On the use of Machine Learning and Deep Learning for Text Similarity and Categorization and its Application to Troubleshooting Automation

Julia Colleoni Couto, Laura Tomaz, Julia Godoy, Davi Kniest, Daniel Callegari, Felipe Meneguzzi, Duncan Ruiz
School of Technology, PUCRS University
{julia.couto, laura.tomaz, julia.godoy, davi.silva01}@edu.pucrs.br;
{daniel.callegari, felipe.meneguzzi, duncan.ruiz}@pucrs.br

Abstract

Troubleshooting is a labor-intensive task that includes repetitive solutions to similar problems. This task can be partially or fully automated using text-similarity matching to find previous solutions, lowering the workload of technicians. We develop a systematic literature review to identify the best approaches to solve the problem of troubleshooting automation and classify incidents effectively. We identify promising approaches and point in the direction of a comprehensive set of solutions that could be employed in solving the troubleshooting automation problem.

1. Introduction

Learning from text is an important subject for machine learning and deep learning models, providing valuable insights based on the execution of different algorithms [1]. For example, we can use models for text similarity recognition and text categorization or classification to learn from text [1]. Being able to identify text similarity and its categorization allows us to solve problems such as helping answer questions in community-based websites [2], predict author gender based on the text they wrote [3], and tasks related to bug-triage [4, 5, 6, 7] to automate the resolution of issues.

Large companies create many internal service requests, known as issues or incidents, and different people work to resolve them. These services have diverse complexities and levels of urgency. Some services are very specific, while a considerable portion is very similar or even contains identical requests. For the latter, companies can automatically replicate the solution employed in the first occurrence of an issue.

Answering these service requests takes time and requires the action of a specialist, who may sometimes not be available full-time. For this reason, the internal customer (and sometimes the external customer) needs to wait until an expert can resolve the request and fix the problem, but waiting usually leads to customer dissatisfaction. In this context, developing a model that suggests solutions for service requests impacts customer satisfaction, time spent by users, collaborators, and specialists, consequently incurring cost reduction.

This paper aims to understand the most used machine learning and deep learning algorithms for text similarity or text categorization applicable to troubleshooting automation. To do so, we perform a systematic literature review using five widely used web search engines to answer the following question: *What are the most used machine learning and deep learning techniques, algorithms, tools, or models for text similarity or categorization?* We start with 957 papers, and in the end, we accept 35 papers that answer our research question.

We classify the papers into eight categories: models and frameworks, proposed methods, preprocessing and dimension reduction, new text representations, attention-based models, multi-label related, comparative approaches, and bug-triage related. Our review leads to two key findings. First, most papers perform experiments based on only one dataset, and the most used public-available datasets are Reuters-21578 [1] and 20 Newsgroups [2]. Second, Support Vector Machine (SVM) is the most used machine learning model, while Convolutional Neural Network (CNN) is the most used deep learning model for text similarity or categorization.

In what follows, we investigate applications of machine learning to troubleshooting automation. Section 3 describes the methodology we followed, and Section 4 presents the results we achieved. Section 5 answers our research question and a discusses our results. Finally, Section 6 summarizes our conclusions and future work.

[1]Available at `https://archive.ics.uci.edu/ml/datasets/reuters-21578+text+categorization+collection`. Accessed in May 2021.

[2]Available at `https://archive.ics.uci.edu/ml/datasets/Twenty+Newsgroups`. Accessed in May 2021.

URI: https://hdl.handle.net/10125/79427
978-0-9981331-5-7

2. Theoretical Foundation

In this section, we introduce basic concepts related to Text Similarity & Categorization, Machine learning (ML) and Deep Learning (DL), and Troubleshooting Automation (TA), focusing on their main features and descriptions.

2.1. Text Similarity and Categorization

Text Similarity and Categorization are techniques for text analysis. Text similarity analysis refers to comparing two or more pieces of text regarding how strong is the semantic or syntactic similarity among them. The computation of text similarity can be based on the word frequency in the text [1]. On the other hand, text categorization or classification requires categories to which the data can be fitted [8]. Categorization is closely related to the clustering problem, but the difference is that in categorization, we have training and testing examples [1].

2.2. Machine Learning & Deep Learning

ML is a branch of artificial intelligence (AI) that combines Computer Science and Statistics. This area focuses on learning rules from data, adapting to changes, and improving performance with experience [9]. ML employs algorithms that parse data and use the new information to improve decisions [9]. In summary, an ML method based on a labeled dataset (supervised learning) or unlabeled dataset (unsupervised learning) can detect patterns and classify future inputs according to the analyzed data.

Deep learning (DL) is the area of ML that deals with complex artificial neural networks. According to Bengio et al. [10], DL uses a hierarchy of concepts to learn complex concepts by connecting them to simpler ones. Artificial neural networks operate through the composition of networks of simple functions (referred to as neurons) connected to each other by weights that represent how one function influences other connected functions. DL can automatically learn representations from data, such as images, video, or text, without depending entirely on human-crafted features [10].

2.3. Troubleshooting Automation

Troubleshooting Automation is a systematic approach to problem-solving in complex machines or systems [11]. In this paper, we survey techniques amenable to automating the resolution of incidents using ML and DL. An *incident* is a service request created by a user to be resolved by a member of the Information Technology (IT) support team that usually works in the same institution. The time people spend resolving incidents can lead to low productivity and increased costs for the institution. Whenever we can automate the resolution of the incidents, we reduce the time to resolve the service request, reducing cost. The purpose of automation, in this case, is for the system to decide which previous incident is the most closely related one and suggest solutions based on the history of solutions to such past incidents.

Table 1. Example of textual features related to the incidents

Feature	Description
Short_description:	Brief description of the incident.
Description:	Complete description of the incident, usually with examples and logs of errors, when available.
Comments:	Messages between the customer and the attendant.
Work_notes:	Same as the comments, but it is only visible for the internal team.
Close_notes:	Brief description of what was done to solve the incident.

We are interested in the data stored in the service desk system, which contains the incident and the related conversation. Some systems generate error codes when a failure occurs, and eventually, the user complements the incident description by attaching application logs, which contain details of the errors.

Different features can be selected as input for ML and DL algorithms, based on the incidents datasets. We provide an example of the features we could use in Table 1. In this example, our target (the features we want to learn) is the *Close_notes*. One example of an incident and solution that could be automated is the following:

- Short_description: I cannot print an order.
- Description: The print button is not enabled for order #1234, although the order is finished and ready to be printed. Could you check that for me?
- Comments: -
- Work_notes: Service *x* stooped again, we have to update it soon.
- Close_notes: Restart the service *x*.

Besides the features above, we could use the incident ID, incident category, creation date, and resolution date. Computational ways of automating troubleshooting begin with the principle of understanding and categorizing the incidents, which, according to our example, can be represented as text, numbers, or data, to then choose some solution (that can also be represented as text) automatically. In this context, ML and DL can be useful in the TA problem, since it enables us to use specific algorithms for text classification and categorization.

3. Material and Methods

We perform a Systematic Literature Review (SLR), as proposed by Kitchenham et al. [12]. Our focus is to identify the main contributions regarding text similarity and classification and provide an overview of models, techniques, algorithms, and tools used in this research area. According to Kitchenham et al. [12], an SLR has three main phases, as follows.

The first phase refers to planning the review—formulating the research questions and protocols. For that purpose, we define one main Research Question (RQ) to guide our development: "What are the most used machine learning and deep learning techniques, algorithms, tools, or models for text similarity or categorization?". We search for the most used because it allows us to infer the most adequate techniques, algorithms, tools, or models. In order to facilitate the elaboration of research definitions, we follow the PICO method proposed by Sackett et al. [13] to formulate the search string:

- Population: papers that describe techniques, algorithms, tools, or models for text similarity or categorization;
- Intervention: using machine learning or deep learning techniques;
- Comparison: -;
- Outcome: techniques, algorithms, tools, models.

During protocol development, we determine a control study (Zaidi et al. [6]) based on the objective of this literature review. We define the following search string for submission to online search engines: *(”machine learning” OR ”deep learning” OR ”neural network”) AND (incident OR issue OR problem OR troubleshooting) AND (embeddings OR ”text similarity” OR ”text categorization”) AND (technique OR algorithm OR tool OR model).*

The search is limited to papers written in English published from 2017 until 2021. The inclusion criteria are: I) Qualitative or quantitative research about the research theme; II) Describes a complete study in electronic format; III) Conference paper, review, or journal. On the other hand, the exclusion criteria are: I) Incomplete or short paper (less than 4 pages); II) Study is unavailable for download; III) Study is not related to the topic of our research; IV) Duplicated study; V) Written in a language other than English; VI) Conference proceedings index; VII) Published more than 4 years ago; VIII) Literature review; IX) Book or book chapter; X) Thesis or dissertation.

Table 2. Papers by search engines

Source	Initial	1st filter	2nd filter	Final
ACM	359	169	14	7
arXiv	7	0	0	0
Google Scholar	26	11	0	0
IEEE Xplore	269	93	31	13
Web of Science	296	198	32	15
Total:	957	469	77	35

The second phase is conducting the review, and it consists primarily of paper selection. We searched for papers in five online search engines: ACM Digital Library, arXiv, Google Scholar, IEEE Xplore, and Web of Science. The search string in the search engines above, resulted in a total of 957 papers. Table 2 shows the number of papers from each search engine.

We use the StArt (State of the Art through systematic review) tool (Fabbri et al. [14]) to facilitate the process of extraction and compilation of the data, resulting in 469 papers. The following phases consist of: i) the initial selection phase, we apply the inclusion and exclusion criteria to the papers based on title and abstract, rejecting 392 papers; ii) the final selection phase, we thoroughly analyze the 77 papers left, using the criteria to decide whether the paper fits the objectives of our research, then accepting 35 papers.

The third and last phase from Kitchenham et al. [12] is document review. In this part, we write and review the report itself, which we report next.

4. Results

To start the analysis of the papers, we group the papers into eight main categories, according to an analysis of the keywords, important terms in the abstracts, as well as similarities between the approaches followed by the authors. The groups we identified are: 1. Models and frameworks, 2. Proposed methods, 3. Pre-processing and dimension reduction, 4. New text representations, 5. Attention-based models, 6. Multi-label related, 7. Comparative approaches, and 8. Bug-triage related. Next, we explain each group along with the related papers.

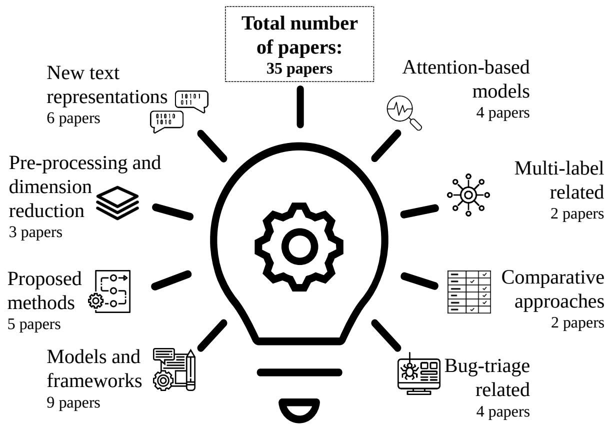
Total number of papers: 35 papers
New text representations 6 papers
Attention-based models 4 papers
Pre-processing and dimension reduction 3 papers
Multi-label related 2 papers
Proposed methods 5 papers
Comparative approaches 2 papers
Models and frameworks 9 papers
Bug-triage related 4 papers

Figure 1. Eight categories of papers.

4.1. Categorization of the papers

In what follows, we classify the accepted papers providing brief description of each category. We conclude the categorization with a summary of the number of papers per category in Figure 1. Most of the papers are related to the Models and Frameworks category (9 papers), and the second most popular category is related to New text representation methods (6 papers).

4.1.1. Models and frameworks

Research that propose different models and frameworks based on text classification or similarity are the majority in the final selected papers. Liu et al. [15] develop a Centroid-Based Classification Model, the Gravitation Model (GM), in which a Vector Space Model (VSM) represents the documents. This model is useful when we work with a class-imbalanced dataset. Like in universal gravitation, the classes represent the objects with the attractive forces in this model.

Zhang, Gao, and Fang [16] propose a model to classify news titles according to related topics. This model is called the Word-Embedding-based Sentence-LDA (WESL) model. The authors first convert the letters to lowercase, then delete stop-words and non-alphabetic characters to later delete words that appear once or twice in the dataset.

Chen et al. [2] develop the Heterogeneous Social Influential Network. They focus on question retrieval for community-based question answer to help users select historical questions that match their new questions, based on semantics or relevancy. Their model learns about the textual content of the questions, information about related categories, and social information of people who ask questions in the system to rank the most similar historical proposed questions to the new question. Consulting previous text answers is relevant to TA since we need to process a new incident and base its solution on historically similar problems.

Adam et al. [17] develop two Long Short-Term Memory (LSTM) models to monitor software use based on the history of the actions of the users in the interface. The authors use an LSTM network with action embeddings (one-hot encoded vectors). They feed the LSTM network with feature vectors for their crash detection task, composed of actions with above-average crash probabilities.

Tellez et al. [18] develop a framework to create a text classifier regardless of both the domain and the language, based only on a training set of labeled examples. The authors approach creating effective text classifiers as a combinatorial optimization problem—a space with all the options of text transformations, tokenizers, and weighting procedures. They use a meta-heuristic (Random Search and Hill Climbing—a procedure to generate/select a good solution to an optimization problem) to produce an effective text classifier in this space. The name of this model selection procedure is μTC (micro–Text Classification).

Kong et al. [19] develop a framework to evaluate the helpfulness of a product review. During pre-experiments, the authors note that using only features learned from a CNN model can achieve better performance than hand-crafted features, and that a CNN model outperforms TransE. Based on the results of the pre-experiments, they build an automatic model based on CNN and TransE. Additionally, the paper from Guo et al. [20] introduces a framework to recommend jobs (text data) that consisted of integrating four different levels: feature-level, model-level, data-level, and approach-level.

Two papers focus their research on entity-related proposals. First, Conover et al. [21] develop Pangloss, a production system for entity disambiguation on noisy text. Pangloss combines a probabilistic linear-time key phrase identification algorithm with a semantic similarity engine based on context-dependent document embeddings to achieve better than state-of-the-art results. Second, Ahmadvand et al. [22] introduce ConCET, a DL algorithm that improves topic classification on human-machine and human-human conversations combining character, word, and entity type embeddings into a single representation.

4.1.2. Proposed methods

Text similarity and classification can be executed in different ways. Bellaouar, Cherroun, and Ziadi [23] based their research on String Subsequence Kernel (SSK). String Kernel is a function used to measure the similarity of pairs of strings; the more similar two strings are, the higher the value of a string kernel. SSK belongs to this family, and the main idea is to compare strings depending on common substrings or subsequences they contain. To improve the time consumption of SSK computation, the authors develop a novel Geometric-based approach by extending the layered-range tree (LRT) data structure to a layered-range sum tree (LRST). Unfortunately, this approach depends on alphabet size, and it is not efficient for a small alphabet. Nevertheless, SSK can be used in many applications, such as text categorization.

Silva, Almeida, and Yamakami [24] present MDLText, a text-classifier method to process high dimensional data quickly. This method is based on the Minimum Description Length (MDL) principle (fewer complex models are preferable, but there is no standard procedure for calculating this). MLDText can prevent overfitting, when a model learns the detail and noise in the training data, negatively affecting the model's performance in the test data. In addition, the authors point out that MDLText has a low computational cost.

Wei et al. [25] propose a graph recurrent neural network for text categorization. In the paper, the authors compare their method to widely used text classification methods, such as CNN, LSTM, RNN, and achieve higher accuracy than the other methods using GloVe with 300-dimensional embeddings to initialize word representation.

Goudjil et al. [26] focus on text categorization using what the authors call active learning, in which the main objective is to reduce the labeling effort and maintain accuracy by selecting samples to be labeled. Their algorithm is based on SVM with a probabilistic output to be able to deal with multi-class labels. Active learning is used by selecting a batch of informative samples to be labeled by a domain expert. Their proposed approach is called AL-MSVM, based on the posterior probability estimated by a set of SVM classifiers.

The work of Gadri and Moussaoui [27] addresses the problem of automatic topical text categorization. The authors use a new approach based on the k-Nearest Neighbor (k-NN) algorithm, as well as a new set of pseudo-distances (distance metrics) known in the field of language identification. This constitutes a simple and effective method to improve the quality of performed categorization.

4.1.3. Pre-processing and dimension reduction

When dealing with text tasks such as classification, data size can be massive. This necessitates dimensionality reduction methods, which the authors do as follows. First, Xu et al. [28] develop a Deep Clustering via Variational Auto-Encoder (DC-VAE) of mutual information maximization. Deep clustering refers to the process of guiding clustering methods jointly with automatic learning representation from the high-semantic and high-dimensional data via deep neural networks. Compared to k-NN, which is computationally costly due to its lazy learning pattern, DC-VAE performs well with high-dimensional space features.

Chen et al. [3] propose an approach that combines the Latent Semantic Indexing (LSI) method (dimension reduction purpose) with k-NN to predict the gender, based on a real-life collection of posts on actual blog pages. Das Gollapalli and Jung-Jae [29] tackle the task of classifying wordmarks, which are essentially a small set of words (less than five most of the time). They describe how to prepare the data using different syntactic representations simultaneously as features for best performance, then compare the classification performance with a large variety of methods.

4.1.4. New text representations

Text representation is an important step for similarity or classification tasks. Different authors propose different models or techniques for text representation to be used in such tasks. For example, Hongpeng and Jia [30] develop a sentence vectorization method based on task contribution that uses the improved information gain feature selection method (IIG-SIF). IIG-SIF vectorization can be used in two tasks: i) text categorization based on neural networks; ii) text similarity using a specific formula described by them.

Li et al. [31] present a novel framework called Text Concept Vector, which uses both neural networks and a knowledge base to produce a high-quality representation of text. They test their framework in sentence pair similarity and sentiment classification, performing well against popular text representations.

Shuang et al. [32] develop a text representation model named Convolution–Deconvolution Word Embedding (CDWE). CDWE is a multi-prototype and end-to-end fusion embedding that composes specific information for the context and tasks and extracts semantic and syntactic information. The authors focus on polysemy (terms having more than one meaning) and task unawareness (the type of task before analyzing the words). They apply their model to word embedding generation to then use it in machine translation and text classification.

Jiang et al. [33] propose a model for Latent Topic Text Representation (LTTR) based on word embedding (word2vec-CBOW model) to learn a text distance measure in the framework of a statistical manifold (information geometry/probability distribution). They use a Gaussian Mixture Model to describe the word distribution, where each Gaussian represents a potential topic. This kind of text classification requires a distance metric to measure how much the texts differ, of which they describe four different measurement theories and introduce a distance metric using statistical manifold learning. They compare the results of the proposed method with other state-of-the-art methods, using the SVM and k-NN classifiers on the test sets.

Zheng et al. [34] introduce Hierarchical Collaborative Embedding (HCE) for a context-aware recommendation. To overcome the bag-of-words limitation, which does not capture semantic meaning, and RNN gradient vanishing problem, in which in a long input word sequence RNN tends to forget previous words, they use HCE to learn hierarchical item embeddings from the textual content. Because it is hierarchical, HCE derives a better item embedding, achieving more accurate recommendation results. Hierarchical Recurrent Network (HRN) is a component of the proposed HCE. HRN learns embeddings of documents associated with items. To validate HCE effectiveness, they compare it with five baseline models—HCE consistently outperforms all baselines.

Hourali, Zahedi, and Fateh [35] present a new coreference resolution approach. A coreference occurs when two or more expressions in a text refer to the same person or thing. The authors incorporate RoBERTa embeddings with a neural Multi-Criteria Decision Making (MCDM) method. RoBERTa uses syntactic and semantic information and extracts correct mentions with different lengths from the text, and is used for word embedding to obtain better contextual information. MCDM is accurate for ranking candidate antecedent and better detection rate of co-referent mentions. Thus, their proposal manages the problem of coreference resolution with the lowest error rate.

4.1.5. Attention-based models

Attention mechanisms consider the importance of words in a domain/task. The importance of words depends on the data domain, and the text task goal impacts on the results of the models. The attention mechanism allows the setting of different weights to highlight important information from the context. The following authors use attention mechanisms for analyses and classification purposes. Liu and Guo [36] develop an architecture for text classification based on an attention-based BiLSTM with a convolutional layer (AC-BiLSTM) in the pre-processing step, for sentiment and question classification.

Zheng and Zheng [37] propose Bidirectional Recurrent Convolutional Neural Network Attention-Based Model (BRCAN), a model that combines the bidirectional LSTM for context, and the CNN with the attention mechanism for keywords with Word2Vec. Their model can assign different weights to words in sentences according to their importance to the classification given by attention mechanisms. BRCAN got results that outperformed traditional ML models.

Shi and Lu [38] build a bidirectional hierarchical LSTM network model (HBLSTM-ATT) based on the attention mechanism and calculate the correlation before and after the sentence. At the same time, their model focuses on document-level classification, using hierarchical structures to build document-level vectors from word vectors. Yuan [39] considers that BiLSTM has been widely used in the field of text classification but still has room for improvement in accuracy and feature extraction. In response, the author proposes a BiLSTM-WS attention model, which achieves good results on the used datasets.

4.1.6. Multi-label related

Categorization problems require a more complex classification due to multi-label classes and non-unique correct answers. The following two papers focus on multi-label tasks. Imrattanatrai, Kato, and Yoshikawa [40] develop a model based on neural networks for multi-label sentence classification that classifies sentences as representing properties given a target entity. They introduce zero-shot learning to train sentences when properties are unavailable.

Gargiulo et al. [41] propose a methodology to regularize data labels named Hierarchical Label Set Expansion (HLSE). HLSE expands the label set of each document integrating all the missing labels along the label hierarchy, and analyzes the impact of different semi-supervised word embedding models.

4.1.7. Comparative approaches

Understanding the overall differences in methods/models is important, leading the following authors to compare the approaches for text classification. Surkova, Skorynin, and Chernobaev [42] analyze and compare two commonly used linguistic models (cognitive approach and word embeddings) by their classification capacity. The authors conclude that opting for either of the models does not necessarily guarantee improved classification quality.

Zhu and Wong [43] focus on text categorization using five known algorithms (k-NN, Naive Bayes (NB), SVM, AdaBoost, and Multi-Layer Perceptron (MLP)) to evaluate an automatically generated labeled dataset. In addition, they use five types of feature selection applied to their dataset: Chi-Square (CHI), information gain (IG), mutual information (MI), odds ratio (OR), and GSS coefficient.

4.1.8. Bug-triage related

Four of the 35 selected papers use "bug-triage" as term for categorizing problems found in the software domain. Despite the purpose of that categorization, they are all based on the principle of text classification. For example, Zaidi et al. [6] propose a CNN framework to assign a bug classified from their textual information (e.g., summary and description) to a specific developer, which, in essence, is the same as categorizing a problem and assigning it to someone or something related to that category. To compare their performances, they test the framework using word embeddings, such as Word2Vec, GloVe, and ELMo.

Lee et al. [4] use a similar approach with DL to reach the same objective. In addition, they develop an architecture with word embedding using Word2Vec in a multi-language context, as it helps deduce latent meaning from the regular words and jargon.

Next, we review two more applications of text categorization in bug-related problems. First, Ardimento and Mele [7] use Bidirectional Encoder Representations from Transformers (BERT) to predict the time for bug resolution in the bug-tracking system. Second, Phetrungnapha and Senivongse [5] analyze user reviews on popular application stores to classify them between new feature requests and bug reports to automatically generate tickets in an issue-tracker system. Their approach first classifies user reviews, then determines which ones are duplicates to generate the issue ticket. According to their experiments, the best classifier is the Extra Tree ensemble model, using doc2vec embedding to represent comments from the users.

4.2. Characteristics of the datasets used in the experiments

Based on the selected papers, we identify the most used datasets for text similarity or categorization. Most papers report using only one dataset (14 papers), but the amount varies from 1 to 45 datasets. The papers that perform experiments on more than three datasets are: Silva, Almeida, and Yamakami [24] (45 datasets), Liu et al. [15] (12 datasets), Liu and Guo [36] and Tellez et al. [18] (7 datasets), Zheng and Zheng [37] and Xu et al. [28] (6 datasets), and Conover et al. [21] (5 datasets), Shi and Lu [38] and Wei et al. [25] (4 datasets). We also identify that eight papers analyze three datasets, and five papers analyze two datasets. The papers we select analyze 95 different datasets. Some datasets are used to validate approaches in more than one paper. The most used datasets are Reuters-21578, a dataset that contains news (10 papers) and 20 Newsgroups, a dataset with newsgroups (7 papers).

The reason for the authors that choose to present their experiments based on only one dataset varies. Lee et al. [4], Phetrungnapha et al. [5], Zhu et al. [43] claim the reason for adopting one dataset is because of the originality of their studies. Gadri et al. [27] and Surkova et al. [42] decided to use well-known benchmark corpus (Reuters-21578 and 20 Newsgroup) to ease replicability. Guo et al. [20] had to use a dataset from the ACM RecSys Challenge 2017 [44], since they produced the paper for the challenge. Adam et al. [17] show from their results that selecting a larger dataset would be better to take advantage of the learning capacity of the DL model used by the authors.

Chen et al. [3], Yuan [39], and Gargiulo et al. [41] stated they used only one dataset because they needed domain-specific data. In two papers, they present their experiments with only one dataset because they needed handcrafted features (Das Gollapalli et al. [29], and Kong et al. [19]). Finally, Ardimento and Mele [7] state that their choice to use only one dataset is a threat to validity in their research since it could not significantly represent the domain and, therefore, difficult replicability.

5. What are the most used machine learning and deep learning techniques, algorithms, tools, or models for text similarity or categorization?

To answer this question, we analyze the algorithms of the 35 selected papers used as baselines and the ML or DL technique they used. Among the 35 mapped papers, we identify 29 ML techniques and 42 DL techniques. The most used ML technique is SVM. SVMs [45] are discriminative linear classifiers based on the concept of decision planes that defines decision boundaries learned between different classes, linear or nonlinear. The second most used ML technique is Naive Bayes (NB) classifier. NB assumes that a particular feature in a class is unrelated to any other feature. In other words, it assumes that each input variable is independent [37]. The next one is k-NN. k-NN is instance-based learning, which means the function is locally approximated, and all computation is deferred until classification. Its main idea is to measure the similarity of new instances with training instances in an n dimensional space [3].

As for DL techniques, the most common one is CNN. CNNs are specialized in processing data in the form of multiple arrays, such as images (2D), audio and video, or volumetric data (3D) [10]. The first use of CNNs was to recognize simple shapes, in this case, handwritten digits [46]. The second most used DL technique is LSTM. LSTM is a type of recurrent neural network (RNN) that has feedback connections. It addresses the problem of vanishing gradient in RNNs by replacing self-connected hidden units with memory blocks [36]. The standard LSTM network only exploits historical context, but the lack of future context may lead to an incomplete understanding of the problem. Therefore, BiLSTMs aims to exploit both historical and future contexts by combining a forward hidden layer and a backward hidden layer [36]. The last DL technique is BERT. BERT is a new language representation model based on a bidirectional contextualized representation of words, which allows generating word embeddings that retain information about the context of words within the sentence [7].

Table 3 lists the top three ML, and DL techniques used more than once among the papers. Other ML techniques, such as logistic regression, multi-layer perceptron, and extra trees, appeared respectively 4, 4, and 2 times among authors, but are not part of the top three. We omit from the table ML and DL techniques used only once.

Our specific goal is to find the most used ML and DL approaches that deal with TA tasks, even when applied with a different context. In troubleshooting tasks, possible problems tend to be repetitive or even similar: depending on the complexity of each solution, it can be automatically solved because the problems are alike or the same.

In the final list of selected papers, the ones in the bug-triage section are well related to this research, especially Zaidi et al. [6] and Lee et al. [4]. Bug as a software problem can be considered an issue. These authors use textual information of bugs classified by ML or DL to later assign it to a proper developer, a specialist in the classified problem.

Chen et al. [2] also develop an approach connected with the possible use for TA, in which their model can find similarities between old answered questions, and a new question from the user, leading us to think in a good use for finding associations between old and new

Table 3. Top 3 most used Machine Learning and Deep Learning techniques.

Classifier	Amount	Papers
Machine Learning		
- SVM	15	[5, 6, 16, 17, 18, 19, 24, 27, 31, 32, 33, 36, 37, 38, 43]
- Naive Bayes	10	[3, 5, 19, 24, 27, 30, 36, 37, 39, 43]
- k-NN	5	[3, 24, 32, 33, 43]
Deep Learning		
- CNN	12	[4, 6, 19, 22, 25, 31, 32, 36, 38, 39, 40, 41]
- LSTM / BiLSTM	10	[17, 25, 31, 32, 35, 36, 37, 39, 40, 42]
- BERT	2	[7, 35]

problems, to then execute or even propose the same solution. When it comes to the task of identifying those solutions, we can use similar techniques but now applied to categorizing the resolution or merging neural networks with a knowledge base, as Li et al. [31] presented, to then create a high-quality text representation of them.

Finally, we summarize the approaches to text analysis that can be used in the TA problem. All selected papers have at least one possible link to TA: different models and frameworks based on textual tasks, attention-based and multi-label related models, ways of pre-processing data, and reducing their dimensionality, novel text representations, and methods used for classification and similarity.

Table 4 groups the selected papers into two classes according to their approach to TA and categorizes them based on their application. Most papers fit "Classifying the issue" since they are related to topic classification or text similarity. However, "Identifying double-meaning terms" contains articles related to specific terms, such as wordmarks and polysemy, that—like others in this class—can be incorporated into a TA system to improve its operation.

Among the papers related to text categorization, [25], [33], and [24] can be applied to the feature *Close_notes*, as shown in Table 1, to classify the incident into a category with similar instances. In many cases, the description contains the informal text. On this topic [21] introduces a new model for information retrieval, taking into account noisy text and entity disambiguation. Another automatic approach to TA is analyzing the semantic similarity between

Table 4. Approaches to Troubleshooting Automation

TA application	Papers
Directly approaches TA	
- Classifying the issue	[2, 3, 4, 5, 6, 16, 21, 22, 23, 24, 25, 26, 28, 30, 31, 33, 36, 37, 38, 39, 40, 41, 42, 43]
- Dealing with imbalanced data	[15]
Indirectly approaches TA	
- Analyzing historical interactions data	[17, 20, 34]
- Identifying double-meaning terms	[29, 32, 35]
- Identifying languages	[18, 27]
- Receiving feedback	[19]
- Predicting resolution time	[7]

incident *Close_notes*, as proposed by methods described in [31] and [30]. Once we find these correspondences, the system can direct the user to a solution or conduct the solution process itself.

6. Conclusion

This paper provides a systematic literature review on machine learning and deep learning for text similarity and categorization, applied to the troubleshooting automation problem. We selected 35 papers that answer our research question, and we grouped the papers into eight categories, according to their similarity: models and frameworks, proposed methods, preprocessing and dimension reduction, new text representations, attention-based models, multi-label related, comparative approaches, and bug-triage related.

One of our key findings regards the datasets used in the experiments for text classification or categorization. Specifically, most papers analyze only one dataset (37% of the papers), while a single paper used 45 datasets in its experimental validation ([24]). Our results suggest that Reuters-21578 and 20 Newsgroup datasets could be a good point for validating a text classification or categorization model since they were the most widely used ones in the papers selected.

Even though we did not list all ML and DL models used by each paper, we detected a wide variety of models. The most used ML and DL models—SVM and CNN—are widely employed for text classification or categorization. Finally, we identify how the approaches from the papers we review could be used in the troubleshooting automation problem. All the algorithms we found can be useful for Troubleshooting Automation.

As future work, we plan to develop and validate a model for troubleshooting automation, based on a real dataset that contains the features we discussed earlier and the results we collected in this SLR. We plan to validate this model using Focus Groups in a widely known multinational company to evaluate and discuss the results with domain experts.

ACKNOWLEDGEMENTS

This work was supported by Dell Brazil using incentives of the Brazilian Informatics Law (Law no 8.2.48, year 1991).

References

[1] C. C. Aggarwal, *Machine learning for text*. Springer, 2018.

[2] Z. Chen *et al.*, "Question retrieval for community-based question answering via heterogeneous social influential network," *Neurocomputing*, vol. 285, pp. 117–124, 2018.

[3] J. Chen *et al.*, "Gender prediction on a real life blog data set using lsi and knn," in *Annual Computing and Communication Workshop and Conf.*, pp. 1–6, 2017.

[4] S.-R. Lee *et al.*, "Applying deep learning based automatic bug triager to industrial projects," in *Joint Meeting on Foundations of SE*, p. 926–931, 2017.

[5] K. Phetrungnapha and T. Senivongse, "Classification of mobile application user reviews for generating tickets on issue tracking system," in *Int. Conf. on Inf. Com. Technology and System*, pp. 229–234, 2019.

[6] S. Zaidi *et al.*, "Applying conv. neural networks with different word representation techniques to recommend bug fixers," *IEEE Access*, vol. 8, p. 18, 2020.

[7] P. Ardimento and C. Mele, "Using bert to predict bug-fixing time," in *Conf. on Evolving and Adaptive Intelligent Systems*, pp. 1–7, 2020.

[8] A. Dhar *et al.*, "Text categorization: past and present," *Artificial Intelligence Review*, vol. 54, no. 4, pp. 3007–3054, 2021.

[9] T. M. Mitchell, "Machine learning. 1997," *Burr Ridge, IL: McGraw Hill*, vol. 45, no. 37, pp. 870–877, 1997.

[10] Y. Bengio, I. J. Goodfellow, and A. Courville, "Deep learning," *Nature*, vol. 521, pp. 436–444, 2015.

[11] D. H. Jonassen and W. Hung, "Learning to troubleshoot: A new theory-based design architecture," *Educational Psychology Review*, vol. 18, no. 1, p. 77, 2006.

[12] P. Brereton *et al.*, "Lessons from applying the systematic literature rev. process within the se domain," *Journal of Systems and Software*, vol. 80, no. 4, pp. 571–583, 2007.

[13] R. B. Haynes *et al.*, "Evidence-based medicine: How to practice & teach ebm," *Canadian Medical Association. Journal*, vol. 157, no. 6, p. 788, 1997.

[14] S. Fabbri *et al.*, "Improvements in the start tool to better support the systematic review process," in *Int. Conf. on Evaluation and Assessment in SE*, 2016.

[15] C. Liu *et al.*, "A new centroid-based classification model for text categorization," *Knowledge-Based Systems*, vol. 136, pp. 15–26, 2017.

[16] F. Zhang, W. Gao, and Y. Fang, "News title classification based on sentence-lda model and word embedding," in *Int. Conf. on Machine Learning, Big Data and Business Intelligence*, pp. 237–240, 2019.

[17] C. Adam *et al.*, "Dynamic monitoring of software use with recurrent neural networks," *Data & Knowledge Engineering*, vol. 125, p. 101781, 2020.

[18] E. S. Tellez *et al.*, "An automated text categorization framework based on hyperparameter optimization," *Knowledge-Based Systems*, vol. 149, pp. 110–123, 2018.

[19] L. Kong *et al.*, "Predicting product review helpfulness a hybrid method," *IEEE Transactions on Services Computing*, pp. 1–1, 2020.

[20] C. Guo *et al.*, "How integration helps on cold-start recommendations," in *RecSys Challenge*, 2017.

[21] M. Conover *et al.*, "Pangloss: Fast entity linking in noisy text environments," in *Int. Conf. on Knowledge Discovery & Data Mining*, p. 168–176, 2018.

[22] A. Ahmadvand *et al.*, "Concet: Entity-aware topic classification for open-domain conversational agents," in *Int. Conf. on Information and Knowledge Management*, p. 1371–1380, 2019.

[23] S. Bellaouar, H. Cherroun, and D. Ziadi, "Efficient geometric-based computation of the string subsequence kernel," *Data Mining and Knowledge Discovery*, vol. 32, no. 2, pp. 532–559, 2018.

[24] R. M. Silva, T. A. Almeida, and A. Yamakami, "Mdltext: An efficient and lightweight text classifier," *Knowledge-Based Systems*, vol. 118, pp. 152–164, 2017.

[25] X. Wei *et al.*, "Recurrent graph neural networks for text classification," in *Int. Conf. on Software Engineering and Service Science*, pp. 91–97, IEEE, 2020.

[26] M. Goudjil *et al.*, "A novel active learning method using svm for text classification," *Int. Journal of Automation and Computing*, vol. 15, no. 3, pp. 290–298, 2018.

[27] S. Gadri and A. Moussaoui, "Application of a new set of pseudo-distances in documents categorization," *Neural Network World*, vol. 27, no. 2, p. 231, 2017.

[28] C. Xu *et al.*, "Deep clustering by maximizing mutual information in variational auto-encoder," *Knowledge-Based Systems*, vol. 205, p. 106260, 2020.

[29] S. Das Gollapalli and K. Jung-Jae, "Effective identification of distinctive wordmarks," in *Companion Proceedings of the Web Conf.*, p. 471–477, 2020.

[30] T. Hongpeng and J. Jia, "Sentence embedding model based on feature selection," in *Int. Conf. on Computer Engineering and Application*, pp. 659–663, 2020.

[31] Y. Li *et al.*, "Incorporating knowledge into neural network for text representation," *Expert Systems with Applications*, vol. 96, pp. 103–114, 2018.

[32] K. Shuang *et al.*, "Convolution–deconvolution word embedding: An end-to-end multi-prototype fusion embedding method for natural language processing," *Information Fusion*, vol. 53, pp. 112–122, 2020.

[33] B. Jiang *et al.*, "Latent topic text representation learning on statistical manifolds," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 11, pp. 5643–5654, 2018.

[34] L. Zheng *et al.*, "Hierarchical collaborative embedding for context-aware recommendations," in *Int. Conf. on Big Data*, pp. 867–876, 2017.

[35] S. Hourali, M. Zahedi, and M. Fateh, "Coreference resolution using neural mcdm and fuzzy weighting technique," *Int. Journal of Computational Intelligence Systems*, vol. 13, pp. 56–65, 2020.

[36] G. Liu and J. Guo, "Bidirectional lstm with attention mechanism and conv. layer for text classification," *Neurocomputing*, vol. 337, pp. 325–338, 2019.

[37] J. Zheng and L. Zheng, "A hybrid bidirectional recurrent convolutional neural network attention-based model for text classification," *IEEE Access*, vol. 7, pp. 106673–106685, 2019.

[38] X. Shi and R. Lu, "Attention-based bidirectional hierarchical lstm networks for text semantic classification," in *Int. Conf. on Information Technology in Medicine and Education*, pp. 618–622, 2019.

[39] Y. Yuan, "Research on text classification algorithm based on bilstm-wsattention," in *Advanced Information Technology, Electronic and Automation Control Conf.*, vol. 5, pp. 2235–2239, 2021.

[40] W. Imrattanatrai, M. P. Kato, and M. Yoshikawa, "Identifying entity properties from text with zero-shot learning," in *Int. Conf. on Research and Development in Information Retrieval*, pp. 195–204, 2019.

[41] F. Gargiulo *et al.*, "Deep neural network for hierarchical extreme multi-label text classification," *Applied Soft Computing*, vol. 79, pp. 125–138, 2019.

[42] A. Surkova, S. Skorynin, and I. Chernobaev, "Word embedding and cognitive linguistic models in text classification tasks," in *Int. Scientific Conf. Communicative Strategies of the Inf. Society*, p. 6, 2019.

[43] D. Zhu and K. W. Wong, "An evaluation study on text categorization using automatically generated labeled dataset," *Neurocomputing*, vol. 249, pp. 321–336, 2017.

[44] F. Abel *et al.*, "Recsys challenge 2017: Offline and online evaluation," in *RecSys Challenge*, pp. 372–373, 2017.

[45] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach.* Prentice Hall, third ed., 2010.

[46] Y. LeCun *et al.*, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.