Enhancing Relevance Ranking of Q&A Systems

Mikhail Boulgakov, Silvia Jinrong Gong, Shan Hannadige, Lavanya Satyan

December 16, 2017

1 Research Context and Problem Statement

With the advent of the digital age, the online shopping scene is at an all time high. E-Commerce trend analytics from 2017 reveal that a staggering 51 percent of Americans prefer shopping online. In fact, over 80 percent of Americans have made online purchases in the last month. Companies such as Amazon, Alibaba, and eBay have contributed to the growth of this online community by incorporating review and Q&A systems into their product purchasing platforms.

However, it can be difficult for users to find answers to their questions when the information they seek is not readily available in the Q&A section or the product description. Manually searching hundreds or even thousands of reviews can be time-consuming and inefficient.

This problem with Q&A systems has only partially been addressed by existing work. Numerous researchers have used document summarization and opinion mining techniques on product reviews [1, 2] to search for readily-available recapitulations of the product's specifications and similar previously-answered questions. Others have employed a ranking system that promotes reviews that are expected to be helpful to the top [3]. Unfortunately, while these methods are successful in filtering out low quality and "troll" reviews and answering simple, common questions, all of these approaches have little value when the answer to a query is hidden in non-featured reviews.

McAuley et al. [4] describe an approach to this problem in their framework *Moqa*, which uses machine learning techniques to match user questions with relevant reviews. However, their algorithm views reviews as discrete and unrelated entities and thus ignores metrics such as user reputation or the community-generated rating of each review. These metrics are significant when dealing with large datasets where low quality and spam reviews make up a significant portion of all reviews.

We will be solving the problem of finding reviews that are most relevant to a user's binary (yes/no) query and that are likely to contain correct information. We will then process these reviews to generate a succint yes/no answer to the question.

2 Proposed Solution

We propose building an extension to *Moqa* [4] that finds a binary answer to a user's question using product reviews. We will do so by finding high quality reviews which are likely to contain a correct answer to a user's query by taking into consideration the metadata associated with each review. Similar to McAuley et

al.'s model, our model will iteratively run the following two functions on every review for a particular product:

1. Voting Function: Determines the probability that the answer to the user's query is "yes" based on the information in this review.

2. Relevance Function: Determines the relevance of this review to the query.

We will then obtain a probability (0 to 1) that the final answer, based on all the reviews, is yes by performing a weighted average of the output of the voting function on each review using the output of the relevance function as the weight. A probability of 0.0-3.0 will signify that the answer to the user's query is likely No, 0.3-0.7 will signify that the answer is non-binary, and 0.7-1.0 will signify that the answer is likely Yes. Although the high-level structure of our model is similar to that of McAuley et al., we will be using different techniques to determine how relevant a review is to a query.

2.1 Voting Function

The first component of our proposed model is the voting function, which outputs the probability, based on a review, that the answer to a question is "yes". As an input, the voting function takes the question and review encoded in a bag-of-words representation—a data structure that stores the count of each word in a one-dimensional array, where each index corresponds to the word's position in a dictionary. The function then generates a probability from 0 to 1 that indicates whether the information contained in the review suggests that the answer to the question is yes.

For example, if a user query is "Can I take this speaker to the beach?", the voting function, when run on review "This item is not waterproof", will return 0 ("No"), while returning 1 ("Yes") on the review "Took item to the beach, had a blast". On the other hand, a review such as "This item can be submerged for 10 minutes, but after that it breaks" will get a score of around 0.5 ("Maybe"). Because this review contains two conflicting pieces of information, the algorithm is only 50% sure that the answer is yes (and, likewise, 50% sure that the answer no).

At the extremes, an output of 0 stands for a 0% chance that the answer is "yes" (meaning it is "no"), and an ouput of 1 stands for a 100% chance that the answer is "yes". However, most of the probabilities generated by the voting function will be somewhere in between.

To train the voting function, we will use a dataset of 1000 hand-labeled question-answer pairs that we have obtained from our research advisor. We will then perform a multiple regression in which the independant variable is each question and review pair (encoded as a bag-of-words) and the dependent variable is the labeled yes/no answer. The same method was used by McAuley et al. for their voting function [4].

2.2 Relevance Function

The relevance function works in conjunction with the voting function and returns how useful and correct an individual review is. In the previous example with the speaker, all 3 reviews will get a high relevance score, since they all answer the question, albeit in contradicting ways. On the contrary, the review "This speaker is red" will get a low score because it does not answer the query.

A good relevance function should be able to find reviews that contain an answer to a question. We will find relevant reviews by finding similarities between the words in the review and those in the query. To do this, we will represent the query and every single review as bag-of-words. We will then do a word by word comparison to determine which reviews are most similar to the question.

Reviews ranked highly by the relevance function also need to contain a correct answer to the question (as opposed to simply having an answer that may or may not be correct). We will determine how correct a review is by taking into account the validity of the information stated in the review when generating our relevance score. A high score would imply that a review is very likely to contain truthful information, and, as such, will be weighted higher when processing the reviews than a potentially more similar review with a lower relevance score. This score is influenced by a multitude of features, as described in the table below:

Feature	Description & Significance
Helpfulness of Review	Online platforms allow users to mark reviews that were helpful to them. Reviews that are given high "helpful" scores are very likely to contain correct and useful information.
User reputation	We can assign a reputation score to each user by analyzing their posting history and taking into account the number of reviews they have posted, their reviewer ranking, and the total number of helpful votes they have received on their reviews. In doing so, we can differentiate between experts, whose opinions we can trust, and trolls, whose reviews contain false and misleading information.
Time & Date	The time and date that a review was written is significant because the correctness of a review may fluctuate over time. For example, updates may improve the quality of a product, alter existing features, or add new ones.

Star Rating	The purpose of looking at the star rating left by the reviewer is to avoid overly biased information. A review with 1 star may contain useful information that is relevant to a query, but it may also contain lies written in a fit of rage by a disgruntled customer.
Tone	We can perform sentiment analysis to determine the tone of a review and determine which reviews are factual and which are mainly opinion motivated.

The final relevance function will contain a weighted combination of 1) the pairwise word similarity between the query and the review and 2) the confidence that the review contains the correct answer. We will determine how strongly to weight each of these subscores when testing and refining our model.

To train our relevance function, we will use supervised learning on the Q&A section extracted from Amazon product pages. For each question, we have two answers: the one generated by our model and the correct answer, which will be found in the hand labeled dataset. Ideally, we want these two answers to give us the same result. We will create an error function E that gives us the difference between these two answers, and we will slightly modify the parameters of our model—specifically the weights used in the voting and relevance functions—until we get the smallest error value (as close to zero as possible). To fully optimize these parameters, we will use a technique known as gradient descent, a common process in supervised training which finds the minimum value of a function.

It is important to note that not all products have enough relevant reviews to provide an answer to every possible question. If we find that all reviews have an unsatisfying relevance score in relation to a query, we consider it an edge case and make no prediction.

By treating our reviews as part of a larger structure and incorporating new features into our relevance measures, we hope to improve the accuracy of our methods and thus get a smaller percentage of misclassification than the *Moqa* model.

3 Evaluation and Implementation Plan

3.1 Evaluation Plan

Our evaluation plan consists of 2 main parts: We will 1) refine our model by testing its accuracy on a test dataset, and 2) compare our model to other question-answering approaches.

To refine our model, we will run it on a test dataset that we will web mine from Amazon. Specifically, we will break up all of the products we have extracted into 8 different categories—electronics, home and kitchen, sports and outdoors, tools and home improvement, automotive, cell phones, health and personal care, and garden products—and run our model on every single question contained in the Q&A section. Doing so will allow us to see which categories, if any, our model struggles with and which ones it excels at. We will then compare the binary (yes/no) answer generated by our model to the correct answer. We assume that the correct answer is the highest upvoted community answer found in the Q&A section. To convert it into binary form (so that we can compare it to the model's output), we use the approach described in "Summarization of yes/no questions using a feature function mode" [5]. This method helps us discard open-ended questions and has a 97% precision, which, although worse than a hand-labeled dataset, is accurate enough for our purpose.

Our goal when refining our model is to achieve the highest possible recall (proportion of all questions whose answer is "yes" that were correctly labeled by the model) and precision (proportion of all questions that the model answered as "yes" that were correctly labeled by the model). To do so, we want to determine the ideal combination of features to use in our relevance function (see the previous section for a detailed list of these features). We will perform a series of feature ablation experiments, a type of experiment where we systematically exclude features from our relevance function one at a time to determine which features are significant in improving the recall and precision of our model, and which ones may actually hinder it and overfit the training data. After performing these experiments to determine the ideal set of features, we should have a much smaller difference between the answers provided by our model and the correct answers.

To evaluate our model in relation to other models, we will compare it to other state-of-the-art methods that perform Q&A based tasks, including Moqa, Cosine Similarity, and Okapi BM25. We will run all of these models on the same dataset used in the first part of the evaluation process, and calculate the recall and precision for each category of products.

Our research would be a success if the improved relevance function's prediction is more accurate in terms of recall and precision than that of *Moqa*. *Moqa* is, on average, correct in 88% of cases [4]. Although relatively high (25% more accurate than cosine similarity and 5% more accurate than a linear combination of ROUGE and Okapi BM25), this percentage conveys that over 1 in 10 questions are wrongly answered. Ideally, we would want to drop this error to 1 in 100 or less; however, for this project, we are striving for at least a 5% improvement over *Moqa*.

3.2 Implementation Plan: Timeline

Fall Quarter 2017

- Weeks 2-3: Literature Search & Data Collection

  - Find relevant research papers from reputable publications on natural language processing, data mining, and machine learning
  - Extract the reviews, product descriptions, and Q&A pairs for each product on Amazon

- Weeks 4-10: Learn background machine learning techniques
  - Watch CSE 158 Podcasts (Recommender Systems and Web Mining) and complete the homework assignments
  - Attend CSE 291 (Trends in Recommender Systems and Human Behavioral Modeling)
  - Get familiar with Python and modules such as scipy and sklearn
- Weeks 5-8: Write, edit, and submit the research proposal
- Weeks 9-10: Prepare for the final proposal presentation
- Thursday, December 14: Present proposal for ERSP class + advisors

Winter Break

- Practice implementation of machine learning techniques on sample dataset (bilinear regression, classification, SVMs)

Winter Quarter 2018

- Weeks 1-4: Implement the Mixture of Experts model described by McAuley et al. [4] as our base model
- Weeks 5-8: Add new features to the model
  - Incorporate the following features into the relevance function: the review's helpfulness, the author's reputation, and the time the review was published
  - Train our model using a variety of different learning algorithms
    * Bilinear Regression
    * Support Vector Machines
    * K-Nearest Neighbors
    * other similar methods
- Weeks 7-10: Test the current model on the Amazon dataset and use the results to improve it (correctness testing)
  - Compute the recall and precision for each of the learning algorithms and relevance function variations to find out what works and what does not work

  - Continue reading literature and potentially try out new techniques

Spring Quarter 2018

- Weeks 1-2: Perform incremental improvements on model
  - Add new features to the model if time permits
  - Add more comprehensive tests
- Weeks 3-5: Evaluate the performance of our model
  - Compare the performance of our model to that of Moqa [4]
  - Compare our model to various other state-of-the-art relevance ranking approaches
    * Cosine similarity
    * Okapi BM25
    * Bilinear regression
    * other similar methods
- Weeks 6-10: Prepare for the Final Presentation
  - Make poster for presentation
  - Prepare for the final oral presentation

4 References

[1] Xinfan Meng and Houfeng Wang. 2009. Mining user reviews: from specification to summarization. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers* (ACLShort '09). Association for Computational Linguistics, Stroudsburg, PA, USA, 177-180.

[2] 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* (CIKM '05). ACM, New York, NY, USA, 84-90.

[3] Anindya Ghose and Panagiotis G. Ipeirotis. 2007. Designing novel review ranking systems: predicting the usefulness and impact of reviews. In *Proceedings of the ninth international conference on Electronic commerce (ICEC '07).* ACM, New York, NY, USA, 303-310.

[4] Julian McAuley and Alex Yang. 2016. Addressing Complex and Subjective Product-Related Queries with Customer Reviews. In *Proceedings of the 25th International Conference on World Wide Web* (WWW '16). International World Wide Web Conferences Steering Committee, Republic and Canton of

Geneva, Switzerland, 625-635.

[5] J. He and D. Dai. 2018. Summarization of yes/no questions using a feature function mode. In *Journal of Machine Learning Research*.

5 Revisions

Writing Style:

- Got rid of this and these in places where they were unnecessary.
- Fixed some sentences to be in active voice.
- Restructured some sentences to have better flow and transitions.

Proposed Solution:

- Discussed how we will be training our voting function (the classifier) in more detail.
- Discussed what the value outputed by the voting function is supposed to signify (how sure we are that the review is saying that the answer to the question is yes).
- Added a paragraph about how we will be combining the two parts of the relevance function (using a weighted average).

Evaluation Plan:

- Went into more detail about the dataset that we will extract (Q&As) and how we will use it to evaluate our model's output.
- Talked about the use of the voting function in converting user generated answers into binary form.