# VIZ-Wiki: Generating Visual Summaries to Factoid Threads in Community Question Answering Services

Tanya Chowdhury
IIIT-Delhi
tanya14109@iiitd.ac.in

Aashay Mittal
IIIT-Delhi
aashay14001@iiitd.ac.in

Tanmoy Chakraborty
IIIT-Delhi
tanmoy@iiitd.ac.in

## ABSTRACT

In this demo, we present VIZ-Wiki, a browser extension which generates an overview of summarizable threads in Question Answering forums. It reduces a user's effort to go through lengthy text-based, sarcastic and highly critiqued answers. Our tool can be used to collect community opinion from popular discussion sites like Quora, Yahoo! Answers, Reddit etc. as well as topic-centric ones such as Askubuntu, Stackoverflow. We rely on textual information of these forums to extract insightful summaries for a reader.

VIZ-Wiki provides users a pie-graph view marking popular choices when such a question link is raised. A button guides them to detailed statistics and relevant list of answers. It further highlights sentences relevant to an answer choice in the text. VIZ-Wiki deals with answers contradicted by other users, prioritizes highly-recommended ones and avoids sarcasm. We test our model on the factoid questions dataset of Yahoo! Answers and obtain a macro precision of 0.6 on displayed answers and a macro recall of 0.69, beating the baseline significantly. To the best of our knowledge, *VIZ-Wiki is the first attempt to visualize answers for questions in community question answering services.* **In the spirit of reproducibility, we have released the code and a demonstration video public at** http://goo.gl/cyx3EF **and** https://youtu.be/XNmRa_jtmC8 **respectively**.

## CCS CONCEPTS

• **Information systems → Information retrieval query processing**; *Information retrieval*; Information extraction;

## KEYWORDS

Factoid questions; Summarization; CQA services

## 1 INTRODUCTION

Community question answering (CQA) services have of late gained a lot of popularity. Readers browse through Quora, Reddit in leisure time. Stackoverflow is immensely popular amongst the programming community to discuss common errors and optimal strategies. Community forums such as Stackexchange and Mathoverflow exist to cater to specific scientific communities.

**Motivation:** Most of these forums are open to join and post. They are usually unmoderated and often unstructured. Few questions have answers running in hundreds, and it is difficult for the readers to browse through all to obtain opinion. Also being open, these forums are often subject to slurs, hatred, and used as a medium for propaganda. Most forums allow users to support an answer through an 'upvote', and the most upvoted answer is awarded the best answer. Users often write lengthy paragraphs for factoid queries and deviate from the original topic. Little work exists to summarize these threads in order to filter out useful information from these knowledge bases.

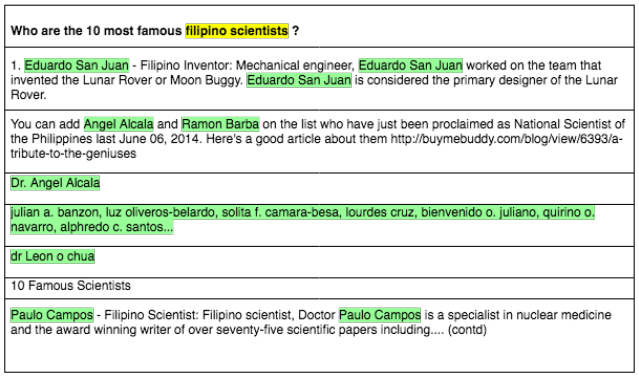


**Figure 1: Question thread from Yahoo! Answers (question noun headword in yellow and annotated answers in green).**

The concept of Answer-Wikis[1] in CQA was introduced by Quora. It is a bulleted list allowing the community to collaborate on an answer summary or an aggregated answer for a particular question. According to Quora, these wikis are aimed at building a comprehensive collection of uncontroversial, factual information. A bulleted list of answers and their visual representation for factoid questions, would make it easier for a user to understand the answers at a glance. To the best of our knowledge, Answer Wikis on Quora are manually curated, which requires unanonymous edits and staff moderation[2]. A user is able to create an Answer Wiki only when the question has at least 3 answers.

**System Architecture:** VIZ-Wiki is an online tool which automatically generates a visual summary of answers for factoid questions posted in CQA forums (as shown in Figure 1). It follows a novel architecture, right from the data-preprocessing step to the presentation of summary results (see Figure 2). When a CQA question is loaded with VIZ-Wiki, a button is shown, by clicking on which the question type is determined and the answers are parsed. The sentiment of the question is gauged and answer sentences with opposite sentiments are removed. An attempt is made to find candidate answers via Named Entity Recognizer (NER), Anaphora Resolver and Dependency Parser. Answer phrases with *nsubj* relation with

[1]https://www.quora.com/What-is-an-Answer-Wiki-on-Quora-1
[2]https://goo.gl/m9LrRY


*WWW '18 Companion, April 23–27, 2018, Lyon, France*

ACM ISBN 978-1-4503-5640-4/18/04.
https://doi.org/10.1145/3184558.3186986

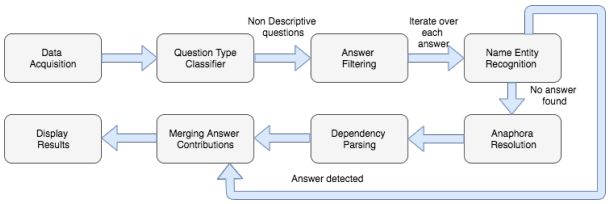


**Figure 2: System architecture of VIZ-Wiki**

the noun headword are marked as candidate answers. Candidate answers from different user-written answers are merged, and the most popular choices in the community are displayed through a pie-graph (see Figures 3 and 4).

Since we did not receive permission to scrape Quora, we build a generic model and test it on Yahoo! Answers (with suitable permission). Experimental results with human judgment show that VIZ-Wiki performs significantly well in extracting a meaningful list of answers, achieving a macro precision of 0.6 on displayed answers and a macro recall of 0.69 .

## 2 RELATED WORK

Previous work mostly attempted to find similar questions in CQA forums [5]. A *similar question database*[3] was also made available by Quora for analysis on this problem. However, it only comprises of similar question pairs, and it is not permissible to scrape corresponding answers from Quora. Liu et al. [7] analysed the quality of answers and suggested an optimal strategy to choose the best answer. Nakov et al. [10] focused on ranking of answers in these online forums based on their content and quality.

Not much work has been done in generating answer overviews for questions in CQA forums. Pande et al. [12] attempted to solve the problem by adding missing valuable information to the "best answer". Song et al. [13] summarized non-factoid answers in CQA services. They employed a sparse coding based summarization strategy based on document expansion.

A significant amount of research has been done in extracting answers for factoid questions given a corpus of relevant documents. Wang et al. [14] wrote a survey on answer extraction techniques for factoid question answering.

VIZ-Wiki is different from others in many aspects. It analyzes the sentiment of the question to remove opposing sentiment answers, which are otherwise detected by a Named Entity Recognizer. To deal with question-type classifier's and NER's failure in detecting an answer, VIZ-Wiki uses anaphora to resolve the text followed by dependency parsing. Unlike previous work, VIZ-Wiki creates visual summaries and bulleted lists which can be understood at a glance.

## 3 METHODOLOGY AND FRAMEWORK

In this section, we describe individual modules of VIZ-Wiki. Figure 2 shows the overall system framework of VIZ-Wiki.

[3]https://data.quora.com/First-Quora-Dataset-Release-Question-Pairs

### 3.1 Data Acquisition

We choose our dataset from the official factoid Queries dataset[4] provided by *Yahoo! Answers*[5] (with suitable permission). Also since the official factoid questions dataset primarily contains one correct answer per question (not survey questions), we took only 15% of our final dataset from them and the rest was randomly taken from Yahoo! Answers. When a user clicks on the VIZ-Wiki button on a question link page, we scrape the question and pass it on to the next module (question type classifier).

### 3.2 Question Type Classifier

Li et al. [6] suggested that all factoid questions can be classified into 6 coarse classes. We build a supervised question type classifier to identify which class a question belongs to. We use the dataset provided by Li and Roth, containing 5,500 hand-tagged questions to train our model. Each question is labeled as one of the following 6 coarse-grained categories – 'abbreviation', 'entity', 'description', 'human', 'location', and 'numeric'. Here, we train a supervised model on this dataset. We use training features such as words of the question, their POS tags, named-entities, the Wh-word, the noun-head chunk (the first noun following the Wh-word) and the verb-head chunk (the first verb following the Wh-word). We also add Word2Vec [9] information to take into consideration semantic similarity amongst different question words. We feed these features to an SVM classifier[6] and use standard grid-search for hyper-parameter optimization. We validate our model with a test set of 500 questions released in TREC'10[7]. We then run this pre-trained model on our dataset. If the question type is identified as 'description', our pipeline does not attempt to find an answer for that question; otherwise we proceed to the next module (answer type identification).

### 3.3 Answer Filtering

Certain answers in CQA forums may deviate from the actual point. E.g., one of the answers to the question "What is your least favorite Oscar winning film?" [8] goes like: "Faves: Halle Berry's for Monster's Ball. Even though she forgot to mention ...(contd)". As a result, we analyse the sentiment of the question and answer statements and rule out mismatched sentences from further processing. We use Serendio [11] to classify the question into 'Positive', 'Negative' or 'Neutral'. Serendio offers a normalized sentence sentiment score per class using Sentiwordnet [1] on feature words. The winner class is the class with the maximum score. We define a sentiment as strongly 'Positive' or 'Negative' if the dominant class score in a sentence is greater than 0.4. For such sentiment polar questions, we analyze the sentiment of its answer statements. We remove occurrences of strongly 'Negative' sentiment sentences from questions classified strongly 'Positive' and vice-versa.

Answers usually comprise of: one word, one phrase, list of phrases or paragraphs. We directly mark one word, one phrase

[4]VIZ-Wiki can be employed on Quora threads too. However, scraping Quora isn't permissible. Hence we were unable to include our analysis on the Quora dataset.
[5]https://answers.yahoo.com/
[6]We experimented with many classifiers such as decision Tree, Logistic Regression, K-NN, and got the best results from SVM.
[7]http://trec.nist.gov/data/web10.html
[8]https://answers.yahoo.com/question/index?qid=20130218073751AAG6zGa

and list of phrases as candidate answers. It is impossible to process huge paragraphs of text at run time. As a result, we extract important sentences from lengthy answers. We use a naive algorithm for important sentence extraction as follows. We run a POS-tagger on the answers. Next, we assign a score to all sentences based on its constituent words. We assign a weight of 2 to each proper noun and a weight of 1 to each common noun (singular/plural) [9]. We define a threshold and assign it a value proportional to the answer length and calculated score. Sentences with score above this threshold are marked important. We then proceed with a list of important sentences to the next module.

## 3.4 Named Entity Recognition

Wang et al. [14] mentioned that all answers to factoid questions could be detected by NER. In this step, we apply Stanford NER [4] to: one phrase long answers, answers with a bulleted-list of phrases and important sentences extracted from paragraph-based answers. Stanford NER provides three types of named-entity classifiers – a 3-class NER (Organization, Location, Person), a 4-class NER (Organization, Location, Person, Misc) and a 7-class NER (Location, Organization, Date, Money, Person, Percent, Time). We observed that if our question type belongs to one of the classes provided by the 3-class classifier, then an answer was more likely to be detected by the 3-class NER than the 4 or 7-class NERs. As a result, if the question type is one of 'human', 'entity' or 'location', we use the 3-class NER. If the question type is 'entity' and we do not find an answer with the 3 class tagger, we roll back and employ the 7-class NER. In other cases, we directly use the 7-class NER. This model seems to give the best results (F-score of 0.7) on our data.

## 3.5 Anaphora Resolution

We use BART toolkit [2] on paragraph-based answers for anaphora resolution. Note that we consider entire paragraphs for anaphora resolution instead of important sentences, in case no answer is found yet. The REST-based web service of BART returns co-reference chains with each chain having a unique SetID attribute. We replace pronouns in the text with the nouns that they refers to. The next module, i.e., the dependency parser is used to extract answers in anaphora resolved text.

## 3.6 Dependency Parsing

Next we identify the noun headword in the Question. For this we run a dependency parser on the question statement. The word which has a subject relation with the Wh-word, directly or via a determiner, is tagged as the noun headword. In case there exists no Wh-word in the question, we set hand-based rules to determine the noun head chunk.

We then convert the answer to a list of sentences. We iterate over this list and generate dependency trees for each sentence. These dependency trees are stored as a list of dependency relations, i.e., three element tuples. We look for dependency relations common between the question and answer sentences. In case there is a match, we look for the noun headword in the sentence. If the noun headword exists, then the word with which it has a subject relation is a candidate answer. In case a direct subject relation does not exist,

[9] https://medium.com/@acrosson/summarize-documents-using-tf-idf-bdee8f60b71

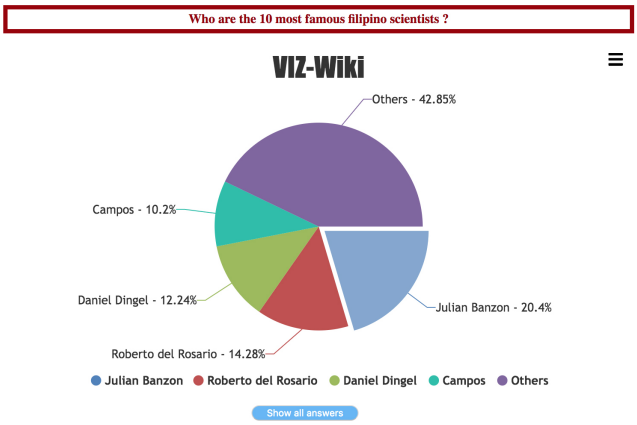


**Figure 3: Primary view of VIZ-Wiki UI.**

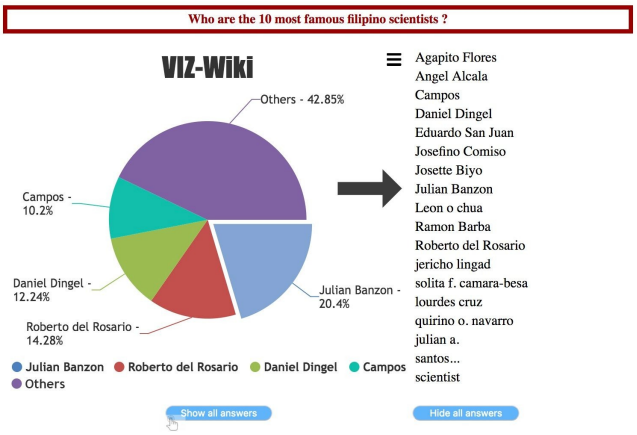


**Figure 4: Extended view of VIZ-Wiki UI.**

we add all nearest noun candidates via determiners to the list of candidate answers.

We tested with two dependency parsers: Stanford Dependency Parser [3] and Stanford CoreNLP Dependency Parser [8]. The CoreNLP parser seemed to perform marginally better in our case.

## 3.7 Merging of Answers

At the end of the pipeline, we obtain a list of candidate answer tuples for a particular question; one tuple from each community-user answer. We now create a histogram with every candidate-answer as the key, and a count of number of actual answers in which it has appeared as the value. In this process, we find that often candidate answers, due to spelling variation, receive different keys. As a result we set a minimum edit distance (2), below which we club histogram keys and add their counts. We then send the candidate answers and their counts to the display module of VIZ-Wiki.

## 3.8 Display

We pick the top answers that cumulatively contribute to approximately 60% of the results and display their names and percentages in a pie chart. We club all other answers into an 'Others' category. In case of less that 5 answers, we plot all answers in the chart.

We name this sub-class of candidate answers as display answers. This is followed to avoid overcrowding of pie chart with multiple low-frequency candidate answers. A button directs the user to a comprehensive list of less popular answers (see Figures 3 and 4). We use *CanvasJs*[10], a responsive HTML5 charting library, to visualize pie-charts in VIZ-Wiki. When the user clicks on one of the pie chart labels, we highlight content related to that answer choice, in the text.

## 4 EVALUATION

For evaluating VIZ-Wiki, we built a dataset of 500 factoid questions from Yahoo! Answers. Since in the official Yahoo! answers factoid questions dataset, each question primarily has one correct answer (not survey questions), we took only 15% of our final dataset from them and the rest was randomly generated from Yahoo! Answers. Three annotators manually tagged the questions into one of the six categories as mentioned in Section 3.2 (inter-annotator agreement was 0.79). This forms the ground-truth for question type classification. Similarly, six annotators generated wikis for our dataset answers. The generated wikis do not take into account candidate answer popularity and ranking. Answers with multiple occurrences have been reported only once. An answer is marked to be a candidate answer if it is favorably referred to in at least half its occurrences. Again we consider this set of annotated Wikis as the ground-truth in the last stage, for evaluating final answers. We consider the output of Stanford NER as a **baseline system**.

| Sl. No. | Features Used | F-Score |
|---|---|---|
| (1) | Sentence unigrams | 0.52 |
| + (2) | POS-Tags of words | 0.58 |
| + (3) | Word2Vec vectors of sentence unigrams | 0.67 |
| + (4) | Noun & verb chunks and Wh-Words | 0.71 |

**Table 1: Performance of the question type classification module of VIZ-Wiki after the addition of each feature.**

### 4.1 Evaluation of Question Type Classifier

As stated earlier, VIZ-Wiki uses unigrams, POS tags of unigrams, Word2Vec representation of words, weighted noun & verb chunks and Wh-words features for question type classification. Table 1 shows the F-score of VIZ-Wiki Question Type Classifier with the addition of each feature. The Question Type classifier module of VIZ-Wiki achieves a F-score of 0.71.

### 4.2 Evaluation of Answer Extraction Model

We compare VIZ-Wiki with the baseline model by two relevance measures - (i) **Precision:** We find the precision of displayed answers as the fraction of correct answers among the set of answers returned. (ii) **Recall:** We calculate the recall value of the computed candidate answers against the hand tagged ground truth wikis as the fraction of correct answers which are returned. Note that we ignore answer ranking and popularity while computing precision and recall.

The accuracy is measured for each question. Table 2 shows the micro and macro statistics of the competing models. We observe that VIZ-Wiki outperforms the baseline by a significant margin for all the measures.

[10]https://canvasjs.com/

| Model | Mi-P | Ma-P | Mi-R | Ma-R |
|---|---|---|---|---|
| Baseline | 0.60 | 0.59 | 0.51 | 0.52 |
| VIZ-Wiki | 0.60 | 0.60 | 0.67 | 0.69 |

**Table 2: Performance of baseline and VIZ-Wiki for extracting final answers w.r.t Micro Precison (Mi-P), Macro Precision (Ma-P), Micro Recall (Mi-R) and Macro Recall (Ma-R).**

## 5 CONCLUSION AND FUTURE WORK

Question answering forums could deploy built-in visual summarizers to ease user browsing. We developed VIZ-Wiki that can automate this summarization process. VIZ-Wiki was further tested by human experts. The proposed framework can be extended to non-factoid description-based question threads.

Often questions on these online forums ask for procedures to do a task (eg: WikiHow). In future, we would try to come up with a model which gives a graph-based depiction of steps, combining significant answers via abstractive summarization.

## ACKNOWLEDGMENTS

The work was partially supported by the Ramanujan Fellowship, SERB-DST, Govt. of India.

## REFERENCES

[1] Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. 2010. SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining.. In *LREC*, Vol. 10. 2200–2204.

[2] Samuel Broscheit, Massimo Poesio, Simone Paolo Ponzetto, Kepa Joseba Rodriguez, Lorenza Romano, Olga Uryupina, Yannick Versley, and Roberto Zanoli. 2010. BART: A multilingual anaphora resolution system. In *Proceedings of the 5th international workshop on semantic evaluation*. ACL, 104–107.

[3] Marie-Catherine De Marneffe, Bill MacCartney, Christopher D Manning, et al. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC*, Vol. 6. Genoa Italy, 449–454.

[4] Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd annual meeting on association for computational linguistics*. ACL, 363–370.

[5] Jiwoon Jeon, W Bruce Croft, and Joon Ho Lee. 2005. Finding similar questions in large question and answer archives. In *Proceedings of the 14th ACM international conference on Information and knowledge management*. ACM, 84–90.

[6] Xin Li and Dan Roth. 2002. Learning question classifiers. In *Proceedings of the 19th international conference on Computational linguistics-Volume 1*. Association for Computational Linguistics, 1–7.

[7] Mingrong Liu, Yicen Liu, and Qing Yang. 2010. *Predicting Best Answerers for New Questions in Community Question Answering*. Springer Berlin Heidelberg, Berlin, Heidelberg, 127–138.

[8] Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP Natural Language Processing Toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*. 55–60.

[9] Tomas Mikolov, Quoc V Le, and Ilya Sutskever. 2013. Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168* (2013).

[10] Preslav Nakov, Doris Hoogeveen, Lluís Màrquez, Alessandro Moschitti, Hamdy Mubarak, Timothy Baldwin, and Karin Verspoor. 2017. SemEval-2017 task 3: Community question answering. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. 27–48.

[11] Prabu Palanisamy, Vineet Yadav, and Harsha Elchuri. 2013. Serendio: Simple and Practical lexicon based approach to Sentiment Analysis. In *proceedings of Second Joint Conference on Lexical and Computational Semantics*. 543–548.

[12] Vinay Pande, Tanmoy Mukherjee, and Vasudeva Varma. 2013. *Summarizing Answers for Community Question Answer Services*. Springer Berlin Heidelberg, Berlin, Heidelberg, 151–161.

[13] Hongya Song, Zhaochun Ren, Shangsong Liang, Piji Li, Jun Ma, and Maarten de Rijke. 2017. Summarizing answers in non-factoid community question-answering. In *WSDM*. ACM, 405–414.

[14] Mengqiu Wang. 2006. A survey of answer extraction techniques in factoid question answering. *Computational Linguistics* 1, 1 (2006).