CHECKPOINT 6.864: SUGGESTING POSSIBLE ANSWERS TO NEW QUESTIONS IN TECHNOLOGY FORUMS

DIEGO CIFUENTES, BOJAN JOVESKI, EZZELDIN HUSSEIN

1. Introduction

Online public forums present a great place for sharing knowledge. Most of the forums are organized around QA pairs, consisting of a single question and various answers and relevant comments from the community. The appeal of these types the forums - the abundance of questions and their distributed nature, is also the source for some of their biggest problems. Many of them suffer from duplicate posts where the user post new questions that have already been answered. Most of the forums deal with this problem manually. They assign moderators that go through the newly posted questions and check if a semantically equivalent question was already asked. Alternatively, forums provide simple key-word search that show possibly related questions to the user. In this project, we will explore various NLP approaches from information retrieval, duplicate finding, and translation models to improve the performance and find relevant questions. A good solution to this problem has two benefits: on one side, it allows the users to easily search through the database, and it also reduces the duplication of the questions, increasing the quality of the forum.

1.1. **Formal problem.** The task in the formal setting is to the following. For a given search query q (e.g. "What is the best free anti-virus for Windows 8"), our task is to find the QA pair that treats the most similar (ideally the same) question. Each QA pair is composed of multiple items: a question (which in term has short title and description), tags (predefined categories), and a set of answers (with the possibility that a particular answer is chosen as the right or best one).

1.2. **Model.** The goal of our problem is to match a new query with an already existing QA pair that might solve the question. There are many information retrieval approaches for looking for a query that might be helpful for it. Unsupervised models typically assign a score to

Date: November 13, 2012.

each possible document depending on some similarity metrics. Once we have this score, the answer that we are looking for is just the QA pair that produces the maximal score. In this project, we are calculating the similarity score in two fundamentally different ways.

In the first attempt, we define various metrics that address various aspects of the forum, such as lexicographic, semantic or category closeness between the search query and the QA pair. Given these individual metrics, we try to find the best linear combination that maximizes the amount of correctly classified pairs, and use this weighted function.

One problem with this approach is that this model assumes that all queries come from the same underlying distribution. Therefore, instead of using weights independent of the search query, we are going to use a model where the weights will be dependent on some property of the query q.

1.3. **Data Set.** The particular forum that we're using is the Super User forum [6], a Question and Answer forum for computer enthusiasts. Super User (part of the Stack Exchange forums) contains various questions ranging from setting up the firewall configuration on Windows 7 to problems relating Adobe Reader. Each post in this forum has rich structure - in addition to having fields such as question, and a set of answers, each posts has additional information: tags (i.e. windows), vote count, state (open, resolved, closed), further comments and so on. Moreover, the data set [8] that we'll be using consists of 3000 duplicate QA pairs, providing the corpus for our analysis. Each duplicate is a separate post with a link to an already existing QA pair that answers the same question.

2. Background information

Basic approaches use a bag of words representation of both the query and the documents and compute the cosine similarity. It is standard to give priority to some terms by using factors such as the term frequency and the inverse document frequency. There are many related ranking functions; Okapi BM25 is a popular variation that shows good performance in practice [1].

A different approach for information retrieval is based in constructing a language model of the documents. The query likelihood model, for instance, attempts to create a language model for each of the possible documents and to find the one that fits better the query. In Kullback-Leibler divergence we also create a model of the query and then we estimate the distance between the probability distributions. These models provide good experimental results after smoothing [2].

In FAQ retrieval it has been shown that including the semantic similarity between documents using NLP approaches further improves the performance. A possible estimation may take into account distance between every pair of synsets in Wordnet [4]. One of such approaches is implemented in the system FAQ finder [3].

Matching the best answer is highly dependent on the specific domain under study. In our case, we are working in technology forums where there are many new words and synsets than in general questions. For example, the word "print" usually means to display in the screen, which is such an specific instance that it does not appear in Wordnet [9]. Past results prove that including domain specific features can provide significant improvement in the medical domain [5].

Most of the literature related to FAQ retrieval does not focus on online forums where the question is usually not well specified and it usually contains a significant amount of background. Some previous works try to detect the specific questions [7], but the settings in our dataset are relatively different.

3. Strategy

3.1. **Data Pre-processing.** One of the problems related to working with online forums is the noise that exists in the data. In order to get better quality data for the metrics using the bag of words approach, we are applying few filters. We will follow some of the approaches explained in [10]:

3.1.1. *Stop word elimination.* We will be using the NLTK stopword directory to get rid of the words such as "the", "as", that don't provide particular meaning to the query. However, we are going to exclude some words from the list (e.g. "what," "how") that might provide information. While this won't improve the scoring of tf-idf metric, we believe that it will contribute to better results for the other metrics.

3.1.2. *Stem words.* A second way to normalize the data is to use only the stems of the words. This approach should provide cleaner lexicographic relation by eliminating the differences between "going" and "goes", for example. We'll be using already built kit that has this option.

3.1.3. *Technology bigrams and acronyms.* One of the peculiarities when working with the Super User data set is that many of the terms exist as bigrams or acronyms. In order to capture their meaning, we will try to extract these forms (e.g. by going through the set of tags) and make

sure that we can differentiate between "Windows 7" and "Windows XP".

3.2. **Metrics.** We are going to implement the following similarity metrics and evaluate their performances.

3.2.1. *Cosine similarity metric.* We will apply the cosine similarity between the given query q and the title part of the question. This provides a rudimentary baseline that the other metrics will be compared to.

3.2.2. *Term frequency - inverse document frequency.* This heuristics, taken from information retrieval provides the relative importance for a given word (the term) to the particular QA pair (the document). We expect that tf-idf yield good performance given the nature of our data - almost all of the questions contain specific terms that greatly determine the semantics of the question.

3.2.3. *Tag based similarity.* All of the QA pairs in our corpus are annotated multiple tags, indicating the categories that the QA pair belongs to. Although the tags give information about the category, too often they are too general or too specific. We are experimenting with different measures relating to this similarity. One of them is the number of the QA tags appearing in the query normalized by the size of the query.

3.2.4. *Wordnet similarity.* This similarity is our first effort at calculating semantic closeness between the query and the question. For this part, we'll be implementing the similarity provided by [4]. Namely the similarity between the query q and the question part r from a QA pair is the following:

$$s(q,r) = \frac{1}{2}\left(\frac{\sum_{q_i \in q} \max ssim(q_i, r)}{|r|} + \frac{\sum_{r_i \in r} \max ssim(r_i, q)}{|q|}\right),$$

where $ssim(r_i, q)$ is defined as the maximum wordnet similarity between the word r_i and each individual word from q. In order to reduce the complexity of the calculation, we are going to make the assumption that each word in the query is associated with the most common meaning in the wordnet dictionary. However, this assumption might significantly affect the performance of this metric because quite often computer terms are overloaded and are don't correspond to the most common definition (i.e. "mouse" as an animal vs. tracking device).

3.2.5. *query language model.* In this metric, we are going to use a machine translation model to estimate the probability of generating the query q given the QA pair. Under this model, the scoring function between the query q and the QA is given by:

$s(q, QA) = \Pi_{q_i \in q}(1 - \lambda)P_{tr}(q_i|QA) + \lambda P_{ml}(q_i|C)$,

where C is the context, and P_{ml} is a smoothened probability obtained with simple bigram count. The translational probability P_{tr} is defined as

$$P_{tr}(q_i|QA) = \sum_{t \in QA} P(q_i|t)P_{ml}(t|D)$$

In this scenario, we estimate the transition probability $P(q_i|w)$ by using the mutual information between the terms q_i and w.

3.3. **From individual metrics to a combined model.** We will implement each of the similarity metrics described above and compare their performance on the testing corpus. To compare the different methods we are going to use the metric "Success at n" (S@n) which indicates whether the right QA pair is in the top n positions. We refer to the objective function as $F(m)$, i.e. the expected value of the S@n using the a metric m.

We will also provide the results for the trained combinations of the above metrics in our experiments to improve the performance. Formally, we assume that for any given query and any given QA pair we can compute several similarity metrics:

$$m_i(q, QA) \text{ for } 1 \leq i \leq k$$

We are interested in computing a linear combination that maximizes our objective function:

$$argmax_{\mu} F\left(\sum \mu_i m_i\right)$$

The above optimization problem is quite complicated, given that S@n is not convex. Therefore, we will use a heuristic objective function:

$$argmax_{\mu \geq 0, \sum \mu_i = 1} E_{q, QA_q}(\sum \mu_i m_i(q, QA_q))^2$$

where QA_q is the best match for a given q. In other words, we try to optimize the similarity value for all pairs (q, QA_q).

3.4. **Deriving a probability model.** We now explain the second approach to the modeling function. We assume that all similarity metrics m_i are normalized, so that they can be considered as probability distributions. We are interested in deriving a model for $P(QA|q)$. We

follow the ideas in [5], assuming that each QA can be considered as a set of independent aspects $\{s_i\}$, such that:

$$P(QA|q) = \sum_i P(s_i|q)P(QA|s_i, q)$$

If we denote $P(QA|s_i, q)$ by $m_i(QA, q)$, the section above can be considered as a particular instance in which $P(s_i|q) = \mu_i$, i.e. the probabilities of each metrics are independent of the query. To derive a more general model, which is dependent on the query, we can use different approaches. [5] uses the EM algorithm; however, the dataset used for the experiment has a considerable amount of repeated queries, which is quite different than our case (we have two paraphrases per QA pair). A possible approach is to define a feature representation of the queries and try to do a regression to estimate the whole similarity vector. We can also try implement a clustering algorithm to identify the queries with close similarity vectors.

4. Experiments

All experiments will be done with a set of 3000 questions, which were closed for being exact duplicates of another question. The dataset will be divided into two groups, one for testing, and one for training. As mentioned before, to measure the performance of a given method we will use the S@n metric. In particular, when $n = 1$ we will be considering the percent of times in which the method returns the associated questions in the top position.

5. Current progress

We investigated current approaches for FAQ retrieval and we identified some useful methods to solve our problem.

We have already downloaded the full Super User dataset available by September of 2011. The dataset is available in xml format and contains an anonymous description of all posts, comments, users and votes in the forum. We have already parsed the data to access it using python code. Our current approach loads into memory all the questions to perform further operations.

We have implemented a preliminary version of a cosine similarity with the standard bag of words approach taking only the titles of the posts into account. We can look for the closest question to a new query in a reasonable amount of time, but we can not do a complete sort of the questions efficiently. We also implemented a preliminary version of the wordnet semantic similarity described before. To disambiguate the

words we are currently run a tagger using the NLTK toolkit [12] and we take the most common synset for it. This method is computationally intensive and our current approach is not fast enough to compute this similarity with all existing questions.

6. Future Work

Our current approach to load into memory the full history of posts is not scalable and can't be done to load all answers in the post. We have to preprocess the data and to make it available in an efficient way for later simulations. In addition, given the high dimensionality of the dataset, we have to use implement a sparse representation of all feature vectors to be able to access them in memory.

We have already implemented two of the similarity metrics mentioned before. However we need to improve their performance. We still have to implement methods for the rest of them.

We have to compute a way to combine the results of multiple similarity metrics and we have to develop a model that allows the combination to be dependent on the specific query.

7. Possible issues

Due to the size of the dataset we might restrict our study to a subset of the questions, for instance, considering only certain set of labels. In addition, the low performance of similarity metrics such as the wordnet similarity or the query language model with translation might be impractical to compute in the full dataset. We might compute this similarity only for the top N results extracted by simpler methods.

Our testing data is considerably small compared to the size of the dataset. In addition, most of the current literature approaches have access to many different queries for a single answer. Our limitations with the testing data might lead to less accurate results.

8. References

[1] C. D. Manning, P. Raghavan, y H. Schütze, Introduction to information retrieval, vol. 1. Cambridge University Press Cambridge, 2008.

[2] C. X. Zhai, Statistical language models for information retrieval a critical review, Foundations and Trends in Information Retrieval, vol. 2, no. 3, pp. 137-213, 2008.

[3] R. D. Burke, K. J. Hammond, V. Kulyukin, S. L. Lytinen, N. Tomuro, y S. Schoenberg, Question answering from frequently asked

question files: Experiences with the faq finder system, AI magazine, vol. 18, no. 2, p. 57, 1997.

[4] W. Song, M. Feng, N. Gu, y L. Wenyin, Question similarity calculation for FAQ answering, in Semantics, Knowledge and Grid, Third International Conference on, 2007, pp. 298-301.

[5] C. H. Wu, J. F. Yeh, y M. J. Chen, Domain-specific FAQ retrieval using independent aspects, ACM Transactions on Asian Language Information Processing (TALIP), vol. 4, no. 1, pp. 1-17, 2005.

[6] Super User Forum: http://superuser.com/

[7] G. Cong, L. Wang, C. Y. Lin, Y. I. Song, y Y. Sun, Finding question-answer pairs from online forums, in Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval, 2008, pp. 467-474.

[8] J. Atwood, Creative Commons Data Dump Sep '11, http://blog.stackoverflow.com/2011/09/creative-commons-data-dump-sep-11/

[9] Princeton University "About WordNet." WordNet. Princeton University. 2010.

[10] D. Bernhard, I. Gurevych, Answering Learners' Questions by Retrieving Question Paraphrases from Social QA Sites, Proceedings of the Third ACL Workshop on Innovative Use of NLP for Building Educational Applications, pp. 44-52, 2008

[11] G. Zhou, L. Cai, J. Zhao, K. Liu Phrase-Based Translation Model for Question Retrieval in Community Question Answer Archives, Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics, pp 653-662, 2011

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

[13] Z. Wu y M. Palmer, Verbs semantics and lexical selection, in Proceedings of the 32nd annual meeting on Association for Computational Linguistics, 1994, pp. 133-138.