex data combinations without programming in a user study over two datasets. We then demonstrate the generalizability of CAVA through two additional usage scenarios. The results of the evaluation confirm that CAVA is effective in helping the user perform data foraging that leads to improved analysis outcomes, and offer evidence in support of integrating data augmentation as a part of the visual analytics pipeline.

Index Terms—Visual Analytics, Information Foraging, Data Augmentation

1 Introduction

Over 20 years ago, Pirolli and Card coined the term "information foraging" to denote the process of seeking and gathering information to apply towards a task [59, 60]. While they focused on foraging for additional entities or rows of a dataset, foraging for additional data attributes or columns can unlock new analytical capabilities. For example, if a dataset contains a list of countries, adding the population of those countries as an attribute enables per-capita analyses.

Traditionally, crafting new attributes and augmenting a dataset with them is done prior to any visual analysis. Incorporating this process into visual analytics workflows can benefit both the augmentation process as well as the underlying tasks of the system in several ways. First, the need for an additional attribute may only arise based on insights generated during analysis; if data augmentation is embedded in the system, users can toggle back and forth between analysis and augmentation. Second, the process of discovering data and crafting a new attribute is an analytical task in its own right, and typically consists of complex querying that can be abstracted away by a visual interface. Lastly, user-driven curation of the attributes of the dataset can serve as an additional medium of communication between user and system: the user can communicate domain knowledge by adding new attributes, and can likewise learn from the attributes that are discovered by the system.

However, there are many complications in integrating data augmentation into visual analytics systems. Finding external data and matching it to the entites in the user's dataset is nontrivial. Once connected to

- *Dylan Cashman, Cong Liu, and Remco Chang are with Tufts University. E-mail: {dcashm01, cong, remco}@cs.tufts.edu.*
- *Shenyu Xu, Subhajit Das, and Alex Endert are with Georgia Tech. E-mail: {das, shenyuxu, endert}@gatech.edu.*
- *Florian Heimerl and Michael Gleicher are with University of Wisconsin, Madison. E-mail: {heimerl, gleicher}@cs.wisc.edu.*
- *Shah Rukh Humayoun is at San Francisco State University. E-mail: humayoun@sfsu.edu.*

Manuscript received 30 Apr. 2020; accepted 14 Aug. 2020. Date of Publication 06 Sep. 2020; date of current version 06 Sep. 2020.

external data, it can be difficult to determine which data is relevant or useful to the user's analysis. Some attributes may require the aggregation of multiple pieces of external data and can require joining through several intermediate entities. These hurdles should be abstracted away so that the user is not required to be a database expert.

Large scale information repositories provide the potential for interactive data augmentation because they put the potential data at the user's fingertips. However, in order to use such repositories a number of challenges must be addressed. If the repository is in the form of a data lake (a large collection of tables, e.g. WikiTables [11]), it can be difficult to determine which tables are relevant and joinable to the user's dataset. Tabular data can often be fragmented as well: for example, if a user is foraging for data on water usage across the United States, each municipality might be responsible for publishing its own data, and the formats may not align, resulting in sparse, error-prone joins. Joining together multiple tables can also result in ambiguities over the type of join (left, right, outer, inner) that are difficult to resolve, and limit the types of attributes that can be constructed without additional data munging. In addition, plenty of external data may be irrelevant or even harmful to the analysis tasks at hand. While there have been great strides in addressing some of the technical issues in using data lakes (see section 2.2), little work has been done to allow a user to explore potential attributes and construct new ones. Knowledge graphs offer an alternative to data lakes because they simplify entity matching and joining ambiguities due to their graph format. They bring different challenges, however, in scalability and complexity, and there is still a dearth of tools available to facilitate foraging over knowledge graphs.

In this paper, we present CAVA, a visual analytics system for Columnar data Augmentation through Visual Analytics. Carefully crafted attributes are synthesized by running queries over knowledge graphs, as illustrated in Figure 1, before being added to a dataset as additional columns for downstream analysis. First, each row of a dataset is mapped to an entity in a knowledge graph. Then, to help the user identify potential information of interest, CAVA explores the local graph neighborhood about entities in the dataset to determine commonly held attributes. Using visualizations of data quality, distribution, and the local topology of the knowledge graph, CAVA guides the user through crafting additional attributes of data without any programming or explicit querying. Users can express complicated operations over the knowledge graph by interacting with these visualizations. In sum, CAVA abstracts away the complexity of searching, retrieving, and joining data so that the user can focus on the exploratory task of determining which attributes are relevant to their analysis.

CAVA shows the promise of integrating column augmentation as a foraging process within a VA system. The design process presented in this work in section 5.2 can be useful for future visual analytics tools to adopt columnar data augmentation as part of their workflow. We claim that a resulting VA system can make the exploration and discovery of additional attributes interactive, and let users construct complex attributes without programming. Users are able to discover new attributes as well as create attributes they had in mind. The construction of the augmented dataset can improve the outcome of downstream analytical tasks, such as insight generation, predictive modeling, or any other number of tasks that rely on the presence of a robust dataset.

We offer evidence of these claims by describing two usage scenarios for insight generation and predictive modeling. In the first usage scenario, we demonstrate how a domain scientist's analysis can be enriched through cycles of in-situ column augmentation and analysis. And in the second usage scenario, we show how a user can craft attributes that result in significantly more accurate machine learning models. We also conduct a preliminary user study on two datasets to assess the usability of CAVA. The results confirm that users are able to both discover and craft relevant attributes easily and quickly.

In this work, we present the following contributions:

- We present a visual analytics system, CAVA, for exploratory data augmentation using knowledge graphs. We also describe the design process of using visualization as a medium for query construction and knowledge graph exploration.
- We provide usage scenarios of CAVA being applied to both insight generation and predictive modeling to demonstrate the generality of our approach.
- We conduct a preliminary user study to assess the usability of our system in joining semantically meaningful external data across two different tasks, offering validation of our design.

2 Background and Related Work

2.1 Knowledge Graphs

Our system uses knowledge graphs as its knowledge representation. Relational databases are one of the most popular formats to store data because they can be implemented and queried effectively and efficiently. However, recently knowledge graphs have gained popularity for storing very large datasets because their entity-based model is conducive to how humans think about data. Knowledge graphs represent data entities as nodes in a graph and the edges between those nodes are relationships between those entities. The term *knowledge graph* has been loosely used to describe collections of information. Several different definitions have been offered in recent years [36]. In practice the term has been used interchangeably with *knowledge base* or *ontology* [23]. Structuring entities in the form of an ontology is also used to organize conceptual spaces, for example for the evaluation of visual analytics systems [15] and visualizations that support machine learning tasks [66].

The current popularity of knowledge graphs to store information can be traced back to 2012, when Google introduced its *Knowledge Graph*. Google's Knowledge Graph is a repository that retrieves facts about entities in search terms on their search results pages [22, 68], and is used to power the Google Assistant [46] and Google Home voice queries [12]. Recent work has demonstrated that the approach can generalize to other artificial intelligence tasks such as image captioning and conversational agents [31, 37, 75].

Since 2012, Wikipedia and other Wikimedia projects have collected information into the *Wikidata* knowledge graph. It contains broad information about tens of millions of entities. Other large public repositories exist, from DBPedia [7], a collection of data resources, to the domain-specific universal protein knowledgebase *UniProt* [1] and the linguistic resource *WordNet* [49]. Many of these graphs are part of the Linked Open Data Cloud [48], a knowledge graph that contains information about other knowledge graphs. All these graphs contain a tremendous wealth of structured data that can be accessed programmatically with little to no data munging.

Knowledge graphs have been become popular in many research areas such as information retrieval, Natural Language Processing (NLP), and text analysis. For example, a number of approaches (e.g., [19, 45, 62, 63]) and tools (e.g., [25, 26]) focus on text-centric information retrieval using entity links in underlying knowledge graphs. Examples of applications include the discovery of emerging entities [34], the structuring of contents into topics [8], extracting relations [51], semantic search [9, 42, 55], or predicting the missing relations between entities [21, 74]. Researchers in NLP and text analysis fields have also utilized knowledge graphs from different perspectives, e.g., for embedding entities and words into a continuous vector space [72], for extracting relation facts form text [35, 65], for the generation of questions and answers pairs automatically [64], and for parsing and interpretation of user natural language semantically [27].

In our system, CAVA, users can tap knowledge repositories as a data source to augment a dataset through interactive visual support. Knowledge graphs have the advantage of providing clean, curated data which means that the need for data cleaning and matching is greatly reduced. In addition, the entity-focused way in which knowledge graphs store information can make it easier to think about the relationship of data objects and thus helps users guide the augmentation process.

2.2 Data Augmentation

In the machine learning community, the goal of data augmentation is to expand a training dataset so that as much of the phenomenon being modeled is present as possible. This is done in two ways: adding objects (rows) to the training data, and adding new attributes (columns) for the

objects. For example, in image datasets, adding new objects in the form of slightly modified versions of the training objects (i.e. rotating, adding noise, cropping, modifying color) can improve a machine learning model's sensitivity to noise in the data [38, 43, 56, 58, 67].

Approaches to add columns to a dataset typically vary in where that data comes from, and whether that data is used in training the model or in model inference. Feature engineering [40] is a common method to derive new attributes from existing columns by applying operations to them, such as the difference of two columns. However, it is limited in that it can only express data that is in the scope of existing attributes. Knowledge graphs have recently been used to incorporate world knowledge into machine learning models [70] for a range of models, including text processing [6], image classification [47], and machine translation [53], but most work incorporates knowledge graphs into the last step of the machine learning pipeline, inference, to gather facts, rather than augmenting an entire dataset. The Python library RDFFrames [52] helps extract data from knowledge graphs to improve machine learning training. It allows practitioners to effectively express queries to knowledge graphs and execute them efficiently. However, it requires extensive data science and programming experience to use.

Automated augmentation of datasets is of interest to the database and the data science communities. In the database community, the goal of data augmentation can manifest in many ways. For example, before augmentation can take place, the first challenge is to find datasets that are suitable for joining. In this scenario, sometimes referred to as a "data lake" [50], the data joining system assumes that there is a finite number of candidate datasets, and the task is to identify which of them can be joined with a user's base dataset. Systems such as Google Goods [28], Infogather [73], Octopus [13], Aurum [24], and work by Sarma et al. [20] examine the attributes of the candidate datasets and learn the relationship between those attributes and the attributes of the input data.

One challenge of automated data augmentation that is an active area of research is entity matching. Entity matching refers to finding the same entity in different datasets, often using machine learning techniques. Once identified, the entity can be removed (for deduplication) or merged (for augmentation). Examples of entity matching systems include Magellan [41], Nadeef [18], Autojoin [76], and work by Mudgal et al. [54].

What the data joining and entity matching approaches have in common is that the systems assume little knowledge about the data. The challenge is therefore to identify the commonalities between the datasets (e.g. schema matching) to determine if the datasets or the entities within are related and therefore joinable. In CAVA, we take a different approach. Instead of assuming little to no knowledge about the candidate data, we use a knowledge graph as the source of "candidate data." Although knowledge graphs can be loosely structured, they contain well-defined relations between entities. This property of knowledge graphs allows CAVA find relevant data for augmentation with certainty and ease.

While automated approaches can discover joinable attributes for a given dataset, they lack semantic knowledge of expert users. This may result in a large number of attributes that are irrelevant to the analysis problem being added to the dataset, hindering the users in further analyzing and gaining insights from the dataset. To make use of their semantic knowledge to choose relevant attributes, users may benefit from an exploratory process facilitated by visual analytics. Beyond the selection of relevant attributes, users may benefit from visual explanations of complex queries. For example, in transitive relationships that go through multiple entities, joins require multiple operations to be specified for aggregating a collection into a single value. CAVA overcomes these issues by involving a user in the process. Attributes are only added to the dataset if the user decides that they are helpful for their analysis.

2.3 Visual Analytics Approaches

Curating, improving, and augmenting an existing dataset has been a popular research topic in visual analytics. This can be done to aid interpretation and sensemaking [17] during analysis. Other visual interactive approaches allow users to query and integrate query data from heterogeneous web or local sources. This includes helping users extract information from textual web sources [32], query knowledge graphs [33], query large metadata-rich heterogenous data stores [69], identify relations at attribute levels in mixed data sets [10], analyse associated categories in large categorial data tables [5], or automatically create visual representations of information stored in knowledge graphs. VAiRoma helps users extract and combine information from Wikipedia articles to create visualizations that provide insight into historical events [16]. Vispedia [14] lets users interactively collect and integrate data from Wikipedia tables to create visualizations and answer analysis questions. Atlasify enables users to relate the query concept, corresponds to a Wikipedia article, to spatial entity in the underlying reference system, e.g., countries, political figure [30]. All of these approaches help users access information stored in different forms, but compared to CAVA do not support the process of augmenting an existing dataset with additional data.

3 Knowledge Graphs

We define our usage of the term, knowledge graph, in this section, and describe the sufficient characteristics of a knowledge graph that we assume for our system. We assume that knowledge graphs contain entities as nodes and express attributes of those entities as edges, connecting the source entity to either another entity (to express a type of relationship) or to a literal (such as a string, number or datetime). The number of edges connected to a node can be a proxy for the number of attributes that are held by that entity. The flexibility of this structure allows for complex relationships to be expressed between entities, including one-to-one (i.e. a country has one head of state), one-to-many (a country is composed of multiple municipalities), and many-to-many (countries share borders with other countries non-exclusively).

CAVA requires that the connected knowledge graph has a data retrieval endpoint that can respond to simple queries about entities and their neighbors. The connected graph must also have some sort of service to map from the values in the dataset uploaded to CAVA to the entities on that knowledge graph. For example, if a dataset with U.S. states is uploaded, there must be an existing service that maps from "New York" or "NY" to the entity in the knowledge graph corresponding to that state. These two requirements allow CAVA to connect a user's data to the knowledge graph and to retrieve relevant data for the user to forage from.

In the experiments in this paper, we connect CAVA to Wikidata. It meets both requirements listed above: it responds to the SPARQL query language so gathering neighborhood data about entities is simple, and it has a service `wbsearchentities` to map strings to entities on the graph. But it also has some extra features that we take advantage of. Wikidata contains a broad set of information making it easy to find related attributes for many different kinds of datasets. It also has additional metadata about the data in its graph including datatype information and human readable descriptions and labels for each node and edge in the graph. Lastly, due to its popularity, there is a large amount of documentation and guidance on constructing complex queries.

In future work, we hope to allow users to connect CAVA to multiple knowledge graphs. To limit scope in this work, we focused only on Wikidata because of its high standards for data quality. In addition, it contains general knowledge that made it an applicable information repository for a number of different datasets and usage scenarios. This also made it easier to recruit users for the study described in section 8. While different knowledge graphs bring different challenges, particularly around data sparsity (see the discussion in section 9.2), Wikidata provided a broad enough testbed to inform the design goals and interactions supported by CAVA.

4 Tasks and Goals

CAVA is developed as part of the DARPA Data-Driven Discovery of Models (D3M) program whose goal is to develop software infrastructure and algorithms to make automated machine learning accessible to general data scientists. Inspired by the observation that both the use of

exploratory visualization and the use of advanced machine learning are fundamentally limited by the input data, CAVA was developed to help a data scientist craft better predictive models by foraging for additional attributes to add to their dataset.

We conducted interview sessions with four teams within the DARPA D3M program developing applications for data exploration and predictive modeling. The goal of the interview sessions was to better understand how a tool like CAVA can help the user to perform data augmentation for the purpose of improving machine learning model performance and accuracy. Each of the interviewed teams was shown an early implementation of CAVA that was able to search for related attributes and return a list of them, but without any visualizations. The participants were then asked about what additional system features and interactions would facilitate the discovery of the types of data that they were interested in for their applications.

4.1 Task Analysis

We distilled a list of tasks from the interviews that would enable a user to meaningfully augment their dataset with additional attributes.

T1: View a list of joinable attributes for any existing attribute in the dataset. Revealing the list of potential attributes that can be joined with a particular column in the dataset helps the user determine what external data is available for augmentation. It is a key aspect of the exploration process in information foraging because it may reveal data that the user wasn't aware of.

T2: Analyze the properties of possibly related attributes *before* the join occurs: Before joining a new attribute into a dataset, users will need to gather information about the new attribute to gauge the new attribute's impact on the quality of the dataset. For this reason, users should be provided with as much information as possible about the potentially joinable attributes before the join. This might include metadata, examples of the attribute, and any available information about the general distribution of the new data. In addition, it is important to communicate whether there will be any missing values in the joined attribute.

T3: Specify aggregations: The relationships between entities in the user's dataset and entities in the external data source are varied - they could be one-to-one, one-to-many, or many-to-many. In order to fit information encoded as plural relationships into a single row of data, aggregations must be specified. For example, a user might want to add the populations of a list of states into the their data. However, because there could be multiple measures of the state's population over the years, the user would need to perform an aggregation function over these results, such as MINIMUM, MAXIMUM, or AVERAGE depending on their analysis need.

T4: Connect through to additional data: Attributes relevant to the user's analysis goals may not always be directly encoded as a property of an entity that exists on the initial dataset. For example, a user may want to augment a dataset of countries with the population of each country's capital. This information may not be directly accessible as a property of each country in the data repository. Instead, population size is linked to the entity that represents the capital, which in turn is linked to the country. To support these cases, users should be supported to join *through* intermediate attributes giving them fine-tuned control over how joins pass through multiple relationships.

4.2 Design Goals

Based on this task analysis, we iterated through several sketches to come up with a set of design goals. In the subsequent section, we describe a system with visualizations and interactions that address these goals. However, this list of design goals should prove to have a broader impact than CAVA alone - they can inform how data augmentation can be integrated into other visual analytics tools.

G1: Attributes as first-class citizen. Because external data is found based on attributes in the original data (**T1**), the list of attributes is the most important data to encode. In early sketches, we used a data table to show the user what the data looked like, and allowed users to click on column headers to view new attributes and view joined data in enriched cells, similar to the *ETable* from Kahng et al [39]. We ultimately decided that a table encoding dedicated too much space to the table formatting and data found in the table cells. Instead, we settled on a primary view that listed all attributes of the dataset. By viewing the list of attributes, users are able to get a sense of what attributes might still be needed for their analysis. This list can update as new attributes are foraged. Additional details about each attribute, such as summary statistics and examples, can be provided on demand.

G2: Visual exploration of joinable data: In order to discover interesting joinable data (**T2**) and specify aggregations on that data to create new attributes (**T3**), users should be able to explore the set of joinable data. Visual cues can help in this exploration, by encoding meaningful information about external data like the join quality or the data distribution. As the dataset evolves, users should be given previews of the data they have added to enable iterations of exploration.

G3: Provide visual examples for complex queries. In practice, building complex queries that stepped through intermediate information on the knowledge graph becomes very complicated because it requires multiple aggregations over the joined data. For these types of complex queries, visualizing the knowledge graph about one example in the data can help a user understand how they are combining and shaping the data.

5 CAVA: A Visual Analytics System

In this section, we describe the function and design of CAVA. The general workflow of CAVA starts when a user uploads a tabular dataset. The user can then search for related attributes of any column in their dataset containing entities that can be found in Wikidata. They can augment their dataset with these related attributes by specifying aggregations. After several iterations, if the user is satisfied with the augmented dataset, they can export their data for further analysis in other visual analytics tools.

In this section, we first describe the design of each component of the interface and their interactions. Then, we explain how these user interactions are translated into queries that can be executed over a knowledge graph.

5.1 User Interface

Figure 2, CAVA has the following main interface components:

Column View: This view shows the attributes and their data types (**G1**, see Figure 2-A). Users can view details on demand about each attribute, including the data distribution of that attribute shown as a histogram chart, several example values, the type of unit (i.e. kilogram or mile, if applicable) (see Figure 2-C).

Related Attributes: A user can search for related attributes of any string attribute in the Column View. CAVA shows the related attributes discovered from the knowledge graph next to the list of attributes of the current table, as seen in Figure 2-B. An estimation of the join quality of each related attribute is encoded visually with a donut chart, and details on demand are also provided for each attribute, including distribution of values. This information can help the user identify whether a related attribute has the potential to help their analysis *before* joining it to their data (**G2**). Both the estimated probability and the estimated attribute distribution are calculated from random sample of possible joinable data, since calculating the true values would require completing the full join for every row in the dataset. For more details on implementation techniques, see section 5.2.

Adding Attributes: To add any attribute, users click the plus symbol, and CAVA will show a drop-down menu revealing various options for join operations. The available join operations correspond to the data type of the relationship. If the relationship is one-to-one, i.e. a country has a single head of government, then a single value will be retrieved for each row of the dataset. If the relationship is one-to-many

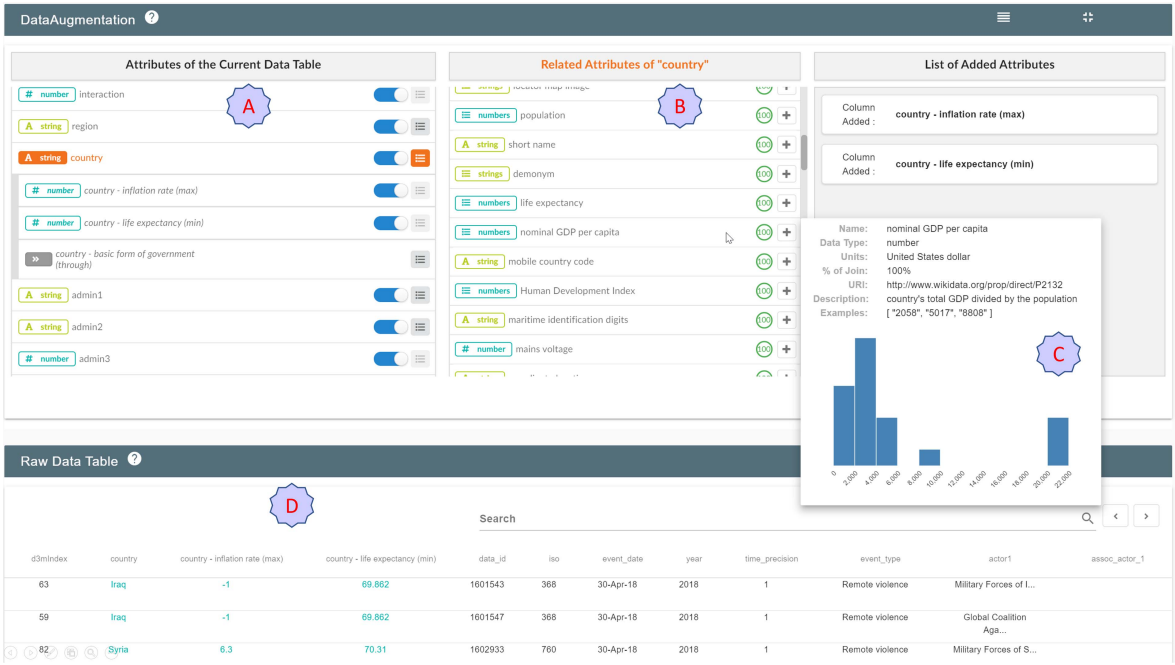

Fig. 2. The user interface of CAVA, showing an analysis session on the ACLED dataset. **A** shows the initial list of attributes in the table as well as augmented attributes, like the maximum inflation rate recorded for each country. The list of attributes related to the *Country* column is shown in **B**. Each related attribute is adorned with a donut chart showing the estimated join quality. Additional details about any attribute is available on demand in a popup, as seen in **C**. As attributes are added to the dataset during the analysis session, a preview of the table is updated as seen in **D**.

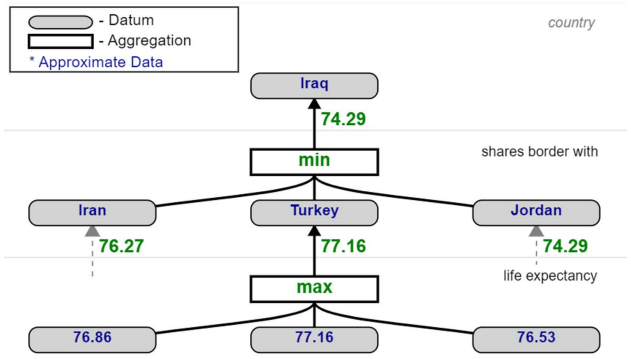

Fig. 3. When a through join is specified, a dialog window pops up to allow the user to specify aggregations at each level of the join. To help them specify that query to the system, CAVA shows them an example of the neighborhood in the knowledge graph that is queried for a single row of the table. In this image, the user would like to add the lowest life expectancy of any neighboring countries because they believe this will help their analysis of conflict. And for each country, several values are available for life expectancy, so the user selects the "max" operator. All six neighbors for Iraq, but only three are shown for simplicity of the illustration.

or many-to-many, an aggregation must be specified. For example, if the attribute is a collection of numbers, such as set of populations recorded for a country, the user can select numerical aggregations such as *count* , *mean*, *max*, *min*, *sum*, or *variance*. If the attribute is a datetime, the user can select from *count*, *max*, or *min*. If the attribute is a string, the user can select either *count* or *through*, a type of placeholder operation that let's the user step through this attribute to form complex multi-hop joins (more details are provided below). In addition to these join operations, for any type of data, the user can select to randomly sample from the collection. This operation is useful if an attribute is generally expected to have only one value per entity, but due to data quality issues, some rows might have it recorded twice, like date of birth.

Extracting information for the whole dataset from the knowledge graph is too computationally demanding to be performed at interactive rates. Gathering the data from the knowledge graph requires the retrieval of many different parts of the knowledge graph in contrast to the retrieval of a single column from a relational database. CAVA mitigates much of that time cost by only joining attributes for the top 10 rows of the dataset. Users are still able to glance at the dataset preview, as seen in Figure 2-D, and get an idea of the data that is being joined to the dataset (**G2**). This lets the user explore the available attributes rapidly. When the exploration phase is completed, the full join can be executed.

Through Joins: In some cases, the user may want to join to data through an intermediate attribute. If the intermediate attribute has a one-to-one relationship with the rows of the original dataset, i.e. a country has one head of government, then the user can simply join the intermediate attribute first, and then use that as their starting point to seek more related attributes. If the intermediate attribute is a one-to-many relationship, then the user must specify multiple levels of aggregation. We call this type of aggregation a *"multi-hop aggregation,"* since it requires aggregating data across multiple hops on the knowledge graph.

As an example, suppose the user wants to determine if a country has a lower life expectancy than its neighbors. Consider how that would be calculated for a single country, Iraq. To gather the required data from the knowledge graph, the user must first join to all bordering countries, then connect through them to reach the life expectancies of those countries. Once all data is selected, aggregations must be specified to produce the desired value. In Wikidata, countries have multiple life expectancies recorded, so the user has to specify that they want the maximum life expectancy recorded. That value is then calculated for each bordering country (i.e. Iran, Turkey, Jordan, etc.). Then, they have to specify that they want the minimum among all bordering countries. Expressing this type of query is complicated enough to explain for a single value, let alone for an entire column.

In CAVA, the user is shown a simplified illustration of the topology

of the knowledge graph to assist them in understanding the aggregations that the user must choose, as seen in Figure 3. For a given row of the dataset, up to three values of the intermediate attribute and the target attribute are sampled from the knowledge graph. The system then displays these values on a graph, representing a small slice of the knowledge graph. Users can then dynamically set the aggregations (i.e. *count*, *mean*) that occur at each level of the join graph, and see the resulting value that would get joined for that row. By viewing the values that will get *passed through* the knowledge graph to ultimately calculate the attribute for one row of their data, users get reinforcement that the complex query they are building in CAVA matches the query they are building in their head (**G3**).

5.2 Backend Implementation

Here, we describe how our visual analytics system augments a dataset by translating the tasks supported by the user interface described above into valid queries over a knowledge graph. CAVA gathers data from the connected knowledge graph in two different subroutines. In all examples in the paper, CAVA connects to Wikidata using the SPARQL query language.[1]

Finding Related Attributes: This subroutine receives a column of data as input and returns a list of attributes along with assorted metadata. Consider a dataset with a Country column, i.e. "Germany," "France," "Austria," etc.. Unlike a column in a SQL table, where entities and their relations might be expected to adhere to a schema, a list of countries in a knowledge graph might not share the same set of attributes. For example, in Wikidata, some Western countries, such as Germany, can have more than a thousand attributes, while other countries have only a fraction of that. In order to provide a user with a consistent set of attributes for these entities, we first need to find their commonalities. As such, the primary goal of this subroutine is to return the list of attributes that are available on as many of the entities as possible.

Checking the related attributes of each entity to gather this list can take prohibitively long, especially if the input data contains thousands to millions of rows. As an optimization step to make this subroutine support a user's interactive analysis, we employ a sampling-based technique to reduce computation time. With this optimization, some subset of the dataset is randomly sampled; in our experiments we have found that 20-30 rows are sufficient to differentiate high-quality attributes from poor-quality ones. Each sampled row is mapped to an entity on the knowledge graph, and a list of related attributes for that row is retrieved. Then, the lists for all the rows are compared, and the 50 attributes that appear on the most lists are returned. In this query, we also gather all details of the attributes that are displayed on demand, such as the data distribution of that attribute shown as a histogram chart, several example values, and the type of unit (i.e. kilogram or mile, if applicable). For the types of exploration done in the user study and in examples given in this paper, retrieving the list of related attributes from Wikidata typically takes 2-5 seconds.

Materializing Joins: This subroutine takes in join instructions and returns an augmented dataset with an additional column. The join instructions specify the path taken through the knowledge graph, along with any aggregation functions, such as "count" or "max." CAVA builds a SPARQL query to crawl the knowledge graph to retrieve the desired data (see Fig 1). The time to retrieve data scales linearly with the number of rows and is limited by the concurrency limits on the knowledge graph API. While the user is interactively exploring attributes, CAVA will only materialize the join for the top 10 rows of the data table to provide the user with a preview of the table in real time (see Fig 2-D). When the foraging is done, all joins are materialized for the entire dataset (rather than just a sample of rows) in the order in which they were constructed by the user, which takes around 10-15 seconds on the datasets explored in the user study in section 8.

CAVA is implemented with a VueJS frontend web application and a NodeJS backend to handle all asynchronous calls. Queries are sent concurrently whenever possible. Source code and installation instructions are available at `https://tuftsvalt.github.io/snowcat/`.

[1]Example queries generated by the system for both subroutines are available in the supplemental material.

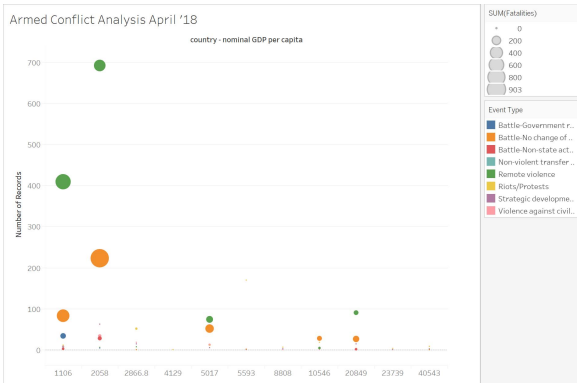

Fig. 4. Visualizing the number of records in the ACLED dataset in April 2018 by per capita GDP, a field extracted from Wikidata. Magnitude of marks encodes number of fatalities, while color encodes event type. Visualization generated with Tableau.

6 Usage Scenario 1: Conflict Data

To demonstrate the value of column augmentation for insight generation, we present a usage scenario in which CAVA is integrated into a typical workflow for the popular visual analytics tool, Tableau. Suppose a political scientist wants to study the factors that lead to armed conflict. They download the Armed Conflict Location & Event Data Project (ACLED) dataset for April 2018, which contains 2,279 records of armed conflicts across many different countries and dates [61]. Initially, they load the data into Tableau to analyze relationships between countries, event types, and fatalities stemming from armed conflict. They see no clear pattern that emerges as they find it difficult to visually group countries without more data.

They load their dataset into CAVA to look for additional attributes that may be relevant to event types. When the system is first loaded, the user sees a list of the attributes in the uploaded table in Figure 2-A. They also can see the first five rows of the data below in Figure 2-D.

The political scientist believes there might be economic metrics that effect the types and severity of conflicts, so they click the *related attributes* button next to "Country" in Figure 2-A. The system returns a list of related attributes found by scraping Wikidata, as seen in Figure 2-B (**T1**). The user scans through the list of related attributes, looking for information about the countries' economies. They see that Wikidata holds the *nominal GDP per capita* for each country. The user adds this attribute to the dataset. While looking for economic data, the user also notices additional interesting attributes for analysis and joins them for future analysis, including the max *inflation rate*, minimum *life expectancy*, the *type of government*, and the mean *Human Development Index* (**T3**).

The user loads this next iteration of the dataset into Tableau, and generates various visualizations including the event types and fatality numbers with their newly-discovered attributes. As can be seen in Figure 4, grouping event types by nominal GDP per capita reveals a trend that certain event types, like *Remote violence* and *Battle - No change of territory*, appear to have more severe fatalities in countries that have lower nominal GDP per capita. In contrast, other event types like *Battle - Government regains territory* and *Battle - Non state actor overtakes territory* don't seem to be very different between high nominal GDP per capita countries and low ones. The additional visual analysis enabled by the added attribute, nominal GDP per capita, has led the user to new hypotheses that can then be further explored. This usage scenario demonstrates how the integration of column augmentation into a traditional insight generation pipeline can unlock new analyses in-situ.

7 Usage Scenario 2: Modeling Poverty

In our second usage scenario, we demonstrate the value of CAVA for a separate type of visual analytics task - the generation of a predictive

model. Many visual analytics systems have been constructed to enable domain experts to interact with and steer the generation of machine learning models on their data. Augmenting a dataset with new attributes is one way in which a user can imbue domain knowledge into the modeling process. They may know that certain attributes can help the prediction while others may mislead. While it may seem unusual to augment a training set with additional attributes, the additional attributes found on the knowledge graph can likewise be added to any data that the resulting model is asked to predict on at test time when the model is deployed.

To demonstrate the use of CAVA in the model building process, we modify CAVA to be able to train a predictive model at any point in the data augmentation process. Users can instruct the system to build a predictive model with the current version of the data. CAVA splits the data into a training and test set in the ratio of 0.8:0.2 and trains a random forest regressor. For each model, CAVA reports the *R Squared* score on the training and test sets, as well as the feature importance scores of the five most important features. This functionality was designed to be representative the iterative workflow of model builders, and to examine the ways CAVA could help in-situ model building.

Consider Andy is a public policy analyst who seeks to build a regression model on a U.S. Census Bureau dataset containing data about poverty in different counties around the U.S. [3]. The dataset contains 3136 rows, and 6 attributes–the dataset index, *FIPS*, *State*, *County*, *RUCCode*, and the number of residents living in the county under the poverty line. Andy seeks to augment this data by adding meaningful columns to accurately predict the *Poverty*. When Andy first loads the data in CAVA, an initial model is trained with an *R Squared* score of 0.577 on training and -1.801 on test set respectively; indicating that the model badly overfitted the training data. Andy seeks to improve the regression model's performance further by augmenting the base data using CAVA's workflow and visual interface. Andy searches for related attributes of the variable *State* from the Column View. In response, CAVA shows a list of related attributes that Andy may consider to add to the data. From these attributes they click on a set of interesting attributes such as *shares border with*, *life expectancy*, *inflation rate*, etc. to see the distribution of values, and its metadata as a text view.

From these set of attributes Andy thinks that the attributes *Inflation Rate* and *nominal GDP per capita* are good attributes to predict *Poverty* and thus decides to add them to the data using the join operation *Mean*. They notice that the newly added column is displayed on the Table View. However looking at a few rows of the table Andy finds quite a few cells that are empty indicating that the joined data may have missing values. Nonetheless, Andy clicks a button to construct a new regression model using the augmented data. Andy notices that the *R Squared* score marginally improves (new score: 0.586 and -1.810 on training and test set respectively). They hover the mouse over the model metric card to see the list of "Top 5" attributes with their weights utilised in the regression model. While Andy expected to see a substantial improvement in the model performance, they infer that the marginal improvement is probably due to missing values in the data after the join operation.

Motivated to improve the model further, Andy searches for columns that may directly help to predict poverty per county. Based on prior knowledge, Andy understands that the *population* of a state may be directly proportional to its poverty, and they add that attribute. In the process of searching for other relevant columns, Andy notices the column *Maximum temperature*. Inquisitive to see if a temperature of a state is correlated with its *Poverty* they add it to the table. After constructing a new regression model, Andy notices that the *R Squared* score changed from 0.586 to 0.577 and from -1.810 to 0.541 for the training and test dataset respectively. Happy with the progress so far they decide to remove any column that may not be contributing to the prediction task. First, they remove the columns *RUC Code*, and *nominal GDP per capita* (by triggering the slider on the Column View) and then triggers CAVA to construct a new regression model. As expected Andy notices that the *R Squared* score did not change. Next Andy searches for related attributes of the column *County*. They explore the set of choices shown in the Column View. They choose to add the column *County-population* (by median operation), and *shares border with* (by count operation). Andy constructs a new regression model to see the models' train and test *R Squared* score improved considerably (0.807, and 0.746 respectively). Content with the improvement, they export the model and the data to continue analysis.

In a short time, Andy has constructed a regression model with substantially better R2 score than the model trained on the base data. They have also gained insight into which attributes are relevant to their modeling problem, which may help their understanding of the resulting machine learning model.

8 Preliminary User Study

We evaluate CAVA in a preliminary user study. The purpose of the evaluation is to validate CAVA both in terms of its usability and its effectiveness in helping a user improve a machine learning model through data augmentation. Specifically, we hypothesize that:

- **H1**: CAVA allows users to accurately join external data given a written description of that data.
- **H2**: CAVA is able to help users discover additional data that can improve the predictive quality of machine learning models trained on the dataset.

We recruited 6 participants (3 Female, 3 Male), between the age of $23-36$. We required each participant to have at least an elementary knowledge of machine learning and data analysis. Due to COVID-19, we were not able to conduct an in-person study. Instead we conducted an online study using Bluejeans[2]. The participants interacted with CAVA on their own computer while sharing their screen. We provided the participants with a url to our system that was hosted on our local machine that is exposed using the Ngrok remote tunneling software[3]. The study took approximately $50-60$ minutes and we compensated the participants with a $10 Amazon gift card.

8.1 Study Design

Before the study we asked participants to fill out a background information questionnaire regarding their name, age, gender, machine learning expertise and various use cases in which they use machine learning. We began the study by showing the participants a tutorial video of CAVA, explaining the workflow, interface GUI elements, and its interaction capabilities that support various join operations. Next we asked the participants to perform three tasks, the first of which was a practice task to ensure that they were sufficiently knowledgeable about CAVA to perform the experimental trials. We proceeded to the experimental sessions only when we observed that the participants were confident and able to use CAVA on their own. In the next two tasks we asked the participants: (1) To add a set of specified attributes to a given dataset. These attributes can be added to the data using various join operations supported by CAVA. (2) To freely augment the data such that they can improve the performance metric of a machine learning model (metric being a R Squared Score of a regression model). We used the Scikit Learn machine learning library [57] to construct *Random Forest Regression* models, in the same manner as described in Section 7. We collected the following data from each experimental trial: (1) *Task competition time*, (2) *Task Accuracy*, (3) *Model performance metric* e.g., R Squared Score, and (4) *User ratings* collected using a post-study Likert-scale questionnaire.

8.2 Datasets

For the tutorial video and practice task we used the same U.S. Census Bureau dataset as section 7. For the first task in our experimental session we use the IMDB Movies dataset [2] containing 500 movies and 28 attributes such as *Director-name*, *Duration*, *Movie-Title*, *Movie Cast Facebook Likes*, etc. The second task in the experimental session which required users to augment data and construct regression models, used the unemployment rate dataset [4]. This dataset contained 1200 rows where each row corresponded to a county's poverty rate measured

[2]https://www.bluejeans.com/

[3]https://ngrok.com/

in a given month. This dataset contained only 5 attributes–the dataset index, *Month*, *State*, *County*, and the *Unemployment-Rate* (dependent variable).

8.3 Tasks and procedure

Participants were first asked to complete a practice task. During the practice task, their answers were not recorded and their performance was not included in the analysis described below. For the practice task, using the Poverty dataset, we asked the participants to add three attributes: (1) the area of the county, (2) the population of the state, and (3) the earliest inception date of any county that each county shared borders with. While the first two attributes could be retrieved using a straight-forward *join by value* operation, the third attribute required the *join through* operation to search for the required attribute that was one "hop" away in the knowledge graph. For the first experimental task using the IMDB dataset, participants were given descriptions of four attributes to add: (1) the director's country of citizenship, (2) the number of awards received by the director, (3) the number of cast members in each movie, (4) the sum of the number of awards received by all the producers of each movie. Note that attributes 3 and 4 required the participants to *join through* several attributes.

The final experimental task gave participants 10 minutes to augment the Unemployment rate dataset with as many attributes as they'd like that they believed would aid in building an accurate regression model to predict poverty rate. At any point in the 10 minutes, participants could instruct CAVA to build a regression model with the current dataset to give them insight into whether they were improving the predictive modeling process. Building a model took about 10-20 seconds, and participants would be shown the change in R Squared Score for each new model, as well as the weights of top "5" attributes used in the model. Random Forest regression models were used because their flexibility, speed, and accuracy met the constraints of the experiment.

8.4 Data Collection

After all the tasks were completed, participants were asked to fill out a post-study questionnaire (using Google Forms) that included: (1) Likert-scale [44] rating questions on their use of the system, (2) a NASA-TLX [29] questionnaire to measure task difficulty, and (3) Open-ended descriptive questions asking users about the interface design, the workflow, and other responses related to improve the usability of the system[4]. The Likert-scale questions asked the participants to rate (in a scale of $1-7$, 1 being "strongly disagree") if they found the system: (1) Easy to learn, (2) Intuitive, and (3) Expressive for data augmentation. We used the open-ended descriptive feedback to qualitatively evaluate the usability and the interaction design of CAVA for their tasks. Open-ended prompts included (1) *Describe your thoughts about CAVA*, (2) *Describe things you disliked about the system or the workflow. Elaborate on how you think it could have been improved*, and (3) *Describe your strategy or your process to augment the data for Tasks 1 and 2*. With the consent of the participants, each session was both video and audio-recorded. We encouraged the participants to verbalize their thoughts following a think-aloud protocol. Furthermore, to assess if data augmentation using CAVA led to any change in the regression models performance, we saved: (1) model metric (i.e., R Squared Score), (2) attribute weights, and (3) predicted values from the model. We also saved user mouse-clicks to analyze the set of attributes the user explored to augment the data.

8.5 Result and Analysis

8.5.1 Task Performance

To assess **H1**, we reviewed the number of attributes that participants correctly joined in Task 1. Five out of six participants properly joined all four expected attributes. Only one participants missed one attribute when joining the number of awards received by the director. This indicates that in general participants were able to use CAVA to accurately add new data attributes.

[4]Pre-experiment and Post-experiment surveys are attached as supplemental material

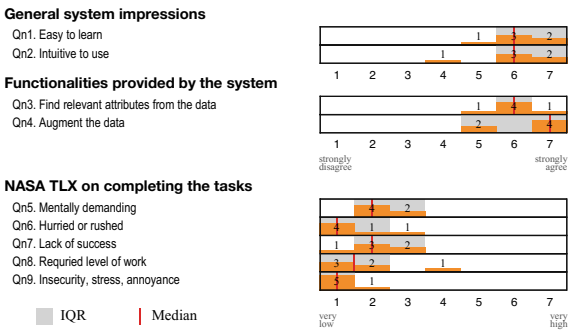

Fig. 5. Participants' responses about the system (Qn1–Qn4, the higher the better) and the overall effort (Qn5–Qn9, the lower the better).

Next, to assess **H2**, we similarly reviewed the *R Squared* score improvement that participants achieved by augmenting additional data in Task 2. On average, participants improved the *R Squared* Score by 0.048 ($\sigma = 0.014$), starting with a baseline *R Squared* Score of 0.292. The area and population of the county are two most frequently joined attributes across all the participants, which is reasonable as these two attributes are usually the most influential for predicting unemployment rate.

As a result of our analysis, we **accept H1** because all participants were able to successfully use CAVA to explore and identify the correct attributes for data joining in Task 1. However, we **neither accept nor reject H2** as the study results did not contain clear evidence of users being able to improve the predictive quality of a regression model in Task 2 beyond trivial improvements.

8.5.2 User satisfaction

We use the post-study questionnaire to assess user satisfaction of the data augmentation process in CAVA. Fig. 5 shows the results of the questionnaire responses. Although the participant size is too small to infer statistical significance using quantitative analysis, we do observe that participants found CAVA easy and intuitive to use based on the likert scale user ratings (Fig. 5, **Qn1,2**). Participants were also satisfied with the two main functionalities provided by CAVA, finding relevant attributes from the data and joining additional data (Fig. 5, **Qn3,4**). The mean ratings of **Qn1–4** were all 6 or above. Furthermore, in analyzing user satisfaction related to the overall process (Fig. 5, **Qn5–9**), we found that participants were generally satisfied with the process of conducting the tasks, as the mean ratings of **Qn5–9** are all 2 or lower.

8.5.3 Qualitative Feedback

To assess **H1** and **H2** from a qualitative perspective, we analysed participants' descriptive feedback collected from post-study interviews. We also observed participants' workflow in using CAVA from audio and video recordings of their computer. We report the following three main qualitative user responses from the study:

Intuitive workflow: All the participants found CAVA's workflow intuitive to find and augment relative data attributes. The primary justification for CAVA's intuitiveness is that CAVA provides a visual interface that is easy to infer and for users to search and add relevant attributes from the data. As *P6* noted, "*I liked the interface of this tool because it was easy to navigate and add/remove columns to the base data*." The intuitiveness helped most participants to easily perform the requested tasks, especially Task 1. For example, *P1* described his experience in doing Task 1 as "*just looked at the question and implemented it*" and "*it was pretty straight forward*."

Two major strategies in model construction: When conducting Task 2, we observed two common strategies used by the participants: (1) searching for relevant attributes based on existing or prior knowledge and (2) exploring all possible combinations of available attributes to find one that results in improvement in model performance. Some participants relied on their existing domain knowledge for augmenting the data with new attributes. This was summarized nicely by *P5*, "*I*

was actually using my domain knowledge. Thinking which factors can improve unemployment rate predictions, and then tried to select the variables based on the choices I had. This actually helped me improve the model's performance." However, a few participants who may lack prior knowledge about the data, instead explored the dataset and tried out all possible combinations of attributes to find a model that is a significant improvement over the model trained on the base data. "*I selected columns one by one to see the change of R-squared score. By doing so, I was able to filter the variables that decrease R-square. In the end, I was able to get a relatively high R-squared model*" (*P6*). The rest of the participants adopted a combination of both strategies. For example, *P2* described his strategies as follows, "*Thought a bit about what could boost performance, then I did some trial and error (manual feed-forward selection of attributes)*."

Need for more powerful system features for expert users: Participants with visual analytics backgrounds expected editable visualizations to represent or encode attribute distribution differently. "*It would be cool to add additional ways of visualizing attributes*" (*P2*). What's more, participants with database backgrounds wanted more details about queries of fetching data to increase flexibility. "*Maybe consider providing the real query of fetching the dataset to the expert users to give them more clear sense and allow them to change the query*" (*P1*). These suggestions are valuable for guiding our further improvement in generalizing CAVA to tackle various problem domains.

8.6 Limitations

While our user study results **support H1** but **neither accept nor reject H2**, the results should be read in the light of the study limitations. The number of the participants for our study are limited, which might be a confounding point of the results. As we conducted the study online, the uncertainty existing in the online setup, such as internet connection, might also confound the results. To improve the performance of the system and the fluidity of user interaction, we limited the number of related attributes that participants could fetch for each parent attribute. Although it did help participants with a smoother user experience during the study, participants also complained about the restriction as it limited their performance of tasks, especially Task 2. Despite the limitations, the user study does help us to better identify the limitations in our system design and improve the system with new features.

9 Discussion

Through our user study and usage scenarios, we have demonstrated the efficacy of information foraging using knowledge graphs within visual analytics systems. The generality of CAVA suggests that data augmentation could be added into traditional visual analytics system workflows to improve the outcome of any embedded task. Many fruitful avenues of research arise when considering how to apply information foraging to the full space of usage scenarios which are served by visual analytics systems.

9.1 Mental Models of Data Augmentation

Knowledge graphs can be difficult to reason about, especially when they are used to find data corresponding to an entire column of a dataset. In CAVA, we address this by showing the user previews of the joined dataset, as well as visualizations of the distribution of joined data and an example of the portion of the knowledge graph that is used to construct a single value (see Figure 3). This approach was based on our conversations with designers of visual analytics applications for building predictive models. A better understanding of the user's mental model of the knowledge graph and the joining process is needed to generalize this approach to other usage scenarios. We generally aimed to hide the complexity of the underlying knowledge graph, but it may be the case that more direct exploration of the knowledge graph is helpful for some cases.

9.2 Sparsity in Knowledge Graphs

In this work, we rely on the high quality of Wikidata to find attributes that have sufficient support for the user's dataset. But in many cases, only 10-15% of the rows of a dataset will have joinable data for a given attribute. Because knowledge graphs do not hold schematic data, they can suffer from data sparsity issues. We feel that the experiments in this work show that there are many popular data types, including geographic entities like countries or states, that have sufficient robust data to discover attributes with full support on a user's dataset. In situations where the attribute of interest is only present for a small percent of rows of the dataset, we suggest two possible solutions. First, the additional data can still be useful as additional information injected into the dataset, and using imputation or setting reasonable default values (using a tool such as the *data tamer* [71] can make that information usable by downstream analyses. Second, a user can explore the reduced dataset in which the sparse attribute is present, and if it results in promising analysis, they can use CAVA to search for other more complete attributes that might be correlated with that sparse attribute, based on their domain expertise. Lastly, we would like to point out that knowledge graphs are not read-only structures, and CAVA may help point out where it would be valuable for an organization to invest in recording more data.

9.3 Design Space for Interactions with Knowledge Graphs

In CAVA, the channel between the user and the knowledge graph is limited to the two subroutines shown in section 5.2, which allow the user to see lists of related attributes and then construct queries given that information. However, there is more potential to improve the user's control over the process and address edge cases by expanding the set of interactions between user and knowledge graph. Automated processes in CAVA could be replaced by collaborations between user and system.

For example, ambiguities in the entity resolution used to retrieve a list of related attributes can be solved by user interaction. A list of countries could refer to the governmental entities they describe, or they could refer to the national soccer teams participating in the World Cup; there will always exist cases where disambiguating the type of an attribute will require domain expertise from a user.

The user could also benefit from more fine-grained control over the process of building aggregation queries. Users may want to join timestamped data, which would necessitate some specification from the user of how to parse and interpret temporal columns. Spatial data offers an additional potential, as the user might want to use geographical data in their dataset to search for the closest weather station or other geographically tagged entity. There are many types of queries that might necessitate different user interactions than have been used in previous visual analytics systems.

10 Conclusion

In this work, we presented CAVA, a visual analytics system for explorative information foraging using knowledge graphs. Knowledge graphs offer a wealth of information on a broad array of topics that could be used to improve the outcome of embedded tasks of visual analytics systems. In our usage scenarios, we demonstrated that the data gathered through CAVA could result in better predictive models and better insight generation on two datasets. And in our experiment, we showed that CAVA is simple to learn and use, as all six of our participants were able to effectively explore and join data on multiple datasets within a 60 minute session.

Acknowledgments

We thank our collaborators in DARPA's Data Driven Discovery of Models (d3m) program, as well as the reviewers for their helpful feedback. This work was supported by National Science Foundation grants IIS-1452977, OAC-1940175, OAC-1939945, DGE-1855886, and 1841349, as well as DARPA grant FA8750-17-2-0107.

References

[1] Uniprot: the universal protein knowledgebase. *Nucleic acids research*, 45(D1):D158–D169, 2017.

[2] IMDB 5000 Movie Dataset. `https://www.kaggle.com/carolzhangdc/imdb-5000-movie-dataset`, 2018. Online; accessed 10-Mar-2020.

[3] Small Area Income And Poverty Estimates (SAIPE) Program state and county estimates. https://www.census.gov/programs-surveys/saipe/data.html, 2018. Online; accessed 10-Mar-2020.

[4] Us unemployment rate by county, 1990-2016. https://www.kaggle.com/jayrav13/unemployment-by-county-us#output.csv, 2020. Online; accessed 10-Mar-2020.

[5] B. Alsallakh, W. Aigner, S. Miksch, and M. E. Groller. Reinventing the contingency wheel: Scalable visual analytics of large categorical data. *IE EE Transactions on Visualization and Computer Graphics*, 18(12):2849–2858, Dec. 2012. doi: 10.1109/TVCG.2012.254

[6] K. Annervaz, S. B. R. Chowdhury, and A. Dukkipati. Learning beyond datasets: Knowledge graph augmented neural networks for natural language processing. *arXiv preprint arXiv:1802.05930*, 2018.

[7] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. Dbpedia: A nucleus for a web of open data. In *The semantic web*, pp. 722–735. Springer, 2007.

[8] N. Balasubramanian and S. Ccucerzan. Beyond ranked lists in web search: Aggregating web content into topic pages. *International Journal of Semantic Computing*, 04(04):509–534, 2010. doi: 10.1142/S1793351X10001103

[9] J. Berant, A. Chou, R. Frostig, and P. Liang. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1533–1544. Association for Computational Linguistics, Seattle, Washington, USA, Oct. 2013.

[10] J. Bernard, M. Steiger, S. Widmer, H. Lücke-Tieke, T. May, and J. Kohlhammer. Visual-interactive exploration of interesting multivariate relations in mixed research data sets. In *Proceedings of the 16th Eurographics Conference on Visualization*, EuroVis '14, p. 291–300. Eurographics Association, Goslar, DEU, 2014.

[11] C. S. Bhagavatula, T. Noraset, and D. Downey. Methods for exploring and mining tables on wikipedia. In *Proceedings of the ACM SIGKDD Workshop on Interactive Data Exploration and Analytics*, pp. 18–26, 2013.

[12] D. Bohn. Google home: a speaker to finally take on the amazon echo. https://www.theverge.com/2016/5/18/11688376/google-home-speaker-announced-virtual-assistant-io-2016, 2016. Retrieved April 30, 2020.

[13] M. J. Cafarella, A. Halevy, and N. Khoussainova. Data integration for the relational web. *Proceedings of the VLDB Endowment*, 2(1):1090–1101, 2009.

[14] B. Chan, L. Wu, J. Talbot, M. Cammarano, and P. Hanrahan. Vispedia: Interactive visual exploration of wikipedia data via search-based integration. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1213–1220, 2008.

[15] M. Chen and D. S. Ebert. An ontological framework for supporting the design and evaluation of visual analytics systems. In *Computer Graphics Forum*, vol. 38, pp. 131–144. Wiley Online Library, 2019.

[16] I. Cho, W. Dou, D. X. Wang, E. Sauda, and W. Ribarsky. Vairoma: A visual analytics system for making sense of places, times, and events in roman history. *IEEE transactions on visualization and computer graphics*, 22(1):210–219, 2015.

[17] N. Cramer, G. Nakamura, and A. Endert. The impact of streaming data on sensemaking with mixed-initiative visual analytics. In *International Conference on Augmented Cognition*, pp. 478–498. Springer, 2017.

[18] M. Dallachiesa, A. Ebaid, A. Eldawy, A. Elmagarmid, I. F. Ilyas, M. Ouzzani, and N. Tang. Nadeef: a commodity data cleaning system. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, pp. 541–552, 2013.

[19] J. Dalton, L. Dietz, and J. Allan. Entity query feature expansion using knowledge base links. In S. Geva, A. Trotman, P. Bruza, C. L. A. Clarke, and K. Järvelin, eds., *The 37th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '14, Gold Coast , QLD, Australia - July 06 - 11, 2014*, pp. 365–374. ACM, 2014. doi: 10.1145/2600428.2609628

[20] A. Das Sarma, L. Fang, N. Gupta, A. Halevy, H. Lee, F. Wu, R. Xin, and C. Yu. Finding related tables. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, pp. 817–828, 2012.

[21] T. Dettmers, P. Minervini, P. Stenetorp, and S. Riedel. Convolutional 2d knowledge graph embeddings. In S. A. McIlraith and K. Q. Weinberger, eds., *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pp. 1811–1818. AAAI Press, 2018.

[22] X. Dong, E. Gabrilovich, G. Heitz, W. Horn, N. Lao, K. Murphy, T. Strohmann, S. Sun, and W. Zhang. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 601–610, 2014.

[23] L. Ehrlinger and W. Wöß. Towards a definition of knowledge graphs. *SEMANTiCS (Posters, Demos, SuCCESS)*, 48, 2016.

[24] R. C. Fernandez, Z. Abedjan, F. Koko, G. Yuan, S. Madden, and M. Stonebraker. Aurum: A data discovery system. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*, pp. 1001–1012. IEEE, 2018.

[25] P. Ferragina and U. Scaiella. Tagme: On-the-fly annotation of short text fragments (by wikipedia entities). In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10, p. 1625–1628. Association for Computing Machinery, New York, NY, USA, 2010. doi: 10.1145/1871437.1871689

[26] E. Gabrilovich, M. Ringgaard, and A. Subramanya. Facc1: Freebase annotation of clueweb corpora, version 1 (release date 2013-06-26, format version 1, correction level 0). 06 2013.

[27] D. Hakkani-Tür, A. Çelikyilmaz, L. P. Heck, G. Tür, and G. Zweig. Probabilistic enrichment of knowledge graph entities for relation detection in conversational understanding. In H. Li, H. M. Meng, B. Ma, E. Chng, and L. Xie, eds., *INTERSPEECH 2014, 15th Annual Conference of the International Speech Communication Association, Singapore, September 14-18, 2014*, pp. 2113–2117. ISCA, 2014.

[28] A. Halevy, F. Korn, N. F. Noy, C. Olston, N. Polyzotis, S. Roy, and S. E. Whang. Goods: Organizing google's datasets. In *Proceedings of the 2016 International Conference on Management of Data*, pp. 795–806, 2016.

[29] S. G. Hart and L. E. Staveland. Development of nasa-tlx (task load index): Results of empirical and theoretical research. In *Advances in psychology*, vol. 52, pp. 139–183. Elsevier, 1988.

[30] B. Hecht, S. H. Carton, M. Quaderi, J. Schöning, M. Raubal, D. Gergle, and D. Downey. Explanatory semantic relatedness and explicit spatialization for exploratory search. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '12, p. 415–424. Association for Computing Machinery, New York, NY, USA, 2012. doi: 10.1145/2348283.2348341

[31] B. Hixon, P. Clark, and H. Hajishirzi. Learning knowledge graphs for question answering through conversational dialog. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 851–861, 2015.

[32] O. Hoeber, A. Sarkar, A. Vacariu, M. Whitney, M. Gaikwad, and G. Kaur. Evaluating the value of lensing wikipedia during the information seeking process. In *Proceedings of the 2017 Conference on Conference Human Information Interaction and Retrieval*, CHIIR '17, p. 77–86. Association for Computing Machinery, New York, NY, USA, 2017. doi: 10.1145/3020165.3020178

[33] P. Hoefler, M. Granitzer, E. E. Veas, and C. Seifert. Linked data query wizard: A novel interface for accessing sparql endpoints. In *LDOW*, 2014.

[34] J. Hoffart, D. Milchevski, and G. Weikum. Stics: Searching with strings, things, and cats. 07 2014. doi: 10.1145/2600428.2611177

[35] R. Hoffmann, C. Zhang, X. Ling, L. Zettlemoyer, and D. S. Weld. Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 541–550. Association for Computational Linguistics, Portland, Oregon, USA, June 2011.

[36] A. Hogan, E. Blomqvist, M. Cochez, C. d'Amato, G. de Melo, C. Gutierrez, J. E. L. Gayo, S. Kirrane, S. Neumaier, A. Polleres, R. Navigli, A.-C. N. Ngomo, S. M. Rashid, A. Rula, L. Schmelzeisen, J. Sequeda, S. Staab, and A. Zimmermann. Knowledge graphs, 2020.

[37] X. Huang, J. Zhang, D. Li, and P. Li. Knowledge graph embedding based question answering. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pp. 105–113, 2019.

[38] A. B. Jung, K. Wada, J. Crall, S. Tanaka, J. Graving, C. Reinders, S. Yadav, J. Banerjee, G. Vecsei, A. Kraft, Z. Rui, J. Borovec, C. Vallentin, S. Zhydenko, K. Pfeiffer, B. Cook, I. Fernández, F.-M. De Rainville, C.-H. Weng, A. Ayala-Acevedo, R. Meudec, M. Laporte, et al. imgaug. https://github.com/aleju/imgaug, 2020. Online; accessed 19-Mar-2020.

[39] M. Kahng, S. B. Navathe, J. T. Stasko, and D. H. P. Chau. Interactive browsing and navigation in relational databases. *Proc. VLDB Endow.*, 9(12):1017–1028, Aug. 2016. doi: 10.14778/2994509.2994520

[40] J. M. Kanter and K. Veeramachaneni. Deep feature synthesis: Towards automating data science endeavors. In *2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 1–10. IEEE, 2015.

[41] P. Konda, S. Das, P. Suganthan GC, A. Doan, A. Ardalan, J. R. Ballard, H. Li, F. Panahi, H. Zhang, J. Naughton, et al. Magellan: Toward building entity matching management systems. *Proceedings of the VLDB Endowment*, 9(12):1197–1208, 2016.

[42] A. Krishnan, D. P., S. Ranu, and S. Mehta. Leveraging semantic resources in diversified query expansion. *World Wide Web*, 21(4):1041–1067, July 2018. doi: 10.1007/s11280-017-0468-7

[43] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105, 2012.

[44] R. Likert. A technique for the measurement of attitudes. *Archives of psychology*, 1932.

[45] X. Liu and H. Fang. Latent entity space: A novel retrieval approach for entity-bearing queries. *Inf. Retr.*, 18(6):473–503, Dec. 2015. doi: 10.1007/s10791-015-9267-x

[46] M. Lynley. Google unveils google assistant, a virtual assistant that's a big upgrade to google now. `https://techcrunch.com/2016/05/18/google-unveils-google-assistant-a-big-upgrade-to-google-now/`, 2016. Retrieved April 30, 2020.

[47] K. Marino, R. Salakhutdinov, and A. Gupta. The more you know: Using knowledge graphs for image classification. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20–28, July 2017. doi: 10.1109/CVPR.2017.10

[48] J. P. McCrae. The linked open data cloud. `https://lod-cloud.net/`, 2020. Online; accessed 22-Apr-2020.

[49] G. A. Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[50] R. J. Miller. Open data integration. *Proceedings of the VLDB Endowment*, 11(12):2130–2139, 2018.

[51] M. Mintz, S. Bills, R. Snow, and D. Jurafsky. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2*, ACL '09, p. 1003–1011. Association for Computational Linguistics, USA, 2009.

[52] A. Mohamed, G. Abuoda, A. Ghanem, Z. Kaoudi, and A. Aboulnaga. Rdfframes: Knowledge graph access for machine learning tools. *arXiv preprint arXiv:2002.03614*, 2020.

[53] D. Moussallem, A.-C. Ngonga Ngomo, P. Buitelaar, and M. Arcan. Utilizing knowledge graphs for neural machine translation augmentation. In *Proceedings of the 10th International Conference on Knowledge Capture*, K-CAP '19, p. 139–146. Association for Computing Machinery, New York, NY, USA, 2019. doi: 10.1145/3360901.3364423

[54] S. Mudgal, H. Li, T. Rekatsinas, A. Doan, Y. Park, G. Krishnan, R. Deep, E. Arcaute, and V. Raghavendra. Deep learning for entity matching: A design space exploration. In *Proceedings of the 2018 International Conference on Management of Data*, pp. 19–34, 2018.

[55] F. Nanni, S. P. Ponzetto, and L. Dietz. Entity-aspect linking: Providing fine-grained semantics of entities in context. In *Proceedings of the 18th ACM/IEEE on Joint Conference on Digital Libraries*, JCDL '18, p. 49–58. Association for Computing Machinery, New York, NY, USA, 2018. doi: 10.1145/3197026.3197047

[56] D. S. Park and W. Chan. SpecAugment. `https://ai.googleblog.com/2019/04/specaugment-new-data-augmentation.html`, 2020. Online; accessed 19-Mar-2020.

[57] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[58] L. Perez and J. Wang. The effectiveness of data augmentation in image classification using deep learning. *arXiv preprint arXiv:1712.04621*, 2017.

[59] P. Pirolli and S. Card. Information foraging. *Psychological review*, 106(4):643, 1999.

[60] P. Pirolli and S. Card. The sensemaking process and leverage points for analyst technology as identified through cognitive task analysis. In *Proceedings of international conference on intelligence analysis*, vol. 5, pp. 2–4. McLean, VA, USA, 2005.

[61] C. Raleigh, A. Linke, H. Hegre, and J. Karlsen. Introducing acled: an armed conflict location and event dataset: special data feature. *Journal of peace research*.

[62] D. Rao, P. McNamee, and M. Dredze. *Entity Linking: Finding Extracted Entities in a Knowledge Base*, pp. 93–115. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013. doi: 10.1007/978-3-642-28569-1_5

[63] H. Raviv, O. Kurland, and D. Carmel. Document retrieval using entity-based language models. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '16, p. 65–74. Association for Computing Machinery, New York, NY, USA, 2016. doi: 10.1145/2911451.2911508

[64] S. Reddy, D. Raghu, M. M. Khapra, and S. Joshi. Generating natural language question-answer pairs from a knowledge graph using a RNN based question generation model. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pp. 376–385. Association for Computational Linguistics, Valencia, Spain, Apr. 2017.

[65] S. Riedel, L. Yao, and A. McCallum. Modeling relations and their mentions without labeled text. In J. L. Balcázar, F. Bonchi, A. Gionis, and M. Sebag, eds., *Machine Learning and Knowledge Discovery in Databases*, pp. 148–163. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[66] D. Sacha, M. Kraus, D. A. Keim, and M. Chen. Vis4ml: An ontology for visual analytics assisted machine learning. *IEEE transactions on visualization and computer graphics*, 25(1):385–395, 2018.

[67] C. Shorten and T. M. Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):60, 2019.

[68] A. Singhal. Introducing the Knowledge Graph: Things, not Strings. `https://googleblog.blogspot.com/2012/05/introducing-knowledge-graph-things-not.html`, May 2012. Online; accessed 22-Apr-2020.

[69] G. Smith, M. Czerwinski, B. Meyers, D. Robbins, G. Robertson, and D. S. Tan. Facetmap: A scalable search and browse visualization. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):797–804, Sept. 2006. doi: 10.1109/TVCG.2006.142

[70] Y. Song and D. Roth. Machine learning with world knowledge: The position and survey. *arXiv preprint arXiv:1705.02908*, 2017.

[71] M. Stonebraker, D. Bruckner, I. F. Ilyas, G. Beskales, M. Cherniack, S. B. Zdonik, A. Pagan, and S. Xu. Data curation at scale: the data tamer system. In *Cidr*, vol. 2013, 2013.

[72] Z. Wang, J. Zhang, J. Feng, and Z. Chen. Knowledge graph and text jointly embedding. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1591–1601. Association for Computational Linguistics, Doha, Qatar, Oct. 2014. doi: 10.3115/v1/D14-1167

[73] M. Yakout, K. Ganjam, K. Chakrabarti, and S. Chaudhuri. Infogather: entity augmentation and attribute discovery by holistic matching with web tables. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, pp. 97–108, 2012.

[74] B. Yang, W. Yih, X. He, J. Gao, and L. Deng. Embedding entities and relations for learning and inference in knowledge bases. In Y. Bengio and Y. LeCun, eds., *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[75] W. Zheng, J. X. Yu, L. Zou, and H. Cheng. Question answering over knowledge graphs: question understanding via template decomposition. *Proceedings of the VLDB Endowment*, 11(11):1373–1386, 2018.

[76] E. Zhu, Y. He, and S. Chaudhuri. Auto-join: Joining tables by leveraging transformations. *Proceedings of the VLDB Endowment*, 10(10):1034–1045, 2017.