Healthcare Data Analytics Challenge

Zhiguo Yu[1], Byron C. Wallace[2], and Todd R. Johnson[1]

I. INTRODUCTION

Online patient/caregiver support forums such as, cancer compass, ehealthforums, and patientslikeme, allow patients and caregivers to post health-related questions. In many of these forums, there is a significant volume of repetitive questions. One possible reason for this repetition could be that as forums grow longer, patients and caregivers do not have the time or patience to read through previous questions before posting their own question. The challenge here is to design and implement a system that, for a new question q, identifies a maximum of three existing questions that are most similar to q.

In this challenge, we experimented with a variety of methods and representations to address this task, including approaches that leveraged topic modeling, distributional semantics (word2vec), and term frequency-inverse document frequencies (TF-IDF) to induce the vector representation of questions. For similarity measures, we used cosine similarity and the rescaled dot product over these feature spaces. Despite our experimentation with more recent methods, we found that simple TF-IDF with stemming using cosine similarity seemed to result in the best performance.

II. METHODS

A. Topic Modeling

We experimented with both standard Latent Dirichlet Allocation (LDA) [1]and Twitter-LDA [2].

LDA is perhaps the most commonly used topic model. LDA assumes that each document is generated from a mixture of topics, and that each topic corresponds to a distribution over the words in the corpus. We lumped both 95 sample questions corpus and 10 sample test questions together. We used the LDA-c code (*http://www.cs.princeton.edu/ blei/lda-c/index.html*) to perform topic modeling using variational inference. We ran LDA twice with 10 and 20 topics respectively. We used default values in the LDA-c code for all other parameters.

We experimented with two different approaches to determining similar questions based on the results of LDA. First, we treated each question's inferred latent topic distribution as its vector representation. Cosine (eq. 1) and rescaled dot product [6] (eq. 2) were then used to compute each test question's similarity with every question in the sample question corpus. Additionally, we used topic distributions over words to create a 'semantic' vector for each question. Specifically, we followed the following steps:

- Infer the relevant topics for a given question using the estimated document-topics distributions.
- For each word of every question, locate its position in each of the relevant topics'(from the previous step) ranked words distribution and retrieve the 5 closest words as its 'semantic vector'.
- For each question, add all the words' semantic vectors together to generate a vector representation.
- Compute question similarities between vector representations using Cosine and re-scaled dot product.
- Rank test questions with respect to similarity to sample questions and choose the three most similar questions.

Cosine similarity and the re-scaled dot product (our similarity measures) are calculated as follows:

$$cos(\theta) = \frac{\sum_i^n A_i \times B_i}{\sqrt{\sum_i^n A_i^2} \times \sqrt{\sum_i^n B_i^2}} \quad (1)$$

$$Rescaled\ Dot\ Product = \frac{P * Q - d_{Min}}{d_{Max} - d_{Min}} \quad (2)$$

$$d_{Max} = \overrightarrow{P} * \overrightarrow{Q} \quad ; \quad d_{Min} = \overrightarrow{P} * \overleftarrow{Q}$$

Where $\overrightarrow{P}$ and $\overrightarrow{Q}$ are vectors consisting of weights corresponding to all unique words sorted in descending order by weight, and $\overleftarrow{Q}$ is a vector of weights for all unique words sorted in ascending order by weight.

In LDA, a "document" is assumed to have been generated from multiple topics. This mixture assumption may not work well with very short "documents", such as patient questions or, similarly, tweets from Twitter, because such short questions are more likely about only one topic. Twitter-LDA has been proposed to address this issue. T-LDA assumes each twitter user has a distribution of topics about which they tweet. The words comprising any given tweet are then constrained to be generated from a single topic drawn form the user's topic distribution and a background word distribution, which is shared with every tweet.

A Patient's question may be similar to a tweet. To apply T-LDA (*https://github.com/minghui/Twitter-LDA*) to our dataset, we assumed each question was posted by a unique patient. We ran T-LDA on the dataset with the number of topic set to 30, 40, and 50 respectively. The reason that we set the number of topics so high is that we did not want too many questions clustered to the same topic.

For each test question, we identified its assigned topic first and then used this topic to locate those sample questions that

[1]Zhiguo Yu, MS and Todd R. Johnson, PhD, the University of Texas School of Biomedical Informatics at Houston, TX

[2] Byron C. Wallace, PhD, School of Information, University of Texas at Austin, TX

were also assigned with the same topic. These were deemed to be similar questions.

For each question q, we then used the word distribution associated with the topic from which q was inferred to have been generated as this question's supporting word vector. We combined both the question's original tokens and its supporting word vector as its new vector representation. Then we used cosine and re-scaled dot product to compute similarities between each test question and each sample question.

B. Word2Vec

Word2Vec is an unsupervised algorithm for learning the distributional semantics underpinning words. Given a large amount of unannotated plain text, Word2Vec learns relationships between words automatically and represent each word as a vector with remarkable linear relationships with others [3].

We used three different databases to train Word2Vec: PubMed, PubMed-and-PMC, and Wikipedia-PubMed-PMC. The trained word vectors induced from these databases have been made available by others (*http://bio.nlplab.org/*).

For each word in our question dataset, we generated a word vector from these trained word vector models. For each question, we pooled all the words' word vectors together as its new vector representation. Then we used cosine and rescaled dot product to compute similarities between each test question and each sample question.

C. TF-IDF

TF-IDF is a classic text representation scheme still commonly used in information retrieval and data mining tasks [4], [5]. This representation implicitly encodes (via weighting) how important a word is to a document in a corpus. The importance increases proportionally to the number of times a word appears in the document but is offset by the frequency of the word in the corpus overall. TF-IDF can be written as follows:

$$TFIDF(t,d,D) = TF(t,d) * log(\frac{N}{DF(t,D)}) \quad (3)$$

where, $TF(t,d)$ is the term frequency in document d, N is the total number of documents in corpus D, and $DF(t,D)$ is the document frequency of term t in D.

Before computing the TF-IDF computation, we conducted the stemming to reduce the words sparsity over questions. For example, the term 'juice' and 'juices' will be treated as the same term using stemming. In this dataset, each question has a title and a body. To emphasize the importance of the title, we doubled the term frequency in the tile. For example, if term t in d appears 1 time in the title and 1 time in the body, the $TF(t,d)$ will be 3.

We again used cosine and rescaled dot product as similarity measures between each test question and sample question based on the TF-IDF representations. For each test question, we chose the top three sample questions with the highest similarity scores.

III. Discussion

We used cosine and TF-IDF without stemming as our baseline. Comparing these two similarity measures' performance, it would seem cosine performs slightly better than the rescaled dot product. Among all the methods, cosine similarity over TF-IDF with stemming derived vectors appeared to perform best. For example, this is the only strategy that identified (correctly, in our opinion) the following two questions as similar questions.

Test question [19]: 'Need Suggestions for Natural Juices. Can anyone suggest any Natural Juice - Rich in Vitamin C & good for Type 2 Diabetics?'

Target question [191]: 'Is it safe to drink unsweetened/non-concentrated apple juice? It says it is low on the glycemic index but I was wondering if it is too close to soda the way it affects blood sugar.'

We note that the topic modeling method was good at identify similar questions with small number of co-occurring words, such as:

test question [12]: Exercise Advice. What kind of exercise is most beneficial to diabetics? Should I do cardio or weights or swimming or a combination?

target question [177]:Yogic postures. Are there specific yogic postures and breathing exercises that are recommended for diabetics?

But it also resulted in noise due to the similarity calculation. For example, for *Test question [19]*, it identified the following question as its similar question instead of *Target question [191]*, which was identified by TF-IDF with stemming.

target question [188]: Can I drink light beer? Is there a beer we Type II can drink? Can you suggest one?

Using Word2Vec derived representations resulted in performance similar to using topic modeling. Among the three different semantic databases, Wikipedia-PubMed-PMC achieved the best performance. This is intuitively agreeable, as the language in Wikipedia is probably closer to patients' language than that found in the scientific articles that compose PubMed and PMC.

In the future, we would like to test the topic modeling and Word2Vec's performance using the diabetes patients' questions and answers database.

References

[1] Blei, David M., Andrew Y. Ng, and Michael I. Jordan. "Latent dirichlet allocation." the Journal of machine Learning research 3 (2003): 993-1022.

[2] Zhao, Wayne Xin, et al. "Comparing twitter and traditional media using topic models." Advances in Information Retrieval. Springer Berlin Heidelberg, 2011. 338-349.

[3] Mikolov, Tomas, et al. "Efficient estimation of word representations in vector space." arXiv preprint arXiv:1301.3781 (2013).

[4] Sparck Jones, Karen. "A statistical interpretation of term specificity and its application in retrieval." Journal of documentation 28.1 (1972): 11-21.

[5] Wu, Ho Chung, et al. "Interpreting tf-idf term weights as making relevance decisions." ACM Transactions on Information Systems (TOIS) 26.3 (2008): 13.

[6] Chuang, Jason, et al. "Topic model diagnostics: Assessing domain relevance via topical alignment." Proceedings of the 30th International Conference on Machine Learning (ICML-13). 2013.