SyntaViz: Visualizing Voice Queries through a Syntax-Driven Hierarchical Ontology

Md Iftekhar Tanveer
University of Rochester
Rochester, NY, USA
itanveer@cs.rochester.edu

Ferhan Ture
Comcast Applied AI Research
Washington DC, USA
ferhan_ture@comcast.com

Abstract

This paper describes SYNTAVIZ, a visualization interface specifically designed for analyzing natural-language queries that were created by users of a voice-enabled product. SYNTAVIZ provides a platform for browsing the ontology of user queries from a syntax-driven perspective, providing quick access to high-impact failure points of the existing intent understanding system and evidence for data-driven decisions in the development cycle. A case study on Xfinity X1 (a voice-enabled entertainment platform from Comcast) reveals that SYNTAVIZ helps developers identify multiple action items in a short amount of time without any special training. SYNTAVIZ has been open-sourced for the benefit of the community.

1 Introduction

Voice-driven interactions with computing devices are becoming increasingly prevalent. Amazon's Alexa, Apple's Siri, Microsoft's Cortana, and the Google Assistant are prominent examples. Google observed that mobile devices surpassed traditional computers in terms of search traffic (Sterling, 2015), and that 20% of mobile searches are voice queries (Pichai, 2016). As opposed to the keyword-based shorter queries received by web-based search engines, voice-enabled natural language processing (NLP) systems deal with longer, natural-language queries. This raises the question of how such data should be utilized for continuous improvements of the underlying methods.

In this paper, we introduce SYNTAVIZ, a web interface for visualizing natural-language queries based on a syntax-driven ontology, thereby enabling its user to quickly gain insights on the statistical and structural properties of large-scale customer-generated data. We provide use cases of SYNTAVIZ on a dataset of 1 million unique voice queries issued by users of the Xfinity X1 entertainment platform of Comcast—one of the largest cable companies in the United States with approximately 22 million subscribers. We are planning to make the source code of SYNTAVIZ freely available as a contribution to the community.

2 Related Work

There is a growing body of literature that deals with various text visualization techniques: Paulheim and Probst (2012) provided a survey of various ontology-enhanced user interfaces. Most prior work utilizes topic modeling to group and/or partition the collection in a way that lexically different yet semantically similar queries are clustered together. Wei et al. (2010) proposed using Latent Dirichlet Allocation (LDA) to induce topic models for an exploratory text analytics system. Hoque and Carenini (2016) utilized topic modeling and sentiment analysis to visualize blog texts and associated comments. A combination of latent topic analysis, discriminative feature selection and various ranking methods was developed by Singh et al. (2017) as part of a visualization interface for large text corpora. Dasiopoulou et al. (2015) developed a predicate-argument based ontology for exploring text elements and their relations. Their framework was demonstrated on excerpts from patent documents. Despite the similarities with our proposed approach, none of the techniques directly suits the use case described in the next section.

3 Methods

SYNTAVIZ is a web interface that visualizes a collection of queries by grouping them in human-understandable clusters. The clusters are formed in a hierarchy based on the dependency parse of the tree of the natural language queries. The design rationale and an elaborate description of solution are given in the following subsections.

3.1 Need Finding

The project stemmed from a need to optimize the engineering process of the existing intent understanding system. We arranged an informal discussion with the engineering team to understand the current workflow for developing the software that understands the intent of the users from their voice queries. The discussion provided clues that the current design is based on an "open-loop" design process, where the team starts by coming up with many variations of voice queries that would describe a certain intent. For example, for the CHANNELTUNE intent, the first step would be to imagine query patterns such as "tune to ...", "switch over to ...", or simply "channel ...". The next step is to implement an algorithm to convert every such query to its corresponding intent (e.g., "tune to HBO" $\Rightarrow$ CHANNELTUNE).

The design process does not involve any analysis of historical query logs. This creates the risk of potential gaps between the capabilities of the developed software and the users' expectations. Even though the engineers agreed that such data analyses would help design a better product, the lack of suitable techniques and/or a visualization/analysis tool resulted in this suboptimal process. Our conclusion from these discussions was that the development team was not able to gain useful insights about the incoming voice queries. For example, assigning the word "show" to carry an intent of watching *TV shows* (noun) might negatively impact the queries where the users say "show me ..." (verb). It is difficult to identify these adverse effects without a way to group and visualize similar queries.

Therefore, the need finding exercise revealed the importance of grouping or clustering the queries into semantic structures that reduces the informational overburden (Jones et al., 2004) and ranks the failure points of the system in proportion to real query traffic. There are at least three benefits we have identified:

1. Software developers can access the information needed to design the most effective NLP algorithms.
2. Editors can identify bugs and patterns of logical errors in the deployed system.
3. Product managers can prioritize development efforts in a data-driven fashion, by easily identifying real cases that degrade user experience.

Ideally, all the data should be summarized and presented through a visual interface with minimum cognitive burden. We sought to find an ontology suitable for these needs. Although "topic modeling" is used as the *de facto* ontology for many text visualization projects (Wei et al., 2010; Hoque and Carenini, 2016; Singh et al., 2017), it is not suitable for this case because the voice queries are much shorter in length compared to the typical text documents used in topic modeling. In addition, one of the reasons to form clusters of sentences is to visualize the systemic mistakes made by the intent understanding system. From this point of view, it appeared natural to cluster the syntactically similar queries together.

3.2 Ontology

Based on the need finding analysis, we realized that syntactic structures would be a great basis for summarizing the voice queries for the current scenario. Dependency parse trees (Jurafsky and Martin, 2014) capture the syntactic relationships by representing the dependencies among the words with directed arrows. Figure 1 shows the dependency trees for two example sentences. Noticeably, the dependent words (represented by arrow heads) modify or complement their parents (e.g. *watch* complements *want* because it answers "want what?"). It is possible to summarize a number of voice queries into hierarchical clusters using these structures, by putting the queries containing the same *tree fragments* into the same cluster. Each cluster then could be divided into subclusters based on the structures of the subtrees within. This ontology allows the users to (1) identify common patterns of error by the NLP system because syntactically similar queries are likely to all be victims of the same weaknesses, and (2) observe the statistical distributions of the queries to the level of granularity they wish to see.

3.3 Dataset

We collected real voice queries issued by the users of Xfinity X1 TV control system within a period of seven days—totaling 32 million queries with an average length of 2.1 words. As we are interested in more "natural" queries, we filtered out the sentences with fewer than five words. In ad-

Algorithm 1 Building the hierarchical clusters.

```
Input: Query $q$, Dependency tree $tree$, Cluster $clust$
Output: Updated cluster $c'$
  procedure BUILD_CLUST(q, tree, clust)
      for node in tree do
          Get (word, POS, dependency) triplet from node
          Put q in clust[word, POS, dependency]
          if node has subtree then BUILD_CLUST(q, subtree, clust[word, POS, dependency])
```

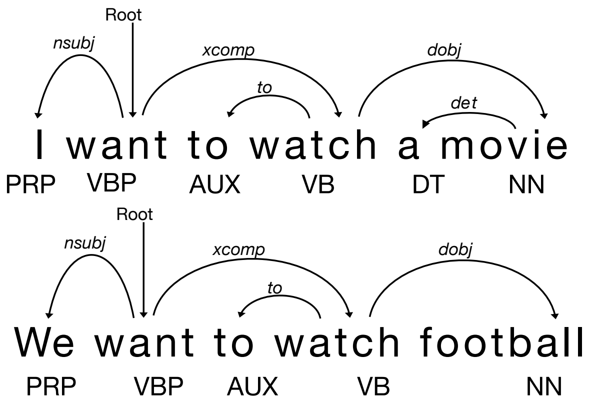


Figure 1: Examples of dependency trees.

dition, we used trigram language model (Kneser and Ney, 1995) probability to obtain queries with higher natural language probabilities. The final dataset contains 1.02 million unique queries.

3.4 Clustering

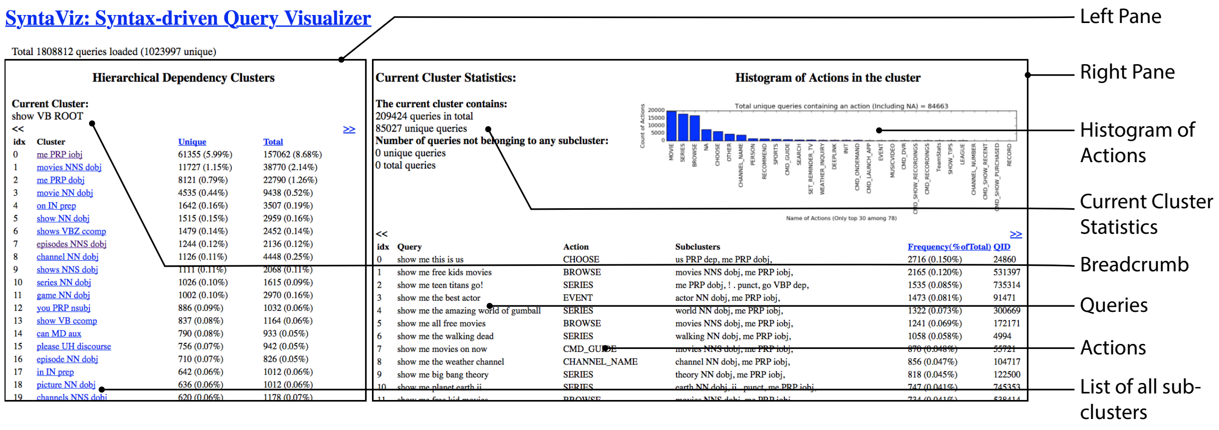
The algorithm for formulating the clusters is shown in Algorithm 1. For the convenience of discussion, let us consider the following two sentences: "I want to watch a movie" and "We want to watch football games". We extract the corresponding dependency tree using Google's "SyntaxNet" (Andor et al., 2016) parser as shown in Figure 1. Representing the tree as a directed acyclic graph, each token becomes a node and each dependency relation is a directed edge between two nodes.

Our clustering algorithm (Algorithm 1) is designed to group queries together if they have the same subpath (including the part-of-speech) in their dependency tree. Clusters are formed in a recursive manner, starting from the root. At every step, new sub-clusters are constructed based on the neighboring nodes (i.e. nodes receiving an edge from the current node). The landing node and the outgoing edge becomes the label of the sub-cluster: a triplet of the word, the part-of-speech, and the dependency relation (e.g., `a/DT/det`), which contains all queries with the same path starting from the root. For example, notice that both sentences in Figure 1 fall in the same cluster named `want/VBP/ROOT`. Under this cluster, there are three sub-clusters: `watch/VB/xcomp`, `I/PRP/nsubj`, and `we/PRP/nsubj` where the first sub-cluster contains both sentences but the other two contains only one sentence each. The sub-cluster `watch/VB/xcomp` is again divided into smaller sub-clusters. SYNTAVIZ allows the user to jump back-and-forth between the different levels of granularity through its novel interface.

3.5 Interface

The SYNTAVIZ interface is shown in Figure 2. It consists of two panes: the left pane lists labels of all the sub-clusters for the current cluster; the right pane lists all queries within the current cluster associated with other relevant information. We sort the list of sub-clusters on the left side by number of unique queries (descending)—this helps to prioritize clusters with highest impact. Clicking on the label of a sub-cluster allows the user to navigate deep into the clusters containing increasingly (syntactically) similar queries. A breadcrumb shows the current position in the hierarchical cluster and also allows the users to navigate back to the upper level clusters. In the right pane, each query contains the action taken by the existing intent understanding system. This allows the users to quickly explore the failure points of the system. The queries are also accompanied by the frequencies of their occurrences in the dataset, and a list of sub-clusters that each query belongs to.

Given the syntax-driven query ontology, we needed to decide the ranking of the sub-clusters (left pane) and queries (right pane) as well. In order to prioritize high-impact queries, we sort the list on the right side by a descending frequency of occurrences in the dataset. This arrangement provides an idea of the distribution of the queries to the user because the most prevalent ones are ranked higher in the order. In addition, a histogram of all the actions taken by the intent understanding system is shown on the upper right, which makes it possible to quickly notice any red flags in terms of system correctness.



SyntaViz: Syntax-driven Query Visualizer

Total 1808812 queries loaded (1023997 unique)

Hierarchical Dependency Clusters

Current Cluster:
show VB ROOT
<< >>

idx	Cluster	Unique	Total
0	me PRP iobj	61355 (5.99%)	157062 (8.68%)
1	movies NNS dobj	11727 (1.15%)	38770 (2.14%)
2	me PRP dobj	8121 (0.79%)	22790 (1.26%)
3	movie NN dobj	4535 (0.44%)	9438 (0.52%)
4	on IN prep	1642 (0.16%)	3507 (0.19%)
5	show NN dobj	1515 (0.15%)	2959 (0.16%)
6	shows VBZ ccomp	1479 (0.14%)	2452 (0.14%)
7	episodes NNS dobj	1244 (0.12%)	2136 (0.12%)
8	channel NN dobj	1126 (0.11%)	4448 (0.25%)
9	shows NNS dobj	1111 (0.11%)	2068 (0.11%)
10	series NN dobj	1026 (0.10%)	1615 (0.09%)
11	game NN dobj	1002 (0.10%)	2970 (0.16%)
12	you PRP nsubj	886 (0.09%)	1032 (0.06%)
13	show VB ccomp	837 (0.08%)	1164 (0.06%)
14	can MD aux	790 (0.08%)	933 (0.05%)
15	please UH discourse	756 (0.07%)	942 (0.05%)
16	episode NN dobj	710 (0.07%)	826 (0.05%)
17	in IN prep	642 (0.06%)	1012 (0.06%)
18	picture NN dobj	636 (0.06%)	1012 (0.06%)
19	channels NNS dobj	620 (0.06%)	1178 (0.07%)

Current Cluster Statistics:

The current cluster contains:
209424 queries in total
85027 unique queries
Number of queries not belonging to any subcluster:
0 unique queries
0 total queries

<< >>

idx	Query	Action	Subclusters	Frequency(%ofTotal)	QID
0	show me this is us	CHOOSE	us PRP dep, me PRP dobj,	2716 (0.150%)	24860
1	show me free kids movies	BROWSE	movies NNS dobj, me PRP iobj,	2165 (0.120%)	531397
2	show me teen titans go!	SERIES	me PRP dobj, ! . punct, go VBP dep,	1535 (0.085%)	735314
3	show me the best actor	EVENT	actor NN dobj, me PRP iobj,	1473 (0.081%)	91471
4	show me the amazing world of gumball	SERIES	world NN dobj, me PRP iobj,	1322 (0.073%)	300669
5	show me all free movies	BROWSE	movies NNS dobj, me PRP iobj,	1241 (0.069%)	172171
6	show me the walking dead	SERIES	walking NN dobj, me PRP iobj,	1058 (0.058%)	4994
7	show me movies on now	CMD_GUIDE	movies NNS dobj, me PRP iobj,	870 (0.048%)	55721
8	show me the weather channel	CHANNEL_NAME	channel NN dobj, me PRP iobj,	856 (0.047%)	104717
9	show me big bang theory	SERIES	theory NN dobj, me PRP iobj,	818 (0.045%)	122500
10	show me planet earth ii	SERIES	earth NN dobj, ii . punct, me PRP iobj,	747 (0.041%)	745353
11	show me free kid movies	BROWSE	movies NNS dobj, me PRP iobj,	724 (0.041%)	528414

Figure 2: The complete SYNTAVIZ interface.

4 User Study

We conducted quantitative, qualitative and heuristic experiments with the users of the SYNTAVIZ interface. There were 8 participants in the study among which four were editors/annotators from the voice NLP and the customer service project, three were developers of the voice NLP project, and one was the project manager of the customer service project. All of the participants were familiar with the voice queries and the existing intent understanding methods.

4.1 Quantitative Evaluation

Several metrics were computed based on responses to statements shown in Table 1 where the participants rated whether they agree, disagree, or remain neutral with respect to each statement. We use Q1 for measuring the usefulness, Q2, Q3 for usability, Q4, Q5 for efficiency, and Q6 for the practical value of the system.

The results of the evaluation are shown in Figure 3. Six out of eight participants agreed that SYNTAVIZ helped them to quickly find the major patterns of mistakes. This shows that SYNTAVIZ provides useful information to achieve its intended use—finding patterns of mistakes in the existing system. Seven participants disagreed with the statement that the interface is complex (Q2) and six agreed that no help is needed to use the interface (Q3). The evidence that SYNTAVIZ does not cause cognitive overload or require outside help implies that it is intuitively usable. In terms of efficiency, six participants agreed that it does not take much time to learn (Q4), half of the total participants seem to be confused at some point while using the interface. We analyzed this issue in the subjective evaluation which is described in the following subsection. Finally, the majority of the participants (5 out of 8) mentioned that they will use SYNTAVIZ in their regular workflow.

4.2 Qualitative Evaluation

We asked for subjective opinions from the participants for a qualitative evaluation of SYNTAVIZ. Each participant was asked to write about the positive features, negative features, important future additions, and the impact of the interface in their current workflow. These comments provided us a spectrum of information regarding the SYNTAVIZ interface which are discussed below.

Positive Aspects. A number of positive aspects of the SYNTAVIZ interface were mentioned. Almost all the participants appreciated that the combination of syntactic ontology and the action histogram makes it easier to understand where the existing system is failing.

> "The interface makes it clear that syntactic information is very useful for drilling down the long tail of VREX commands. Once I have drilled down a few levels deep, I start to see utterances where the system failed."

They also appreciated the frequency-based ranking scheme which helped them to "know the weight of the queries and decide priorities". In addition, every participant commented positively about the simple interaction experience.

> "Simple, easy-to-learn design. Did not take long to get used to the layout and multiple sorting and drill-down features. Almost no load time. Useful and specific information."

Measures for Quantitative Evaluation
Q1: SYNTAVIZ allowed me to quickly find major patterns of mistakes in the deployed system
Q2: The interaction was unnecessarily complex.
Q3: I did not need assistance to use SYNTAVIZ.
Q4: SYNTAVIZ does not take much time to learn.
Q5: I never felt confused/lost while using the interface.
Q6: If available, I'll use SYNTAVIZ in my regular workflow.
Measures for Qualitative Evaluation
Please write some useful aspects of SYNTAVIZ
Please write some aspects of the interface that could be improved upon. Please also suggest why the proposed approach would be an improvement over the existing solution.
Please enlist some possible features that should be added in the future iterations of the interface. Please also explain why.
What immediate changes/benefits can this tool bring in your current workflow?

Table 1: All the measures for quantitative and qualitative evaluation of SYNTAVIZ.

Negative Aspects. The quantitative analysis revealed that some of the users felt confused or lost while using the interface. The underlying reason for such rating became clear in the qualitative evaluation. Many participants mentioned that the labels of the clusters are not immediately understandable. The cluster names included parts-of-speech tags from Penn Treeback Project (Santorini, 1990) (e.g. NN for Noun, JJ for Adjective), and dependency type information from Universal Dependency Project (Nivre et al., 2016). The participants commented that these codes might confuse the users. They suggested a pop-up legend or glossary would be helpful in this situation.

> "I believe that the naming can be hard to understand for many people. QID is not necessarily intuitive and for anyone who operates on a higher level, they may not be aware of what a subcluster is or what the "nn root" means after the groups on the first page."

A few participants mentioned that it is problematic to navigate back to the parent clusters by clicking the *back* button multiple times, suggesting a *Back to Home* button. In addition, these comments suggest that the existence or use of the breadcrumb is not obvious in the interface.

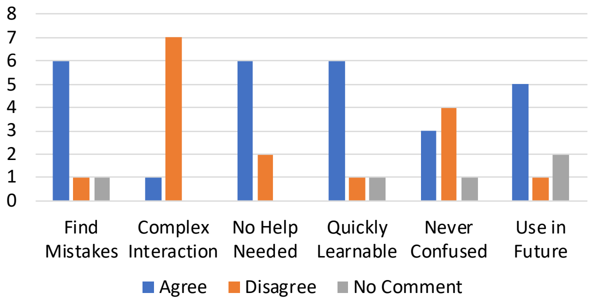


Figure 3: Results of quantitative evaluation of the SYNTAVIZ interface.

Possible Future Additions. We asked the users to suggest future additions to the interface and received a number of ideas that can improve the interface and user experience. For example, several users asked for a search capability within the interface. Interaction on SYNTAVIZ is currently exploratory and thus the experience is similar to browsing through a tree. However, the users might need to search for a specific keyword and navigate quickly to the corresponding clusters.

> "Allow the user to search for a word (or words) eg the word 'card': `card NN ROOT|credit NN nn|` and `change VB ROOT|card NN dobj|` — I wanted to find instances of 'credit card' as some of those queries may have different labels. Allow the user to filter on Action so that the user can view only queries for that action; Allow users to export queries that may need to be fixed"

Participants also mentioned some other useful features; for instance, adjusting the time span for the dataset, searching and filtering for a specific action, and comparing multiple clusters. All these ideas are implementable in a future iteration while staying within the design of the proposed visual framework.

Impact. Many participants mentioned that SYNTAVIZ will allow them to identify outliers in the data or systemic errors in the deployed system, and that it will help them to identify unsupported patterns. This was considered useful for feature prioritization from a product point of view:

> "Feature prioritization will be a big problem to solve from a product point of view. From research point of view this give great understanding of how the data

is getting clustered among queries and find potential bugs and loop holes in our NLP system."

4.3 Heuristic Evaluation

In this section, we describe a usage scenario based on the informal discussions with the developers of the X1 platform: Upon studying the ranked lists, one developer noticed a cluster labeled `record/VB/ROOT`, which refers to the group of 4587 unique queries that share the root verb "record". One of the largest sub-clusters is `oscars/NNS/dobj`, referring to the group of queries where the customers were interested in recording the Oscars event.[1] Clicking on this sub-cluster provides great insight into how customers use voice, as well as queries mishandled by the deployed system. The histogram on the upper right immediately revealed that around 60% of queries were categorized by the system as a `SEARCH` intent, whereas only 20% were linked to the correct (`RECORD`) intent. Discussions among engineers resulted in several action items to fix such issues and provide a more consistent user experience.

These errors might have seemed obvious after a short session with SYNTAVIZ, but such information is usually buried deep under the large pile of popular queries. We observed first-hand that a syntax-driven ontology is very useful at highlighting many other patterns in how customers use natural language to query the platform.

5 Conclusions

In conclusion, SYNTAVIZ is a novel visualization interface for datasets with a large number of unique natural-language queries. It is based on a hierarchical clustering of all queries, designed from a syntax-driven ontology of the dataset.

Evaluation of SYNTAVIZ shows strong evidence that the system provides useful information for finding the errors in the existing system. Based on real needs of a major voice-enabled entertainment platform, our preliminary implementation has shown great promise to improve software development cycles by providing useful analytics that help fix bugs, as well as prioritize future efforts in a data-driven fashion.

By releasing the source code of SYNTAVIZ, we are hoping to share these benefits with many other teams across the industry. The complete source code of SyntaViz is available in
https://github.com/Comcast/SyntaViz

[1]2017 Oscars occurred during the week of our collected dataset.

References

Daniel Andor et al. 2016. Globally normalized transition-based neural networks. In *ACL'16*, volume 1, pages 2442–2452.

Stamatia Dasiopoulou et al. 2015. Representing and visualizing text as ontologies: A case from the patent domain. In *VOILA@ ISWC*, page 83.

Enamul Hoque and Giuseppe Carenini. 2016. Multiconvis: A visual text analytics system for exploring a collection of online conversations. In *IUI'16*, pages 96–107. ACM.

Quentin Jones et al. 2004. Information overload and the message dynamics of online interaction spaces: A theoretical model and empirical exploration. *Information systems research*, 15(2):194–210.

Dan Jurafsky and James H Martin. 2014. *Speech and language processing*, volume 3 (In Prep.) Chap. 14. Pearson London.

Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on*, volume 1, pages 181–184. IEEE.

Joakim Nivre et al. 2016. Universal dependencies v1: A multilingual treebank collection. In *LREC*.

Heiko Paulheim and Florian Probst. 2012. Ontology-enhanced user interfaces: A survey. *Semantic-Enabled Advancements on the Web: Applications Across Industries*, 214.

Sundar Pichai. 2016. Google i/o keynote. Presentation.

Beatrice Santorini. 1990. Part-of-speech tagging guidelines for the penn treebank project (3rd revision). *Technical Reports (CIS)*, page 570.

Jaspreet Singh et al. 2017. Structure-aware visualization of text corpora. In *Proceedings of the 2017 Conference on Conference Human Information Interaction and Retrieval*, pages 107–116. ACM.

Greg Sterling. 2015. It's official: Google says more searches now on mobile than on desktop. http://searchengineland.com/its-official-google-says-more-searches-now-on-mobile-than-on-desktop-220369. Accessed: 2017-08-16.

Furu Wei et al. 2010. Tiara: a visual exploratory text analytic system. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 153–162. ACM.