UNIVERSITÀ DI PISA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

This is a preprint. Please cite this work as:

Alfeo, A.L., Cimino, M.G.C.A. & Vaglini, G. Technological troubleshooting based on sentence embedding with deep transformers. Journal of Intelligent Manufacturing (2021). DOI: https://doi.org/10.1007/s10845-021-01797-w

---

Machine Learning and Process Intelligence

Research Group

Technological troubleshooting based on sentence embedding with deep transformers

Antonio L. Alfeo[1], Mario G. C. A. Cimino[1], Gigliola Vaglini[1]

Received: 28 Dec 2020 / Accepted: DD MM YYYY

Abstract

In nowadays manufacturing, each technical assistance operation is digitally tracked. This results in a huge amount of textual data that can be exploited as a knowledge base to improve these operations. For instance, an ongoing problem can be addressed by retrieving potential solutions among the ones used to cope with similar problems during past operations. To be effective, most of the approaches for semantic textual similarity need to be supported by a structured semantic context (e.g. industry-specific ontology), resulting in high development and management costs. We overcome this limitation with a textual similarity approach featuring three functional modules. The data preparation module provides punctuation and stop-words removal, and word lemmatization. The pre-processed sentences undergo the sentence embedding module, based on Sentence-BERT (Bidirectional Encoder Representations from Transformers) and aimed at transforming the sentences into fixed-length vectors. Their cosine similarity is processed by the scoring module to match the expected similarity between the two original sentences. Finally, this similarity measure is employed to retrieve the most suitable recorded solutions for the ongoing problem. The effectiveness of the proposed approach is tested (i) against a state-of-the-art competitor and two well-known textual similarity approaches, and (ii) with two case studies, i.e. private company technical assistance reports and a benchmark dataset for semantic textual similarity. With respect to the state-of-the-art, the proposed approach results in comparable retrieval performance and significantly lower management cost: 30-minute questionnaires are sufficient to obtain the semantic context knowledge to be injected into our textual search engine.

Keywords deep learning · sentence embedding · textual similarity · remote technical assistance

Introduction

Lately, several industries are transitioning to the *smart manufacturing* model by adopting technologies such as the internet of things (IoT), cloud computing, and machine learning to increase their productivity and competitive advantage (Tao et al. 2018). Indeed, machine learning can provide automatic knowledge extraction from manufacturing big data to increase production efficiency, reduce management costs (O'Donovan et al. 2015), and drive technological innovation. In this context, the paradigm of *Knowledge Management 4.0* (Ansari 2019) emphasizes the business value creation achieved by extracting and providing accessibility to manufacturing domain-specific knowledge obtained by coupling human experiences and data-driven approaches (North et al. 2018). As an example, the data obtained through IoT devices can be analyzed via a machine learning approach to detect production anomalies (Alfeo et al. 2020), while their management can be supported by considering past human-driven maintenance operations to collect best practices and improve the maintenance processes (Navinchandran et al. 2021).

[1] Department of Information Engineering, University of Pisa, Largo L. Lazzarino 1, 56127 Pisa, Italy

✉ Antonio L. Alfeo
luca.alfeo@ing.unipi.it

Specifically, both in-place maintenance operations and remote technical assistance are digitally tracked in the form of textual reports stored in the ERP system (Usmanij et al. 2013), containing the details of the performed inspection and the adopted solutions for the occurred technical problems. Those result from the investigation, experiences, and recommendations of domain-aware technicians. If accessible and exploitable, such knowledge base can be shared and reused to provide effective support for the operators in training (Costa et. al 2016) and may result in a faster diagnosis and management of machines' technical problem. The former can significantly improve production efficiency by reducing machines' downtime. Indeed, the time spent diagnosing the problem and finding a possible solution, is often larger than the time spent fixing it (Sexton et al. 2017).

To take advantage of such a knowledge base, it is essential having an effective tool to find solutions that are relevant to a given problem, i.e. adopted with similar problems. According to (Sunilkumar et al. 2019), the approaches for textual similarity can be organized into four main groups: (i) string-based approaches determine the similarity between two text strings by comparing them as two sequences of characters and words; (ii) corpus-based approaches find the similarity based on corpus statistical analysis, e.g. checking words co-occurrence via cosine similarity or n-grams; (iii) knowledge-based approaches depend on a handcrafted semantic structure for the specific domain concepts, e.g. the shortest path length between the two concepts in a knowledge graph represents their similarity; (iv) approaches based on deep sentence embeddings are used to

automatically build sentences' representation in a semantic space, in which the distance between two vectors is correlated to the similarity of the corresponding sentences.
Still, there is a lack of applications aimed at retrieving technical assistance reports (Ansari 2020), since the effectiveness of such applications may be easily constrained by (i) inconsistencies and inaccuracies in the data due to the informal language used by the operators, (ii) the inability to distinguish suboptimal solutions proposed by technicians with lower expertise, and (iii) the highly unstructured nature of textual reports (Nemeth et al. 2019). This is especially true with the first three groups of approaches. Indeed, to be effective, they require handcrafted features for similarity assessment or a structured representation of the domain-specific semantic context (Aarnio et al. 2016). These processes consist of time-consuming manual activities aimed at collecting and organizing the domain knowledge. Moreover, to have an effective technical assistance reports (TAR) retrieval, those procedures may be repeated during the design of the search engine as well as, every time the semantic context changes, i.e. due to the introduction of new services, machines, or product. This is evident from the business process model notation (BPMN) diagram in Fig. 1, representing the main activities for maintaining a TAR retrieval application.

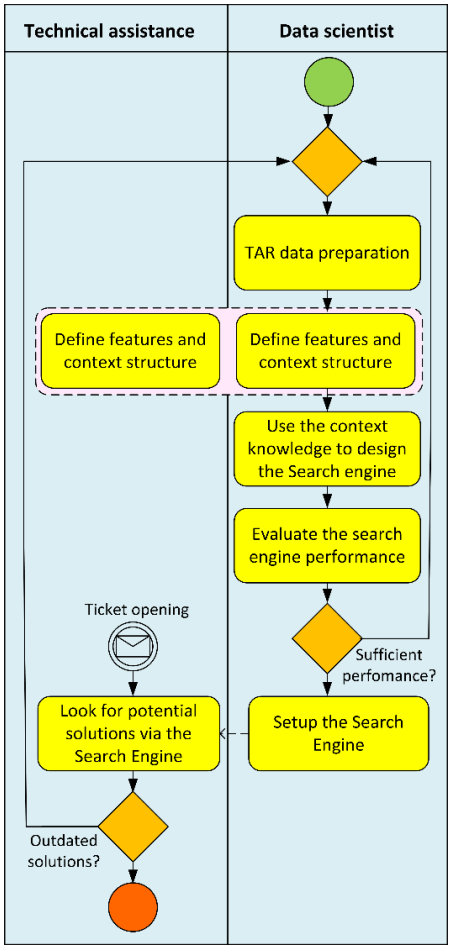

Fig. 1 BPMN diagram of the maintenance process for a TAR retrieval application

In Fig. 1 each lane corresponds to an actor involved in this process. The process starts with the data scientist preparing the unstructured textual data to be processed by the search engine. Then, it follows a joint activity with the technical assistance aimed at defining the features and the semantic context of the TAR data, on which the search engine will be based. According to the retrieval performance achieved during the tests, different rounds of specification of the semantic context and the features may be required. Once satisfactory performances have been obtained, the search engine can support all the requests received by technical assistance (the circle with the envelope in Fig. 1). The system operates as long as the proposed solutions are still applicable and up-to-date, then the setting must be repeated.

To obtain reliable performances while decreasing the system management costs, we propose a deep learning approach to retrieve potential solutions for a given technical problem by employing TAR data and no other structured representation of the domain-specific semantic context. Specifically, we transform the TAR problem descriptions in vectors of the semantic latent space, compute the proximity between those and the vector obtained from a new technical problem, and use this similarity score to rank the problems in the TAR database and propose the corresponding solutions as a potential one for the new problem. Our approach employs a data preparation module to preprocess the textual data. Then, the embedding module is used to transform sentences of arbitrary length into fixed-length vectors and it is based on Sentence-BERT (Reimers et al. 2019). Those vectors can be compared via the scoring module, i.e. processing their cosine similarity via a multilayer perceptron to match the expected similarity between the two original sentences. The system has been tested in two distinct case studies: private company TAR and a benchmark dataset for sentence similarity (Cer et al. 2017). The effectiveness of the proposed approach is compared against one of the best-in-class competitors, i.e. Universal Sentence Encoder (USE) (Cer et al. 2018), and two well-known bag-of-words approaches for sentences' similarity assessment. The paper is structured as follows. In Section 2, we present the literature review. In Section 3, we detail our approach. Section 4 presents the case study. The obtained results are discussed in Section 5. Finally, Section 6 summarizes conclusions and future works.

State of the Art

The literature review presented in this section focuses on the effectiveness and management cost of the approaches for TAR retrieval. Indeed, as introduced in Section 1, effective approaches for TAR retrieval often result in a huge management cost, resulting in limited use of these approaches in real-world industrial applications. Such management costs are mainly associated with the activities aimed at injecting the domain-specific semantic context into the search engine. The cost can be considered *low* if it involves simple activities such as the definition of a domain-specific ontology, a dictionary of terms specific to an application domain, or establishing a categorization for the problems in the dataset. Indeed, these activities can be performed through interviews with domain experts, do not require the modification of the data entry process, or an explicit labeling activity on the existing data, e.g. evaluating the similarity of problem pairs. The management cost increases (*medium* cost) if the activity connected to the injection of the semantic context requires a modification of

the report storage process, e.g. to add metadata about the effectiveness of the solutions or to identify the relevant parts of a maintenance report to be the focus of the analysis. The management cost can be considered *high* if the approach requires the manual labeling of many problem pairs in the dataset since this requires a number of human-driven similarity evaluations equal to the square of the number of problems.

According to (Lan et al. 2018), from a methodological point of view the approaches for TAR retrieval can be organized in two main categories: (i) Information Retrieval (IR) approaches, retrieve potential solutions according to the degree of similarity in the underlying semantics to the original problem (Kathuria et al. 2016), whereas (ii) Question Retrieval and Answering (QR) approaches, rank candidate problem-solution pairs according to their significance to a given problem (Shtok et al. 2012). The effectiveness of both IR and QR approaches can be measured by considering how many of the highly ranked retrieved solutions are useful or relevant to the problem described. To this aim, the most used measures are the Mean Reciprocal Rank (MRR) and the Mean Average Precision (MAP) (Metzler et al. 2005). MAP considers the relevant solutions *i* according to their position r_i in the ranking *R* for the query *q*. The higher the rank of a relevant solution, the more it contributes to the score computation. MRR considers only the highest position of relevant solutions in the rank *R* for query *q*. Both these measures are between 0 and 1 (the higher the better).

$$MAP = \frac{1}{|Q|}\sum_{q=1}^{|Q|}\frac{\sum_{i=1}^{|R_q|}\frac{i}{r_i}}{|R_q|}$$

$$MRR = \frac{1}{|Q|}\sum_{q=1}^{|Q|}\frac{1}{max(R_q)}$$

A clear example in which different management costs result in different retrieval performances is (Guo et al. 2019). The authors compare the effectiveness of twenty-three different approaches for textual similarity ranking based on neural networks, resulting in (i) the best performances among the QR approaches (MAP 0.77), and (ii) an average performance (MAP 0.502) among the IR ones (Table 1). However, these results are associated with different management costs. Specifically, the best QR approach is based on a fast and lightweight sentence embedding deep learning architecture that requires a large number of sentence pairs with their corresponding degree of similarity to be trained, thus resulting in a high management cost. On the other hand, the IR approach needs much fewer labeled sentence pairs and proves experimentally how it can be trained on the data of a given year and reused with the data of the next year.

A more effective (MAP 0.609) IR approach with an even lower management cost is (Heilman et al. 2010), in which the authors represent each textual edit operation as a node in a sequence of transformations, the minimum length of the sequence to make two sentences semantically identical determines the measure of similarity of the sentences. This method requires the design of a set of general-purpose edit operations and domain-specific constraints for their sequence, e.g. which edits cannot be used one after the other. Defining this set of transformations and constraints requires an analysis of the semantic domain of the case study, no explicit labeling activity has to be performed thus resulting in a low management cost.

Authors in (Tong et al. 2015) propose an IR approach for troubleshooting retrieval based on word co-occurrence and domain-specific categories. Given a problem query and categories, the troubleshooting search system can retrieve the relevant information of interest to the selected categories, resulting in a MAP equal to 0.571. This approach requires (i) metadata about the broken asset's component to be used as domain-specific categories, (ii) a mapping describing the semantic relationship between words and categories, as well as the relationship between categories, and (iii) the manual labeling of the important parts of each maintenance report, resulting in a medium management cost.

In (Gupta et al. 2018) the authors propose a replicated Siamese Long Short-Term Memory (LSTM) to evaluate the similarity between asymmetric text pairs. This IR approach is used with an industrial ticketing system to retrieve a relevant solution for an input query using the tickets' knowledge base, resulting in a MAP equal to 0.4. Moreover, this approach implies a labeling campaign resulting in 421 pairs of sentences with their degree of similarity, i.e. a management cost ranging from medium to high.

Authors in (Zhou et al. 2015) employ a look-up table to transforms each word into a vector, then aggregate and process them via an embedding procedure. This procedure is constrained to obtain fixed size vectors from different length sentences while matching a given sentences' categorization. The distance between the sentences' embeddings in the latent space is used to assess their similarity. This IR approach is quite effective (MAP 0.69), yet it employs data consisting of thousands of pair query-solution manually labeled as relevant or not relevant. Given the required (i) definition of the look-up table and the categories, and (ii) manual labeling activity, the management cost of this approach may be considered high.

In (Pang et al. 2017) authors propose an IR approach based on a deep learning architecture to assess the semantic matching of textual data, resulting in a MAP equal to 0.49. This approach demands (i) the definition of weights to represent the importance of words in the query, and (ii) more than one hundred thousand labeled query-document pairs, resulting in a high management cost.

Many QR approaches have been extensively used in the field of community question answering (cQA), in which users ask their web community how to address a specific technical issue they are experiencing. To reduce the number of unanswered questions, the potential answer can be automatically retrieved by employing past similar queries (Guo et al. 2019, Zhou et al. 2015). Even if the domain is slightly different, the technology employed works on the same assumptions and for similar aims, thus, can be exploited in the context of industry 4.0 for retrieving the most suitable technical assistance report with respect to a given problem.

The QR approach presented in (Chahuara et al. 2016) ranks similar questions from question-answer archives differentiating them by topics to cope with the different vocabularies used within questions of different topics. Specifically, each topic is associated with a particular dictionary, and a semantic mapping is established between different topics and between pairs of sentences of specific topics. Although the number of sentence pairs required is not large and the mapping can also be partially automated by leveraging statistical approaches, the number of steps required to employ this approach may correspond to a medium/high management cost. On the other hand, this management cost allows achieving very good effectiveness, i.e. MAP equal to 0.75.

Authors in (Das et al. 2016) retrieve questions that are similar to a given query, via a QR approach leveraging the distance between the query and its topic in the latent vector space. To train the model one hundred queries are used. For each of these 20 plausible solutions are provided, together with the metadata on the effectiveness of each solution and its topic. The introduction of this metadata in a business process has a very low cost, but this does not apply to the selection of plausible solutions for a hundred queries, resulting in an overall medium/high management cost.

In (Baldwin et al. 2016) the authors retrieve similar questions, despite their difference in length, by employing a convolutional neural network combined with a Naive-Bayes classifier and a support vector machine each trained over lexical similarity features. This effective (MAP 0.702) QR approach employs the metadata about the relevance of the proposed solutions and requires a manual labeling process to obtain the similarity query-solution pairs, resulting in a high management cost.

The same management cost is required by the best performing method tested in (Lan et al. 2018), i.e. a QR approach based on deep learning and tested on eight publicly available datasets consisting of a large number of sentence pairs and their similarity.

In (Othman et al. 2019) authors use word embeddings to capture semantic and syntactic information from textual contexts and vectorize the questions. The embedding vectors feed a Siamese LSTM neural network, and the similarity between the questions is obtained as the Manhattan distance of the final LSTM hidden states. In this study, 1624 sentences and 256 queries are used. Human annotators are employed to evaluate and find 2 to 30 relevant solutions for each query, resulting in a high management cost. This QR approach is extended in (Othman et al. 2020) by enhancing the neural network architecture with an attention mechanism to determine which words in the questions should receive more attention during the embedding phase. Again, the approach requires a large number of manually labeled pairs of queries and relevant sentences, about 30% more than the previous study.

Table 1 provides an overview of the best MAP and MRR performances reported in the research works presented in this section together with their management cost and requirements.

Table 1 Best MAP and MRR performances reported in the research works presented in Section 2, together with their management cost (L=low, M=medium, H=high) and the cost's motivations in terms of system requirements.

SHORT REF.	IR/ QR	MAP	MRR	COST	REQUIREMENTS
Guo, 2019	QR	0.770	-	H	· Very large number of labeled pairs
Chahuara, 2016	QR	0.750	-	M/H	· Domain-specific topics · Topic-specific dictionary · Large number of labeled pairs
Lan, 2018	QR	0.739	0.795	H	· Very large number of labeled pairs
Baldwin, 2016	QR	0.702	0.8	H	· Metadata about solutions' relevance · Large number of labeled pairs
Zhou, 2015	IR	0.690	-	H	· Words-vector look-up table · Domain-specific categories · Large number of labeled pairs
Heilman, 2010	IR	0.609	0.692	L	· Edit operations and constraints
Othman, 2020	QR	0.579	-	H	· Very large number of labeled pairs
Othman, 2019	QR	0.574	-	H	· Very large number of labeled pairs
Tong, 2015	IR	0.571	0.643	M	· Domain-specific categories · Words-categories semantic mapping · Selecting documents' relevant parts
Das, 2016	QR	0.532	0.574	M/H	· Solutions' effectiveness metadata · Large number of labeled pairs
Guo, 2019	IR	0.502	-	M/H	· Large number of labeled pairs
Pang, 2017	IR	0.497	-	H	· Selecting important words in the query · Very large number of labeled pairs
Gupta, 2018	IR	0.400	0.287	M/H	· Large number of labeled pairs

As summarized in Table 1, there is a lack of effective textual similarity approaches characterized by low management costs. For this reason, many of the TAR retrieval solutions in the literature are more of a proof of concept rather than real-world applications (Ansari 2020). One of the few

examples of applications can be found in (Wang 2010), in which the authors describe an intelligent semantic labeler to retrieve a potential solution for a new problem by exploiting past problem-solution pairs. In (Sipos et al. 2014, Xu et al. 2020) natural language processing (NLP) approaches are used to cluster maintenance reports and provide information retrieval support for maintenance technicians. Despite the lack of many real-world applications, some of them have proven to be highly cost-effective. As an example, the authors (Ray et al. 2020) provide a QR application able to distinguish symptoms, activities, actions, and advises of problem-solution pairs. The authors report that such an approach reduced the time to response of technical assistance service by 29%, leading to an overall cost savings of 25% per year. This confirms the great potential of these applications, whose availability, however, is strongly limited by a hard-to-reach trade-off between effectiveness and management cost. To reduce the management cost, it may be useful an approach based on the most recent sentence embedding techniques. Sentence embedding can transform arbitrary long sentences into fixed-sized vectors whose distance is correlated to the similarity of the original sentences, resulting in a simple and effective manner to represent textual semantic similarity (Passaro et al. 2020). An example, authors in (Khabiri et al. 2019) classify maintenance reports by exploiting an industry-specific taxonomy, extracting reports' corpus, and obtaining an embedding from each document. Also, consider that both the most effective IR and QR approaches shown in Table 1 (Zhou et al. 2015, Guo et al. 2019) are obtained with approaches based on sentence embedding, even if those correspond to high management costs.

In contrast with those, the recent sentence embedding technique (Reimers et al. 2019) is pre-trained on a huge amount of publicly available data, thus does not require (i) many labeled sentence pairs to learn how to distinguish similar and different sentences in any semantic context, or (ii) any structured context representation such as ontologies or taxonomy graphs. In the next section, we present an effective TAR information retrieval real-world application employing such sentence embedding technique and characterized by low management costs.

Architectural design of the sentence similarity model

In this section, we present the design of the proposed approach. Considering the approaches presented in the last section, the proposed architecture relies on deep transformers for sentence embedding to avoid the explicit modeling of the application's semantic context. Moreover, the latest literature provides different pre-trained models that are extremely useful in the absence of large training datasets, as it frequently happens in real-world manufacturing. It is impossible to effectively train the whole NLP model with a limited amount of data, whereas those data can be effectively used to finetune a pre-trained model.

The proposed architecture consists of a data preparation module, a sentence embedding module, and a scoring module (Fig. 2). Firstly, the data preparation module provides punctuation removal, stop-words removal, and word lemmatization. With stop-words removal, each sentence is transformed in a list of individual words, and the ones which are not adding meaning to the sentence are removed, e.g. 'a', 'the'. With words lemmatization, each word is reduced to its common base form, e.g. from 'studies' to 'study' (Vijayarani et al. 2015). The pre-processed sentences undergo the embedding module, which is in charge of transforming each sentence into a fixed-length vector. This transformation should preserve the semantic relationships between the sentences, i.e. the distance between two vectors should be correlated to the semantic dissimilarity between the original sentences.

The sentence embedding module employs Sentence-BERT (Reimers et al. 2019), a state-of-the-art sentence embeddings technique pre-trained with more than 570,000 sentence pairs (Bowman et al. 2015). S-BERT is an extension of BERT specifically designed for assessing the semantic similarity between sentences. BERT (Devlin et al. 2018) is an NLP transformer-based machine learning technique developed by Google that learns contextual relations between words (or sub-words) in a text. BERT is pre-trained on more than 3.3 billion words obtained via different web sources, e.g., the English Wikipedia. To measure the similarity between two sentences using BERT, two sentences have to be concatenated using a particular token (a separator character) and processed via BERT as a whole. This makes the comparison impractical in real-world information retrieval applications, e.g. to find the two most similar sentences in a given set of *n* sentences would require *n(n-1)/2* operations. S-BERT extends BERT to handle this limitation. By adding a pooling operation to the output of BERT, S-BERT produces a fixed-length (1024) embedding vector for each sentence regardless of its length, resulting in a number of operations equal to the number of sentences analyzed. The embeddings resulting from two sentences will be spatially close (distant) if the corresponding sentences are semantically similar (different). Thus, the comparison can then be obtained via standard and efficient distance-based measures such as the cosine similarity (△ in Fig. 2) computed between the embeddings of two sentences. As an example, in (Reimers et al. 2019) the authors employ the pair-wise similarity between 10,000 sentences for a clustering procedure: it results in 65 hours execution time with BERT, and 5 seconds with S-BERT.

It follows the formulae of the cosine similarity, a bounded measure of similarity between two non-zero vectors. It is defined as the ratio between the dot product and the magnitude of those vectors, to equal the cosine of the angle between them.

$$\triangle_{A,B} = \frac{e_A \cdot e_B}{\| e_A \| \cdot \| e_B \|}$$

As explained in its documentation, S-BERT model can be specialized for a specific semantic context by means of its specific fine-tuning procedure, by using a set of sentence pairs together with their degree of similarity.

The proximity between the embeddings may not always effectively match the similarity between the original sentences, since the embeddings generated by BERT-based models tend to occupy a narrow cone in the vector space (Li et al. 2020). To have a more effective mapping between the proximity between embeddings and the expected similarity

between sentences, we introduce the scoring module. The scoring module consists of a multilayer perceptron (MLP) aimed at processing the cosine similarity between two embedding vectors to match the expected similarity between the two original sentences.
The scoring module is trained using embedding pairs together with the expected similarity of the original sentences. Once trained, the scoring output is considered as the similarity score between two sentences.

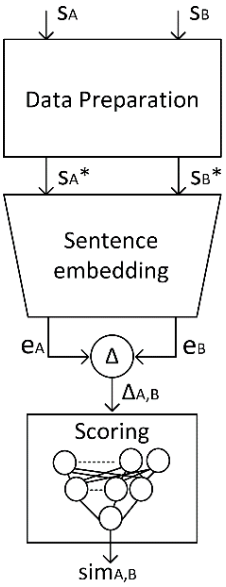

Fig. 2 The architecture of the proposed sentence similarity model

Industrial application and case study

Our case study consists of the analysis of TAR of real-world manufacturing company that produces and sells machinery. In case of a technical issue with the machinery, the customer can request for remote technical assistance, whose process is represented in Fig. 3. During this process, a technical assistance operator oversees and supports the maintenance operations via a dedicated video communication channel. The operator records the motivation of the call, the remotely observed condition of the machinery, as well as the proposed solutions. Those can be searched manually or via a search engine. If no solution results effective, the company schedules an on-site maintenance activity. The collected information is stored as a digital report in the company's database. For each report in the database, we extract the call timestamp, the customer ID, the call ID, if the call was followed by a physical inspection, and three textual descriptions addressing the motivation for the call, the remote inspection, and the proposed solution. We account for the issues affecting the textual analysis with TAR data introduced in Section 1. Then, we account for inconsistencies and inaccuracies in the text by focusing on pairs of problem descriptions and proposed solutions, both provided by the remote technical assistance operator. Indeed, most of the call motivations contain a very generic description of the actual problem, e.g. 'the machine stopped due to overheating'. This is also confirmed by looking at the average number of words used for each textual description: 23.3 for the call motivation, and 35.7 for the problem descriptions. We also account for the inability to distinguish suboptimal solutions by selecting the instances that did not lead to a physical inspection, assuring to keep only the problem-solution pairs in which the solution proposed was effective.

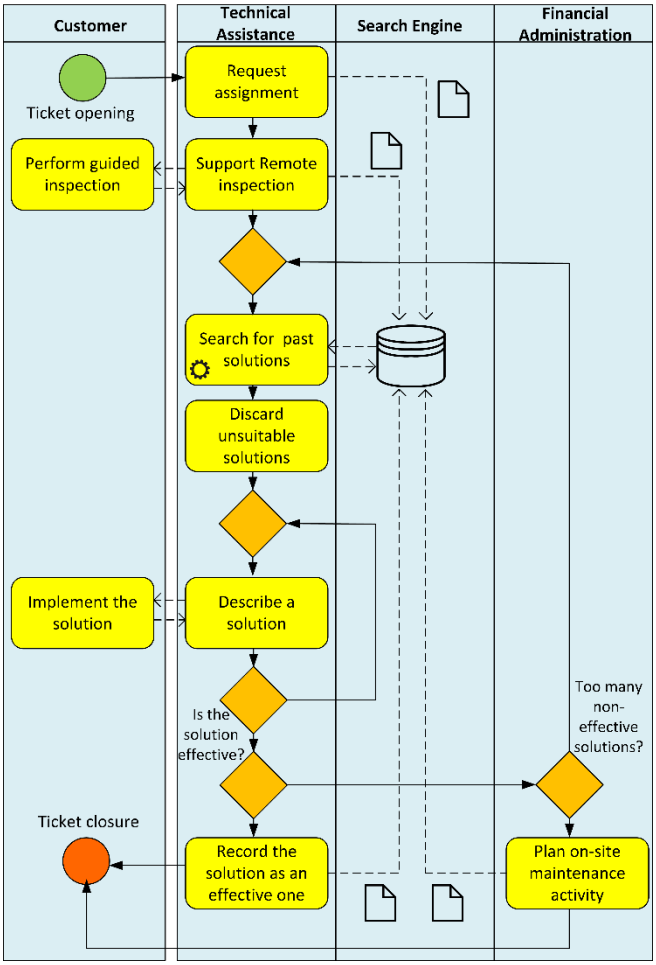

Fig. 3 BPMN diagram of the remote technical assistance process

The resulting TAR dataset is made of 308 problem-solution pairs. We build a training set made of pairs of problem descriptions and their degree of similarity. This is obtained with the support of the domain experts of the company, i.e. the remote technical assistance operators. We selected some representative technical problems, to cover the main topics in the whole set of problem descriptions. To enable the system to distinguish different degrees of similarity, for each representative problem (RTP), 5 other problems have been chosen. Those are characterized by different degrees of similarity with respect to the RTP. Specifically, two were chosen to be similar to the RTP, two different from the RTP, and one problem was chosen at random. In this context, two problems are considered similar (different) if they require to perform similar (different) technical operations on the same (other) asset component to be fixed. We asked three technical assistance operators to rank those 5 problems according to the similarity with the latter. The obtained ranks for each problem pair were averaged and normalized resulting in a similarity score between 1 (very similar problem) and 0 (completely different problem). Finally, this set of problem pairs was treated with a data augmentation

procedure leveraging a synonyms dictionary. The synonyms dictionary provides up to 4 synonyms for 30 different domain-specific technical terms. If a problem description contains one or more of those terms, each combination of their possible synonyms generates a new problem description while preserving the overall sentence's meaning. The final TAR training set is composed of 2440 pairs of problems and their similarity score (examples in Table 2).

Table 2 Examples of problem pairs' in the training set

PROBLEMS DESCRIPTION	SIMILARITY
"the customer cannot restart the line, the embosser nip roll axis goes in alarm state." "the customer complains about the simotion of the tail sealer, they can't connect to it."	0.25
"the log saw is not running even if the operator press the orange push button." "the log saw is in stop state and is not possible restart the machine."	0.917

To validate the results of our approach we also need to know which solution is relevant for a given problem. We build a *solution-relevance* dataset by collecting 10 representative problems descriptions, covering the main topics in the TAR dataset. For each one of those, we rank the most suitable 5 potential solutions. We asked 8 remote assistance technicians to label each solution as "Relevant" or "Not relevant" with respect to the problem. The solution-relevance dataset can be used to evaluate the effectiveness of our application by using two well-known performance measures: MAP and MRR.

A given retrieval approach may provide solutions not included in the *solution-relevance* dataset. We account for those unlabeled solutions by evaluating how them impact MAP and MRR according to the probability (*P*) that an unlabeled solution is relevant for the problem described. Specifically, we consider the probabilities of 0, 0.25 and 0.5. Worst case scenario: if an unlabeled solution is retrieved, it is considered not relevant. Best case scenario: if an unlabeled solution is retrieved, it is considered relevant with a 50% probability.

In our application, the activity aimed at building the training set was performed by submitting a 30-minute questionnaire to the technical assistance technicians. The same amount of time was required to label the *solution-relevance* dataset, and to collect the domain-specific synonyms for the data augmentation. These activities aim at including the domain-specific knowledge in the system by replacing the second and third activities shown in the BPMN diagram in Fig. 1, thus resulting in reduced management costs of such application.

Both the embedding module and the scoring module can be fine-tuned or trained with the TAR training set described above (example in Table 2). Specifically, those modules are trained sequentially, allowing the sentence embedding module to generate the embedding vectors used to train the scoring module. The expected similarity of those vectors corresponds to the similarity of the original sentences. Once the system is trained, a new sentence (i.e. a new problem description *p*) can be pre-processed by the data preparation module and transformed into an embedding vector e_p. Then, the cosine similarity between e_p and the embeddings corresponding to each pre-processed problem description stored in the TAR database can be computed and processed by the scoring module. The obtained degrees of similarity can be used to rank the problems in the database. The solutions corresponding to the highest-ranking problems can be proposed as possible solutions for the problem described via *p*.

Experimental results

In this section, we present our experimental results. Each result is obtained with 10 repeated trials and presented as a 99% confidence interval. Each repetition is performed by randomly sampling 70% of the dataset as training set and the remaining 30% as testing set. The hardware platform used for our experiments employs an *AMD EPYC CPU* (8 cores, 16 Threads, 2195MHz), 23 Gigabyte RAM, and an *NVIDIA Tesla T4* GPU.

To test our approach beyond the industrial case study corresponding to the TAR dataset, we also employ the 2012-2017 '*SemEval SEM STS*' dataset (Cer et al. 2017), a well-known benchmark dataset for semantic textual similarity tasks (Ranasinghe et al. 2019, Belinkov et al. 2019). The STS dataset consists of 8,628 sentence pairs including captions, news, and forum posts. Each pair corresponds to a human-labeled degree of similarity.

Each architecture module is built in Python, by using well known machine learning libraries, e.g. s*klearn* and *tensorflow*. The embedding module leverages the publicly available[2] source code of S-BERT, i.e. the pre-trained version *bert-large-nli-stsb-mean-tokens*. In our experiments, the scoring module is based on a MLP with four feedforward hidden layers consisting of 50, 40, 10, 30 neurons, respectively. As a loss function for the MLP we use the mean square error (MSE), as neurons' activation function we use *relu* (rectified linear unit), and as optimization strategy we use *adam* due to its computational efficiency and little memory requirements (Zhu et al. 2017).

Firstly, we test the ability of the embedding module to adapt to a specific domain context according to the number of training epochs. For this test, we use the Semantic Textual Similarity (STS) dataset, due to its topics variety. The embedding module is fine-tuned using a number of epochs equal to 1, 5, 10, 25 and 100. The scoring module is trained with an early stop strategy. Fig. 4 presents the confidence interval at 99% of the MSE obtained with these trials. Clearly, the confidence intervals of the MSE obtained with less than 25 epochs are consistently higher. On the other hand, there is no significant difference in terms of performances between 25 and 100 epochs. This suggests that 100 training epochs for finetuning the embedding module are sufficient to provide good performances, especially for an assessment task simpler than STS such as the TAR dataset presented in Section 4, which has fewer samples, and a smaller topics variety.

[2] https://github.com/UKPLab/sentence-transformers

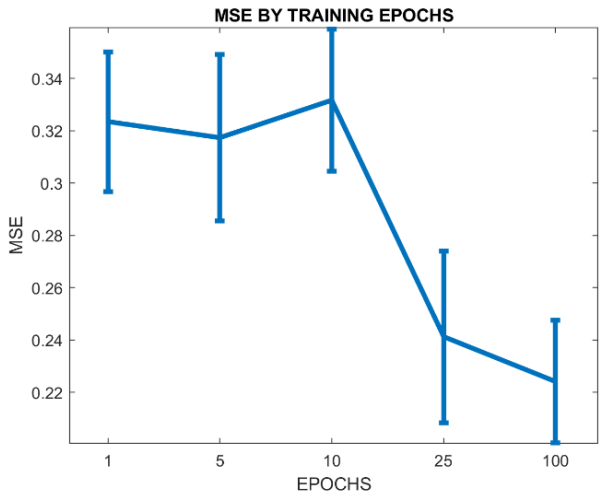

Fig. 4 MSE of the similarity assessment with STS dataset according to the number of fine-tuning epochs.

By using 100 fine-tuning epochs, we compare the effectiveness of the proposed approach against the one obtained with a differently implemented sentence embedding modules. Specifically, we replace S-BERT with USE (Cer et al. 2018) and employ a well-known bag-of-words approaches, such as the cosine similarity and BM25 (Wijewickrema et al. 2019). USE represents the state of the art for sentence embedding approaches based on transformers (Ahmed et al. 2019). The employed version of USE is publicly available on TensorFlow Hub[3]. It can transform sentences into 512-dimensional vectors, and it is pre-trained on a very large set of textual data including sources like Wikipedia, web news, web question-answer pages, and discussion forums. Table 3 shows how S-BERT provides a lower MSE with both STS and TAR datasets. Moreover, by considering the results shown in Fig. 4, the embedding module with S-BERT provides better performances than the other approaches, even with 1 fine-tuning epoch.

Table 3 MSE obtained with STS and TAR dataset by employing retrieval approaches based on different technologies.

	STS	**TAR**
Cosine	0.970 ± 0.004	0.059 ± 0.001
BM25	1.105 ± 0.013	0.055 ± 0.003
USE	0.827 ± 0.033	0.059 ± 0.003
S-BERT	**0.224 ± 0.024**	**0.0003 ± 0.0001**

Finally, we validate the effectiveness of the proposed approach by employing the *solution-relevance* dataset described in Section 4. We compute MAP and MRR by considering the top 5 solutions provided by our approach, the approach based on USE, and two bag of word approaches based on cosine similarity and BM25. Table 4 and 5 show the 99% confidence intervals of the resulting MAP and MRR by considering 0, 0.25 and 0.5 as the probability (*P*) that an unlabeled solution is relevant for the problem described.

[3] https://www.tensorflow.org/hub

Table 4 MAP@5 obtained with TAR dataset via different retrieval approaches.

	P = 0	**P = .25**	**P = .50**
Cosine	0.300 ± 0.063	0.508 ± 0.067	0.648 ± 0.057
BM25	0.288 ± 0.070	0.475 ± 0.075	0.637 ± 0.061
USE	0.383 ± 0.001	0.533 ± 0.019	0.667 ± 0.019
S-BERT	**0.706 ± 0.027**	**0.731 ± 0.016**	**0.760 ± 0.011**

According to Table 4 and Table 5, employing an embedding module based on S-BERT results in similar performances to the top ones reported in Section 2 but does not require any formalization of the application-specific semantic context.

Table 5 MRR@5 obtained with TAR dataset via different retrieval approaches.

	P = 0	**P = .25**	**P = .50**
Cosine	0.290 ± 0.061	0.556 ± 0.094	0.706 ± 0.093
BM25	0.285 ± 0.068	0.529 ± 0.068	0.704 ± 0.091
USE	0.388 ± 0.001	0.556 ± 0.018	0.687 ± 0.025
S-BERT	**0.750 ± 0.032**	**0.791 ± 0.025**	**0.823 ± 0.016**

If compared against the approaches based on bag-of-words methods (i.e. cosine similarity and BM25) and USE, the proposed approach results in a greater MAP and MRR. Finally, by considering the variability of MAP and MRR according to *P*, it is evident that the proposed approach results in a smaller number of unlabeled solutions among different trials.

Conclusions

In this work, we presented an application aimed at retrieving possible solutions for new problems by searching for similar problem stored in remote technical assistance reports. Despite the wide adoption of remote technical assistance service and the strategic advantages that this analysis can provide, there is a lack of such applications in the literature. Indeed, most of these approaches need the support of a structured semantic context to be effective, resulting in a huge management cost.

The application presented in this work overcomes this issue by adopting an architecture based on (i) a data preparation module aimed at providing punctuation and stop word removal, and word lemmatization, (ii) a sentence embedding module based on Sentence-BERT, and (iii) a scoring module, aimed at processing the distance between sentences' embeddings to produce a similarity score.

The obtained results show that: (i) the proposed approach provides better retrieval performances than the two well-known methods for information retrieval and USE, (ii) this occurs even with less than ten finetuning epochs, and (iii) the proposed approach corresponds to the lowest fluctuation when varying the probability that a new solution is effective, thus producing more consistent results for similar queries across different trials. The performances obtained with the proposed approach are higher than the best IR one presented in Section 2, yet it requires a very low management cost to

inject the domain-specific semantic context into the search engine. Indeed, the effort needed is aimed at collecting a few labeled pairs of sentences to finetune the search engine, rather than the large number of pairs that would be needed to train it from scratch.

Given the encouraging results, we aim at (i) improving the retrieval performances by employing a classifier based on deep learning, and (ii) including more heterogeneous textual data to study under what circumstances (e.g. topic diversity) the proposed approach requires retraining.

Acknowledgments

The authors thank the Körber Lab and the Customer Service Team of the Fabio Perini S.p.A. The former for the contribution as domain experts, the latter for supporting the validation of the proposed approach. The authors thank Lorenzo Nannini for his work on the subject during his master thesis.

Declarations

Conflicts of interest/Competing interests: the authors, as well as the industrial and academic partners involved in this research, declare that they have no interests related to the work submitted for publication, nor directly or indirectly.

Funding: this work is partially supported by (i) the company Fabio Perini S.p.A - Körber Tissue in the project "Intelligent Data Analysis algorithms for monitoring mono and multi-agent systems: Digital Applications in the Tissue Industry", and by (ii) the Italian Ministry of Education and Research (MIUR) in the framework of the CrossLab project (Departments of Excellence).

Availability of code, data and material: unavailable due to the industrial partner privacy policy.

References

Aarnio, P., Vyatkin, V., and Hästbacka, D. (2016, September). Context modeling with situation rules for industrial maintenance. In 2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA) (pp. 1-9). IEEE.

Ahmed, M., & Mercer, R. E. (2019, May). Efficient Transformer-Based Sentence Encoding for Sentence Pair Modelling. In Canadian Conference on Artificial Intelligence (pp. 146-159). Springer, Cham.

Ansari, F. (2019). Knowledge Management 4.0: Theoretical and Practical Considerations in Cyber Physical Production Systems. IFAC-PapersOnLine, 52(13), 1597-1602.

Ansari, F. (2020). Cost-based text understanding to improve maintenance knowledge intelligence in manufacturing enterprises. Computers and Industrial Engineering, 141, 106319.

Antonio L. Alfeo, Mario G.C.A. Cimino, Giuseppe Manco, Ettore Ritacco, Gigliola Vaglini (2020). Using an autoencoder in the design of an anomaly detector for smart manufacturing, Pattern Recognition Letters, vol. 136, pp 272-278, ISSN 0167-8655

Baldwin, T., Liang, H., Salehi, B., Hoogeveen, D., Li, Y., and Duong, L. (2016, June). UniMelb at SemEval-2016 Task 3: Identifying similar questions by combining a CNN with string similarity measures. In Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016) (pp. 851-856).

Belinkov, Y., & Glass, J. (2019). Analysis methods in neural language processing: A survey. Transactions of the Association for Computational Linguistics, 7, 49-72.

Bowman S. R., Angeli G., Potts C., and Manning C. D. (2015) A large, annotated corpus for learning natural language inference. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Daniel Cer D., Mona Diab M., Eneko Agirre E., Iñigo Lopez-Gazpio I., and Lucia Specia L. (2017) SemEval-2017 Task 1: Semantic Textual Similarity Multilingual and Cross-lingual Focused Evaluation Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval 2017). Available at: http://ixa2.si.ehu.eus/stswiki

Cer, D., Yang, Y., Kong, S. Y., Hua, N., Limtiaco, N., John, R. S., and Sung, Y. H. (2018). Universal sentence encoder. arXiv preprint arXiv:1803.11175.

Chahuara, P., Lampert, T., and Gancarski, P. (2016, September). Retrieving and ranking similar questions from question-answer archives using topic modelling and topic distribution regression. In International Conference on Theory and Practice of Digital Libraries (pp. 41-53). Springer, Cham.

Costa, R., Lima, C., Sarraipa, J., & Jardim-Gonçalves, R. (2016). Facilitating knowledge sharing and reuse in building and construction domain: an ontology-based approach. Journal of Intelligent Manufacturing, 27(1), 263-282.

Das, A., Shrivastava, M., and Chinnakotla, M. (2016, April). Mirror on the wall: Finding similar questions with deep structured topic modeling. In Pacific-Asia Conference on Knowledge Discovery and Data Mining (pp. 454-465). Springer, Cham.

Devlin J., Chang M., Lee K., Toutanova K., (2018). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". arXiv:1810.04805v2

Guo, J., Fan, Y., Pang, L., Yang, L., Ai, Q., Zamani, H., and Cheng, X. (2019). A deep look into neural ranking models for information retrieval. Information Processing and Management, 102067.

Gupta, P., Andrassy, B., & Schütze, H. (2018, August). Replicated Siamese LSTM in Ticketing System for Similarity Learning and Retrieval in Asymmetric Texts. In Proceedings of the Third Workshop on Semantic Deep Learning (pp. 1-11).

Heilman, M., and Smith, N. A. (2010, June). Tree edit models for recognizing textual entailments, paraphrases, and answers to questions. In Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (pp. 1011-1019).

Kathuria, M., Nagpal, C. K., and Duhan, N. (2016, March). A survey of semantic similarity measuring techniques for information retrieval. In 2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom) (pp. 3435-3440). IEEE.

Khabiri, E., Gifford, W. M., Vinzamuri, B., Patel, D., and Mazzoleni, P. (2019, November). Industry Specific Word Embedding and its Application in Log Classification. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management (pp. 2713-2721).

Lan, W., and Xu, W. (2018, August). Neural network models for paraphrase identification, semantic textual similarity, natural language inference, and question answering. In Proceedings of the 27th International Conference on Computational Linguistics (pp. 3890-3902).

Li, B., Zhou, H., He, J., Wang, M., Yang, Y., & Li, L. (2020, November). On the Sentence Embeddings from BERT for Semantic Textual Similarity. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP) (pp. 9119-9130).

Metzler, D. A., Croft, W. B., & Mccallum, A. (2005). Direct maximization of rank-based metrics for information retrieval. CIIR report, 429.

Navinchandran, M., Sharp, M. E., Brundage, M. P., & Sexton, T. B. (2021). Discovering critical KPI factors from natural language in maintenance work orders. Journal of Intelligent Manufacturing, 1-19.

Nemeth, T., Ansari, F., and Sihn, W. (2019). A Maturity Assessment Procedure Model for Realizing Knowledge-Based Maintenance Strategies in Smart Manufacturing Enterprises. Procedia Manufacturing, 39, 645-654.

North, K., Maier, R., and Haas, O. (2018). Value Creation in the Digitally Enabled Knowledge Economy. In Knowledge Management in Digital Change (pp. 1-29). Springer, Cham.

O'Donovan, P., Leahy, K., Bruton, K., and O'Sullivan, D. T. (2015). An industrial big data pipeline for data-driven analytics maintenance applications in large-scale smart manufacturing facilities. Journal of Big Data, 2(1), 25.

Othman, N., Faïz, R., and Smaïli, K. (2019, October). Manhattan Siamese LSTM for Question Retrieval in Community Question Answering. In OTM Confederated International Conferences" On the Move to Meaningful Internet Systems" (pp. 661-677). Springer, Cham.

Othman, N., Faiz, R., and Smaïli, K. (2020, June). Improving the Community Question Retrieval Performance Using Attention-based Siamese LSTM. In International Conference on Applications of Natural Language to Information Systems (pp. 252-263). Springer, Cham.

Pang, L., Lan, Y., Guo, J., Xu, J., Xu, J., and Cheng, X. (2017, November). Deeprank: A new deep architecture for relevance ranking in information retrieval. In Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (pp. 257-266).

Passaro, L., Bondielli, A., Lenci, A., Marcelloni, F.: UNIPI-NLE at CheckThat! 2020: approaching fact checking from a sentence similarity perspective through the lens of transformers. In: Cappellato, L., Eickhoff, C., Ferro, N., Névéol, A. (eds.): Working Notes of CLEF 2020–Conference and Labs of the Evaluation Forum (2020)

Ray, A., Aggarwal, P., Hadhazi, C., Dasgupta, G., and Paradkar, A. (2020, April). Question Quality Improvement: Deep Question Understanding for Incident Management in Technical Support Domain. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 34, No. 08, pp. 13196-13203).

Ranasinghe, T., Orasan, C., & Mitkov, R. (2019, September). Semantic textual similarity with siamese neural networks. In Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019) (pp. 1004-1011).

Reimers, N., and Gurevych, I. (2019). Sentence-bert: Sentence embeddings using siamese bert-networks. arXiv preprint arXiv:1908.10084.

Sexton, T., Brundage, M. P., Hoffman, M., and Morris, K. C. (2017, December). Hybrid datafication of maintenance logs from ai-assisted human tags. In 2017 ieee international conference on big data (big data) (pp. 1769-1777). IEEE.

Shtok, A., Dror, G., Maarek, Y., Szpektor, I.: Learning from the past: answering new questions with past answers. In: Proceedings of the 21st International Conference on World Wide Web, pp. 759–768, WWW 2012 (2012)

Sipos, R., Fradkin, D., Moerchen, F., and Wang, Z. (2014). Log-based predictive maintenance. Proceedings of the 20th ACM SIGKDD international conference on knowledge discovery and data mining (pp. 1867–1876).

Sunilkumar, P., & Shaji, A. P. (2019, December). A Survey on Semantic Similarity. In 2019 International Conference on Advances in Computing, Communication and Control (ICAC3) (pp. 1-8). IEEE.

Tao, F., Qi, Q., Liu, A., and Kusiak, A. (2018). Data-driven smart manufacturing. Journal of Manufacturing Systems, 48, 157-169.

Tong, B., Yanase, T., Ozaki, H., and Iwayama, M. (2015). Information Retrieval Boosted by Category for Troubleshooting Search System. In GSB@ SIGIR (pp. 28-32).

Usmanij, P. A., Khosla, R., & Chu, M. T. (2013). Successful product or successful system? User satisfaction measurement of ERP software. Journal of Intelligent manufacturing, 24(6), 1131-1144.

Vijayarani, S., Ilamathi, M. J., & Nithya, M. (2015). Preprocessing techniques for text mining-an overview. International Journal of Computer Science & Communication Networks, 5(1), 7-16.

Wang, D., Li, T., Zhu, S., and Gong, Y. (2010). iHelp: An intelligent online helpdesk system. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 41(1), 173-182.

Wijewickrema, M., Petras, V., & Dias, N. (2019). Selecting a text similarity measure for a content-based recommender system. The Electronic Library.

Xu, X., Zhou, S., Xiao, Y., Chang, W., Wei, F., and Yang, M. (2020, January). Text Mining-based Research on Aircraft Faults Classification and Retrieval Model. In 2020 Annual Reliability and Maintainability Symposium (RAMS) (pp. 1-7). IEEE.

Zhou, G., He, T., Zhao, J., and Hu, P. (2015, July). Learning continuous word embedding with metadata for question retrieval in community question answering. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers) (pp. 250-259).

Zhu, A., Meng, Y., Zhang, C., 2017. An improved adam algorithm using lookahead, in: Proceedings of the 2017 International Conference on Deep Learning Technologies, ACM. pp. 19–22.