Teddy: A System for Interactive Review Analysis

Anonymous
for Submission
City, Country
e-mail address

ABSTRACT

Reviews are integral to e-commerce services and products. They contain a wealth of information about users' opinions and experiences, which can help better understand consumer decisions and improve user experience with products and services. Today, data scientists analyze reviews by developing rules and models to extract, aggregate, and understand information embedded in review text. However, working with thousands of reviews, which are typically noisy incomplete user-generated text, can be daunting without proper tools. Here we first contribute results from an interview study that we conducted with fourteen data scientists who work with review text, providing insights into their practices and challenges. Results suggest data scientists need interactive systems for many review analysis tasks. Towards a solution, we then introduce Teddy, an interactive system that enables data scientists to quickly obtain insights from reviews and improve their extraction and modeling pipelines.

Author Keywords

Interactive systems; visualization; data science; contextual interviews; review analysis; text mining; opinion mining; sentiment analysis; schema generation.

INTRODUCTION

Consumer reviews have become an integral part of e-commerce services and products, such as hotels, restaurants, and job listings. Their prevalence is largely spurred by aggregator services such as booking sites for hotels, or restaurants, where reviews can help consumers decide between hotels or restaurants. Reviews are full of useful information, including consumer experiences, facts, tips, and more, and the abundance of reviews can provide reliable and relevant signals about the quality of services and products and how to improve them. Consumers regularly check reviews to inform their purchasing choices, online marketplace platforms display reviews along with summaries for consumers as well as sellers to facilitate their decision making, and service/product owners use them to track consumer feedback and adjust their products and services. Consequently, deriving insights from review text is a fundamental challenge.

CHI'20, April 25–30, 2020, Honolulu, HI, USA

ISBN 978-1-4503-6708-0/20/04…$15.00

DOI: https://doi.org/10.1145/3313831.XXXXXXX

Researchers across multiple fields, including data mining and natural language processing (NLP), have investigated the challenge of extracting, summarizing, and aggregating information from text and developed techniques for opinion mining and sentiment analysis [26, 32]. Today, many e-commerce companies, in particular those providing aggregation and search services, employ data scientists to analyze, extract, and summarize information from reviews. However, review text generated by consumers is notoriously noisy and often sparse in informational content. For example, a review about a hotel typically mentions only a couple of aspects about the hotel out of dozens of possible aspects. Reading and searching through thousands of sparse, noisy short texts in order to analyze, understand, and interpret them is a daunting task without effective tools.

In this paper, we first contribute results from an interview study that we conducted with fourteen participants to better understand the workflows and rate-limiting tasks of data scientists working on reviews. Our results suggest that data scientists spend most of their time in data preparation, providing additional evidence for similar findings from earlier general studies (e.g., [19]). We find that data scientists are less concerned about developing new models or tuning hyper-parameters and are typically satisfied with using existing models such as BERT [7]. On the other hand, they are challenged by a lack of tools that would help across different stages of data preparation, ranging from labeling and crowdsourcing, to interactive exploration, to schema generation (where a schema is defined as a domain-specific set of attributes that users care about, for example a schema for the cell-phone domain might include price, weight, camera quality, etc.). In particular, our findings suggest that data scientists need interactive tools to quickly obtain insights from reviews and to inform their extraction and modeling pipelines.

To address this need, we also contribute Teddy (Fig 1), an interactive visual analysis system for data scientists to quickly explore reviews at scale and iteratively refine extraction schemas of opinion mining pipelines. Informed by the results of our study, Teddy enables similarity-based multiscale exploration of reviews using fine-grained opinions extracted from them. Teddy is extensible, supporting common text search, filtering, regular expression matching, sorting operations and their visualizations through programmatic extensions. Teddy also sustains an interactive user experience for the analysis of large numbers of reviews by using a combination of pre-computation, indexing, and user-controlled separation of front- and back-end computations. Finally, Teddy enables data scientists to interactively revise and expand domain-specific schemas used for aspect-based opinion mining. We demonstrate the utility of Teddy through two in-depth use cases involving exploratory

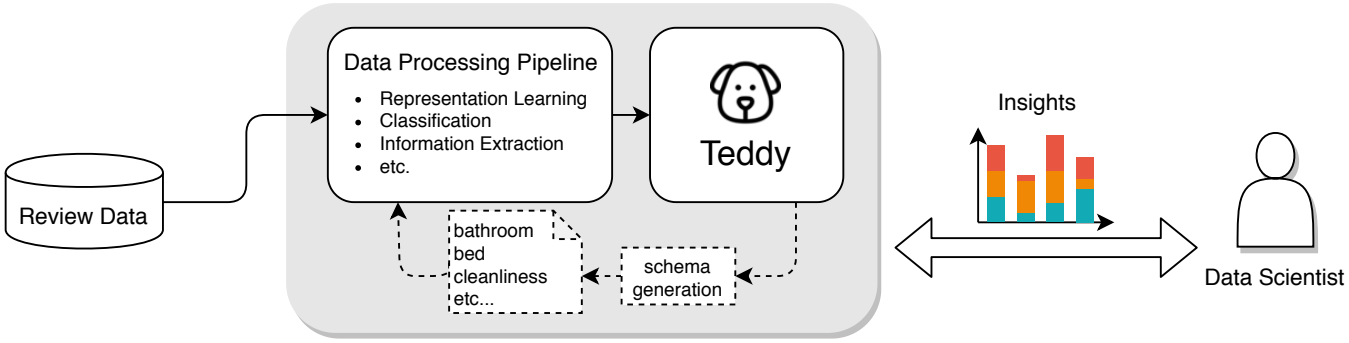


Figure 1: The Teddy review exploration pipeline. The user can customize the data processing pipeline based on their task, whether it is classification, opinion extraction, or representation learning, and use Teddy to gain insights about their data and model. They can also use the application to iterate on the data processing pipeline, for example by creating a new schema that describes attributes of their review corpus.

analysis of hotel reviews and iterative schema generation for opinion extraction from restaurant reviews, respectively.

We have made all of our research artifacts, including raw and aggregated data collected from the interview study and source code for Teddy, along with a running copy deployed as a Web application, publicly available at https://github.com/teddyauthors/teddy.

RELATED WORK

Our work builds on earlier research in interactive systems for visual review analysis, review text mining, and formative interview studies into data science practices.

Visual Analysis of Reviews Prior research introduces several tools and techniques for interactive review analysis and visualization. The general approach underlying prior work is to first extract features from review data and then focus the affordances and visualizations around the derived attributes and their values (e.g., sentiments, frequent words, etc.). Therefore, the differences among prior tools are in part characterized by the differences in their extractors.

Initial work on visual review analysis relies on document or sentence level coarse-grained sentiment extraction. Morinaga et al. [29] propose a 2D scatterplot called positioning map to show groups of sentences with positive or negative sentiments. Gregory et al. [13] use rose plot to display sentiments such as positive, negative, pleasure, pain, and conflict. Pulse [12] visualizes topics and average sentiment values extracted from reviews using a treemap. Chen et al. [5] present a visual analysis system with multiple coordinated views including decision trees and term variation graphs to help users understand conflicting opinions in product reviews and their explanations present in the review data. Draper and Riesenfeld [9] introduce an interactive tool to enable users to visually construct queries and their results. Review Spotlight [41] visually summarizes reviews for rapid comprehension using adjective-noun word pairs.

Reviews contain richer information than document level or sentence level opinion visualizations can provide. Overall sentiments expressed in reviews can be sliced into finer-grained sentiments on domain-specific aspects. With the development of feature-based opinion mining, researchers introduced feature-level opinion visualizations. For example, Liu et al.[25] propose a method to extract feature-level opinions from customer reviews and use bar charts for visual comparison of extracted opinions. Oelke et al. [31] discuss several visualization techniques including visual summary reports, cluster analysis, and circular correlation maps to facilitate visual analysis of customer feedback data at the feature level. Opinion Observer [25] allows users to compare sentiments extracted from product across aspects extracted from the text as well as the structured fields. SentiVis [8] coordinates simple scatterplot and lineplot visualizations for given an aspect and an entity (restaurant). OpinionSeer [40] visualizes hotel reviews across multiple dimensions including demographics, temporal and geolocation information using a radial layout. OpinionBlocks [2] provides an overview of aspect based sentiments in customer reviews. Hao et al. [14] propose pixel-based sentiment visualization for temporal and geospatial visualization of user feedback at scale. To reduce clutter while displaying a large number of sentiments in a single view, the authors use a SOM (self organizing map) based clustering. ViTA-SSD [38] is another system that allows users to select and visualize the distribution of various structured fields, as well as to create and compare keyword clusters using word clouds. TextTile [10] builds on structured search interfaces, allowing users to query review text as well as derived structured and meta data associated with reviews.

In line with recent approaches, Teddy also uses a fine-grained opinion extractor [24] for an in-depth feature based review exploration. Teddy takes multiple visualization techniques generally used in isolation by earlier approaches and combines them in a novel, information-rich configuration to enable the visual analysis of raw review text alongside fine-grained and aggregated opinion extractions and meta data. Unlike previous tools, which were typically designed for end users (e.g., customers), we designed Teddy for data scientists. It is extensible and supports common text operations needed by data scientists and their visualizations through programmatic extensions. In a novel approach, Teddy also facilitates an iterative improvement of opinion extractor schemas, extractions of which it already visualizes.

Note that our work here also falls into broader fields of sentiment/opinion visualization and visual text analytics, for which

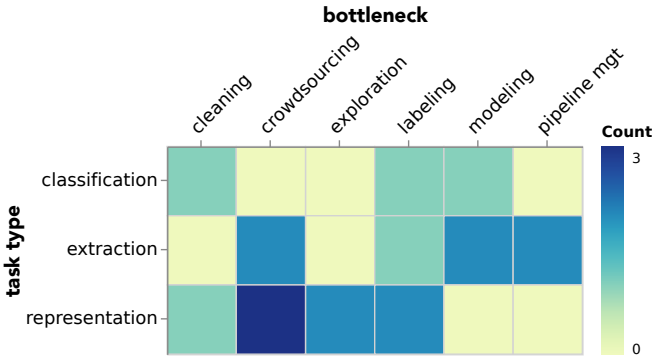


Figure 2: We identify some of the most common bottlenecks reported by data scientists working with review data, separated by the type of task. Bottleneck types are listed along the x-axis, and task-type groups are on the y-axis. We find most bottlenecks involve crowdsourcing, labeling and exploration, especially among those working on entity representation.

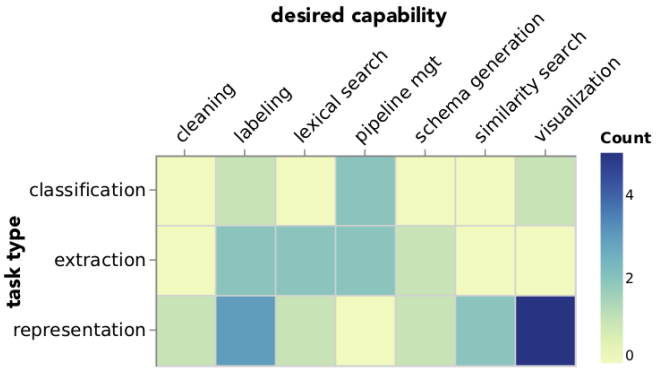


Figure 3: Responses identifying desired capabilities for review-text analysis tools, grouped by participants' task types on the y-axis. We find that there is high demand for tools with better data visualization capabilities, particularly among those working in entity representation tasks, which motivates our application design.

several surveys [22, 27] provide a comprehensive discussion of the existing work.

Review Mining Prior research applies sentiment analysis and opinion mining to extract and compile opinions and facts within large collections of review text [26, 32]. Sentiment analysis aims to quantify the sentiments expressed in text at different levels (sentence, paragraph, document, etc.). On the other hand, opinion mining builds on sentiment analysis to aggregate extracted sentiments into effective summaries to inform various end user tasks. Earlier research in sentiment analysis and opinion mining proposes many approaches for mining the overall opinion at the document and sentence levels [21, 33]. Unsurprisingly, later research increasingly focuses on fine-grained extractions to derive opinions (including subjective judgments, facts, tips, etc.) per aspect or feature [17, 18, 35]. We use OpineDB [24] to extract opinions about domain-specific aspects (e.g., cleanliness, service, location, etc. in the hotel domain) from review text. The OpineDB extractor relies on BERT [7], a pre-trained state-of-the-art language model, to perform fine-grained aspect-based extraction with small human labeling effort. Although Teddy benefits from OpineDB's high-quality interpretable extractions, it doesn't depend on it. Any other feature extractor for text can be plugged into our data preparation pipeline with minimal effort.

Interview Studies with Data Scientists Conducting interviews is a well-established and effective qualitative data collection method [1, 4, 16, 23, 28] for eliciting user feedback as well as understanding problem domains in depth. Researchers have recently conducted semi-structured interviews to better understand different aspects of data science or machine learning practices and challenges (e.g., [6, 19, 20, 30, 34]). In the current work, we also carry out semi-structured interviews with data scientists to gain insights into review text analysis pipelines, informing the design of Teddy as well as research at large. While our methodology is similar to earlier approaches [6, 19, 20], the current interview study differs by focusing on the specific domain of review text analysis. To the best of our knowledge, the interview study presented here is the first interview study focusing on review analysis in particular.

INTERVIEW STUDY

To better understand data science practices and challenges in review analysis and mining, we conducted an interview study with data scientists working with review text corpora.

Participants We interviewed 14 researchers (11 male and 3 female) solicited from our professional networks. Participants worked at either AI/data-science research labs (8), review aggregating companies (2), or hospitality-sector data processing companies (2). Participants were geographically split between the United States (8), Japan (4), and Germany (2).

Participants held job titles such as "research scientist," "data analyst," "data scientist," "research intern," "postdoctoral research fellow" or "research scientist/software engineer." One participant was a "senior data scientist", and one was a "project manager." Henceforth, we will refer to all of the participants as "data scientists." Most participants held PhDs (7 in computer science, 1 in industrial engineering). The rest were either PhD students (2) or held Master's degrees (2 in computer science, 2 in computational linguistics). One participant with a Master's degree in computer science also held a business degree.

Past experience with review corpora ranged from 0 to 18 years with a mean of 2.68 years (std=4.7). Participants' past experience with data analysis in general varied across participants: participants' past data analysis experience ranged from 0 to 20 years with a mean of 6.3 years (std=5.49).

Most participants (9) worked with English-language corpora. The next most common language (4) was Japanese. Only 3 out of the 14 participants reported using multi-language corpora.

Methodology Interviews were conducted in 1-hour sessions. With one exception, we interviewed one participant at a time, and with 1-3 researchers asking questions, taking notes, and recording audio. Notes were compared and summarized between researchers. Once summaries were compiled, we sent them back to each corresponding participant and encouraged them to look over the summaries and reply with revisions or additional comments. Whenever possible, we interviewed participants in person, otherwise resorting to video conferencing.

We asked participants open-ended questions about their work experiences with review corpora, and asked them to walk us through specific examples of projects that they have recently worked on. We designed a rubric of thirteen questions that would identify key elements of data scientists' workflows. The rubric included broad questions such as *"How do you prepare your data for use in your pipeline?* and *"What do you spend most of your time on?"*, and more specific questions such as *"What data sources and formats do you use?"* We also asked participants additional questions that arose from their other answers. At the end of each interview, we asked participants for additional comments if they felt that we had missed important information about their experiences working with reviews.

Once interviews were completed, we used an iterative coding method to analyze the notes. We collected common experiences and tasks into groups, refining these categories as further data was analyzed.

Results

Task Taxonomy We identified 3 overarching task types that participants described in their interviews.

Classification, where analysts develop algorithms to classify either entire reviews or individual sentences into predefined categories (e.g., based on sentiment or between a set of topics).

Extraction, where analysts develop algorithms to detect relevant entities from reviews, as well as descriptive text elsewhere in the reviews that directly modify or describe the entities (e.g., extracting opinions about the quality of specific types of food from restaurant reviews).

Representation, where analysts build graph representations or database architectures to accommodate data and insights related to review text corpora (e.g., developing a schema of amenities offered by specific hotels).

The distribution of participants between these categories of work was roughly even: entity extraction and classification tasks were the primary work categories for 4 participants each, entity representation was the primary work category for 5 participants. A final participant described a work pipeline that included equal amounts of entity representation and extraction tasks.

Data Participants generally sourced data directly from the client companies commissioning the technologies that the participants were developing. For participants engaged in more open-ended research, public domain data was used. In several cases, participants used both proprietary and public domain data in their work. Almost all participants reported needing to look at raw data points (i.e., review text) frequently as part of their analysis. Roughly half of the participants reported that their review analysis tasks required data cleaning as a preliminary task.

Dataset sizes varied across participants, ranging from sets with thousands of data points to sets with tens of millions of data points. Participants using datasets with fewer than 1 million examples generally used CSV files to store their data; those working with larger datasets used SQL databases. Data scalability was named as a significant concern for pipeline management for 5 out of the 14 participants. Counter-intuitively, those working on the largest datasets with tens of millions of entries did not report any concerns about the scalability of their pipelines.

We believe that this discrepancy is due to the nature of the tasks being performed by the analysts: both of the participants that reported having no scalability concerns despite using datasets with tens of millions of entries were working on entity representation tasks that did not require the training of computationally expensive machine learning models. These findings suggest that there is a need for future work that rigorously analyzes the different needs of data scientists working on different tasks.

Tools Python was ubiquitous as the primary programming language for our participants, with some also using Java or SQL for database interfacing. This finding is consistent with a trend reported by a previous interview study of data scientists [6]. Participants generally used a combination of Gensim[36], NLTK[3], and SpaCy[15] for natural language processing functions and word embeddings. One participant specifically mentioned that their team was transitioning from NLTK to SpaCy, as they found SpaCy to outperform NLTK in all of their use-cases: "The quality and performance [of tokenization and other NLP functions] are much better on SpaCy." Participants working with Japanese corpora used SudachiPy [39] for their core NLP work. Participants working directly with neural network models used BERT [7] as their model framework. Roughly one quarter of participants (4) reported using sentiment extraction as part of their analysis, using NLTK, SpaCy, or self-developed algorithms to perform this extraction.

Four of the participants reported using Jupyter Notebook as a collaboration tool.

Model Training While most participants (12) had some sort of model training as part of their pipeline, most were interested in applying off-the-shelf models rather than designing new ones. Only two participants, working on the same project, specified parameter tuning as part of their work pipeline, and only three participants (including the aforementioned two) reported having several different options for which models to use for their pipeline. By contrast, half of the participants reported that they regularly conducted manual reviews of individual data points as part of a model debugging process.

When asked to specify which metrics were their top priorities for improvement when optimizing their models, participants were equally likely to report either precision or recall as their top priority. One participant instead said that they were focusing on finding higher quality data, rather than trying to improve their model's performance on their existing dataset.

Challenges Participants reported a variety of different tasks as being particularly challenging bottlenecks to their work (See Fig 2).

Data Cleaning Roughly half of the participants who included data cleaning as part of their pipeline (3/8) described data cleaning as a bottleneck. For our purposes, we define "data cleaning" as tasks that involve editing or removing data at the outset of the pipeline, such as removing reviews that are too short, censoring foul language, or applying spellchecking functions to the data. Difficulties with data cleaning often stemmed from the scale of the data to be cleaned, and the necessity of subjective judgments as part of the cleaning process, making automation difficult or impossible: *"you have this problem which is data*

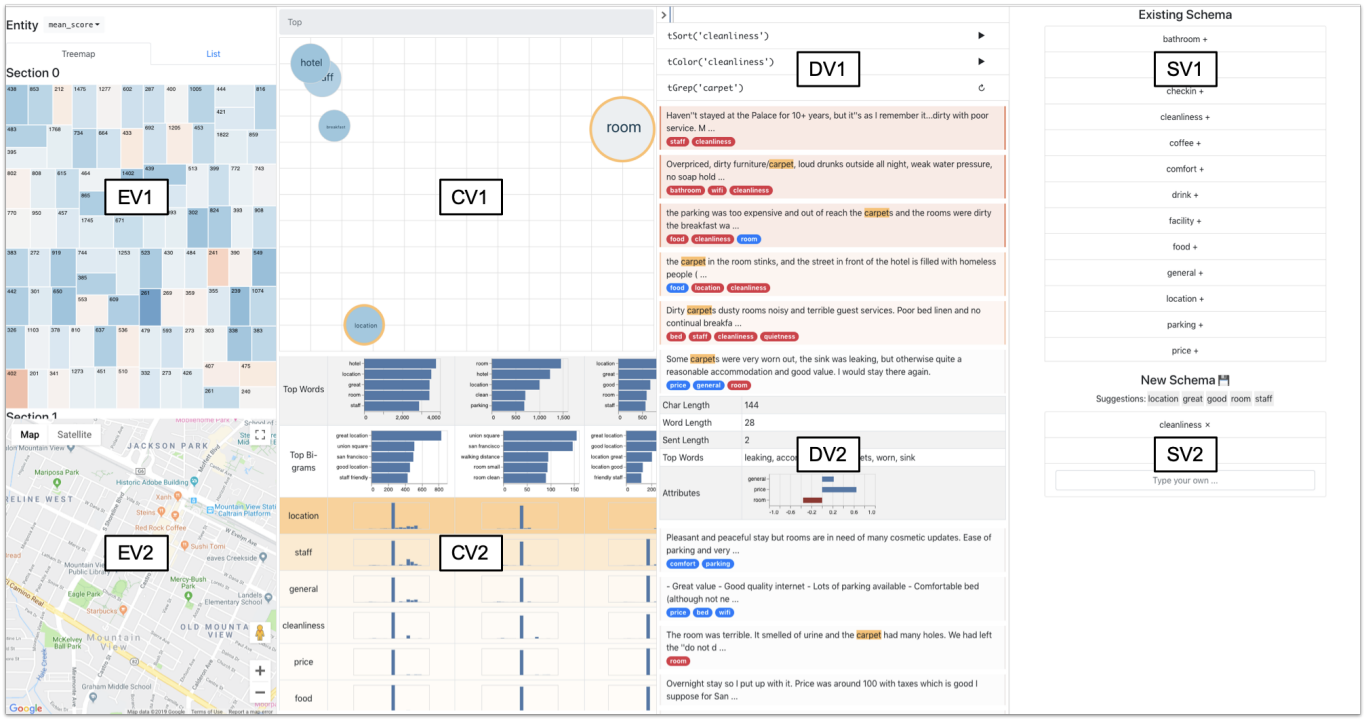


Figure 4: Screenshot of the Teddy user interface. From left to right we have: the Entity View (EV) displaying the entities mentioned in reviews, the Cluster View (CV) for exploring aggregate statistics over clusters of reviews, the Detail View (DV) for viewing and filtering/sorting individual reviews, and the Schema Generation View (SV) for recording aspects of interest from the reviews. These views are described in the Section **Teddy System User Interface**.

sparsity. Consider suggestion mining, out of millions of examples, only one or two percent of them are suggestions. It is very inefficient. Our goal in cleaning is to make the dataset more focused and ignore the negative sentences."

Data Labeling Roughly half of the participants (6/14) identified data labeling as a bottleneck across all task categories. We also found that data scientists with more seniority were less likely to identify data labeling as a bottleneck than more junior data scientists. A possible explanation for this phenomenon is that labeling tasks are more likely to be delegated to less experienced team members.

Crowdsourcing For those utilizing crowdsourcing (9/14), one third reported that crowdsourcing was a bottleneck, due to the difficulties of designing proper crowdsourcing tasks and verifying that crowdsourced labels were accurate. Participants working on entity representation were the most likely to identify crowdsourcing as a bottleneck.

Data Exploration Four participants, working in either the entity extraction or entity representation tasks reported that initial data exploration was a significant bottleneck. One participant identified the particularly high stakes associated with this task, which further slowed down its implementation: *"once we agree on a schema with the client we can't change it anymore, so we need to be really careful to get it right the first time."*

Modeling Two participants working on entity extraction tasks identified model training as a bottleneck to their workflow, expressing frustration with slow runtimes for training algorithms. An additional participant specified setting up an in-house GPU environment as their main bottleneck, but mentioned that training the model might become a more significant bottleneck in itself once the GPU has been set up. Generally, participants did not seem to consider model training to be a significant bottleneck to their work, even when asked about it specifically. One participant said *"once you start the model, it takes one hour, two hours, and then you're done."*

Even when working to debug machine learning models, participants seemed more focused on fine tuning training datasets than on changing model architectures or hyperparameters for training algorithms. These responses suggest to us that while data scientists will certainly still benefit from advances in the state of the art of machine learning algorithms, there is also a need for tools that better facilitate the collection, preparation, and exploration of data.

Pipeline Management Two participants, both working in entity extraction tasks, identified bottlenecks in managing the various parts of their pipeline. In particular, they found it tedious to switch between tasks, which would often require them to go from using command line interfaces to writing scripts or examining data files manually and back again. They also expressed frustration with having to manage multiple overlapping data files, and losing track of how the files corresponded to each other.

Desired Capabilities Participants identified a variety of software features that they felt would help them with their data analysis work (See Fig 3)

Data Cleaning One data scientist identified that they would like additional tools for data cleaning, specifically saying that they wanted a tool to *"filter out content-poor phrases"* that are not relevant to the extraction task at hand.

Data Labeling Six out of the twelve data scientists working with labeling specified that they would like to use software with better data labeling capabilities. Similar to the pattern identified in the bottlenecks section, we found that data scientists with fewer years of experience were more likely to express a need for better labeling tools.

Data Exploration The four participants that identified data exploration as a bottleneck also expressed a need for tools that would allow them to either generate or quickly reference a schema corresponding to their dataset.

Advanced Search Functions Three participants, working in either entity extraction or entity representation tasks, requested integrated search functions that would allow them to search for specific lexical features in their dataset, as opposed to simple string-matching. Two participants also expressed a need for a search function that would allow them to find reviews similar to a selected review.

Pipeline Management Despite the relatively small number of participants identifying pipeline management as an explicit bottleneck to their work, requests for pipeline management tools were rather common when we asked participants to describe capabilities that they wished they had, particularly for participants working on either classification or entity extraction tasks. We also found that more experienced data scientists were more likely to express a need for pipeline management tools. Similar to the opposite trend in data labeling, this trend could potentially be explained by the delegation of broad-scoped project management tasks to more senior data scientists.

Visualizations Data visualization tools were a common request. This need was even more frequently expressed by participants working on entity representation tasks.

DESIGN CONSIDERATIONS

Informed by the interview study results above and our own experience along with a prior data exploration paradigm [37], we derived 5 design considerations for the Teddy prototyping system.

D1 Users should be able to explore, inspect, and compare clusters of review text based on the similarity of the texts as well as the opinions expressed within them.

D2 Users should be able to explore domain-specific aspects of reviews, and revise and build on existing extractor schemas in an iterative manner.

D3 Users should be able to manipulate the data programatically. In particular, through programmatic extensions, the prototype should support text search, regular expression matching, filtering, and sorting operations, and their visualizations.

D4 Users should be able to get an overview of all reviews or of reviews from a single entity.

D5 Users should be able to zoom in and out the data visually as well as semantically in a multi-scale fashion, accessing details for entities and reviews on demand.

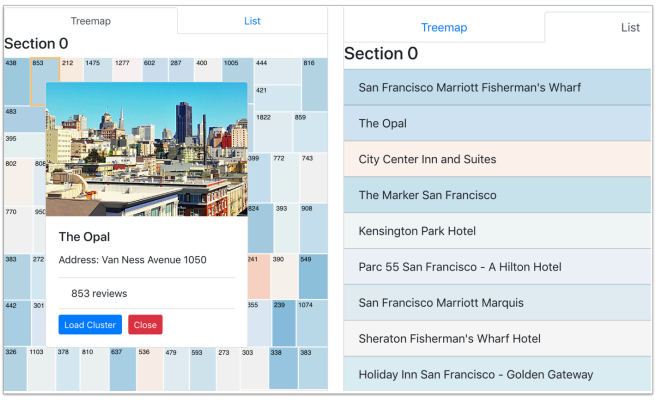


Figure 5: Entity View is used to load reviews and clusters for a specific product/service/location. Left: Treemap visualization and the entity card; Right: Entity list. The user can choose a heuristic to color the entities from the drop-down menu, for example overall rating or average sentiment score for a specific aspect.

TEDDY SYSTEM USER INTERFACE

With the proposed design considerations in mind, we developed Teddy, an interactive system that runs as a single-page web app (SPA), along with a backend server that serves data and provides various text analytics functionalities. The backend is implemented with Flask, Pandas, and scikit-learn, while the frontend is implemented with D3.js, Vega-Lite, and the React framework.

Teddy is designed to work with review datasets from any domain as long as the datasets have entities (e.g. hotels, restaurants, cell phones, mechanics, etc.) and associated review texts.

Backend Data Preparation

In order to sustain interactivity, Teddy precomputes much of the information displayed. We start with raw review text and (optionally) meta-information for the entities that reviews belong to, for example the entity names and locations. In the absence of entity information, Teddy disables the treemap and map views, but the rest of the application stays functional.

First, we feed the reviews into a semantic attribute extractor to generate feature vectors with semantic meanings on each dimension. A schema is required in this step to guide the attribute extractor, but it can be very basic since a goal of the system is to facilitate the iterative formation of schemas (Fig 1). At the end of extraction, each review will have a feature vector with dimensions corresponding to the aspects specified in the schema. Values of a review vector are, typically, continuous sentiments extracted for each aspect from the review.

After extraction, we independently apply the following procedure to the set of all reviews and the sets of reviews belonging to the same entity: Given a set of reviews, we run the k-means algorithm with $k=5$ on review vectors to create an initial clustering, which represents the first zoom level for the set. We then iteratively apply the k-means algorithm to individual clusters to create sub-clusters, forming additional zoom levels. For each cluster we also compute the following summaries: the average number of characters, words, and sentences; top N words and

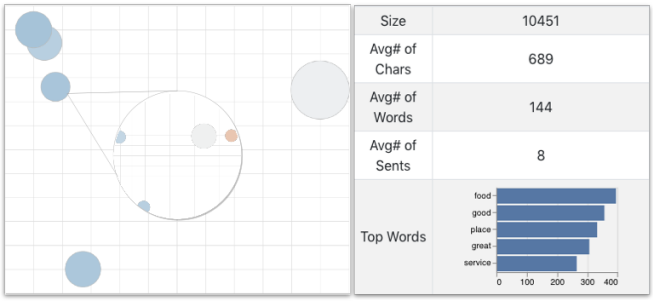


Figure 6: The Cluster View is a scalable way to explore aggregate statistics of reviews. Double clicking on a cluster shows a new clustering on the selected subset of reviews (shown here as an inset). The color represents the average sentiment of reviews in the cluster, while the size represents the number of reviews. When a cluster is selected, data such as frequently occurring words and histograms over the sentiment for each aspect are displayed (example on the right), and the user can select a second cluster to compare these statistics (shown in Fig 8).

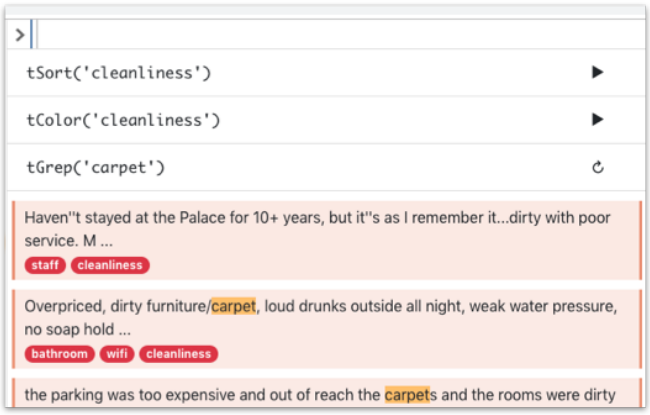


Figure 7: Teddy command line interface for operations on reviews (outlined in Table 1). In this example we explore the relationships between cleanliness and how people talk about the carpet condition of a hotel. We filter reviews with the keyword "carpet", then sort and color the reviews based on the cleanliness attribute, revealing that people write about the carpet in mostly negative contexts.

bi-grams (based on TF-IDF); histogram distributions of each attribute, and the averaged attribute scores. Since each dimension has semantic meaning, the summarized information is helpful to quickly understand a cluster. At the end of this step, we have a k-means based hierarchical clustering for the whole dataset as well as the sets of reviews for each entity. The entities are also clustered with k-means for the treemap view (Fig 5), using the average review vectors per entity.

Frontend Layout and Functionalities

Once the data ready for analysis, the user can start the application. Teddy's interface has four main sections or "Views". Teddy's layout is different from spreadsheet tools or computational notebooks, which have linear layouts of either data or code. We designed it this way to allow a more flexible representation, and to provide more information to users in one glance. The views, shown in Fig 4, are outlined below.

Entity View shows the entities and allows users to select and load entity-specific clusters. Users can toggle between two kinds of visualizations with the tabs on the top (Fig. 4-EV1). Treemap visualization groups similar entities and allocates different sizes of blocks based on how many reviews inside, while the list visualization allows the user to clearly see entity names. Each entity item is colored by an averaged attribute score selected from a drop-down menu. When the user clicks on an entity, the map on the bottom displays its location (Fig 4-EV2). An information card also pops up with a preview picture of the entity, the entity address, and a button to load hierarchical clusters of all the reviews for that entity (Fig 5).

Cluster View enables efficient navigation of the dataset by showing clusters of similar reviews and their statistical summaries. Users can explore the cluster hierarchy by zooming in and out, or compare different clusters to get insights (D5). Initially Teddy shows the clusters of the entire dataset for a quick overview (D4). These clusters are plotted as circles of different colors in a 2D space, based on PCA projections of the cluster's centroids (Fig 4-CV1). The radius of the circle shows the size of the cluster, the color (from red to blue) shows the averaged sentiment score of the review texts in the cluster, and the word in the middle of the circle shows the most frequent word in the cluster. When the user clicks on a cluster circle, Teddy shows the summarizing information in the bottom table (Fig 4-CV2). By default the table also shows the same set of statistics for the whole dataset. By clicking on two clusters while also pressing the "Command" key on the keyboard, the user can compare them side-by-side in the table. Teddy will calculate the histogram distances of each attribute, and highlight the ones that are the most different. This indicator can help users to locate interesting parts to keep an eye on later when they see the raw review texts. To zoom into the next level of clusters (show the clusters inside a cluster), users can simply double click on the circle (Fig 6). To zoom out, users can click on the directory style links on the top (D5).

Detail View shows raw review texts sampled from whichever cluster(s) the user has clicked on. To avoid flooding the screen, Teddy requests 10 reviews at a time, and a "Load More" button is available to request 10 more review items. The reviews are truncated into 100 characters to improve readability, but clicking them shows the full text and a panel with summarizing information similar to the clusters (Fig 4-DV2). The background color of each review item shows the weighted average of all available attributes, which are also displayed in tags at the bottom of each item. This helps the user to do quick analysis without reading the entire text. The user can also sort or filter the reviews based on an attribute or specific pattern. Teddy provides a set of commands (shown in Table 1) to programmatically query reviews (D3). Users can type commands in the input prompt (Fig 4-DV1) at the top then hit "Command+Enter" to run. Teddy first runs the command on the current review flow on the frontend. This helps the user to debug and tweak their command input quickly. When the user is satisfied with the command, they can use the "Remote Run" button on the right to evaluate the command on the server end and get results from the whole dataset. Every new command will be performed on

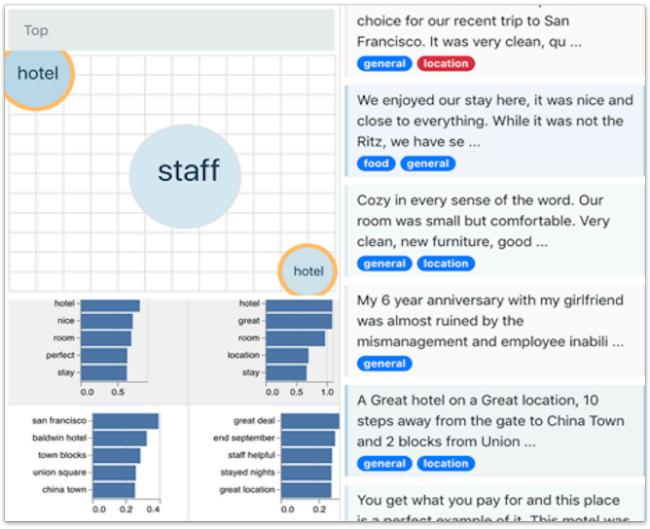
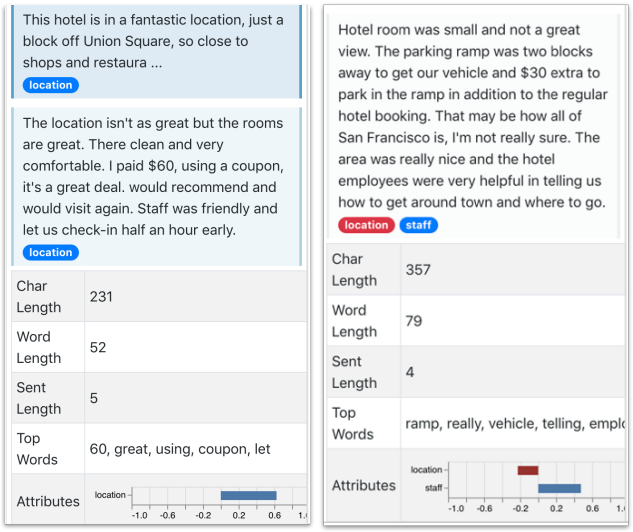


Figure 8: Two clusters have been selected from the visualization in the top left corner. In the bottom left, we can see histograms corresponding to the top-5 words and top-5 bigrams for each selected cluster. Along the right, we see excerpts of reviews tagged with their sentiment category bubbles.



(a) (b)

Figure 9: (a) The review display. One review has been expanded to display its full text as well as metadata. (b) Another review display.

the results of the last one, so the user can gradually build a set of operations to re-produce the discovery process.

Schema Generation View is a workspace for building a new schema. Whenever users discover new attributes, they can fill them in the "New Schema" section (Fig 4-SV2), or they can import attributes from the existing schema (Fig 4-SV1). To aid this discovery, Teddy displays the most frequent words in the selected cluster(s) as new schema suggestions. Users can edit the new schema and export it to be used as an input to re-run the data preparation step. By repeating this process, users can refine the schema to better understand the dataset (D2).

USAGE EXAMPLES

As an evaluation of Teddy, we observed two data scientists, Rosa and Chao, in our lab apply the system for completing various review analysis tasks. Neither Rosa nor Chao was a participant in our interview study.

Exploratory Analysis of Hotel Reviews

We tasked Rosa with analyzing customer reviews of a set of hotels to identify specific areas in which the hotels are exceeding expectations or under-performing, and to extract quotes from reviews that exemplify the hotels' strengths and weaknesses. Additionally, we asked her to help debug an attribute extractor that we are developing, and which is being used in conjunction with Teddy, and to provide suggestions for a new attribute schema so that other researchers on her team can develop more fine grained quality scores for her dataset.

Command	Functionality	Remote
tSort	*Sort reviews based on an attribute*	Y
tFilter	*Filter reviews based on an attribute and a predicate function*	Y
tGrep	*Search for reviews which texts have a certain pattern*	Y
tColor	*Change the background color of review items based on a given attribute*	N
tReset	*Reset previous operations*	Y

Table 1: Teddy commands for customizing the reviews shown. The "Remote" column corresponds to whether the command can also be run offline by the server on all the reviews.

Upon opening the app, Rosa sees that the data has been sorted into three clusters, two of which are labeled "hotel" and one of which is labeled "staff". Rosa wants to learn more about the differences between the two "hotel" clusters, so she command-clicks on them to get a side by side comparison (Fig 8). While the "top words" histograms are similar, the "top bigrams" histograms show a significant difference: cluster 1 has more bigrams about locations, whereas cluster 2 has more bigrams about the hotel's services. Rosa wants to find information about the hotels' rooms, she zooms in on cluster 2 by clicking on it.

Now, Rosa focuses her attention to the reviews listed from this cluster. She wants to focus on reviews that mention room-quality, so she searches for "room" in the search bar and clicks on the first review. She sees a useful quote and copies it for her project: *"the rooms aren't huge, granted - but they are clean and well tended."* (Fig 9a).

Rosa now wants to find some fields where the hotels are under-performing. She looks at the average attribute ratings for each cluster, and notices that while still positive, cluster 3 has a much lower than average sentiment score for "location". She starts looking through the reviews themselves, and notices that several have been flagged with a red "location" bubble, which means that the attribute extractor that she is using has identified that the review expresses a negative opinion about the location of the hotel. However, when she clicks on one review to read it in full, she sees that the review actually has no complaints about the hotel's location, and even says that it is a "convenient

location and good value" (Fig 9b). She sees that the review actually complains about the view; Rosa looks at her schema and notices that "view" is a category separate from "location". She makes a note of this error and sends it to her coworkers working on the extraction algorithm.

Finally, Rosa wants to define a new schema of attributes relevant to the hotel so that her coworkers can improve the performance of their attribute extractor. She notes based on the average attribute histogram that "facility," "food," "general," "location," and "staff" are important features that have strong sentiments associated with them, so she copies them into the new schema field (Fig 10). She also notices that a lot of reviews discuss public transit options near the hotel, but her current schema does not support this feature. Thus, she adds "public-transit" to her schema, and clicks save, which generates a text file that she can send to her coworkers. Armed with these insights, Rosa is ready to help the attribute extraction team build a better algorithm.

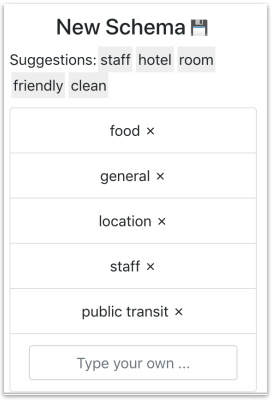


Figure 10: The new schema field.

Designing Extraction Schema for Restaurant Reviews

We asked Chao to prepare a quick demonstration, using a dataset of restaurant reviews, to demonstrate an attribute extractor and review summarization algorithm that he has been working on. Chao's algorithm already worked for hotel reviews but he needed to design a new schema for the restaurant domain in order to start applying his algorithm restaurant reviews.

Chao's extractor operates in two stages: discovering aspect-opinion phrase pairs, and matching the pairs to categories in a domain-specific schema. For example, an extracted pair can be (*"carpet"*, *"a bit filthy"*) and categorized into the "cleanliness" hotel schema attribute. To apply his extractor to restaurant reviews, he needs to design a new schema, a set of attributes (or features) along with opinion scales, specific to the restaurant domain in order for his extractor to provide useful results. First, Chao applies only the first stage of his hotel extractor to the restaurant reviews. Based on his knowledge of restaurants, he creates a very basic schema consisting of "food", "drink", and "facility", and trains a model to categorize the extractions. He loads these reviews with extractions into Teddy to evaluate the representation power of this schema. The first thing Chao notices is that one of his review clusters has "service" as a commonly occurring word. A grep search for "service" confirms his suspicion that people discuss this aspect frequently, so he adds it to his new schema. By clicking on some restaurants and going through a similar discovery process of common terms people use, Chao discovers that his schema needs more granularity: "food" is divided into "food-quality" and "healthiness", "drink" is divided into "drink" and "alcohol", and by exploring reviews that mention service he discovers the aspect "wait-time".

Next, Chao labels some examples and trains a classifier for his new and improved schema. Again, he loads these new extractions into Teddy.

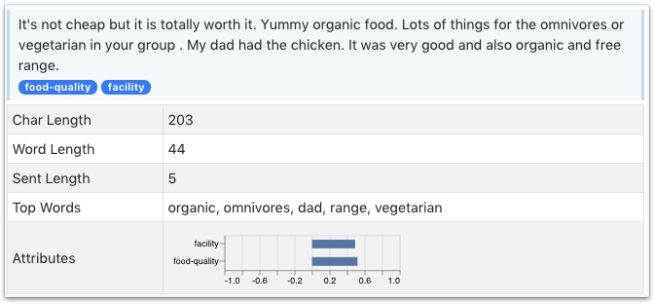


Figure 11: A quick scan of the review section shows that for this restaurant, most of the "food-quality" extractions mention vegetarian food, which should be separated to its own category in the schema.

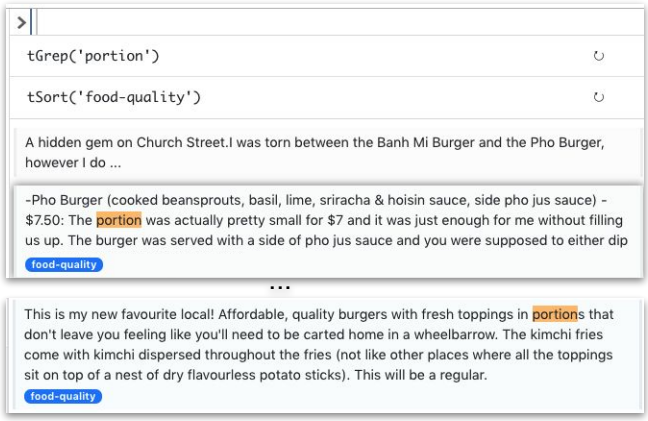


Figure 12: A grep search and sort operation shows that varied opinions in "food-quality" can be explained by this newly-discovered aspect relating to portion size.

Based on the frequently occurring yet varied terms used for the same valency, Chao decides that "food-quality" is still too general. He selects a restaurant that is an outlier with the highest score in it's cluster, and sorts the reviews by "food-quality." Several of the reviews mention "portion size", and a grep search confirms that this is a popular aspect, which he further confirms with another grep search over the full review dataset (Figure 12). Now Chao selects another restaurant with many "food-quality" extractions and finds that most of the reviews specifically mention plentiful vegetarian options (Figure 11). He additionally decides on a "location" aspect, aided by the map feature which helps him discover that this topic is mentioned more in certain neighborhoods.

Chao re-trains his extraction classifier and evaluates the results on his test set. Table 2 shows the accuracies of the restaurant attribute classifiers when trained on three schema versions with increasing number of attributes. Each classifier is trained on 5,000 aspect-opinion pairs by fine-tuning the uncased 12-layer BERT model [7] for 10 epochs and is evaluated on 1,000 labeled pairs. The three versions consist of 3, 7, and 10 attributes created as described above. Each schema also contains a special attribute "others" indicating examples that are not covered. The coverage(%) column shows the percentage of test examples that are not labeled as "others". By increasing the number of at-

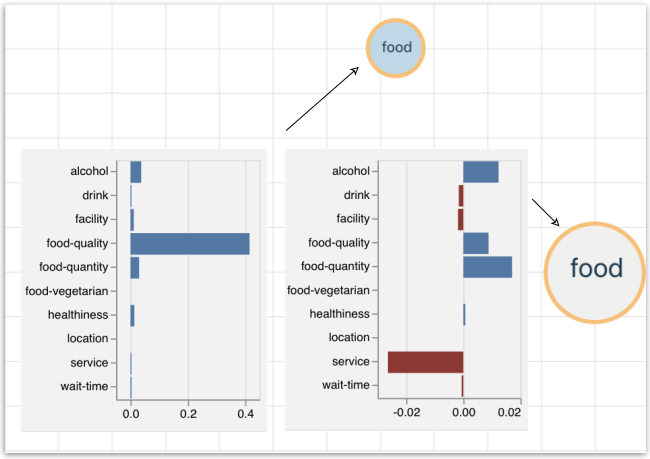


Figure 13: The newly-designed schema reveals the differences between the two largest clusters, which were unclear in previous iterations but are now largely explained by differences in service.

tributes from 3 to 10, the coverage increases by 14.2%. The last 3 columns show the classifiers' accuracy. Each column "Level-*i*" indicates the accuracy when evaluated at the granularity of schema-*i*. When a classifier is evaluated at a less fine-grained (easier) level, each predicted label is mapped to the attribute that contains the predicted label (e.g., "food-quality" to "food", or "alcohol" to "drink"). When evaluated at a finer-grained (harder) level than the classifier's schema, each predicted label is mapped to a single attribute (For example when schema-1 is evaluated on Level-3, a "drink" or "facility" prediction is only correct if those are the ground-truth labels, but a "food" prediction is only correct if the ground-truth is "food-quality", since "food" is not in the level-3 schema). Chao's results clearly show that as the schema becomes more carefully designed, the accuracy of the BERT classifier increases significantly (by 12.9% at Level-2 and by 16.9% at Level-3).

	#attr	coverage(%)	Level-1	Level-2	Level-3
Schema-1	3	78.4	90.9	72.6	68.0
Schema-2	7	92.6	88.9	84.5	80.3
Schema-3	10	**92.6**	90.6	**85.5**	**84.9**

Table 2: BERT classifiers' accuracy using different schemas.

Loading these new extractions into Teddy, he notices more granular differences between clusters. For example, the two largest clusters were already clearly separated by average sentiment, but now Chao observes that much of the difference in sentiment can be attributed to the aspect "service" (Fig 13). With his new schema, Chao can now prepare an extraction and summarization demonstration for a prospective client.

DISCUSSION AND CONCLUSION

The size and availability of user generated text as reviews on the Web are rapidly increasing with the dramatic expansion of e-commerce applications. The wealth and the potential utility of information in reviews has created interest in review analysis and mining.

In this paper, we first contribute findings from an interview study conducted with fourteen data scientists to better understand the current review analysis practices and challenges. To the best of our knowledge, the study presented here is the first interview study focused on review text analysis and mining.

Results suggest that data scientists performing review analysis are not very concerned about new model development, architectures, or parameter tuning and that they are largely satisfied with using existing language models. They often operate on an assumption that improving the quality of their training data is the most effective means to improve the performance of their models. On the other hand, they are handicapped by the lack of tools that would help them across different stages of data preparation.

Also, perhaps unsurprisingly, our study led us to the conclusion that it is unlikely to build a single tool that fits all data scientists' needs. Challenges and priorities vary across the roles and analysis goals of data scientists. For example, pipeline management and provenance is an important concern motivated by the cost of context-switching among senior/lead data scientists. On the other hand, junior data scientists are more concerned about data labeling and crowdsourcing as bottlenecks than senior data scientists are.

Results also indicate that data scientists spend most of their time on data preparation, lending additional support for the currently accepted general wisdom and extending it to the special case of review text analysis. However, our results also contribute further, finer-grained insights on data preparation for the review domain. For example, we find that the most time consuming or challenging data preparation steps are labeling, designing and specifying crowdsourcing tasks, and interactive exploration, not necessarily cleaning.

Through our interview study, we also find that data scientists lack interactive tools with which to quickly explore large collections of reviews together with results of extractors and models on these reviews. In response, we contribute Teddy, an interactive system to help data scientists gain insights into review text at scale along with fine-grained opinions expressed in reviews and enable to data scientists iteratively refine and improve their extraction schemas and models. We demonstrate the utility of Teddy in depth through two use cases carried out by data scientists. Informed by the interview study results, we believe that Teddy addresses an important need in review analysis pipelines of data scientists.

Increasingly, products or services purchased on the Web are converging to be defined by their reviews. Reviews (or any other user generated text referring to user experiences with entities, for that matter) are becoming distributional representations of these products or services, analogous to distributional semantics [11]. Research efforts such as Teddy into better tools for understanding and mining the rich information embodied in user generated text is therefore essential to the future of improved user experience on the Web. To encourage future research, we make Teddy as an open-source software and all the data collected from our interview study available at https://github.com/teddyauthors/teddy.

REFERENCES

[1] Anne Adams, Peter Lunt, and Paul Cairns. 2008. A qualititative approach to HCI research. In *Research Methods for Human-Computer Interaction*. Cambridge University Press.

[2] Basak Alper, Huahai Yang, Eben Haber, and Eser Kandogan. 2011. OpinionBlocks: Visualizing consumer reviews. In *IEEE VisWeek Workshop on Interactive Visual Text Analytics for Decision Making*.

[3] Steven Bird, Ewan Klein, and Edward Loper. 2009. Natural Language Processing with Python. O'Reilly Media.

[4] Ann Blandford. 2013. Semi-structured qualitative studies. In *The Encyclopediaof Human-Computer Interaction*. Interaction Design Foundation.

[5] Chaomei Chen, Fidelia Ibekwe-SanJuan, Eric SanJuan, and Chris Weaver. Visual analysis of conflicting opinions. In *2006 IEEE Symposium On Visual Analytics Science And Technology*. IEEE, 59–66.

[6] Çağatay Demiralp, Peter J. Haas, Srinivasan Parthasarathy, and Tejaswini Pedapati. 2017. Foresight: Rapid Data Exploration Through Guideposts. In *Proc. Workshop on Data Systems for Interactive Analysis (DSIA) at IEEE VIS*.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL*. 4171–4186.

[8] Luigi Di Caro and Matteo Grella. 2013. Sentiment analysis via dependency parsing. *Computer Standards & Interfaces* 35, 5 (2013), 442–453.

[9] Geoffrey Draper and Richard Riesenfeld. 2008. Who votes for what? a visual query language for opinion data. *IEEE transactions on visualization and computer graphics* 14, 6 (2008), 1197–1204.

[10] Cristian Felix, Anshul Vikram Pandey, and Enrico Bertini. 2016. TextTile: an interactive visualization tool for seamless exploratory analysis of structured data and unstructured text. *IEEE transactions on visualization and computer graphics* 23, 1 (2016), 161–170.

[11] John R. Firth. 1957. A Synopsis of Linguistic Theory, 1930-1955. In *Studies in Linguistic Analysis: Special Volume of the Philosogical Society*. Blackwell.

[12] Michael Gamon, Anthony Aue, Simon Corston-Oliver, and Eric Ringger. 2005. Pulse: Mining customer opinions from free text. In *international symposium on intelligent data analysis*. Springer, 121–132.

[13] Michelle L Gregory, Nancy Chinchor, Paul Whitney, Richard Carter, Elizabeth Hetzler, and Alan Turner. 2006. User-directed sentiment analysis: Visualizing the affective content of documents. In *Proceedings of the Workshop on Sentiment and Subjectivity in Text*. Association for Computational Linguistics, 23–30.

[14] Ming C Hao, Christian Rohrdantz, Halldor Janetzko, Daniel A Keim, Umeshwar Dayal, Lars Erik Haug, Meichun Hsu, and Florian Stoffel. 2013. Visual sentiment analysis of customer feedback streams using geo-temporal term associations. *Information Visualization* 12, 3-4 (2013), 273–290.

[15] Matthew Honnibal. 2015. spaCy: Industrial-strength Natural Language Processing (NLP) with Python and Cython. (2015).

[16] Siw Elisabeth Hove and Bente Anda. 2005. Experiences from conducting semi-structured interviews in empirical software engineering research. In *Proc. METRICS*. IEEE.

[17] Minqing Hu and Bing Liu. 2004a. Mining and summarizing customer reviews. In *KDD*. ACM, 168–177.

[18] Minqing Hu and Bing Liu. 2004b. Mining opinion features in customer reviews. In *AAAI*, Vol. 4. 755–760.

[19] Sean Kandel, Andreas Paepcke, Joseph M Hellerstein, and Jeffrey Heer. 2012. Enterprise data analysis and visualization: An interview study. *IEEE Transactions on Visualization and Computer Graphics* 18, 12 (2012), 2917–2926.

[20] Eser Kandogan, Aruna Balakrishnan, Eben M Haber, and Jeffrey S Pierce. 2014. From data to insight: work practices of analysts in the enterprise. *IEEE computer graphics and applications* 34, 5 (2014), 42–50.

[21] Soo-Min Kim and Eduard Hovy. 2004. Determining the sentiment of opinions. In *ACL*. 1367.

[22] Kostiantyn Kucher, Carita Paradis, and Andreas Kerren. 2018. The state of the art in sentiment visualization. In *Computer Graphics Forum*, Vol. 37. Wiley Online Library, 71–96.

[23] Jonathan Lazar, Jinjuan Heidi Feng, and Harry Hochheiser. 2017. *Research methods in human-computer interaction*. Morgan Kaufmann.

[24] Yuliang Li, Aaron Feng, Jinfeng Li, Saran Mumick, Alon Halevy, Vivian Li, and Wang-Chiew Tan. 2019. Subjective Databases. *PVLDB* 12, 11 (2019), 1330–1343.

[25] Bing Liu, Minqing Hu, and Junsheng Cheng. 2005. Opinion observer: analyzing and comparing opinions on the web. In *WWW*. ACM, 342–351.

[26] Bing Liu and Lei Zhang. 2012. A survey of opinion mining and sentiment analysis. In *Mining text data*. Springer, 415–463.

[27] Shixia Liu, Xiting Wang, Christopher Collins, Wenwen Dou, Fangxin Ouyang, Mennatallah El-Assady, Liu Jiang, and Daniel Keim. 2018. Bridging text visualization and mining: A task-driven survey. *IEEE transactions on visualization and computer graphics* 25, 7 (2018), 2482–2504.

[28] John Lofland and Lyn H Lofland. 1971. Analyzing social settings. (1971).

[29] Satoshi Morinaga, Kenji Yamanishi, Kenji Tateishi, and Toshikazu Fukushima. 2002. Mining product reputations on the web. In *KDD*. ACM, 341–349.

[30] Michael Muller, Ingrid Lange, Dakuo Wang, David Piorkowski, Jason Tsay, Q Vera Liao, Casey Dugan, and Thomas Erickson. 2019. How Data Science Workers Work with Data: Discovery, Capture, Curation, Design, Creation. In *CHI*. ACM, 126.

[31] Daniela Oelke, Ming Hao, Christian Rohrdantz, Daniel A Keim, Umeshwar Dayal, Lars-Erik Haug, and Halldór Janetzko. 2009. Visual opinion analysis of customer feedback data. In *2009 IEEE Symposium on Visual Analytics Science and Technology*. IEEE, 187–194.

[32] Bo Pang and Lillian Lee. 2008. Opinion Mining and Sentiment Analysis. *Foundations and TrendsÂő in Information Retrieval* 2 (2008), 1–135.

[33] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up?: sentiment classification using machine learning techniques. In *EMNLP*. 79–86.

[34] Samir Passi and Steven J Jackson. 2018. Trust in Data Science: Collaboration, Translation, and Accountability in Corporate Data Science Projects. *Proceedings of the ACM on Human-Computer Interaction* 2, CSCW (2018), 136.

[35] Ana-Maria Popescu and Oren Etzioni. 2005. Extracting Product Features and Opinions from Reviews. In *EMNLP*. 339–346.

[36] Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. ELRA, Valletta, Malta, 45–50. `http://is.muni.cz/publication/884893/en`.

[37] Ben Shneiderman. 1996. The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations. In *Proceedings of the 1996 IEEE Symposium on Visual Languages*.

[38] Axel J Soto, Ryan Kiros, Vlado Kešelj, and Evangelos Milios. 2015. Exploratory visual analysis and interactive pattern extraction from semi-structured data. *ACM Transactions on Interactive Intelligent Systems (TiiS)* 5, 3 (2015), 16.

[39] Kazuma Takaoka, Sorami Hisamoto, Noriko Kawahara, Miho Sakamoto, Yoshitaka Uchida, and Yuji Matsumoto. 2018. Sudachi: a Japanese Tokenizer for Business. In *Proc. LREC*.

[40] Yingcai Wu, Furu Wei, Shixia Liu, Norman Au, Weiwei Cui, Hong Zhou, and Huamin Qu. 2010. OpinionSeer: interactive visualization of hotel customer feedback. *IEEE transactions on visualization and computer graphics* 16, 6 (2010), 1109–1118.

[41] Koji Yatani, Michael Novati, Andrew Trusty, and Khai N Truong. 2011. Review spotlight: a user interface for summarizing user-generated reviews using adjective-noun word pairs. In *CHI*. ACM, 1541–1550.