LEAP: LLM-powered End-to-end Automatic Library for Processing Social Science Queries on Unstructured Data

Chuxuan Hu
UIUC
Urbana, IL
chuxuan3@illinois.edu

Austin Peters
University of Chicago
Chicago, IL
austinpeters@uchicago.edu

Daniel Kang
UIUC
Urbana, IL
ddkang@illinois.edu

ABSTRACT

Social scientists are increasingly interested in analyzing the semantic information (e.g., emotion) of unstructured data (e.g., Tweets), where the semantic information is not natively present. Performing this analysis in a cost-efficient manner requires using machine learning (ML) models to extract the semantic information and subsequently analyze the now structured data. However, this process remains challenging for domain experts.

To demonstrate the challenges in social science analytics, we collect a dataset, QUIET-ML, of 120 real-world social science queries in natural language and their ground truth answers. Existing systems struggle with these queries since (1) they require selecting and applying ML models, and (2) more than a quarter of these queries are vague, making standard tools like natural language to SQL systems unsuited. To address these issues, we develop LEAP, an end-to-end library that answers social science queries in natural language with ML. LEAP filters vague queries to ensure that the answers are deterministic and selects from internally supported and user-defined ML functions to extend the unstructured data to structured tables with necessary annotations. LEAP further generates and executes code to respond to these natural language queries. LEAP achieves a 100% pass @ 3 and 92% pass @ 1 on QUIET-ML, with a $1.06 average end-to-end cost, of which code generation costs $0.02.

PVLDB Reference Format:
Chuxuan Hu, Austin Peters, and Daniel Kang. LEAP: LLM-powered End-to-end Automatic Library for Processing Social Science Queries on Unstructured Data. PVLDB, 18(2): 253 - 264, 2024.
doi:10.14778/3705829.3705843

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/uiuc-kang-lab/leap.

1 INTRODUCTION

With increasingly accessible *unstructured* datasets [18, 55, 56, 113], social scientists are able to answer questions that were previously beyond scope over unstructured data [28, 73], ranging from macroeconomic questions like "Is the public mood correlated or even predictive of economic indicators?" [9] to sociolinguistic questions like "How can conversational behavior reveal power relationships?" [23]. However, domain experts face two major challenges in answering such queries.

The first challenge is that, since this data is unstructured (e.g., raw texts), the semantic information that the domain experts want to analyze (e.g., emotions in the texts) is not readily available [120]. Manually annotating these datasets could cost hundreds to tens of thousands of dollars when using human labor, so social scientists have turned to machine learning (ML) models [6, 17, 26, 30, 36, 108, 120, 126, 128]. However, applying these ML models is demanding: it involves selecting the correct ML functions and mastering their interfaces, as well as determining appropriate function execution orders when multiple annotations are required [41, 109].

The second challenge is that these domain experts must turn their questions (often in natural language) into actual queries, whether using SQL, dataframe libraries, or statistics libraries. However, this process can be difficult because it involves complex data analytic operations that require advanced programming skills, and the natural language is often underspecified (i.e., *vague*).

To highlight these challenges, we created a new dataset, QUIET-ML, that consists of real-world social science research questions, the corresponding unstructured data, and the ground-truth answers to these questions. QUIET-ML covers all the topics in the Stanford SALT lab's survey [126], which identifies core subject areas in social science using ML, as well as the Stanford CS 224C: NLP for Computational Social Science course [117]. Among the 120 queries we collected, over half (61) of them require executing two or more ML models, and over a quarter (33) of them are vague.

Existing tools struggle to answer these queries. For example, natural language to SQL (NL2SQL) systems [22, 35, 64] are unable to handle ML model executions and perform poorly on vague queries with success rates as low as 3% (Section 5.2), even for NL2SQL systems specifically designed for vague queries at the table schema level [8]. Generic applications built on large language models (LLMs) also fall short in answering these domain-specific social science questions, with failure in generating responses in 10 different attempts when querying Q_{11} in QUIET-ML (Table 1): "I want to find the emotion triggers for all posts" [120] over a CSV file containing 1817 posts in plain text using ChatGPT (GPT-4) [78, 79].

To address these issues, we propose LEAP, an end-to-end library that assists social scientists in analyzing unstructured data. With the raw unstructured data and natural language queries as inputs, LEAP automatically parses the natural language, applies the necessary internally supported and user-defined ML functions, and executes the query over the results of the ML models, i.e., structured tables with adequate semantic information.

To accomplish this, LEAP uses LLM to parse the natural language query, decide on the ML functions to apply, and generate the code

Proceedings of the VLDB Endowment, Vol. 18, No. 2 ISSN 2150-8097.
doi:10.14778/3705829.3705843

to perform the necessary analysis. However, effectively using LLM in LEAP requires overcoming several challenges.

First, LEAP must handle the vague queries that prior NL2SQL systems fail to respond to. To do this, LEAP incorporates a filter (Section 4.1) that identifies vague queries, terminates library executions, and suggests specified alternatives to facilitate exploratory processes, which achieves a success rate of 96% on vague queries.

Second, automatically selecting an appropriate ML function chain is difficult, especially in situations involving complex function dependencies. Answering Q_{11} requires first applying the emotion classifier f_{emotion} to the posts and then propagating its outputs to the emotion trigger identifier f_{trigger}. When directly passing Q_{11} as the user message in the function calling interface provided by OpenAI using `gpt-4-0613` [80] with three functions (f_{emotion}, f_{trigger}, and named entity recognizer f_{ner}) as candidates, functions are called in incorrect orders in 10 different attempts.

Third, in terms of efficiency, the query cost is high due to the extensive variety of supported ML functions. Q_{11} takes 9,084 input tokens with LEAP's internally supported function list as candidates, exceeding the token limit of 8,192 supported by `gpt-4-0613` [79].

To improve effectiveness, LEAP integrates a forward planning mechanism that achieves 98% accuracy in identifying ML function chains. In addition, LEAP incorporates doubly linked lists that connect functions with mutual dependencies such as f_{emotion} and f_{trigger} in Q_{11}, which increases the accuracy of queries with implicit function calls from 20% to 87.7%. To improve efficiency, LEAP structures the supported function list into a function tree to pass only the functions in a single leaf node as candidates for the function calling interface and inserts alias check blocks that determine whether the annotations to be generated already exist before executing ML functions to prevent redundant executions, saving query costs by 55%. We introduce the details of these components in Section 4.2.

We demonstrate the performance and cost efficiency of LEAP in Section 5. LEAP achieves 100% pass @ 3 and 92% pass @ 1 across all 120 queries in QUIET-ML. The success rate of LEAP in responding to vague queries is over 30 times higher than existing NL2SQL systems. The average cost per query is $1.06, with only $0.02 spent on code generation. The end-to-end cost, from vague query reformulation and data annotation to code generation and execution, of LEAP is less than 0.1% of what traditional social science research spends on data annotation alone.

2 QUIET-ML: A SOCIAL SCIENCE RESEARCH QUESTION DATASET

We introduce QUIET-ML, a dataset containing social science **q**ueries on **u**nstructured data **i**nvoking **e**xtended **t**ables with **ML** models. QUIET-ML consists of queries that cover all the topics addressed in the Stanford SALT lab's survey [126] and the Stanford CS 224C course [117]. QUIET-ML includes 120 queries, spanning 9 prominent social science domains and 25 popular topics, addressing social science problems across 68 sources. We present the details in Table 1.

Among these 120 queries, 78 are non-vague queries without unspecified numerical values, 9 are non-vague queries with unspecified numerical values, and 33 are vague queries. The 33 vague queries cover the following three common causes of ambiguity in social science research questions:

(1) **Lack of context**: social scientists pose vague queries with unspecified contexts. For instance, a psychologist might ask "Provide cognitive behavioral therapies for these negative thoughts" (Q_{16}), implicitly assuming that "cognitive behavioral therapies" refers to positive reframing [93].
(2) **Data insufficiency**: social scientists formulate vague queries that cannot be answered from available data. For instance, "Is the public mood correlated with, or even predictive of, economic indicators?" (Q_2) [9] is not answerable by simply analyzing the provided Tweets without incorporating relevant economic statistics.
(3) **Informal or unconventional expressions**: natural language queries may include non-rigid expressions. For instance, a social media researcher can ask "I want to predict whether the conversation will get out of hand" (Q_{24}) [122], where "get out of hand" informally denotes "become toxic". While the two can be considered equivalent in everyday conversation, the lack of precision in "get out of hand" poses challenges for computational analysis.

QUIET-ML provides the unstructured data for each query, consisting of 22,323 data points on average. This data covers a diverse range of unstructured data types, from text and PDF documents to videos. On average, each query needs 2.0 semantic annotations for each data point, where 61 out of the 120 queries require two or more annotations. For evaluation, QUIET-ML includes the ground-truth query results on the provided unstructured data.

3 USE CASES

We demonstrate how an end-to-end library helps social scientists handle the three types of queries in QUIET-ML.

Non-vague queries without unspecified numerical values. A media researcher collects Tweets with dog whistles and asks "For each (targeted) persona/in-group, I want to know the number of each type of dog whistles." (Q_3) [75]

The library first loads the Tweets as a single-column table, then proceeds to select and apply ML functions to extend the table with necessary semantic information to answer the query. In this case, $f_{\text{p/i}}$ that identifies the targeted persona/in-group and $f_{\text{dw-type}}$ that classifies dog whistle types should be executed as explicitly stated in the query. However, f_{dw} that extracts the dog whistle terms should first be applied to the Tweets, since the output of f_{dw} implicitly serves as the input of $f_{\text{p/i}}$ and $f_{\text{dw-type}}$. The detailed dependencies can be viewed in Figure 4.

When the table is extended with adequate semantic information, the query is translated into code similar to the following SQL code.

```
SELECT persona_or_ingroup, type, COUNT(*) AS count
FROM table
GROUP BY persona_or_ingroup, type
```

Once the extended table and code are generated, the library executes the code and displays the execution results.

Non-vague queries with unspecified numerical Values. A sociolinguist collects Reddit posts where people argue and aims to filter the posts according to persuasion effect scores by asking "Which posts are persuasive?" (Q_{17}) [108]

Table 1: Social Science Domains and Topics Covered in QUIET-ML

Domain	Topic	Example
Persuasion	Persuadability [108] (Q_1, Q_{41}, Q_{81}, Q_{82})	Recognize the "malleable" cases.
	Persuasiveness [3, 116, 120] (Q_{17}, Q_{37}, Q_{47}, Q_{77})	Which posts are persuasive?
	Attackability [46, 54, 108] (Q_{34}, Q_{74})	I want to see how sentiment affects the attackability of a sentence.
Emotion	Emotion Classification [9, 33, 44, 58, 100] (Q_2, Q_{42})	Is the public mood correlated or even predictive of economic indicators?
	Emotion Triggers [120] (Q_{10}, Q_{11}, Q_{50}, Q_{51})	I want to find the triggers for all posts of emotion that has the maximum quantity.
Social Bias	Dog Whistles [75] (Q_3, Q_{43})	For each persona/in-group, what is the number of each type of dog whistle?
	Hate Speech [30, 63, 72, 95, 96] (Q_6, Q_7, Q_8, Q_9, Q_{46}, Q_{47}, Q_{48}, Q_{49})	Which posts contain hate speech?
	Equality [10, 39, 87] (Q_{38}, Q_{78}, Q_{113})	I want to quantify how the bias of words evolves.
	Offensiveness [5, 11, 20, 21, 32, 97] (Q_{111}, Q_{112})	Are the defense to these texts effective?
Misinformation	Fake News [36, 62] (Q_4, Q_{14}, Q_{44}, Q_{54})	Which news headlines contain misinformation?
	Imaginary Stories [49, 66, 98, 99, 105] (Q_{25-28}, Q_{65-68})	I want to get the imaginary stories generated based on the recalled stories.
	Deceptive Videos [42, 84, 110] (Q_{36}, Q_{39}, Q_{76}, Q_{79})	I want to extract all fake videos.
Attitude	Stance [76] (Q_5, Q_{45})	What is the difference between stance and sentiment?
	Ideology [6, 50, 53, 88] (Q_{29}, Q_{30}, Q_{69}, Q_{70})	I want to find the percentage of right political ideology.
Linguistics	Figurative Language [14, 52, 77, 106] (Q_{12}, Q_{13}, Q_{52}, Q_{53})	Retrieve the explanations of the premise that entails the figurative sentences.
	Dialect Features [26, 127] (Q_{15}, Q_{55})	I want to retrieve posts with the most common dialect features.
	Semantics [85] (Q_{18}, Q_{58})	I want to retrieve example pairs of the same verb but with different semantics.
	Discourse Acts [48, 121] (Q_{19}, Q_{59})	I want to classify comments in online discussions into a set of coarse discourse acts toward the goal of better understanding discussions at scale.
	Echo Chamber Effect [1] (Q_{114}, Q_{115})	Extract tweets with low Echo Chamber Effect.
Psychology	Mental Health [4, 69, 74, 94, 103, 128] (Q_{16}, Q_{20}, Q_{21}, Q_{32}, Q_{33}, Q_{56}, Q_{60}, Q_{61}, Q_{72}, Q_{73})	Provide cognitive-behavioral therapies for these negative thoughts.
	Social Psychology [12, 24, 65, 115, 122] (Q_{22}, Q_{23}, Q_{24}, Q_{62}, Q_{63}, Q_{64})	I want to identify all impolite posts.
Social Roles	Trope [7, 19] (Q_{31}, Q_{71})	I want to find all characters that are chanteuse.
	Relationship [17, 23] (Q_{35}, Q_{40}, Q_{75}, Q_{80})	I want to study how conversational behavior can reveal power relationships.
Legal Services	Documents[102] (Q_{90}, Q_{91}, Q_{94-97}, $Q_{102-110}$, $Q_{116-120}$)	I want to summarize the documents.
	Texts [31] (Q_{83-89}, Q_{92}, Q_{93}, Q_{98-101})	I want to give an overview of the cases.

To decide whether a post is considered "persuasive", the user should specify a persuasion effect score as the criterion. However, the user may be unsure about this value before knowing the score distributions. Therefore, the library issues a warning and lets the user decide whether to proceed with this value remaining unspecified or to input a new query with specified numerical values such as "Which posts have a persuasion effect score > 0.9?"

Regardless of which query the user chooses to proceed with, the library first loads the posts as a single-column table and appends an additional column containing persuasion effect scores from applying the persuasion effect score calculator f_{pe} on the posts. For code generation, the library generates code similar to the following SQL code.

```
SELECT * FROM table
WHERE persuasion_effect_score > X
```

where X is either the user-specified value, if provided, or the library chooses a reasonable value based on the data distribution. Finally, the library executes the code and displays the results.

Vague queries. An economist collects Tweets to extract emotions and analyze their correlation with economic conditions by posing an exploratory query: "Is the public mood correlated with, or even predictive of, economic indicators?" (Q_2) [9]

The library detects the vagueness, terminates the execution, and suggests alternative queries that (1) are specific enough and (2) align with user intent. For example, an alternative query could be "I want to compute the emotion distribution of the posts."

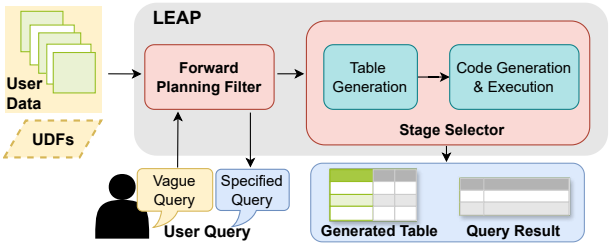

Figure 1: Workflow of LEAP.

With the recommended non-vague query, the library first loads the Tweets as a single-column table and appends an additional column containing emotion classes from applying the emotion classifier f_{emotion} on the posts. The library then generates code similar to the following SQL code.

```
SELECT emotion, COUNT(emotion) AS count
FROM table
GROUP BY emotion
```

Finally, the library executes the code and displays the results.

4 LEAP: AN END-TO-END LIBRARY FOR SOCIAL SCIENCE RESEARCH QUESTIONS

In this section, we propose LEAP, an **L**LM-powered **e**nd-to-end **a**utomatic library for **p**rocessing social science research questions. We provide an overview of the workflow of LEAP in Figure 1.

LEAP takes user-provided data $\mathcal{D}$, a potentially vague query in natural language q on $\mathcal{D}$, and optionally, the user-defined functions (UDFs), as input. It processes these inputs to generate result r in response to q together with $\mathcal{T}$, a structured table that contains adequate semantic information. $\mathcal{D}$ can either be unstructured data like the raw data in QUIET-ML, which LEAP loads as a single-column table, or it can readily be a structured table containing partial or full semantic information.

LEAP consists of (1) a forward planning filter (Section 4.1) that determines if q is vague, and (2) a stage selector (Section 4.2) that selects among candidate stages of *table generation* (Section 4.2.1), *code generation* (Section 4.2.2), *code execution* (Section 4.2.3), and *result display* (Section 4.2.3). We introduce the user interface of LEAP, including the integration of UDFs, in Section 4.3.

4.1 Forward Planning Filter

To prevent users from multiple aimless retries when giving vague queries, LEAP starts with a forward planning filter *FP*. Given the full function list $\mathcal{F}$, which consists of LEAP's internally supported functions and any UDFs, and user-provided data $\mathcal{D}$, *FP* decides whether the user query q is vague through a query check.

FP prompts `gpt-4-0613` [80] and checks (i) whether there exists a well-defined function chain to annotate the data (i.e., whether $\mathcal{D}$ can be extended into deterministic structured tables), (ii) whether q can be answered through a deterministic sequence of SQL operations given a table with adequate information, and (iii) whether there are any unspecified numerical values (i.e., whether q can be translated into executable code). $FP(q, \mathcal{F}, \mathcal{D})$ falls into three cases:

(1) When q is clear in terms of (i), (ii), and (iii), *FP* passes the query check and generates a planned function chain C.
(2) When q satisfies only (i) and (ii), *FP* generates a planned function chain C and a warning message where users can decide whether to proceed or not. If users decide to proceed, *FP* passes the query check and lets the *code generation* (Section 4.2.2) stage handle the unspecified numerical values.
(3) When q does not meet the criteria of either (i) or (ii), *FP* identifies q as a vague query and fails the query check by terminating the execution and generating an alternative query list Q based on $\mathcal{F}$, $\mathcal{D}$, and q.

In cases (1) and (2), *FP* falls back to returning an empty function chain $C = \emptyset$ when $\mathcal{D}$ already contains adequate semantic information, such that q can be directly translated into SQL code. For example, if q extracts Tweets containing a specific `keyword`, it can be answered with code similar to

```
SELECT * FROM table WHERE Tweet LIKE '%keyword%'
```

without applying ML functions to `Tweet`.

We use chain of thoughts (COT) [114], a forward planning prompt technique that decomposes large tasks into small steps, to help *FP* plan for the function chain C, as well as few-shot learning [13], a prompt technique that provides a limited number of examples to guide the LLMs, to improve query recommendations.

4.2 Stage Selector

The user query q that passes *FP* and enters the stage selector is non-vague. The stage selector follows the typical steps of data analysis in social science: first, annotate the raw data with the necessary semantic information; then, write code based on the research question; when the code is ready, execute it in the correct setup; once the result is ready, display it for social scientists to view and evaluate. The stage selector maintains a progress record R to track the executed stages. The stage selector automatically selects the next stage to be executed according to q, the current table, and R. The stage selector enters the *table generation* stage (Section 4.2.1) when the current table is incomplete, and enters the *code generation* stage (Section 4.2.2) otherwise. If given a complete table and executable code, LEAP proceeds to the *code execution* stage (Section 4.2.3). Upon detecting an execution result, LEAP moves to the final *result display* stage (Section 4.2.3). LEAP automatically terminates after entering the same stage three times consecutively, providing detailed feedback for users to refine q. We use the function calling interface provided by OpenAI API with `gpt-4-0613` [80], where each stage is wrapped as an individual function call.

4.2.1 Table Generation. In this stage, LEAP generates a structured table $\mathcal{T}$ and corresponding column descriptions by extending the current table with columns annotated with ML functions, ensuring adequate and correct information is included to answer q (Figure 2).

We use the function calling interface provided by OpenAI API with `gpt-4-0613` [80]. Inspired by the idea of COT [114], we deploy a step-to-step example as guidance. Specifically, we integrate an example query "I want to count the number of positive paragraphs in the PDF document." as part of the prompt. For each progressive status indicated by existing column descriptions, the prompt includes the ground-truth function selections in order (f_{ocr}: OCR

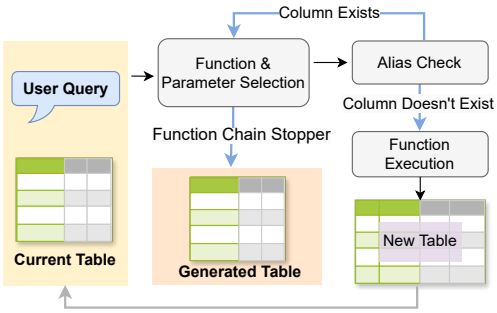

Figure 2: Workflow of the table generation stage.

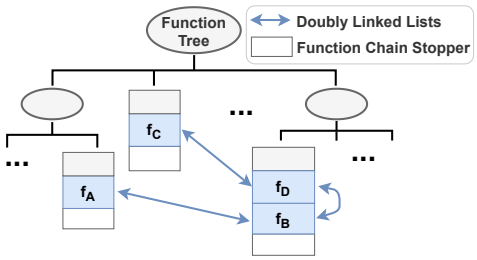

Figure 3: Structure of the supported function list $\mathcal{F}$ in LEAP.

function that translates from pdf files to texts; f_{para}: paragraph separator; $f_{\text{sentiment}}$: sentiment analyzer applied on each paragraph; f_{stopper}: function call chain stopper), together with the ground-truth parameter selections (i.e., the columns to apply these functions).

To further optimize function call accuracy and minimize query costs, the function list $\mathcal{F}$ is organized into a *function tree*, where mutually dependent functions form *doubly linked lists* (Figure 3).

Function tree. To support a wide range of user queries, the function list size $|\mathcal{F}|$ is enormous such that it exceeds the token limit of 8,192 for `gpt-4-0613` even without any UDFs when the entire list becomes candidates. To overcome the issue, we organize $\mathcal{F}$ into a function tree where functions are grouped into subgroups based on their types and stored in leaf nodes (Figure 3). Every leaf node includes a stopper function f_{stopper} for the function chain to end when the current table includes adequate information. LEAP goes through a tree search process based on the hierarchical structure, and only the functions in the targeted leaf node become the candidates for the function calling interface.

Doubly linked lists. The ML function executions can be mutually dependent, but user queries are often implicit with such dependencies. For example, a social media researcher asks "I want to get the readers' actions of headlines that have a high rate of likelihood to spread" (Q_{14}) [37]. To get readers' actions using $f_{\text{reader-action}}$, the writers' intent should first be identified using $f_{\text{writer-intent}}$, and to assess a headline's likelihood of spreading using $f_{\text{spread-likelihood}}$, the reader perception should first be inferred using $f_{\text{reader-perception}}$.

To resolve the issue, we connect dependent functions with doubly linked lists (Figure 3), where a forward pointer $f_A \rightarrow f_B$ represents that the outputs of f_A is necessary for the execution of f_B, and a backward pointer $f_C \leftarrow f_D$ represents that the execution of f_D depends on the outputs of f_C. This design reveals the underlying relationships among functions, and connects different leaf nodes based on function dependencies, resulting in a more accurate tree path search. The doubly linked lists allow users to obtain the correct results without having to know or explicitly state the implicit function dependencies, which are easy to neglect.

Alias check blocks. ML functions can be repeatedly called, which is especially common in cases where (1) complex function dependencies exist, and/or (2) $\mathcal{D}$ contains information derived from ML functions. For example, a literature scholar can ask "I want to get the imaginary stories generated based on the recalled stories." [98]. To do so, one first summarizes the provided recalled stories using the summarizer f_1, and then generates imaginary stories based on the summaries using the story generator f_2. Given that summarizing texts is easier and cheaper than generating them, users can readily provide the summaries of the recalled stories, i.e., the outputs of f_1, in $\mathcal{D}$. However, due to the function dependencies indicated by $f_1 \leftarrow f_2$, where the execution of f_2 depends on the outputs of f_1, executing f_2 can result in repeated calls to f_1. Although the correctness of the final results is not affected by such repetitive execution, it is inefficient both in resources, since ML functions are expensive to execute, and in time, since users have to wait for additional rounds of ML function execution time.

To avoid this, LEAP inserts an alias check block before the execution of each function to determine if any column c_{current} already contains information that exactly matches the outputs to be generated. In such cases, columns are aliased as $c_{\text{new}} = c_{\text{current}}$ instead of generating new ones. This also generates more organized tables $\mathcal{T}$, preventing confusion in subsequent stages.

4.2.2 Code Generation. In this stage, LEAP translates the users' natural language query q into executable code on the extended table $\mathcal{T}$. The code generator generates code that assigns the final execution result to a user-defined variable. The code generator is an NL2SQL system using `gpt-4-0613` [80]. It takes the column descriptions and, if any, sample values of $\mathcal{T}$ along with q as input, and produces executable code that answers q on $\mathcal{T}$ as output. If q contains unspecified numerical values, the code generator automatically chooses a reasonable value based on the data distribution.

4.2.3 Code Execution and Result Display. With the extended table $\mathcal{T}$ and corresponding code prepared to address user queries, LEAP directly executes the code and displays the execution results. The code executor executes the generated code on the extended table $\mathcal{T}$. The result display function reads and displays the result.

4.3 User Interface

After installing LEAP via commands like `pip install`, which are typically familiar to users, they can initiate and execute the entire process with a single function signature:

```
result, table = leap(query, data, description),
```

where `result` is the response to the user query in natural language `query` based on the user-provided data `data`, and `table` is the table containing adequate semantic information annotated by ML functions. `description` contains descriptions of `data`.

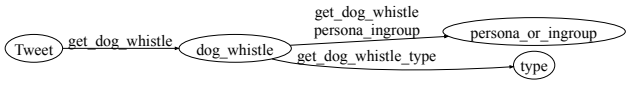

Figure 4: LEAP table generation stage's display for Q_3: "For each (targeted) persona/in-group, I want to know the number of each type of dog whistle." [75]

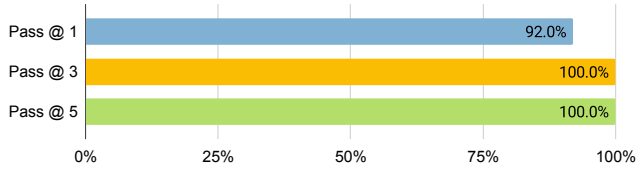

Figure 5: Pass @ k of LEAP over QUIET-ML.

LEAP displays the current stage during its execution to indicate the progress. During the table generation stage, LEAP dynamically updates the column mapping relation graph (Figure 4).

To include UDFs to answer a query, users simply need to pass a list containing function metadata, `udf_metadata`, as a parameter in the `leap` function. LEAP integrates UDFs with the internally supported function list and executes them in user space.

We invited a legal researcher to use LEAP for their research. They found that LEAP streamlines legal document annotations and enhances time efficiency for vague queries. They also found LEAP easy to install, load, and use with an intuitive interface.

5 EXPERIMENTS

In this section, we evaluate the applicability of LEAP and its components. We introduce the experiment setup in Section 5.1. We demonstrate the applicability of LEAP in social science research in Section 5.2. We explore the effectiveness of the critical components in Section 5.3 and conduct ablation studies in Section 5.4.

5.1 Experiment Setup

We outline the metrics and baselines for evaluation.

Metrics. We use *pass @ k* [16, 59], where a query is considered correctly answered if the result from any of the k executions matches the ground truth, as our primary performance metric. We select $k = 1, 3, 5$. With the goal of improving efficiency in social science research, we place a greater emphasis on $k = 1$. We also track the average API cost for all requests in answering each query [82].

Baselines. To demonstrate the superiority of LEAP compared to existing NL2SQL and question answering systems, we select 9 representative baselines. We select **(1)** TAPAS large model fine-tuned on WikiTable Questions [29, 40, 45, 83] (TAPAS), the state of the art for table question answering tasks. WikiSQL [125] and Spider [119] are the two major datasets for training and evaluating NL2SQL systems. We select **(2)** TAPEX [68], the state of the art on WikiSQL, **(3)** T5 finetuned on WikiSQL (T5-WikiSQL) [90, 92], the most downloaded model on Huggingface as of 02/25/2024 finetuned on WikiSQL for NL2SQL tasks, **(4)** DAIL-SQL + GPT-4 + Self-Consistency (DAIL-SQL) [38], the state of the art on Spider, and **(5)** T5 finetuned on Spider (T5-Spider) [2, 90], the most downloaded model on Huggingface as of 02/25/2024 finetuned on Spider for NL2SQL tasks. To compare the performance of LEAP with existing NL2SQL systems designed for vague queries, we select **(6)** LogicalBeam [8], which successfully handles vague queries at the table schema level. To show the effectiveness of the design of LEAP, we select **(7)** `gpt-4-0613` (GPT-4), the LLM prompted by LEAP's code generator, including a performance comparison with **(8)** `gpt-3.5-turbo-0125` (GPT-3.5), both prompted with the official NL2SQL prompt [81]. To compare LEAP with question answering systems on unstructured data, we select **(9)** RoberTa + Parallel + Adapters (Roberta) [15, 70], the latest model on SQuAD2.0 [91], a benchmark dataset consisting of question and answer pairs, including unanswerable questions, based on Wikipedia articles.

5.2 LEAP Achieves Reliable Performance With Low Cost

We demonstrate the performance and cost efficiency of LEAP.

LEAP achieves a 92% accuracy. We first examine the performance of LEAP in various social science domains. We run LEAP 5 times over each of the 120 queries in QUIET-ML to obtain pass @ 3, pass @ 5, and the average pass @ 1. For vague queries, a run is successful if LEAP correctly rejects the query and recommends alternatives, one of which yields a result that matches the ground truth. For non-vague queries with unspecified numerical values, a run is successful if LEAP generates a warning, and the result of the query with specified numerical values matches the ground truth.

As we show in Figure 5, LEAP achieves an average of 92% pass @ 1, where 79 out of 120 queries achieve a 5 out of 5 success rate, 34 achieve a 4 out of 5 success rate, and the remaining 7 achieve a 3 out of 5 success rate. The average pass @ 1 achieves 91.5% for non-vague queries, and 93.3% for vague queries. LEAP achieves 100% for both pass @ 3 and pass @ 5. LEAP consistently performs well despite the large variance in query complexity across different social science research questions.

LEAP outperforms baselines in responding to vague queries. We compare the performance of LEAP to 7 NL2SQL systems and 2 question answering systems on vague queries. For LEAP and baselines (1)-(8), we run the 33 vague queries in QUIET-ML on the pre-generated structured tables with adequate information. Roberta is provided with texts following its original configuration. We perform 5 runs for each query.

As we show in Figure 6, LEAP succeeds in producing correct results in 96.97% of all runs, GPT-3.5 succeeds in 41.21% of all runs, GPT-4 succeeds in 39.39% of all runs, DAIL-SQL succeeds in 29.09% of all runs, TAPAS succeeds in 24.24% of all runs, and TAPEX, T5-Spider, LogicalBeam, and T5-WikiSQL have accuracy of 15.15%, 12.12%, 9.09%, and 3.03% respectively. Roberta has an accuracy of 3.03%, while LEAP achieves an accuracy of 93.3% on the 33 vague queries when also provided with unstructured data.

LEAP outperforms existing systems due to its capability to identify common vagueness in social science queries and generate alternative specified queries. The execution results of LogicalBeam and Roberta indicate that the ambiguity in social science queries

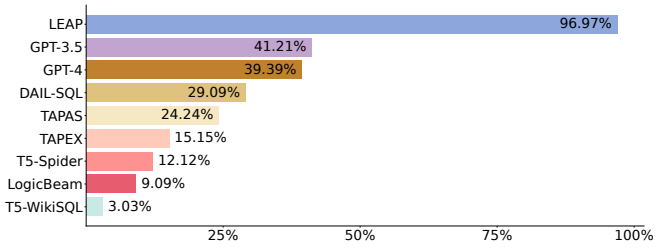

Figure 6: Pass @ 1 of LEAP and baselines over the 33 vague queries in QUIET-ML on structured tables.

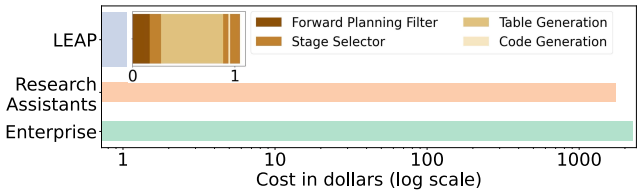

Figure 7: Cost breakdown of LEAP and comparison with traditional social science research methods.

extends beyond the table schema level and a limited number of text segments, where LEAP demonstrates a significant advantage.

LEAP is cost-efficient. We compare the query cost of LEAP with traditional social science research methods. The end-to-end execution cost of LEAP on QUIET-ML queries is $1.06 per query on average. Specifically, the cost of the forward planning filter is $0.17, the stage selector is $0.22, the table generation stage is $0.66, and the code generation stage is $0.02 (Figure 7).

When estimating the costs of traditional social science research methods, we consider only the data annotation costs, excluding all other costs such as labor and time for code writing and execution, and assuming a specified research question is readily formulated. We estimate traditional data annotation costs based on the average size of the unstructured data with 22,323 data points, and on average each query requires 2.0 new annotations for each data point. We calculate the cost in two distinct ways: (1) with professional label providers from enterprises like Scale [101], each annotation costs $0.05, yielding a total cost of $2,265.78 per query; and (2) by hiring research assistants (RAs) at the minimum wage in Illinois of $14 per hour. Assuming it takes 10 seconds for an RA to provide an annotation, which is conservative given the extensive volume of videos and documents, it costs $1,736.23 per query.

As we show in Figure 7, it costs less than 1/1000 to answer a query using LEAP than using traditional social science research methods. This low cost results from LEAP's efficient prompt designs with various structures, as we demonstrate in Sections 5.3 and 5.4.

5.3 Component Studies

We study the effectiveness of the critical components in LEAP.

Forward planning filter achieves over 96% accuracy. We study the accuracy of the results of the forward planning filter. We first define which results are considered accurate case by case.

(1) For non-vague queries without unspecified numerical values, the forward planning filter should not signal unnecessary warnings or incorrectly label these queries as vague, and should accurately identify function chains.
(2) For non-vague queries with unspecified numerical values, the forward planning filter should issue warnings inquiring whether the user intends to proceed, and should accurately identify function chains.
(3) For vague queries, the forward planning filter should detect the vagueness and terminate the execution. The forward planning filter should recommend alternative queries, at least one of which satisfies condition (1) or (2).

We run LEAP 5 times for each query, and the accuracy for all 120 queries achieves 96.5%. The accuracy for the non-vague queries without unspecified numerical values achieves 96.9%, the accuracy for the non-vague queries with unspecified numerical values achieves 97.8%, and the accuracy for the vague queries achieves 95.8%. The accuracy for vague queries is lower due to the uncertainty introduced by generating specified queries, including those that do not match the user's original intentions.

Stage selector achieves 99% accuracy. We study the accuracy of the stage selector. A run is accurate if all necessary stages are selected and executed without errors or repetitions. We run LEAP 5 times over all 120 queries. The accuracy of the stage selector achieves 98.5%. Out of the 9 failure cases, only 1 is due to redundant selection, and the remaining 8 are due to inaccurate selection or erroneous execution. The stable performance of the stage selector ensures the workflow of LEAP is logically and efficiently divided.

Function selection achieves 98% accuracy. To evaluate the capability of LEAP in calling appropriate functions, we study the accuracy of function selection. The function selection of a query is considered accurate if (1) the function tree search engine identifies correct paths to the tree leaves, (2) the correct functions are called in appropriate orders, and (3) the function chain stopper f_{stopper} is correctly called. We run LEAP 5 times over all 120 queries. The accuracy of function selection achieves 97.8%. The prompt designs for function calling in LEAP's table generation stage ensure correct data annotations in social science research.

Parameter selection achieves 99% accuracy. We further explore LEAP's capability in mapping relationships among columns by analyzing the accuracy of parameter selection. The parameter selection is considered accurate if the correct columns are selected to derive the new columns. We run LEAP 5 times for all 120 queries. The accuracy of parameter selection achieves 98.8%. The table column descriptions are properly maintained during LEAP's table generation stage to ensure correct column mappings.

Alias check blocks work with forward planning filter to ensure 0% redundancy. We study the capability of LEAP to minimize unnecessary ML function executions. We select the 20 queries in QUIET-ML that require multiple function calls with mutual dependencies, and extend the data with intermediate annotations. For tests in Section 5.2 that provide unstructured data `X`, and where the ML functions annotate this data into a table with columns `[X,` f_1`(X),` $f_2(f_1$`(X))]`, we provide a structured table with columns

[X, Y], where Y = f_1(X). We examine whether the ML functions that generate these intermediate annotations (i.e., the f_1's) are redundantly executed. We run each of these 20 queries 5 times.

Of the 94 runs (accuracy 94%) that produces correct results, all (100%) successfully avoid redundant ML function executions. The forward planning filter avoids the calling of these functions by excluding them from the planned function chains in 54.3% of the runs, and alias check blocks avoid the execution of these functions in all the remaining 45.7% when they are redundantly called. In 11 out of 20 queries, alias check blocks avoid the execution of unnecessary ML functions in at least 3 out of the 5 total runs. LEAP avoids all (100%) of the unnecessary function executions when given multiple intermediate columns. By accurately identifying redundancy in annotations, LEAP minimizes ML function execution costs.

5.4 Ablation Studies

We conduct ablation studies to reveal the impact of each component.

Forward planning filter doubles accuracy. We study how the forward planning filter affects LEAP's performance by removing it and running the same tests as Section 5.2 on all 120 queries.

With the forward planning filter removed, the average pass @ 1 drops from 92% to 43.7%, where only 19 out of 120 queries achieve a 5 out of 5 success rate, which is less than a quarter of the number of LEAP with the forward planning filter. Specifically, the average pass @ 1 for non-vague queries drops from 91.5% to 54.7%, and the average pass @ 1 for vague queries drops from 93.3% to 14.5%. The pass @ 3 drops from 100% to 59.2%, where pass @ 3 for non-vague queries drops to 73.6%, and pass @ 3 for vague queries drops to 21.2%. The pass @ 5 drops from 100% to 66.7%, where pass @ 5 for non-vague queries drops to 78.2%, and pass @ 5 for vague queries drops to 36.4%. This indicates that the forward planning filter contributes both to specifying vague queries and identifying complex function chains, with the former having greater influence.

Doubly linked lists raise accuracy by 4×. We examine how doubly linked lists help answer queries that require executing mutually dependent functions. We remove all doubly linked lists and run the same tests as Section 5.2 on all 13 queries whose function chains involve functions in these doubly linked lists.

Doubly linked lists provide explicit information about function dependencies, which otherwise can only be inferred from column descriptions. With all doubly linked lists removed, the average pass @ 1 drops from 87.7% to 20%, the pass @ 3 drops from 100% to 46.2%, and the pass @ 5 drops from 100% to 61.5%.

Function tree halves query costs. We study how the function tree structure lowers query cost. We remove the function tree search engine and pass all internally supported ML functions as candidates for the function calling interface. We measure query costs in terms of the total number of prompt tokens used in table generation, the stage where the function tree structure contributes.

Instead of the entire function list, LEAP only selects from a small subset of functions (i.e., the selected tree leaf nodes) for each function call. This reduction in prompt tokens outweighs the additional prompts used by the tree search engine, especially when multiple function calls (i.e., multiple annotations) are required. By removing the function tree structure, the average number of prompt tokens increases from 20,861 to 46,318, resulting in the total query cost being 122.03% higher than that of the original LEAP with the function tree structure in the table generation stage.

Function tree raises accuracy by 5×. We study how the function tree structure affects LEAP's performance by modifying LEAP as in the previous section and running the same tests as Section 5.2 on all 120 queries.

The function tree's removal impedes the selection of the function chain stopper f_{stopper} among the extensive pool of candidates. The average pass @ 1 drops from 92% to 16.2%, pass @ 3 drops from 100% to 34.2%, and pass @ 5 drops from 100% to 39.2%.

6 RELATED WORK

We review related work from the following two aspects.

Computational social science. As available data scales, social scientists increasingly quantify complex social science problems and leverage computational resources for solutions [61]. This leads to the development of the computational social science field [28, 60], enabling social scientists to derive deep insights from large amounts of data [47, 67, 123]. Additionally, ML models [25] achieve satisfactory performance in quantitatively analyzing social science problems across a wide range of popular domains [6, 17, 26, 30, 36, 108, 120, 126, 128]. However, a significant gap remains between the advancements in ML models and their practical deployment in social science research due to their complexity [41, 109].

NL2SQL. Natural language access to databases is a key area of research [22]. One branch is table question answering systems that understand and answer questions directly based on data presented in tabular form [29, 45, 51, 83, 125]. As database sizes and complexity grow, the scalability of NL2SQL systems makes them more viable options [71]. Existing NL2SQL systems achieve high accuracy [57] by (1) improving the representation of table schema [27, 112], and (2) improving the mapping between the intent of natural language queries and the translated SQL codes [43, 118]. The generalizability of existing NL2SQL systems has also been widely studied [107] and improved [35, 104]. The development of LLMs makes NL2SQL systems more effective for handling complex database queries [64, 86, 124]. Existing NL2SQL systems and benchmarks addressing vague queries focus on the table schema level [8, 34, 111]. However, in actual social science research [19, 62, 76, 94, 98–100, 103, 108, 115, 122], the causes of ambiguity are more complex, impeding the deployment of NL2SQL systems in social science research [89].

7 CONCLUSION

In this work, we present QUIET-ML, a dataset that comprehensively covers 120 popular queries in social science research. Along with QUIET-ML, we introduce LEAP, an end-to-end library designed to support social science research by automatically analyzing user-collected unstructured data in response to their natural language queries. LEAP incorporates a forward planning filter that handles vague queries and generates function chains effectively. By integrating innovative structures such as the function tree, doubly linked lists, and alias check blocks, LEAP achieves 100% pass @ 3 and 92% pass @ 1 with an average end-to-end cost being $1.06 per query on QUIET-ML, significantly outperforming the baselines.

REFERENCES

[1] Faisal Alatawi, Paras Sheth, and Huan Liu. 2023. Quantifying the Echo Chamber Effect: An Embedding Distance-Based Approach. arXiv:2307.04668 [cs]

[2] Gauss Algorithmic. 2023. https://huggingface.co/gaussalgo/T5-LM-Large-text2sql-spider Accessed: February 25, 2024.

[3] Tim Althoff, Cristian Danescu-Niculescu-Mizil, and Daniel Jurafsky. 2014. How to Ask for a Favor: A Case Study on the Success of Altruistic Requests. *ArXiv* abs/1405.3282 (2014). https://api.semanticscholar.org/CorpusID:8809599

[4] Akari Asai, Sara Evensen, Behzad Golshan, Alon Halevy, Vivian Li, Andrei Lopatenko, Daniela Stepanov, Yoshihiko Suhara, Wang-Chiew Tan, and Yinzhan Xu. 2018. HappyDB: A Corpus of 100,000 Crowdsourced Happy Moments. arXiv:1801.07746 [cs.CL]

[5] Mana Ashida and Mamoru Komachi. 2022. Towards Automatic Generation of Messages Countering Online Hate Speech and Microaggressions. In *Proceedings of the Sixth Workshop on Online Abuse and Harms (WOAH)*. Association for Computational Linguistics, Seattle, Washington (Hybrid), 11–23. https://aclanthology.org/2022.woah-1.2

[6] Ramy Baly, Giovanni Da San Martino, James Glass, and Preslav Nakov. 2020. We Can Detect Your Bias: Predicting the Political Ideology of News Articles. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (Eds.). Association for Computational Linguistics, Online, 4982–4991. https://doi.org/10.18653/v1/2020.emnlp-main.404

[7] David Bamman, Brendan O'Connor, and Noah A. Smith. 2013. Learning Latent Personas of Film Characters. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Hinrich Schuetze, Pascale Fung, and Massimo Poesio (Eds.). Association for Computational Linguistics, Sofia, Bulgaria, 352–361. https://aclanthology.org/P13-1035

[8] Adithya Bhaskar, Tushar Tomar, Ashutosh Sathe, and Sunita Sarawagi. 2023. Benchmarking and Improving Text-to-SQL Generation under Ambiguity. arXiv:2310.13659 [cs.CL]

[9] Johan Bollen, Huina Mao, and Xiaojun Zeng. 2011. Twitter mood predicts the stock market. *Journal of Computational Science* 2, 1 (2011), 1–8. https://doi.org/10.1016/j.jocs.2010.12.007

[10] Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. 2016. Man is to Computer Programmer as Woman is to Homemaker? Debiasing Word Embeddings. In *Advances in Neural Information Processing Systems*, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett (Eds.), Vol. 29. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2016/file/a486cd07e4ac3d270571622f4f316ec5-Paper.pdf

[11] Luke Breitfeller, Emily Ahn, David Jurgens, and Yulia Tsvetkov. 2019. Finding Microaggressions in the Wild: A Case for Locating Elusive Phenomena in Social Media Posts. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, Hong Kong, China, 1664–1674. https://doi.org/10.18653/v1/D19-1176

[12] Penelope Brown and Stephen C. Levinson. 1987. *Politeness: Some universals in language usage*. Vol. 4. Cambridge University Press.

[13] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. arXiv:2005.14165 [cs.CL]

[14] Tuhin Chakrabarty, Arkadiy Saakyan, Debanjan Ghosh, and Smaranda Muresan. 2022. FLUTE: Figurative Language Understanding through Textual Explanations. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (Eds.). Association for Computational Linguistics, Abu Dhabi, United Arab Emirates, 7139–7159. https://doi.org/10.18653/v1/2022.emnlp-main.481

[15] Branden Chan, Timo Möller, Malte Pietsch, and Tanay Soni. 2023. https://huggingface.co/deepset/roberta-base-squad2 Accessed: June 23, 2024.

[16] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021. Evaluating Large Language Models Trained on Code. arXiv:2107.03374 [cs.LG]

[17] Minje Choi, Ceren Budak, Daniel M. Romero, and David Jurgens. 2021. More than Meets the Tie: Examining the Role of Interpersonal Relationships in Social Networks. arXiv:2105.06038 [cs.SI]

[18] Eric Chu, Akanksha Baid, Ting Chen, AnHai Doan, and Jeffrey Naughton. 2007. A relational approach to incrementally extracting and querying structure in unstructured data. In *Proceedings of the 33rd international conference on Very large data bases*. 1045–1056.

[19] Eric Chu, Prashanth Vijayaraghavan, and Deb Roy. 2018. Learning Personas from Dialogue with Attentive Memory Networks. arXiv:1810.08717 [cs.CL]

[20] Yi-Ling Chung, Elizaveta Kuzmenko, Serra Sinem Tekiroglu, and Marco Guerini. 2019. CONAN - COunter NArratives through Nichesourcing: a Multilingual Dataset of Responses to Fight Online Hate Speech. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Florence, Italy, 2819–2829. https://doi.org/10.18653/v1/P19-1271

[21] Yi-Ling Chung, Serra Sinem Tekiroğlu, and Marco Guerini. 2021. Towards Knowledge-Grounded Counter Narrative Generation for Hate Speech. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Online.

[22] Ann Copestake and Karen Sparck Jones. 1990. Natural language interfaces to databases. *The Knowledge Engineering Review* 5, 4 (1990), 225–249.

[23] Cristian Danescu-Niculescu-Mizil, Lillian Lee, Bo Pang, and Jon Kleinberg. 2012. Echoes of power: Language effects and power differences in social interaction. In *Proceedings of WWW*. 699–708.

[24] Cristian Danescu-Niculescu-Mizil, Moritz Sudhof, Dan Jurafsky, Jure Leskovec, and Christopher Potts. 2013. A computational approach to politeness with application to social factors. In *Proceedings of ACL*.

[25] Richard D De Veaux and Adam Eck. 2021. Machine Learning Methods for Computational Social Science. *Handbook of Computational Social Science, Volume 2: Data Science, Statistical Modelling, and Machine Learning Methods* (2021).

[26] Dorottya Demszky, Devyani Sharma, Jonathan H. Clark, Vinodkumar Prabhakaran, and Jacob Eisenstein. 2021. Learning to Recognize Dialect Features. arXiv:2010.12707 [cs.CL]

[27] Xiang Deng, Ahmed Hassan Awadallah, Christopher Meek, Oleksandr Polozov, Huan Sun, and Matthew Richardson. 2021. Structure-Grounded Pretraining for Text-to-SQL. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.naacl-main.105

[28] Achim Edelmann, Tom Wolff, Danielle Montagne, and Christopher A. Bail. 2020. Computational Social Science and Sociology. *Annual Review of Sociology* 46, 1 (2020), 61–81. https://doi.org/10.1146/annurev-soc-121919-054621 arXiv:https://doi.org/10.1146/annurev-soc-121919-054621

[29] Julian Martin Eisenschlos, Syrine Krichene, and Thomas Müller. 2020. Understanding tables with intermediate pre-training. arXiv:2010.00571 [cs.CL]

[30] Mai ElSherief, Caleb Ziems, David Muchlinski, Vaishnavi Anupindi, Jordyn Seybolt, Munmun De Choudhury, and Diyi Yang. 2021. Latent Hatred: A Benchmark for Understanding Implicit Hate Speech. arXiv:2109.05322 [cs.CL]

[31] Nora Freeman Engstrom, David Freeman Engstrom, Jonah Gelbach, Austin Peters, and Aaron Schaffer-Neitz. 2024. Secrecy by Stipulation. *Duke Law Journal* (May 2 2024). https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4811151

[32] Margherita Fanton, Helena Bonaldi, Serra Sinem Tekiroğlu, and Marco Guerini. 2021. Human-in-the-Loop for Data Collection: a Multi-Target Counter Narrative Dataset to Fight Online Hate Speech. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

[33] Emilio Ferrara and Zeyao Yang. 2015. Measuring Emotional Contagion in Social Media. *PLOS ONE* 10 (06 2015). https://doi.org/10.1371/journal.pone.0142390

[34] Avrilia Floratou, Fotis Psallidas, Fuheng Zhao, and et al. [n.d.]. NL2SQL is a solved problem... Not! https://www.cidrdb.org/cidr2024/papers/p74-floratou.pdf

[35] Han Fu, Chang Liu, Bin Wu, Feifei Li, Jian Tan, and Jianling Sun. 2023. CatSQL: Towards Real World Natural Language to SQL Applications. *Proceedings of the VLDB Endowment* 16, 6 (2023), 1534–1547.

[36] Saadia Gabriel, Skyler Hallinan, Maarten Sap, Pemi Nguyen, Franziska Roesner, Eunsol Choi, and Yejin Choi. 2022. Misinfo Reaction Frames: Reasoning about Readers' Reactions to News Headlines. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (Eds.). Association for Computational Linguistics, Dublin, Ireland, 3108–3127. https://doi.org/10.18653/v1/2022.acl-long.222

[37] Saadia Gabriel, Skyler Hallinan, Maarten Sap, Pemi Nguyen, Franziska Roesner, Eunsol Choi, and Yejin Choi. 2022. Misinfo Reaction Frames: Reasoning about Readers' Reactions to News Headlines. *ACL* (2022).

[38] Dawei Gao, Haibin Wang, Yaliang Li, Xiuyu Sun, Yichen Qian, Bolin Ding, and Jingren Zhou. 2023. Text-to-SQL Empowered by Large Language Models: A Benchmark Evaluation. *CoRR* abs/2308.15363 (2023).

[39] Nikhil Garg, Londa Schiebinger, Dan Jurafsky, and James Zou. 2018. Word embeddings quantify 100 years of gender and ethnic stereotypes. *Proceedings of the National Academy of Sciences* 115, 16 (April 2018). https://doi.org/10.1073/pnas.1720347115

[40] Google. 2023. https://huggingface.co/google/tapas-large-finetuned-wtq Accessed: February 25, 2024.

[41] Justin Grimmer, Margaret E. Roberts, and Brandon M. Stewart. 2021. Machine Learning for Social Science: An Agnostic Approach. *Annual Review of Political Science* 24, 1 (2021), 395–419. https://doi.org/10.1146/annurev-polisci-053119-015921 arXiv:https://doi.org/10.1146/annurev-polisci-053119-015921

[42] Matthew Groh, Ziv Epstein, Chaz Firestone, and Rosalind Picard. 2021. Deepfake detection by human crowds, machines, and machine-informed crowds. *Proceedings of the National Academy of Sciences* 119, 1 (Dec. 2021). https://doi.org/10.1073/pnas.2110013119

[43] Jiaqi Guo, Zecheng Zhan, Yan Gao, Yan Xiao, Jian-Guang Lou, Ting Liu, and Dongmei Zhang. 2019. Towards Complex Text-to-SQL in Cross-Domain Database with Intermediate Representation. arXiv:1905.08205 [cs.CL]

[44] E. Hatfield, J.T. Cacioppo, and R.L. Rapson. 1993. *Emotional Contagion.* Cambridge University Press. https://books.google.com/books?id=BbA-BAAAQBAJ

[45] Jonathan Herzig, Paweł Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Martin Eisenschlos. 2020. TAPAS: Weakly Supervised Table Parsing via Pre-training. arXiv:2004.02349 [cs.IR]

[46] Christopher Hidey, Elena Musi, Alyssa Hwang, Smaranda Muresan, and Kathy McKeown. 2017. Analyzing the Semantic Types of Claims and Premises in an Online Persuasive Forum. In *Proceedings of the 4th Workshop on Argument Mining*, Ivan Habernal, Iryna Gurevych, Kevin Ashley, Claire Cardie, Nancy Green, Diane Litman, Georgios Petasis, Chris Reed, Noam Slonim, and Vern Walker (Eds.). Association for Computational Linguistics, Copenhagen, Denmark, 11–21. https://doi.org/10.18653/v1/W17-5102

[47] Chuxuan Hu, Qinghai Zhou, and Hanghang Tong. 2024. Genius: Subteam Replacement with Clustering-based Graph Neural Networks. In *Proceedings of the 2024 SIAM International Conference on Data Mining (SDM)*. 10–18. https://doi.org/10.1137/1.9781611978032.2

[48] Pere-Lluís Huguet Cabot, Verna Dankers, David Abadi, Agneta Fischer, and Ekaterina Shutova. 2020. The Pragmatics behind Politics: Modelling Metaphor, Framing and Emotion in Political Discourse. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, Trevor Cohn, Yulan He, and Yang Liu (Eds.). Association for Computational Linguistics, Online, 4479–4488. https://doi.org/10.18653/v1/2020.findings-emnlp.402

[49] Ali Hürriyetoğlu, Hristo Tanev, Vanni Zavarella, Jakub Piskorski, Reyyan Yeniterzi, Osman Mutlu, Deniz Yuret, and Aline Villavicencio. 2021. Challenges and Applications of Automated Extraction of Socio-political Events from Text (CASE 2021): Workshop and Shared Task Report. In *Proceedings of the 4th Workshop on Challenges and Applications of Automated Extraction of Socio-political Events from Text (CASE 2021)*, Ali Hürriyetoğlu (Ed.). Association for Computational Linguistics, Online, 1–9. https://doi.org/10.18653/v1/2021.case-1.1

[50] Mohit Iyyer, Peter Enns, Jordan Boyd-Graber, and Philip Resnik. 2014. Political Ideology Detection Using Recursive Neural Networks. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Kristina Toutanova and Hua Wu (Eds.). Association for Computational Linguistics, Baltimore, Maryland, 1113–1122. https://doi.org/10.3115/v1/P14-1105

[51] Mohit Iyyer, Wen-tau Yih, and Ming-Wei Chang. 2017. Search-based Neural Structured Learning for Sequential Question Answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Regina Barzilay and Min-Yen Kan (Eds.). Association for Computational Linguistics, Vancouver, Canada, 1821–1831. https://doi.org/10.18653/v1/P17-1167

[52] Arthur M. Jacobs and Annette Kinder. 2018. What Makes a Metaphor Literary? Answers From Two Computational Studies. *Metaphor and Symbol* 33, 2 (2018), 85–100. https://doi.org/10.1080/10926488.2018.1434943

[53] Zubin Jelveh, Bruce Kogut, and Suresh Naidu. 2014. Detecting Latent Ideology in Expert Text: Evidence From Academic Papers in Economics. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Alessandro Moschitti, Bo Pang, and Walter Daelemans (Eds.). Association for Computational Linguistics, Doha, Qatar, 1804–1809. https://doi.org/10.3115/v1/D14-1191

[54] Yohan Jo, Seojin Bang, Emaad Manzoor, Eduard Hovy, and Chris Reed. 2020. Detecting Attackable Sentences in Arguments. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (Eds.). Association for Computational Linguistics, Online, 1–23. https://doi.org/10.18653/v1/2020.emnlp-main.1

[55] Daniel Kang, John Emmons, Firas Abuzaid, Peter Bailis, and Matei Zaharia. 2017. NoScope: Optimizing Neural Network Queries over Video at Scale. arXiv:1703.02529 [cs.DB]

[56] Daniel Kang, John Guibas, Peter D. Bailis, Tatsunori Hashimoto, and Matei Zaharia. 2022. TASTI: Semantic Indexes for Machine Learning-based Queries over Unstructured Data. In *Proceedings of the 2022 International Conference on Management of Data* (Philadelphia, PA, USA) *(SIGMOD '22)*. Association for Computing Machinery, New York, NY, USA, 1934–1947. https://doi.org/10.1145/3514221.3517897

[57] George Katsogiannis-Meimarakis and Georgia Koutrika. 2023. A survey on deep learning approaches for text-to-SQL. *The VLDB Journal* 32, 4 (2023), 905–936. https://doi.org/10.1007/s00778-022-00776-8

[58] Adam Kramer, Jamie Guillory, and Jeffrey Hancock. 2014. Experimental Evidence of Massive-Scale Emotional Contagion Through Social Networks. *Proceedings of the National Academy of Sciences of the United States of America* 111 (06 2014). https://doi.org/10.1073/pnas.1320040111

[59] Sumith Kulal, Panupong Pasupat, Kartik Chandra, Mina Lee, Oded Padon, Alex Aiken, and Percy Liang. 2019. SPoC: Search-based Pseudocode to Code. arXiv:1906.04908 [cs.LG]

[60] David Lazer, Alex Pentland, Lada Adamic, Sinan Aral, Albert-László Barabási, Devon Brewer, Nicholas Christakis, Noshir Contractor, James Fowler, Myron Gutmann, et al. 2009. Computational social science. *Science* 323, 5915 (2009), 721–723.

[61] David MJ Lazer, Alex Pentland, Duncan J Watts, Sinan Aral, Susan Athey, Noshir Contractor, Deen Freelon, Sandra Gonzalez-Bailon, Gary King, Helen Margetts, et al. 2020. Computational social science: Obstacles and opportunities. *Science* 369, 6507 (2020), 1060–1062.

[62] David M. J. Lazer, Matthew A. Baum, Yochai Benkler, Adam J. Berinsky, Kelly M. Greenhill, Filippo Menczer, Miriam J. Metzger, Brendan Nyhan, Gordon Pennycook, David Rothschild, Michael Schudson, Steven A. Sloman, Cass R. Sunstein, Emily A. Thorson, Duncan J. Watts, and Jonathan L. Zittrain. 2018. The science of fake news. *Science* 359, 6380 (2018), 1094–1096. https://doi.org/10.1126/science.aao2998 arXiv:https://www.science.org/doi/pdf/10.1126/science.aao2998

[63] Jens Lemmens, Ilia Markov, and Walter Daelemans. 2021. Improving Hate Speech Type and Target Detection with Hateful Metaphor Features. In *Proceedings of the Fourth Workshop on NLP for Internet Freedom: Censorship, Disinformation, and Propaganda*, Anna Feldman, Giovanni Da San Martino, Chris Leberknight, and Preslav Nakov (Eds.). Association for Computational Linguistics, Online, 7–16. https://doi.org/10.18653/v1/2021.nlp4if-1.2

[64] Jinyang Li, Binyuan Hui, Ge Qu, Jiaxi Yang, Binhua Li, Bowen Li, Bailin Wang, Bowen Qin, Ruiying Geng, Nan Huo, et al. 2024. Can llm already serve as a database interface? a big bench for large-scale database grounded text-to-sqls. *Advances in Neural Information Processing Systems* 36 (2024).

[65] Mingyang Li, Louis Hickman, Louis Tay, Lyle Ungar, and Sharath Chandra Guntuku. 2020. Studying Politeness across Cultures using English Twitter and Mandarin Weibo. *Proc. ACM Hum.-Comput. Interact.* 4, CSCW2, Article 119 (oct 2020), 15 pages. https://doi.org/10.1145/3415190

[66] Sha Li, Heng Ji, and Jiawei Han. 2021. Document-Level Event Argument Extraction by Conditional Generation. arXiv:2104.05919 [cs.CL]

[67] Zihao Li, Dongqi Fu, and Jingrui He. 2023. Everything Evolves in Personalized PageRank. In *Proceedings of the ACM Web Conference 2023, WWW 2023, Austin, TX, USA, 30 April 2023 - 4 May 2023*, Ying Ding, Jie Tang, Juan F. Sequeda, Lora Aroyo, Carlos Castillo, and Geert-Jan Houben (Eds.). ACM, 3342–3352. https://doi.org/10.1145/3543507.3583474

[68] Qian Liu, Bei Chen, Jiaqi Guo, Morteza Ziyadi, Zeqi Lin, Weizhu Chen, and Jian-Guang Lou. 2022. TAPEX: Table Pre-training via Learning a Neural SQL Executor. arXiv:2107.07653 [cs.CL] https://arxiv.org/abs/2107.07653

[69] Siyang Liu, Chujie Zheng, Orianna Demasi, Sahand Sabour, Yu Li, Zhou Yu, Yong Jiang, and Minlie Huang. 2021. Towards Emotional Support Dialog Systems. arXiv:2106.01144 [cs.CL]

[70] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR* abs/1907.11692 (2019). arXiv:1907.11692 http://arxiv.org/abs/1907.11692

[71] Shuai Ma, Wenbin Jiang, Xiang Ao, Meng Tian, Xinwei Feng, Yajuan Lyu, Qiaoqiao She, and Qing He. 2023. Semantic-Driven Instance Generation for Table Question Answering. In *International Conference on Database Systems for Advanced Applications*. Springer, 3–18.

[72] Binny Mathew, Anurag Illendula, Punyajoy Saha, Soumya Sarkar, Pawan Goyal, and Animesh Mukherjee. 2020. Hate begets Hate: A Temporal Study of Hate Speech. arXiv:1909.10966 [cs.SI]

[73] Daniel A McFarland, Kevin Lewis, and Amir Goldberg. 2016. Sociology in the era of big data: The ascent of forensic social science. *The American Sociologist* 47 (2016), 12–35.

[74] K. McRae and J. J. Gross. 2020. Emotion regulation. *Emotion* 20, 1 (2020), 1–9.

[75] Julia Mendelsohn, Ronan Le Bras, Yejin Choi, and Maarten Sap. 2023. From Dogwhistles to Bullhorns: Unveiling Coded Rhetoric with Language Models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (Eds.). Association for Computational Linguistics, Toronto, Canada, 15162–15180. https://doi.org/10.18653/v1/2023.acl-long.845

[76] Saif Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiaodan Zhu, and Colin Cherry. 2016. SemEval-2016 Task 6: Detecting Stance in Tweets. In

Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016), Steven Bethard, Marine Carpuat, Daniel Cer, David Jurgens, Preslav Nakov, and Torsten Zesch (Eds.). Association for Computational Linguistics, San Diego, California, 31–41. https://doi.org/10.18653/v1/S16-1003

[77] Vlad Niculae and Cristian Danescu-Niculescu-Mizil. 2014. Brighter than Gold: Figurative Language in User Generated Comparisons. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Alessandro Moschitti, Bo Pang, and Walter Daelemans (Eds.). Association for Computational Linguistics, Doha, Qatar, 2008–2018. https://doi.org/10.3115/v1/D14-1215

[78] OpenAI. 2023. https://chat.openai.com/ Accessed: October 24, 2024.

[79] OpenAI. 2023. https://platform.openai.com/docs/models/gpt-4-and-gpt-4-turbo Accessed: October 24, 2024.

[80] OpenAI. 2023. https://platform.openai.com/docs/guides/function-calling/supported-models Accessed: October 24, 2024.

[81] OpenAI. 2023. https://platform.openai.com/examples/default-sql-translate Accessed: October 24, 2024.

[82] OpenAI. 2024. https://openai.com/pricing

[83] Panupong Pasupat and Percy Liang. 2015. Compositional Semantic Parsing on Semi-Structured Tables. *CoRR* abs/1508.00305 (2015). arXiv:1508.00305 http://arxiv.org/abs/1508.00305

[84] Verónica Pérez-Rosas, Mohamed Abouelenien, Rada Mihalcea, and Mihai Burzo. 2015. Deception Detection using Real-life Trial Data. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction* (Seattle, Washington, USA) *(ICMI '15)*. Association for Computing Machinery, New York, NY, USA, 59–66. https://doi.org/10.1145/2818346.2820758

[85] Mohammad Taher Pilehvar and Jose Camacho-Collados. 2019. WiC: the Word-in-Context Dataset for Evaluating Context-Sensitive Meaning Representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Jill Burstein, Christy Doran, and Thamar Solorio (Eds.). Association for Computational Linguistics, Minneapolis, Minnesota, 1267–1273. https://doi.org/10.18653/v1/N19-1128

[86] Mohammadreza Pourreza and Davood Rafiei. 2024. Din-sql: Decomposed in-context learning of text-to-sql with self-correction. *Advances in Neural Information Processing Systems* 36 (2024).

[87] Marcelo O. R. Prates, Pedro H. C. Avelar, and Luis Lamb. 2019. Assessing Gender Bias in Machine Translation – A Case Study with Google Translate. arXiv:1809.02208 [cs.CY] https://arxiv.org/abs/1809.02208

[88] Daniel Preoţiuc-Pietro, Ye Liu, Daniel Hopkins, and Lyle Ungar. 2017. Beyond Binary Labels: Political Ideology Prediction of Twitter Users. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Regina Barzilay and Min-Yen Kan (Eds.). Association for Computational Linguistics, Vancouver, Canada, 729–740. https://doi.org/10.18653/v1/P17-1068

[89] Bowen Qin, Binyuan Hui, Lihan Wang, Min Yang, Jinyang Li, Binhua Li, Ruiying Geng, Rongyu Cao, Jian Sun, Luo Si, et al. 2022. A survey on text-to-sql parsing: Concepts, methods, and future directions. *arXiv preprint arXiv:2208.13629* (2022).

[90] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2023. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. arXiv:1910.10683 [cs.LG]

[91] Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know What You Don't Know: Unanswerable Questions for SQuAD. arXiv:1806.03822 [cs.CL] https://arxiv.org/abs/1806.03822

[92] Manuel Romero. 2023. https://huggingface.co/mrm8488/t5-base-finetuned-wikiSQL Accessed: February 25, 2024.

[93] Barbara Rothbaum, Elizabeth Meadows, and Patricia Resick. 2012. Cognitive-Behavioral Therapy. *Journal of Traumatic Stress* 13 (10 2012).

[94] Barbara O Rothbaum, Elizabeth A Meadows, Patricia Resick, and David W Foy. 2000. Cognitive-behavioral therapy. In *Effective treatments for PTSD: Practice guidelines from the International Society for Traumatic Stress Studies*, Edna B Foa, Terence M Keane, and Matthew J Friedman (Eds.). The Guilford Press, 320–325.

[95] Maarten Sap, Dallas Card, Saadia Gabriel, Yejin Choi, and Noah A. Smith. 2019. The Risk of Racial Bias in Hate Speech Detection. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Anna Korhonen, David Traum, and Lluís Màrquez (Eds.). Association for Computational Linguistics, Florence, Italy, 1668–1678. https://doi.org/10.18653/v1/P19-1163

[96] Maarten Sap, Saadia Gabriel, Lianhui Qin, Dan Jurafsky, Noah A. Smith, and Yejin Choi. 2020. Social Bias Frames: Reasoning about Social and Power Implications of Language. arXiv:1911.03891 [cs.CL]

[97] Maarten Sap, Saadia Gabriel, Lianhui Qin, Dan Jurafsky, Noah A Smith, and Yejin Choi. 2020. Social Bias Frames: Reasoning about Social and Power Implications of Language. In *ACL*.

[98] Maarten Sap, Eric Horvitz, Yejin Choi, Noah A. Smith, and James Pennebaker. 2020. Recollection versus Imagination: Exploring Human Memory and Cognition via Neural Language Models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (Eds.). Association for Computational Linguistics, Online, 1970–1978. https://doi.org/10.18653/v1/2020.acl-main.178

[99] Maarten Sap, Anna Jafarpour, Choi Yejin, Noah Smith, James Pennebaker, and Eric Horvitz. 2022. Quantifying the narrative flow of imagined versus autobiographical stories. *Proceedings of the National Academy of Sciences of the United States of America* 119 (11 2022), e2211715119. https://doi.org/10.1073/pnas.2211715119

[100] Elvis Saravia, Hsien-Chi Toby Liu, Yen-Hao Huang, Junlin Wu, and Yi-Shin Chen. 2018. CARER: Contextualized Affect Representations for Emotion Recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (Eds.). Association for Computational Linguistics, Brussels, Belgium, 3687–3697. https://doi.org/10.18653/v1/D18-1404

[101] Scale. [n.d.]. https://scale.com/rapid Accessed: October 24, 2024.

[102] Harvard Law School. 2023. https://case.law/ Accessed: October 24, 2024.

[103] Ashish Sharma, Adam S. Miner, David C. Atkins, and Tim Althoff. 2020. A Computational Approach to Understanding Empathy Expressed in Text-Based Mental Health Support. arXiv:2009.08441 [cs.CL]

[104] Peter Shaw, Ming-Wei Chang, Panupong Pasupat, and Kristina Toutanova. 2020. Compositional generalization and natural language variation: Can a semantic parsing approach handle both? *arXiv preprint arXiv:2010.12725* (2020).

[105] Rachele Sprugnoli and Sara Tonelli. 2019. Novel Event Detection and Classification for Historical Texts. *Computational Linguistics* 45, 2 (June 2019), 229–265. https://doi.org/10.1162/coli_a_00347

[106] Kevin Stowe, Prasetya Utama, and Iryna Gurevych. 2022. IMPLI: Investigating NLI Models' Performance on Figurative Language. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (Eds.). Association for Computational Linguistics, Dublin, Ireland, 5375–5388. https://doi.org/10.18653/v1/2022.acl-long.369

[107] Alane Suhr, Ming-Wei Chang, Peter Shaw, and Kenton Lee. 2020. Exploring Unexplored Generalization Challenges for Cross-Database Semantic Parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (Eds.). Association for Computational Linguistics, Online, 8372–8388. https://doi.org/10.18653/v1/2020.acl-main.742

[108] Chenhao Tan, Vlad Niculae, Cristian Danescu-Niculescu-Mizil, and Lillian Lee. 2016. Winning Arguments: Interaction Dynamics and Persuasion Strategies in Good-faith Online Discussions. In *Proceedings of the 25th International Conference on World Wide Web* (Montréal, Québec, Canada) *(WWW '16)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 613–624. https://doi.org/10.1145/2872427.2883081

[109] Anja Thieme, Danielle Belgrave, and Gavin Doherty. 2020. Machine Learning in Mental Health: A Systematic Review of the HCI Literature to Support the Development of Effective and Implementable ML Systems. *ACM Trans. Comput.-Hum. Interact.* 27, 5, Article 34 (aug 2020), 53 pages. https://doi.org/10.1145/3398069

[110] Catalina L. Toma, Jeffrey T. Hancock, and Nicole B. Ellison. 2008. Separating Fact From Fiction: An Examination of Deceptive Self-Presentation in Online Dating Profiles. *Personality and Social Psychology Bulletin* 34, 8 (2008), 1023–1036. https://doi.org/10.1177/0146167208318067 arXiv:https://doi.org/10.1177/0146167208318067 PMID: 18593866.

[111] Bing Wang, Yan Gao, Zhoujun Li, and Jian-Guang Lou. 2023. Know What I don't Know: Handling Ambiguous and Unanswerable Questions for Text-to-SQL. arXiv:2212.08902 [cs.CL]

[112] Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. 2021. RAT-SQL: Relation-Aware Schema Encoding and Linking for Text-to-SQL Parsers. arXiv:1911.04942 [cs.CL]

[113] Chuangxian Wei, Bin Wu, Sheng Wang, Renjie Lou, Chaoqun Zhan, Feifei Li, and Yuanzhe Cai. 2020. AnalyticDB-V: a hybrid analytical engine towards query fusion for structured and unstructured data. *Proc. VLDB Endow.* 13, 12 (aug 2020), 3152–3165. https://doi.org/10.14778/3415478.3415541

[114] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. arXiv:2201.11903 [cs.CL]

[115] Orion Weller and Kevin Seppi. 2019. Humor Detection: A Transformer Gets the Last Laugh. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (Eds.). Association for Computational Linguistics, Hong Kong, China, 3621–3625. https://doi.org/10.18653/v1/D19-1372

[116] Diyi Yang, Jiaao Chen, Zichao Yang, Dan Jurafsky, and Eduard Hovy. 2019. Let's Make Your Request More Persuasive: Modeling Persuasive Strategies via Semi-Supervised Neural Nets on Crowdfunding Platforms. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and*

Short Papers), Jill Burstein, Christy Doran, and Thamar Solorio (Eds.). Association for Computational Linguistics, Minneapolis, Minnesota, 3620–3630. https://doi.org/10.18653/v1/N19-1364

[117] Diyi Yang, Kaitlyn Zhou, and Wenna Qin. 2023. CS 224C: NLP for Computational Social Science. Stanford University. https://web.stanford.edu/class/cs224c/

[118] Tao Yu, Zifan Li, Zilin Zhang, Rui Zhang, and Dragomir Radev. 2018. TypeSQL: Knowledge-based Type-Aware Neural Text-to-SQL Generation. arXiv:1804.09769 [cs.CL]

[119] Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. 2018. Spider: A Large-Scale Human-Labeled Dataset for Complex and Cross-Domain Semantic Parsing and Text-to-SQL Task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (Eds.). Association for Computational Linguistics, Brussels, Belgium, 3911–3921. https://doi.org/10.18653/v1/D18-1425

[120] Hongli Zhan, Tiberiu Sosea, Cornelia Caragea, and Junyi Jessy Li. 2022. Why Do You Feel This Way? Summarizing Triggers of Emotions in Social Media Posts. *ArXiv* abs/2210.12531 (2022). https://api.semanticscholar.org/CorpusID:253098848

[121] Amy X. Zhang, Bryan Culbertson, and Praveen K. Paritosh. 2017. Characterizing Online Discussion Using Coarse Discourse Sequences. *Proceedings of the International AAAI Conference on Web and Social Media* (2017). https://api.semanticscholar.org/CorpusID:35696952

[122] Justine Zhang, Jonathan Chang, Cristian Danescu-Niculescu-Mizil, Lucas Dixon, Yiqing Hua, Dario Taraborelli, and Nithum Thain. 2018. Conversations Gone Awry: Detecting Early Signs of Conversational Failure. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Iryna Gurevych and Yusuke Miyao (Eds.). Association for Computational Linguistics, Melbourne, Australia, 1350–1361. https://doi.org/10.18653/v1/P18-1125

[123] Jun Zhang, Wei Wang, Feng Xia, Yu-Ru Lin, and Hanghang Tong. 2020. Data-Driven Computational Social Science: A Survey. *Big Data Research* 21 (2020), 100145. https://doi.org/10.1016/j.bdr.2020.100145

[124] Yi Zhang, Jan Deriu, George Katsogiannis-Meimarakis, Catherine Kosten, Georgia Koutrika, and Kurt Stockinger. 2023. ScienceBenchmark: A Complex Real-World Benchmark for Evaluating Natural Language to SQL Systems. arXiv:2306.04743 [cs.DB]

[125] Victor Zhong, Caiming Xiong, and Richard Socher. 2017. Seq2SQL: Generating Structured Queries from Natural Language using Reinforcement Learning. *CoRR* abs/1709.00103 (2017).

[126] Caleb Ziems, William Held, Omar Shaikh, Jiaao Chen, Zhehao Zhang, and Diyi Yang. 2023. Can Large Language Models Transform Computational Social Science? arXiv:2305.03514 [cs.CL]

[127] Caleb Ziems, William Held, Jingfeng Yang, Jwala Dhamala, Rahul Gupta, and Diyi Yang. 2023. Multi-VALUE: A Framework for Cross-Dialectal English NLP. arXiv:2212.08011 [cs.CL]

[128] Caleb Ziems, Minzhi Li, Anthony Zhang, and Diyi Yang. 2022. Inducing Positive Perspectives with Text Reframing. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (Eds.). Association for Computational Linguistics, Dublin, Ireland, 3682–3700. https://doi.org/10.18653/v1/2022.acl-long.257