Comparison of Three Similarity Measures for Selecting an Answer in a Retrieval-Based Chatbot

Olta Cakaj
Bremen, Germany
olta@uni-bremen.de

Vanja Sophie Cangalovic
Bremen, Germany
vanja@uni-bremen.de

G. M. Nazmul Hossain
Bremen, Germany
gmnaz@uni-bremen.de

Mahdi Islam
Bremen, Germany
mahdi@uni-bremen.de

Yvonne Jenniges
Bremen, Germany
yvo_jen@uni-bremen.de

Ayasha Siddiqua
Bremen, Germany
ayasha@uni-bremen.de

Abida Sultana
Bremen, Germany
abida@uni-bremen.de

ABSTRACT
A main challenge of retrieval-based conversational agents is the choice of a model for selecting an answer from a predefined set of responses. For this purpose, different methods are available. In the scope of the development of such an agent, three methods are compared in this work, namely cosine similarity between TF-IDF vectors, inner product between sentence embeddings and LDA. As dataset, discussion threads of online forums on the topic of climate change are used. For a retrieval-based chatbot, a matching algorithm, e.g. a text similarity measure, is used to match the user's input to a question from the corpus. In a second step, the answers to that question are ranked by the similarity measure and the most suitable one is then returned. For the comparison of the three mentioned methods, top-level questions, i.e. questions opening a discussion thread, serve as an input for each method. The output, i.e. the returned response, is judged according to a metric that is based on the depth of the answer in the original conversation thread. As a result of this evaluation, this paper finds that cosine similarity and sentence embeddings performed comparable and achieved higher scores than LDA. This suggests that the former models are, in the chosen configuration, more suitable to serve as a text similarity measure on posts and to operate as answer selection algorithms in a retrieval-based chatbot.

UPDATED—April 30, 2020.

ACM ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

Author Keywords
conversational agents, retrieval-based, discussion platforms, text similarity measures

INTRODUCTION
Conversational agents can be classified into two main categories, according to their intended functionality. On the one hand, task-oriented agents aim to support users in fulfilling a specific task, this setup is called a dialogue system. On the other hand, there are "chatbots", i.e. systems, which do not pursue a specific task, but are meant to have natural and meaningful discussions with people. [27]

For the implementation of a chatbot, different architectures are available, including generative and retrieval-based ones. Generative models assemble an answer given the user input and potentially some context knowledge, whereas retrieval-based models return an answer from the corpus that is ranked as best suited to the user input by some algorithm. [27] Hence, besides the underlying dataset, the selection of an answer matching algorithm that retrieves a relevant answer from the corpus is a crucial step in the development of retrieval-based chatbots.

As dataset, this work uses conversations from online discussion forums. Online discussion platforms are considered an important medium for efficient and extensive participation [17] in discussions for a plethora of topics. One of the main reasons for this is that they engage a high number of potentially diverse people in mutual reflection as well as exchanging viewpoints and knowledge. [10] Thus, by extracting data from online discussion forums, not only a substantial amount of discussions and responses is available, but also people's authentic opinions on a particular issue. Therefore, the data that serves as a base to this work is extracted from various discussion platforms focusing on the topic of climate change. Communication data is valuable in the context of chatbots because its goal is a natural talk with people. Since conversations in discussion forums are often arranged hierarchically, i.e. there can be comments to answers to a question, the corpus was flattened to question-answer pairs and each answer was assigned its original hierarchy level for ease of evaluation.

Measuring text similarity is, according to Gomaa et al., an important part for a range of tasks, including text summarization, text classification, generation and answering of questions, as well as in information retrieval, for instance in retrieval-based chatbots [12]. Gomaa et al. differentiate the existing measures into string-based, corpus-based and knowledge-based ones [12]. The string-based methods perform approximate string comparisons on the base of either terms or characters. Term-based measures include cosine and Jaccard similarity, while e.g. n-grams are considered to belong to the character-based subcategory. The second and the third group of text similarity measures, i.e. the corpus-based and the knowledge-based ones, are semantic measures. This means that they take into account what the context of the words is, the relation of the words to other words and how the words are used. The difference between corpus-based and knowledge-based methods is the data that drives their decision on similarity. While the corpus-based category relies on data from collected texts and/or speeches, knowledge-based measures rely on semantic networks, like WordNet. Examples for corpus-based methods are Latent Semantic Analysis (LSA) and Hyperspace Analogue Language (HAL). For instance, Leacock et al. implemented a knowledge-based measure [15]. Moreover, hybrid approaches combine techniques from multiple of the above mentioned categories. [12]

In this paper, three answer matching algorithms for the retrieval process are compared: the string-based cosine similarity measure between TF-IDF vectors, the corpus-based inner product between sentence embeddings, as well as Latent Dirichlet Allocation (LDA), which is also corpus-based. For the comparison, the models are evaluated by a metric which depends on the hierarchy level of the returned answer. In this way, statements can be made about the relevance of the models' answers without the need for manual annotation. Thus, this paper answers the **research question** (RQ):

> What are the differences in answer relevance of the three answer matching methods (cosine similarity, sentence embeddings, LDA) according to the employed evaluation metric?

The obtained results can help to choose an appropriate model for a retrieval-based chatbot. For this, the evaluation indicates that cosine similarity and sentence embeddings present a more reasonable choice than LDA, considering the present implementation, corpus and parameter choice.

RELATED WORK

For the purpose of this work, text similarity measures for a retrieval-based conversational agent are evaluated. Hence, this section aims to, first, introduce the chatbot architecture. Second, it gives a background on the selected methods, being cosine similarity between TF-IDF vectors, inner product between sentence embeddings and LDA. Additionally, review on existing comparative studies is provided.

Retrieval-Based Chatbot Architecture

Retrieval-based conversational agents are based on an existing corpus of predefined question-answer pairs, which can be created manually or based on already existing information. A retrieval-based chatbot applies a matching algorithm to rank the questions in the corpus and find the most similar one to the user input. The answer to the matched question is then returned to the user. [29] Previous studies [28, 27] support the usage of answer matching methods similar to those compared in the present paper. For instance, Yan et al. propose a retrieval-based conversation system using deep learning to concatenate context statements with the input message as redefined inquiries [28].

Text Similarity Measures

Here, the three mentioned models used to measure text similarity in information retrieval systems are reviewed.

Cosine Similarity Between TF-IDF Vectors

A general scheme in developing a retrieval-based model is to encode the corpus and then extract a relevant response by applying similarity measures. According to Li et al., cosine similarity is an extensively used method to match an appropriate response in information retrieval systems [16]. It calculates the angle between two vectors in a dot-product of two normalized vectors [16]. Term Frequency - Inverse Document Frequency (TF-IDF) is a widely used approach for vectorizing text data in cosine similarity computations [22]. Specifically, the latter is a word occurrence-based method, representing each word in the underlying text via the product of its term frequency (TF) and inverse document frequency (IDF). [2] TF is the frequency of a term in a given document, whereas DF (document frequency) represents the fraction of the arguments containing this word. IDF, on the other hand, is a logarithmically scaled inverse DF [3]. The latter aims to penalize more common words found in several documents, considering them as trivial and not characteristic for specific texts [23].

Neural Network Models

Moreover, deep neural networks can be exploited for the task of encoding a natural language text into continuous-valued vectors. Most notably, word2vec and sentence embedding models have been proposed. [18, 4] Deep neural networks, trained on a variety of (self-)supervised tasks using large corpora of natural language utterances, provide another way to encode given sentences into continuous-valued vector representations. In order to reliably solve the various tasks, which can range from POS tagging, prediction of missing or subsequent words to causal reasoning, the networks need to learn an encoding in which words, or whole sequences of words, that are functionally similar, lie closer together. Thus, statistical and semantic information of the data is encoded. These latent representations can be retrieved as either word embeddings, a prominent example being word2vec (Mikolov et al., 2013), or sentence-level embeddings. The underlying networks are usually trained as language models. Word2vec's neural network of three layers, for example, can be trained via the Continuous Bag of Words and the Skip-Gram method. In these settings, the model's task is to predict either the current word given some surrounding words, or the context words, weighted by their distance, given a current word. [18] Another example is the Bidirectional Encoder Representations from Transformers (BERT) model, which constitutes a state-of-the-art language model for a multitude of NLP tasks. BERT is primarily trained to predict the original value for a masked token, given a sequence of words taken from a large corpus. [11] The general idea being, that such networks are able to create task-agnostic representations for words or whole sentences in their first layer(s), which might then be applied and fine-tuned in other settings. [25] In [4] the authors introduce an improved baseline for sentence embedding through weighted average of word vectors using dimensionality reduction techniques such as PCA/SVD. Results of this paper show enhanced performance in calculating sentence similarity.

Topic Models

Latent Semantic Indexing (LSI) is a broadly used method for topic modelling, aiming to detect low-dimension representation of terms or documents. [9] LSI employs singular-value decomposition (SVD), which utilizes a representation of words or documents in the form of linear combinations of semantic factors. This method assumes that words used in the same context have a similar meaning.[19]

Latent Dirichlet Allocation (LDA) is a probabilistic method for topic modelling, i.e. for extracting topics from texts. It is a form of unsupervised learning that observes documents as bags of words and characterizes a topic by a distribution over words. The assumption that each document consists of multiple topics in different proportions results in a topic distribution for each document. [5]

Minglai et al. offer an enhanced similarity measure using latent topic modelling and word co-occurrence analysis. To refrain from clustering inaccurate words that might have comparable topic probability but different topics nonetheless, linguistic correlation is evaluated through term co-occurrence given that the latter demonstrates an advanced text topic. [24]

Comparative Research

Few studies [8, 23] compare different similarity measures for longer texts. A vast majority relates to the comparison of different vector space models regarding their ability to measure text similarity for short ones. Here, a short text means one of the length of a title, a text of medium length is e.g. an abstract and a long text depicts everything with more characters than that [23]. In the following, related work on similarity measure comparisons for short texts is investigated. For instance, Lau and Baldwin compare doc2vec (document to vector) to two baseline methods, namely averaging word2vec as well as an n-gram model. [13] The former calculates the similarity of forum questions, whereas the latter aims to estimate the similarity between pairs of sentences. Based on this paper's findings, doc2vec performs noticeably better, especially when trained on large corpora. Another paper [20] compares three prominent models used to identify the semantic meaning of words, specifically a topic model (LSA), a neural network (word2vec), and GloVe (Global Vectors for Word Representation), which provides vector representations for words trained via an unsupervised algorithm which builds on the idea of higher co-occurrence probabilities for semantically similar words. Their results indicate that for the task of topic segmentation, word2vec constitutes a better word vector representation model than the other two employed methods. Even though it should be noted that the purpose of the research presented in this paragraph is to compare and review which model performs better for short sections of text.

Research can be found on the comparison of similarity measures for longer texts. One example is the work of Dai et al., which aims to find the most appropriate similarity measure for whole paragraphs, by comparing doc2vec, respectively weighted word2vec representations, TF-IDF and LDA on two corpora, i.e. Wikipedia and arXiv. [8] The study concludes that doc2vec performs significantly better than the other models on the Wikipedia corpus. However, it scarcely outperforms TF-IDF on arXiv. In [23], an evaluation of various vectorization techniques for encoding the semantics of natural language texts is conducted. Similarly to other papers mentioned above, the work examines TF-IDF, a topic model, in this case Latent Semantic Indexing (LSI), and paragraph vectors [14]. The latter are derived from neural networks, which have been trained to predict the words of a given document of variable length. The paper deduces that sophisticated text embedding techniques and extensions to TF-IDF can barely outperform the original and simple TF-IDF method.

This work attempts to review existing methods used to measure similarity of texts that are on average of medium-length, compare them and provide guidance on which of the investigated measures yields better results. Following other research approaches, the compared models are a basic one, a topic model and a neural network. The three different response matching techniques employed are cosine similarity between TF-IDF vectors, LDA and inner product between sentence embeddings. Cosine similarity is a string-based, surface matching model [30], which is frequently employed as a baseline method. LDA is a corpus-based model and, according to the best of the authors' knowledge, less explored as a text similarity measure than LSA/LSI. The inner product between sentence embeddings, another corpus-based method, is not represented in the considered comparative papers. Thus, this work integrates into the series of papers comparing text similarity measures. Contrary to the majority of presented papers, this work compares forum questions to other questions and answers from online discussion forums in the context of a retrieval-based chatbot.

METHODS

In order to answer the RQ, a retrieval-based chatbot is implemented with an interchangeable text similarity measure. As text similarity measures, cosine similarity, sentence embeddings and LDA are programmed. On the basis of data from online discussion forums, the chatbot answers a user input by returning a response that is selected from the dataset by the respective similarity measure. The retrieved response is then evaluated using the evaluation metric presented in section 3.5.

Data Collection and Pre-processing

The data that serves as a base to this work is extracted from various online discussion platforms, namely: Quora, TedTalk, Stack Exchange, Climate Debate, Skeptical Science

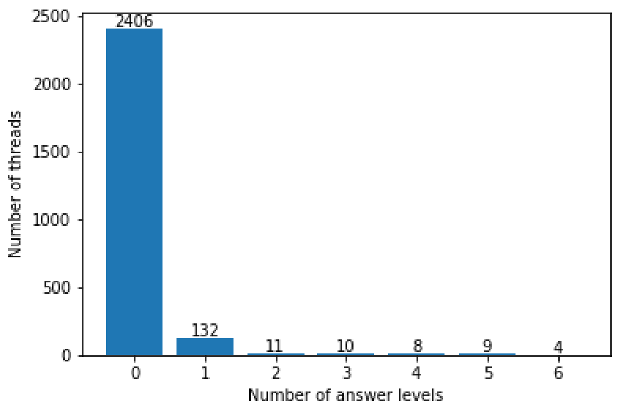

Figure 1: Depth of threads. The bar chart shows the maximum answer depth of the question threads.

and NASA. Climate related topics are extracted from these websites by performing web scraping, mainly using Selenium [1]. The scraped corpus, containing the nested question-answer threads, includes discussions in the timespan from 2010 to 2020.

The pre-processing steps needed for the examined similarity measures are the removal of stopwords, which is based on the idea of eliminating non-discriminative words in order to reduce the dimensionality of the feature spaces to be created, and stemming, which is common practice for the LDA and TF-IDF models.

The conversations are kept as they appear in the online forums. All data taken from the discussion platforms is then used as a question-answer pair, which is considered the main corpus. It consists of 153,540 question-answer pairs from 2,580 different threads. 93.26% of the threads have only direct comments. The others consist of up to seven answer levels (Figure 1). This wide spread is reflected by a high standard deviation of about 900 threads. On average, each thread has 14 answers resulting from a large number of threads with few answers and some outlier threads containing many answers with a maximum of 120 replies. In total, there are 35,598 answers on level 0, i.e. top-level or direct answers, 2,620 answers on level 1, 265 on level 2, 167 on level 3, 135 on level 4, 111 on level 5 and 94 on level 6. Analyzing the length of the texts shows that questions are generally shorter than their answers having a median of 11 words, compared to a median of 141 words for their replies.

Retrieval-Based Chatbot Architecture

For this work, a retrieval-based architecture is implemented. Since there are multiple cases in the scraped dataset, in which one question has more than one answer, the matching algorithm is first applied to all questions and afterwards to all answers from that question's thread. The key challenge, thereby, is to find a matching algorithm that returns a meaningful answer to the user. In the following, three different methods for this task are introduced.

Text Similarity Measures

The implementation of the three methods employed in this work for calculating the similarity between two given natural language texts, i.e. TF-IDF vectorization, sentence embeddings and LDA, are explained in this section. As a programming language, Python 3.7.3 is used.

Cosine Similarity Between TF-IDF Vectors

The idea of using cosine similarity calculated between TF-IDF vectorized texts follows the theory that statements with similar words are also semantically similar. In order to estimate the similarity between a given text and the corpus, pairwise cosine similarity, here used form the scikit-learn library, calculates the angle between the respective vectors and returns a score between 0 and 1, respectively indicating no similarity or identity. Using this traditional approach to approximate semantic similarity, while being extremely simple, has several disadvantages. It employs no knowledge of synonyms, thus texts about the same topic, but expressed with different, semantically related words would not be considered similar. Furthermore, it is a bag-of-words approach, meaning that all, potentially meaningful, syntagmatic structure is lost during the vectorization process.

Inner Product between Sentence Embeddings

In this work, the Universal Sentence Encoder [7] is employed, which is based on the transformer architecture [26], and has been shown to deliver promising results in the domain of semantic textual similarity [6]. Tensorflow's implementation of a pre-trained model with an embedding layer size of 512 is used in this work. The similarity between the resulting vector pairs is then calculated via their orthogonality, i.e. their pairwise inner product.

Topic Modeling - Latent Dirichlet Allocation (LDA)

For the implementation of LDA, this work uses the machine learning library scikit-learn. First, the corpus has to be vectorized for which the count vectorizer from the previously mentioned library is taken. All terms with a document frequency higher than 95% are ignored. Secondly, the LDA model is defined and fitted on the corpus. For the model instantiation, the maximum number of iterations is set to 50 and the online learning method is employed with a learning offset of 20. Moreover, the random_state parameter is assigned to 0 to allow reproducibility. The number of topics is treated as a hyperparameter in this work and is described in section 3.4.

After the model is trained on the present corpus of online discussions and a topic distribution is generated for each post, the LDA model can be used in the chatbot. In detail, the count vectorizer first vectorizes the user input. The output of this stage is then fed into the LDA model to generate a topic distribution. In the default answer matching process, the question from the corpus with the closest topic distribution to the user input topic distribution is selected based on the minimum L1-distance. The topic distributions of the answers to the chosen question are then again compared to the user input distribution. Finally, the answer with the most similar distribution is returned to the user.

Hyperparameters

Number of Topics for LDA

For LDA, an important parameter that has to be manually chosen is the number of topics n. There is no established method to perform this task but several are proposed in the scientific community. Here, a subjective evaluation was performed besides the calculation of perplexity. [31] These methods are employed for $n = 10, 20, 40, 50$ and $n = 100$ topics to cover a broad range. For the subjective evaluation, each list of top-words, i.e. each topic, was assigned a broader topic manually, if possible. The manual check yields similar results for all n: 20 - 30 % of the topics are not interpretable or irrelevant. The perplexity score, which describes the ability of a statistical model to depict a dataset [31], is calculated by saving the `bound_` attribute, provided by sklearn e.g. for LDA, for all implemented LDA models. Generally, a lower perplexity indicates a better generalization ability of the model. For the tested number of topics, the minimum perplexity is reached for $n = 20$ topics. Since the manual evaluation of the topics does not show any significant difference between the models, the perplexity score serves as a means for the topic number selection. Hence $n = 20$ is chosen.

Evaluation Metric

The conversational agent needs to retrieve the most appropriate answer to a user input, given a large corpus of human conversations. In order to evaluate the correctness/accuracy of this task, a quantitative evaluation metric is needed. Since manually annotating data would be both time- and labour-intensive, this work employs an unsupervised method using the corpus at hand. This existing corpus of natural language discussions contains answers to various questions, that have - evidently - been classified as appropriate and relevant by some human, i.e. the answer's author. This fact is utilized by evaluating the three aforementioned similarity metrics on these conversations. Therefore, the correctness of the system's output is determined by its belonging to the original input's thread, which is determined by a numerical score (Eqn. 1). Retrieving one of the direct answers is interpreted as a perfect score of 1, while answers from different threads as a failed 0, and answers further down the conversation hierarchy receive a discounted score. The discount depends on the hierarchy level of the answer, i.e. a direct answer to a question is assigned hierarchy level 0, a comment to a direct answer is on level 1 and so on. Furthermore, the level of the answer is divided by the overall maximum number of levels plus a constant number, whose purpose is to prevent a selected answer from the last hierarchy level of a thread to receive a score of 0. The constant was set to $const = 5$ in this work.

$$score = 1 - \frac{hierarchy\_level}{max\_hierarchy\_level + const} \in [0, 1] \qquad (1)$$

Evaluation Modes

The evaluation was conducted using the top-level questions as input. Thus, the outcome of the three similarity measures can be compared against the ideal answer (a direct comment to the question). In order to analyse the behaviour of the models, four different evaluation modes are applied which are abbreviated with *question binary*, *default answer*, *all answers* and *answer to correct question*. They are explained in the following. Since the first step of the answer retrieval is the matching of the user input to the most similar question from the corpus, the evaluation mode *question binary* checks if the model can find the most similar question to the input out of all top-level questions in the corpus, which would be the exact same question. Thus, the question matching step is tested. The option *default answer* orients itself on how the implemented retrieval-based chatbot retrieves an answer: First, the most similar question is chosen from the top-level questions. Out of the answers to this selected questions, the most suitable one according to the model is returned. The third option is *all answers*, which retrieves the most similar answer out of all the answers in the corpus. This can give insights about the relevance of the question matching step. Lastly, *answer to correct question* retrieves the most similar answer from the thread of the original question to evaluate the output for the case that the question matching was successful.

RESULTS

In order to answer the RQ, the above mentioned evaluation using Eqn. 1 was conducted for all three similarity measures and every evaluation mode, i.e. *question binary*, *all answers*, *answer to correct question* and *default answer*. The results can be seen in Tab. 1a, Tab. 1b, Tab. 1c and Tab. 1d respectively. The tables can be read as follows: For each similarity measure, one row of a table depicts how many of the 2580 input questions resulted in a matched answer with the score denoted in the column "Score". For example, Tab. 1b (evaluation mode *all answers*) shows that cosine similarity returned in 2146 out of 2580 cases an answer that got a score of 0. The mode *question binary* is limited to a score of either 0 or 1 because it states whether the answer was in the original question thread ($score = 1$) or not ($score = 0$). The other three modes, *all answers*, *answer to correct question* and *default answer*, can produce eight different scores according to Eqn. 1 since the present corpus exhibits a maximum of seven answer levels in addition to the score of 0 (representing that the returned answer was not located in the original question thread).

The comparison of the three employed text similarity measures answers the RQ: The difference between the models according to the introduced evaluation metric manifests in the computed scores. While cosine similarity and sentence embeddings achieve an average score of 1 when matching the question, LDA reaches an average score of 0. The answer matching on the base of the correct question, resulted in average scores of 0.9935 (sentence embeddings), 0.9913 (cosine similarity) and 0.9825 (LDA). Overall, cosine similarity and sentence embeddings yield comparable results which are better than those of LDA. This comparison is explored in detail in the following.

Cosine Similarity between TF-IDF Vectors

Calculating the cosine similarity between the same two TF-IDF vectorized texts results, by definition, in a score of 1. Thus, the *question binary* evaluation task is fulfilled with an average score of the returned answers of 1 (Tab. 1a). Due to the accurate results of the *question binary* task, all of the matched answers in mode *default answer* and *answer to correct question* are correctly located in the respective discussion threads, leading to scores larger than 0 (Tab. 1d and Tab. 1c). Thus, these two modes resulted in similar scores and are considered together here. The scores average a value of 0.9918 over the 2580 input questions, and in 93.72% of the cases, first-level answers were selected. The *all answers* task resulted in lower scores, the average being 0.1665, having matched into the original thread in 16.65% of the cases and having selected a first-level answer in 15.08% (Tab. 1b).

Inner Product between Sentence Embeddings

In line with the results reported for cosine similarity of TF-IDF vectorised texts, calculating the inner product between the same two embedded texts achieves perfect results, the average score of *question binary* being 1 which is shown in Tab. 1a. In consequence, the *default answer* and *answer to correct question* tasks resulted in similar scores, averaging a value of 0.9935 over the 2580 input questions. In 95.12% of the cases, first-level answers were selected (Tab. 1d and Tab. 1c). With an average score of 0.1318, this approach resulted in a poorer performance on the more difficult *all answers* task. In 13.29% of the cases, the measure selected an answer from the original thread (Tab. 1b).

LDA

Overall, LDA achieved very low scores. This is already visible when evaluating the mode *question binary* (Tab. 1a): Apart from one exception, the majority of the matched questions did not equal the original thread question and therefore received a score of 0. Since the question is not matched correctly, the answer is picked from another question thread. This entails poor values for the evaluation option *default answer* (Tab. 1d) because the question selection is the first part of the latter evaluation method. For the option *all answers*, LDA performed similar: In most cases, with only one exception, the retrieved answer was not included in the thread of the input question, and thus received a score of 0 (Tab. 1b). Lastly, Tab. 1c shows, given the correct question, which level the retrieved answer has. 83.37% of the answers are on the first level, i.e. direct answers to the top-level question.

Analysis of Results

According to the employed evaluation metric, sentence embeddings and cosine similarity performed similar and outperformed LDA. Since LDA already fails to match the correct question, it is not able to retrieve an answer that would be labeled as suitable by the metric. Thus, it returns lower scores than the other two methods, excluding the case of *answer to correct question*. In the latter mode, LDA achieved comparable results to the other two methods. However, it matched significantly more second level answers, 393 in total, compared to 130 (cosine similarity) and 99 (sentence embeddings). Therefore, the evaluation metric penalizes LDA, indicating that its answer matching capability is inferior to the abilities of cosine similarity and sentence embeddings, which return more direct, i.e. first-level, answers. The high number of first level answers for all three cases can be explained by the fact that the corpus mainly consists of threads which have first level answers only. Cosine similarity and sentence embeddings achieved very similar scores, since they always find the correct question. However, these two models rely on the question matching step for finding a relevant answer, which is inferable from Tab. 1b: Without this step, i.e. when selecting the answer directly from all possible responses, the methods only choose direct answers to the input question in a small fraction of the cases. In this mode, cosine similarity performed slightly better than sentence embeddings, choosing 91 times more often an answer that was in the original question thread. In contrast, sentence embeddings achieved marginally better scores for *answer to correct question* with an average score of 0.9935 compared to an average score of 0.9913 (cosine similarity).

DISCUSSION

Given the evaluation results, this section discusses the impacts on the RQ, contextualizes the results and finalizes by pointing out limitations.

In line with Shahmirzadi et al. and Dai et al., the results of this paper indicate cosine similarity between TF-IDF vectors to be a good choice for measuring the similarity of documents. According to former work, performance and cost of this model justify its usage as a text similarity measure. [23, 8]. In the present paper, the method generally achieves high scores in the different evaluation modes. For *question binary*, the cosine similarity between the same two TF-IDF vectorized texts is calculated and results by definition in a score of 1. Thus, the measure also achieves good scores in the modes *default answer* and *answer to correct question*.

Similar to cosine similarity, the sentence embeddings approach yield high scores in the evaluation. This performance can also be tracked back to the fact that the inner product between the same two embedded texts in the mode *question binary* works perfectly. Contrary to the comparison in this work, the neural network-based paragraph vectors employed by Shahmirzadi et al. performed, similar to their LSI model matching very short texts, better than cosine similarity between TF-IDF vectors. However, this result was achieved only after extensive parameter tuning. [23] The Universal Sentence Encoder employed in this work generally resulted in scores very similar to those of TF-IDF, though it could not outperform this method. This difference might be explained by the varying underlying architectures of the models. While the paragraph vectors are explicitly trained to predict words of a given document, the Universal Sentence Encoder is trained via multi-task learning, creating the sentence embeddings along the way. Another explanation for the different results might be the different corpora, which possibly entail different linguistic challenges.

In the comparison, LDA performed worst according to the presented evaluation metric. The main cause for this is that LDA already fails matching a top-level question to the same question in the corpus, i.e. the evaluation mode *question binary*. One possible reason is the L1-distance measurement between the topic distributions. There are several alternative approaches, e.g. Niraula et al. propose Kullback-Leibler divergence or the Hellinger distance in the context of LDA [21]. However, other work that compares topic models to neural network models and paragraph vectors also find that

Score	Cosine Similarity	Sentence Embeddings	LDA
0.0	0	0	2579
1.0	2580	2580	1
Average Score	1	1	0.0004

(a) Evaluation mode: *question binary*. It show how many out of the 2580 top-level questions each similarity measure matches to a different question from the corpus (*score* = 0) and to the exact same question (*score* = 1).

Score	Cosine Similarity	Sentence Embeddings	LDA
0.0	2146	2237	2579
0.45	0	0	0
0.55	0	0	0
0.64	1	2	0
0.73	0	0	0
0.82	0	0	0
0.91	44	24	0
1.0	389	317	1
Average Score	0.1665	0.1318	0.0004

(b) Evaluation mode: *all answers*. Without limiting the similarity measure to a question thread, this table shows which score the answers achieved, that were retrieved from matching the top-level-questions to answers from the corpus.

Score	Cosine Similarity	Sentence Embeddings	LDA
0.0	0	0	0
0.45	2	2	2
0.55	4	4	4
0.64	5	4	5
0.73	10	6	7
0.82	11	11	18
0.91	130	99	393
1.0	2418	2454	2151
Average Score	0.9918	0.9935	0.9825

(c) Evaluation mode: *answer to correct question*. Given the answer thread of the top-level input question, this table shows the scores assigned to the answers that the similarity measures returned.

Score	Cosine Similarity	Sentence Embeddings	LDA
0.0	0	0	2579
0.45	2	2	0
0.55	4	4	0
0.64	5	4	0
0.73	10	6	0
0.82	11	11	0
0.91	130	99	0
1.0	2418	2454	1
Average Score	0.9918	0.9935	0.0004

(d) Evaluation mode: *default answer*. It shows the scores of the answers that the similarity measures return when operating in the way they are used in the chatbot, i.e. first matching the top-level input question to a question in the corpus and then choosing an answer from the thread of the selected question thread.

Table 1: Scores of the similarity measures according to Eqn. 1 for the four evaluation modes: 1a: *question binary*, 1b: *all answers*, 1c: *answer to correct question* and 1d: *default answer*. The last row denotes the average score of the output text of each similarity method for the depicted evaluation mode.

the topic model is outperformed by the other two methods [23, 8]. Moreover, it should be considered that the LDA model can be improved, by parameter tuning or the combination with other methods. For instance, Minglai et al. show that the value of LDA as a text similarity measure can be enhanced by combining it with word co-occurence analysis [24].

The generally lower results for the task *all answers* can be explained by the fact that its set of possible candidates is much larger. While *answer to correct question* and *default answer* restrict the search space to the texts of a given discussion thread (which contain an average number of 14 answers), for *all answers*, the whole corpus of answers is considered during the similarity calculation process (resulting in 39,990 candidates). Thus, apart from reducing the prior probability of our algorithms selecting a correct answer, this larger set of possible answers greatly increases the challenge as a whole, demanding both fine-grained and robust calculations for finding a correct match to the input. The results in Table 1b indicate that LDA failed this task, whereas TF-IDF and sentence embeddings were able to retrieve a correct answer for about 15% of input questions. Another noteworthy result is that the answer with a score of 0.64 for both the TF-IDF and the sentence embedding method is in fact the same, which further supports the seeming similarity between those measures.

Overall, to answer the initially posed RQ, the obtained results indicate that both sentence embeddings and TF-IDF perform similarly and significantly better than LDA. However, their performance drops for the most challenging task *all answers*.

Limitations

A major limitation of this work is the lack of the number of both discussions in general and deep discussion threads in particular, which severely restricts the relevance of the present evaluation results. In order to achieve more reliable results, a larger corpus would be needed.

Another fundamental limitation concerns the employed evaluation metric itself. Intuitively, the assumption of all answers in a given discussion thread treating the same topic and being relevant to the first question might appear far-fetched to online forum-experienced people. The tendency of human discussions to diverge from the initial topic is reflected in the discounting of the metric. However, it is realistic to assume that there are cases in which answers from different threads might actually be more relevant to a given question than comments from the original thread. This observation results in the possibility of having a high number of false negatives, which could only be detected by manual investigation of the results.

CONCLUSION

This work presents three text similarity measures for returning relevant, i.e. semantically related, answers in a retrieval-based conversational agent: cosine similarity between TF-IDF vectors, inner product between sentence embeddings and LDA. The models are (trained and) tested using question-answer pairs from online discussion forums, focusing on the topic of climate change. Moreover, a metric is introduced to automatically evaluate the three methods. The metric depends both on the hierarchy level of the answer, as well as the maximum number of answer levels in the present corpus. Although TF-IDF vectorization is a simple method, discarding much information, which is deemed crucial for understanding a text's precise semantics, it performs similarly to the inner product between sentence embeddings, whose underlying neural network has potentially learnt layers of semantic information associated with natural language constructions. Furthermore, LDA, despite being a promising measure since it contains, contrary to TF-IDF vectors, a semantic differentiation, performed very poorly.

Future Work

In order to improve the quality of the returned answers, several steps can be taken. One possibility is to improve the implementation and/or hyperparameters of LDA, e.g. by applying other measures for finding an appropriate number of topics. This could be useful because there is no established method to determine this parameter. A different measure could lead to a different number of topics and therefore might change the results for the presented evaluation.

Additionally, along the lines of ensemble methods in machine learning architectures, a combination of the presented similarity measures might be able to yield better results. For example, calculating sentence embeddings and cosine similarity would result in the texts being matched on both a semantic and a word-based statistical level. Alternatively to sentence embeddings, LDA could be employed. However, the conducted evaluation supports the first combination variant. Thus, for a retrieval-based chatbot, cosine similarity could perform the question matching to retrieve the lexically most similar question to the user input, and sentence embeddings the answer matching to find a semantically relevant answer.

REFERENCES

[1] 2020. SeleniumHQ Browser Automation. (2020). `https://www.selenium.dev/` Accessed: 2020-04-16.

[2] Akiko Aizawa. 2003. An Information-Theoretic Perspective of Tf—Idf Measures. *Inf. Process. Manage.* 39, 1 (Jan. 2003), 45–65. DOI: `http://dx.doi.org/10.1016/S0306-4573(02)00021-3`

[3] M. Alodadi and V. P. Janeja. 2015. Similarity in Patient Support Forums Using TF-IDF and Cosine Similarity Metrics. In *2015 International Conference on Healthcare Informatics*. 521–522.

[4] Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2019. A simple but tough-to-beat baseline for sentence embeddings.

[5] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *J. Mach. Learn. Res.* 3, null (March 2003), 993–1022.

[6] Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 Task 1: Semantic Textual Similarity Multilingual and Crosslingual Focused Evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Association for Computational Linguistics, Vancouver, Canada, 1–14. DOI: `http://dx.doi.org/10.18653/v1/S17-2001`

[7] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Universal Sentence Encoder. *arXiv:1803.11175 [cs]* (April 2018). arXiv: 1803.11175.

[8] Andrew M. Dai, Christopher Olah, and Quoc V. Le. 2015. Document Embedding with Paragraph Vectors. *CoRR* abs/1507.07998 (2015). `http://arxiv.org/abs/1507.07998`

[9] Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American society for information science* 41, 6 (1990), 391–407.

[10] Liping Deng, Yang-Hsueh Chen, and Sandy C. Li. 2017. Supporting cross-cultural online discussion with formal and informal platforms: a case between Hong Kong and Taiwan. *Research and Practice in Technology Enhanced Learning* 12, 1 (2017), 5. DOI: `http://dx.doi.org/10.1186/s41039-017-0050-z`

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. (2018).

[12] Wael H.Gomaa and Aly A. Fahmy. 2013. A Survey of Text Similarity Approaches. *International Journal of Computer Applications* 68, 13 (Apr 18, 2013), 13–18.

[13] Jey Lau and Timothy Baldwin. 2016. An Empirical Evaluation of doc2vec with Practical Insights into Document Embedding Generation. 78–86. DOI: `http://dx.doi.org/10.18653/v1/W16-1609`

[14] Quoc V. Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents. (2014).

[15] Claudia Leacock and Martin Chodorow. 1998. *Combining Local Context and WordNet Similarity for Word Sense Identification*. Vol. 49. 265–.

[16] Baoli Li and Liping Han. 2013. Distance Weighted Cosine Similarity Measure for Text Classification. In *Intelligent Data Engineering and Automated Learning – IDEAL 2013*. Springer Berlin Heidelberg, Berlin, Heidelberg, 611–618.

[17] Edith Manosevitch, Nili Steinfeld, and Azi Lev-On. 2014. Promoting online deliberation quality: cognitive cues matter. *Information, Communication & Society* 17, 10 (2014), 1177–1195. DOI: `http://dx.doi.org/10.1080/1369118X.2014.899610`

[18] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. (2013).

[19] Andreea Moldovan, Radu Boţ, and Gert Wanka. 2005. Latent semantic indexing for patent documents. *International Journal of Applied Mathematics and Computer Science* 15 (01 2005), 551–560.

[20] Marwa Naili, Anja Habacha, and Henda Ben Ghezala. 2017. Comparative study of word embedding methods in topic segmentation. *Procedia Computer Science* 112 (12 2017), 340–349. DOI: http://dx.doi.org/10.1016/j.procs.2017.08.009

[21] Nobal Niraula, Rajendra Banjade, Dan Ştefănescu, and Vasile Rus. 2013. Experiments with Semantic Similarity Measures Based on LDA and LSA. In *Statistical Language and Speech Processing*, Adrian-Horia Dediu, Carlos Martín-Vide, Ruslan Mitkov, and Bianca Truthe (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 188–199.

[22] Juan Ramos. 2003. Using TF-IDF to determine word relevance in document queries. (01 2003).

[23] O. Shahmirzadi, A. Lugowski, and K. Younge. 2019. Text Similarity in Vector Space Models: A Comparative Study. In *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*. 659–666.

[24] Minglai Shao and Liangxi Qin. 2014. Text Similarity Computing Based on LDA Topic Model and Word Co-occurrence. In *2nd International Conference on Software Engineering, Knowledge Engineering and Information Engineering (SEKEIE 2014)*. Atlantis Press.

[25] Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to Fine-Tune BERT for Text Classification? (2019).

[26] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. (06 2017).

[27] Yu Wu, Wei Wu, Chen Xing, Ming Zhou, and Zhoujun Li. 2016. Sequential Matching Network: A New Architecture for Multi-turn Response Selection in Retrieval-based Chatbots. (2016).

[28] Rui Yan, Yiping Song, and Hua Wu. 2016b. Learning to Respond with Deep Neural Networks for Retrieval-Based Human-Computer Conversation System. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '16)*. Association for Computing Machinery, New York, NY, USA, 55–64. DOI:http://dx.doi.org/10.1145/2911451.2911542

[29] Zhao Yan, Nan Duan, Junwei Bao, Peng Chen, and Ming Zhou. 2016a. DocChat: An Information Retrieval Approach for Chatbot Engines Using Unstructured Documents. In *54th Annual Meeting of the Association for Computational Linguistics*. Berlin, Germany, 516–525.

[30] Wen-Tau Yih and Christopher Meek. 2007. Improving similarity measures for short segments of text. In *AAAI*, Vol. 7. 1489–1494.

[31] Weizhong Zhao, James J. Chen, Roger Perkins, Zhichao Liu, Weigong Ge, Yijun Ding, and Wen Zou. 2015. A heuristic approach to determine an appropriate number of topics in topic modeling. In *12th Annual MCBIOS Conference*, Vol. 16 Suppl 13. https://www.ncbi.nlm.nih.gov/pubmed/26424364