applied sciences

MDPI

Systematic Review

Short-Text Semantic Similarity (STSS): Techniques, Challenges and Future Perspectives

Zaira Hassan Amur [1,*], Yew Kwang Hooi [1], Hina Bhanbhro [1], Kamran Dahri [2] and Gul Muhammad Soomro [3]

[1] Department of Computer and Information Sciences, Universiti Teknologi PETRONAS, Seri Iskandar 32160, Malaysia
[2] Department of Information Technology, University of Sindh, Jamshoro 71000, Pakistan
[3] Department of Artificial Intelligence, Tomas Bata University, 760 01 Zlín, Czech Republic
* Correspondence: zaira_20001009@utp.edu.my; Tel.: +60-148-142-485

Abstract: In natural language processing, short-text semantic similarity (STSS) is a very prominent field. It has a significant impact on a broad range of applications, such as question–answering systems, information retrieval, entity recognition, text analytics, sentiment classification, and so on. Despite their widespread use, many traditional machine learning techniques are incapable of identifying the semantics of short text. Traditional methods are based on ontologies, knowledge graphs, and corpus-based methods. The performance of these methods is influenced by the manually defined rules. Applying such measures is still difficult, since it poses various semantic challenges. In the existing literature, the most recent advances in short-text semantic similarity (STSS) research are not included. This study presents the systematic literature review (SLR) with the aim to (i) explain short sentence barriers in semantic similarity, (ii) identify the most appropriate standard deep learning techniques for the semantics of a short text, (iii) classify the language models that produce high-level contextual semantic information, (iv) determine appropriate datasets that are only intended for short text, and (v) highlight research challenges and proposed future improvements. To the best of our knowledge, we have provided an in-depth, comprehensive, and systematic review of short text semantic similarity trends, which will assist the researchers to reuse and enhance the semantic information.

Keywords: short text; semantic similarity; natural language processing; deep learning; STSS

Citation: Amur, Z.H.; Kwang Hooi, Y.; Bhanbhro, H.; Dahri, K.; Soomro, G.M. Short-Text Semantic Similarity (STSS): Techniques, Challenges and Future Perspectives. *Appl. Sci.* **2023**, *13*, 3911. https://doi.org/10.3390/app13063911

Academic Editor: Valentino Santucci

Received: 17 January 2023
Revised: 30 January 2023
Accepted: 3 February 2023
Published: 19 March 2023

1. Introduction

In natural language processing, short text semantic similarity is considered the most essential technique. It has been used in a variety of applications such as network analysis, search snippets, question–answering systems (QAS), information retrieval, text categorization, and so on. Earlier, the methods calculated the similarity between longer sentences in a very high dimensional space; thus, the method used has been inefficient and cannot be effective for many NLP tasks [1]. However, the short text reveals information that is important to understand. The short text presents numerous distinct challenges for natural language processing in contrast to longer sentences. Shorter sentences have a larger likelihood of being difficult to understand [2]. These sentences contain many common buzzwords that create noise in the data. However, short text frequently encompasses polysemous and synonymous terms, one word can have several distinct meanings, and it is even possible that two or more than two words may have the same meaning or concept [3,4]. These various aspects make it more complicated for machine learning to retained such information.

In short text semantic similarity, a significant challenge is to identify the semantics of sentences based on the context [5,6]. The context can be developed from many responses. These responses can be student answers, search queries, news headlines, tweets, and user feedback. However, machine learning algorithms still face difficulties in comprehending the meaning of words from text corpora due to the number of drawbacks of short sentences.

1.1. Barriers and Drawbacks of Short Sentences in Semantic Similarity

Figure 1 presents the numerous barriers and drawbacks of short sentences, which degrade the performance of machine learning algorithms.

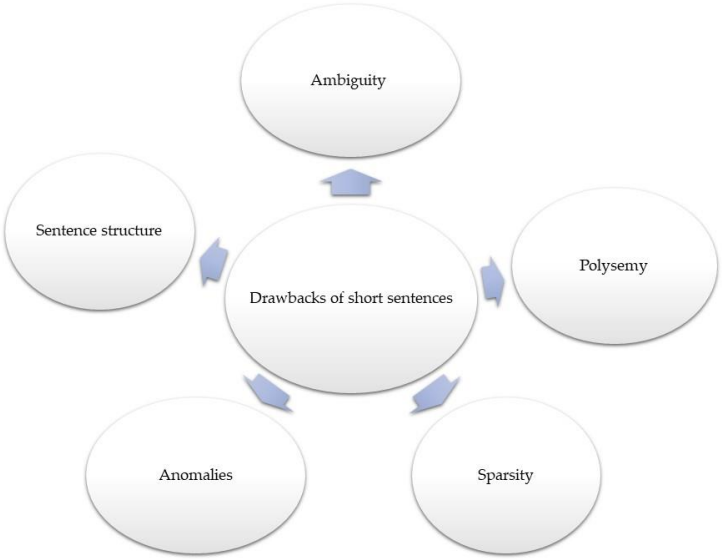

Figure 1. Drawbacks of short sentences.

Ambiguity: Conventional methods, which include parsing and parts-of-speech labeling, cannot be used for short text, because it does not adhere to appropriate syntax in any natural language. Short texts have a very limited amount of information, and most responses in short sentences are five words; however, search queries and tweets are limited to 140 characters [6–9]. Due to this, the short texts lack adequate statistical signals to support text processing. These problems lead the short text toward sentence ambiguity; in addition, multiple new approaches are required to solve these issues.

Polysemy and Sparsity: Social networks, search engines, blogs, and various other platforms produce a huge amount of short text, which is often known as sparse data. The data is presented in the form of synonyms and polysemous terms. Polysemy refers to a term or phrase with several meanings. However, the word polysemy is often related to linguistics, which presents sophisticated issues and vague concepts in short text messages and snippets. In natural language processing, word sense disambiguation (WSD) is usually used to solve the problem of synonymy and polysemy [10]. Moreover, the lexical meaning of word representations in natural language processing and philosophical contexts is difficult to handle for machine learning algorithms.

Anomalies: Earlier projects failed to ensure maximum correctness in the domain of short text, due to the fact that a lot of content is partially available. Short text produces a certain degree of anomalies, such as abbreviations, non-standard terms, misspellings, improper punctuations, grammatical errors, and slang words [11]. Machine learning needs an appropriate strategy to solve these irregularities in a short text.

Sentence Structure: In the short text, there are two components that structure the sentences, including (i) punctuation and (ii) word order. Inappropriate punctuation makes the sentences more complicated to understand, whereas every word includes different meanings that influence the contextual meaning of the text. There are other factors that affect the short text, including sentence fragments and run-on sentences.

Run-on sentences: These sentences use incorrect exclamation marks to form the sections of sentences.

Sentence fragments: Sentence fragments, which lack the necessary details to complete the sentence (one simple example is a predicate) and many which do not include the main verb, including only noun phrases, cannot complete the sentences [10,11]. However, in

short sentences, grammar is not the only factor that affects the structure. Style and rhythm are also key factors that make short sentences more challenging for machine learning algorithms. Moreover, machine learning algorithms use preprocessing pipelines on more sophisticated sentences. Under natural language processing, preprocessing techniques (Figure 2) help to identify the actual text from unstructured raw text.

Figure 2. Basic techniques of preprocessing pipeline.

1.2. Natural Language Preprocessing Techniques in Short Text

Preprocessing is a process in which machines transform raw sentences into meaningful organized sentences and weighted data prior to carrying out any task. Preprocessing involves multiple operations to organize the text into a meaningful structure. Here, we discuss a few of its operations, as depicted in Figure 2.

1.2.1. Tokenization

Tokenization separates text into meaningful pieces of information, which are often known as tokens. There are two types of tokenization in text: (i) word tokenization and (ii) sentence tokenization. To interpret each word's underlying meaning, different words are separated from the raw text using word tokenization. On the other hand, sentence tokenization separates specific sentences from the source text [12]. Apart from sentences and words, punctuation and special characters are also considered tokens that can be extracted from paragraphs and documents. There are various other tokens usually treated as combined tokens that are used in abbreviations and acronyms such as BA, BS, M. Phil., and Ph.D. These tokens are considered the same as punctuation and special characters such as @, #, $, €, etc. There are many words that are used as combined words in sentences, such as Barack Obama, check-in, long-term, and mind-blowing. These words are treated as a single entity in the normal space vector [12]. However, in order to check the semantics, these words can be tokenized separately. Consequently, for that, machines use various algorithms to identify the type of tokenization. Some of the well-known techniques used for tokenization are the wordpiece tokenizer, the sentence piece, and the byte-pair-encoding tokenizer. An example of tokenization is shown in the example below.

Example:
Text: Islamabad is the capital of Pakistan.
Tokenized Text: ['Islamabad', 'is', 'the', 'capital', 'of', 'Pakistan'].

1.2.2. Lemmatization

Lemmatization breaks words into their basic components and examines the words from a morphological viewpoint. The word lemmatizer maintains the original state of the word. In lemmatization, only the inflectional end of words is eliminated, which often returns to their base form. For example, the word "saw", can be either a noun or a verb [7–12]. Based on the parts of speech (POS), the lemmatizer can separate both words. It can convert the saw to its basic form "see" and saw by itself. Consider the following example of lemmatization:

Example:
Text: Ali saw a carpenter, using a saw.
Lemmatized text: Ali sees a carpenter, using a saw.

1.2.3. Stop Word Removal

Natural language processing filters some words from the sentences, which are known as stop words. Examples of stop words include the, a, at, so, an, and so on. These are the most frequently used terms in the language [7–12]. Removing such words from sentences reduces the low-level information, which makes the sentences easy to understand. Moreover, removing the stop words doesn't affect the contextual meaning of sentences, as machines preserve the semantic information of words. The following is an example that illustrates an explanation of stop word removal.

Example:

Text: A computer is an electronic machine.

Stop word removal: ['computer', 'electronic', 'machine'].

2. Research Methodology

In this research study, we have conducted a systematic literature review (SLR) with the purpose of analyzing the challenges, limitations, and recent trends in the domain of STSS in a targeted way. Furthermore, we have embraced the philosophy of SLR from references [13–15] to successfully complete the depth of knowledge for this study. However, the study is further broken down into three phases. Phase (i): describes the planning of this study with the purpose to identify the research questions, describe the rationale of the work, and verify and validate the review of the protocol. Phase (ii): outlines how the information was combined, which studies were chosen, how the data were extracted, and how the inclusion criteria were determined. Phase (iii): covers the writing and validation of a systematic literature review. Moreover, the phases of the systematic review process are also illustrated in Figure 3.

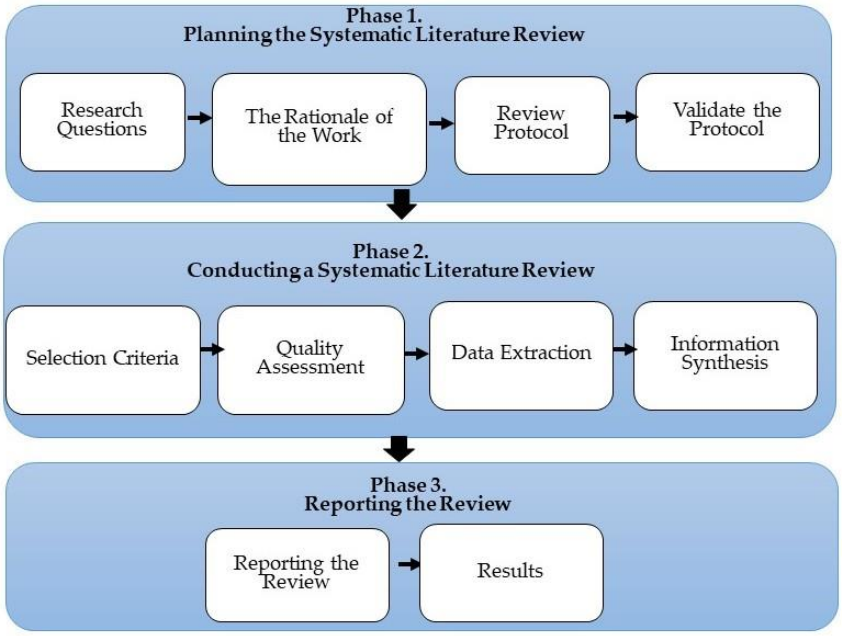

Figure 3. Phases in the systematic literature review (SLR).

2.1. Phase 1: Planning the Systematic Literature Review (SLR)

Phase 1 of this study has been broken into four sections. The research questions are stated in Stage 1. Stage 2 provides justification for the work. Stage 3 evaluates the protocol, and Stage 4 confirms the report.

2.1.1. Research Questions

We have designed the research questions for this study in Table 1. In response to the present demand for a comprehensive review of the literature in the area of short-text semantic similarity.

Table 1. Research questions and objectives of this research study.

Research Question (RQ)	Research Objective (RO)
RQ1. What are the existing deep learning techniques used in short-text semantic similarity (STSS)?	RO1. The objective of this study is to summarize current deep learning techniques used in short-text semantic similarity.
RQ2. Which existing deep learning techniques are most appropriate for generating high-level contextual representations?	RO2. This research question aims to uncover the most appropriate existing studies that extract key features from short sentences based on contextual information.
RQ3. What are the available datasets used for the short-text semantic similarity?	RO3. The purpose of this research question is to determine the available datasets for STSS. However, we are interested in discovering such datasets that support short sentences.
RQ4. What are the current challenges and suggested improvements for short question–answering systems (QAS)?	RO4. The aim of this research question is to discuss challenges and provide future recommendations for short question–answering systems.

2.1.2. Rationale of the Work

In artificial intelligence (AI), short text is an emerging field, which lacks sufficient standards to follow the proper syntax of the language. Short text (ST) contains a limited number of words, and various words affect the intent of other words, which causes inaccurate similarity in return. Words are the main parameters of any language; without words, language would not be a concept [12–14]. Understanding the formation, structure, and semantics of sentences through machine-oriented methods are a challenge for many researchers. In the absence of comprehensive analysis, this study extracts and provides a detailed systematic literature review (SLR) of existing studies. Moreover, this study helps researchers to choose appropriate solutions and propose new systems for complex queries. The techniques provided by this study can further be reused for question–answering systems, short text classification, information retrieval, and keyword analysis.

2.1.3. Review Protocol

This step of the systematic literature review (SLR) is the most critical, since it specifies and evaluates the actions before the conducting phase [14]. When exploring various sources for articles, the review process supports the creation and authentication of pertinent keywords. Moreover, the review protocol is refined throughout the entire selection of studies. Furthermore, to choose the appropriate keywords and key phrases, we followed the guidelines of [13,14], and a number of sources were searched using Boolean operators. The following are the ways by which we have extracted a number of papers from different repositories:

- When searching for journal articles, we used the "AND operator" and "OR operator" to combine keywords.
- Extracted keywords and key phrases from questions.
- Used different terms related to the targeted topic.
- Included various keywords from the number of publications from different repositories.

We have added keywords that describe the specific methods and techniques, which are shown in Table 2, which presents key terms and keywords used to select various publications.

Table 2. Keywords, key terms, and key phrases were used for this study.

No.	Keywords, Key Terms, and Key Phrases
1.	("Short Text" OR "ST") AND ("STSS" OR QAS)
2.	("Short Questions" OR "Short Answers") AND ("Short Text Semantic" OR "Semantics in Sentences")
3.	("DL" OR "STSS" OR "QA") AND ("Sparsity" OR "Ambiguity" OR" Polysemy")
4.	("STSS similarity" OR "STSS semantics") AND ("Keywords" OR "Short Sentences" OR "Similarity")
5.	("Similarity Barriers" OR "Barriers in Sentences") AND ("Sentence Ambiguity" OR "Word order")
6.	("ASAG" OR "short answer grading") AND ("Answer similarity" OR "Question similarity") AND ("QAS" OR "ASAQS" OR "STSS")
7.	("Short text classification" AND "Text classification") AND ("Short Sentence length" OR "Sentence Length")
8.	("Context-Embeddings" Or "Sense information") AND ("Context independent" Or Context-Dependent)
9.	("Short Answer Dataset" OR "QAS dataset") AND ("Multi-Lingual QAS" OR "Domain-independent QAS dataset")
10.	("QAS dataset" OR "Short sentence Dataset") AND ("ASAGs dataset" OR "Scoring Rubric dataset")

2.1.4. Validate Review Protocol

In accordance with Kitchenham et al. [15], we have included the following questions to validate and evaluate the protocol:

- Are the keywords and key phrases derived correctly for search strings?
- Does the extracted data address all questions?
- Have the exclusion and inclusion criteria been applied correctly?

In order to do this, the protocol was created after the changes and revisions made by researchers.

2.2. Phase 2: Conducting a Systematic Literature Review

This phase is divided into four stages. Stage 1 is the selection of appropriate studies from various repositories. Stage 2 assesses the study with quality parameters, Stage 3 is the extraction stage for every question included in this study, and Stage 4 is synthesis of the information.

2.2.1. Selection of Studies

Many repositories have been used for selecting the research publications in Phase 2, such as ACM Digital Library, ProQuest, Google Scholar, IEEE explorer, Web of Science, Science Direct, Scopus, and Springer, as are presented in Figure 4. We selected highly indexed research papers from these databases.

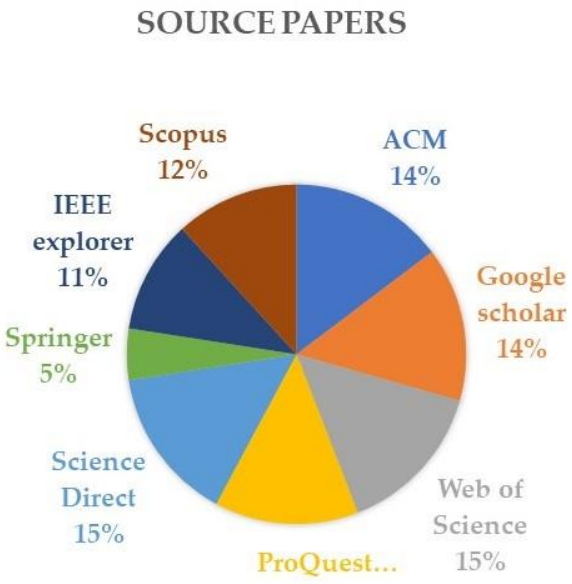

Figure 4. The number of research papers from various repositories.

The selection criteria were further divided with inclusion and exclusion criteria presented in Tables 3 and 4 and a final sample of candidate articles was determined.

Table 3. Inclusion criteria of candidate articles.

Inclusion Criteria for Short Text Semantic Similarity
Studies conducted between 2017–2023
The research was relevant to short-question–answering systems (QAS), short answers, short sentences, short text classification
The selection criteria were only focused to search strings "short text", "short answers", "QAS", and "semantic similarity"
The research used deep learning for short-text semantic similarity (STSS).
The research included journal articles/conference papers and peer-reviewed articles.
For the duplication of the same studies, the most recent and most completed studies were selected.

Table 4. Exclusion criteria of candidate articles.

Exclusion Criteria for Short Text Semantic Similarity
Articles performing semantic similarity and not short text
Unclear and unfocused articles
Articles that focused only on short questions
Articles that used similarity techniques for long paragraphs and essays
Studies that did not meet the objectives

2.2.2. Quality Assessment

According to [14], the quality assessment of research papers is challenging because it involves various factors to measure the effectiveness of different articles. The articles require additional criteria to follow the systematic literature review (SLR). This paper was related to the following guidelines:

- Documentation of Data: The number of surveys, methodologies, and results cited in this paper (e.g., question–answering systems, datasets, data mining, data analytics, and so on).
- Accessibility of Information Sources: Collected data includes various DOI, URLs, databases, and different organizations.
- Description of Methodology: A thorough methodology was used, and the basic axioms and guidelines listed in several research were followed step by step.
- Results: Comprehensive graphics and tables were used to present the results.

2.2.3. Data Extraction and Information Synthesis

The researchers extracted the data from a number of repositories that addressed the research questions and contributions of the current study. For each inserted article, knowledge was created by utilizing and synthesizing the practical data aspects. However, additional information characteristics were added to further encode the data. Moreover, we used narrative synthesis to address our research questions. Table 5 presents the gathered data for each included question.

Table 5. Data Extraction for Each Question.

Short-Text Semantic Similarity
RQ1: Deep learning techniques in short-text semantic similarity.
RQ2: Keyword extraction techniques in STSS
RQ3: Available tools and dataset for short-text semantic similarity
RQ4: Limitations and suggested improvements in deep learning methods

Furthermore, PRISMA guidelines motivated by the studies [13–15] were followed to filter the most appropriate studies. These guidelines help the researchers to conduct the

review in an organized manner. Figure 5 depicts the entire selection of studies divided into five phases. Phase 01 yielded records in a total of 1384; we removed 37 duplicate records before the screening. In phase 02, 1347 records were screened, and 1200 records were excluded. However, phase 03 retrieved 147 records, whereas the number of non-retrieval records was 11. Furthermore, phase 04 sorted records for abstract contemplation, and the number of records was 136; however, we eliminated irrelevant records that did not meet our review criteria, and the number of those records was 34. In phase 05, we included the final sample of 102 studies to complete this work.

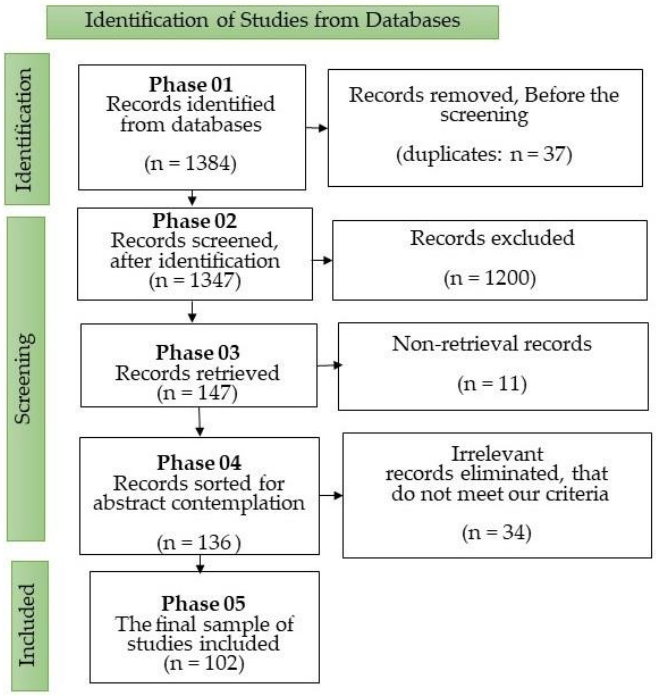

Figure 5. PRISMA guidelines for the selection of systematic literature review.

2.3. Phase 3. Reporting the Review

The reporting phase is the final phase, in which we addressed the research questions from original sources. Moreover, the systematic literature review (SLR) depends on the presentation of the results. Hence, the research paper was well-written, well-structured, and well-documented.

Results

The results were gathered and computed based on research questions. A total of 102 studies, as mentioned in Figure 5, were included to complete the review. Moreover, Table 6 presents the number of studies per research question. A total of 42 research articles explain the deep learning (DL) techniques that are highly recommended for short-text semantic similarity. However, we discovered 76 journal articles, including conference papers, that present a number of methods that extract the semantics of keywords based on the context of short sentences. Moreover, 34 studies introduce different datasets that are currently available for short-text semantic similarity. Finally, we discovered challenges and suggested improvements from 25 studies.

Furthermore, Figure 6 depicts the number of studies from the year 2017 to 2023. We found only four studies which were published in the year 2017. The majority of the studies were related to short sentences and short queries. However, the number of research articles increased from the year 2018 to 2022. In the year 2018, the majority of studies used deep learning models for STSS, and 12 papers were extracted from the duplicate's records; however, in 2019, 18 papers were published in recognized journals to extract the similarity

from short snippets. Furthermore, 21 research articles were mentioned in the year of 2020, 25 and 16 articles for the year of 2021 and 2022, respectively, and 4 for 2023.

Table 6. Number of studies per research question.

Research Questions (RQ)	Number of Studies
DL-based techniques in short-text semantic similarity (STSS)	42
Contextual based keyword selection approaches	76
Datasets for short text similarity	34
Challenges or limitations of different methods in STSS	25

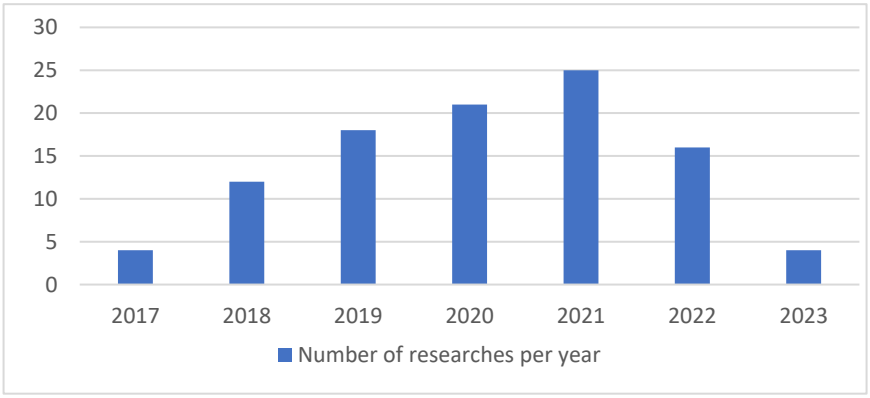

Figure 6. Number of studies from the year 2017 to 2023.

3. Contribution to Work

Previous studies on semantic analysis, computational linguistics, natural language processing, and different statistical methods contribute to understanding various challenges of short-text semantic similarity (STSS). The contribution of our work is methodological. The following are the key contributions of our study:

- We focused on various deep learning techniques that are compact, dense, and appropriate for analyzing the semantic and lexical meaning of short texts.
- We compiled research that mostly deals with short text, short text classification, short question-answers, short tweets, and short sentences.
- This study also included three essential methods for emphasizing the context that underlie the semantics of short text.
- This study also concentrated on performance assessments, short text datasets, and suggested improvements for future initiatives.

The rest of the paper is organized as follows: In order to address the first RQ, Section 4 provides information on several deep learning approaches and their impacts on the semantic similarity of short text. Section 5 explains those deep learning techniques that generate high-level textual representations in order to improve sentence similarity. Furthermore, the datasets for short text similarity tasks are presented in Section 6. Moreover, Section 7 elaborates on the STSS challenges in QAS systems and presents suggested improvements. Section 8 has been written with concluding remarks.

4. What Are the Existing Deep Learning Techniques Used in Short-Text Semantic Similarity (STSS)?

We have included a total number of 42 research articles that explain the methods and techniques used in short-text semantic similarity.

4.1. Convolutional Neural Network (CNN)

In order to answer this research question, approx. 42 research studies were reviewed to find various deep learning techniques that are used in short-text semantic similarity

(STSS) and question–answering systems (QAS). Deep learning is becoming more and more prevalent in the field of natural language processing (NLP). Several models of attention-based neural networks sparked the interest of many academics, such as the convolutional neural network (CNN), the recurrent neural network (RNN), and the bi-directional encoder representation from the transformer (BERT) [1]. Table 7 presents a summary of the convolutional network (CNN) model in various categories of short text, to which Wang et al. [1] contributed their research in the field of short-text semantic similarity (STSS). They used a convolutional neural network (CNN) for short text classification to find the related words, and, further, they used external knowledge to understand the conceptual meaning of answers and words from a short text. However, they also used Jaro–Winkler similarity to find the grammatical errors in sentences. Shih et al. [16], developed short answer grading systems by using a convolutional neural network (CNN). The available model worked as a text classifier to understand the Chinese language from students' answers. However, they also used binary classification to grade the answers as correct or incorrect. Moreover, Xu et al. [17] presented the dual embedding convolutional neural network (DE-CNN) to understand the underlying meaning of the short answer. They used two embedding layers to understand the relevant context of sentence features and also utilized an attention embedding layer to get the concept representations from answers. Perera et al. [18], proposed the CNN model to identify irrelevant answers from web-based question–answering systems (QAS). The length of answers was short, and the category of questions was a factoid. Moreover, the model proposed by that research could not answer the whole set of factoid questions. Surya et al. [19] developed a character-level convolutional neural network (CNN) to understand short answers. Without any prior understanding of language and semantic structures, the model learned to anticipate the target information from answers. Additionally, the study ran into a number of difficulties when trying to score short answers, due to a lack of tools and generic strategies. In order to understand the short text semantically, Wang et al. [20] developed a brand new semantic hierarchical paradigm for categorizing short texts. However, they further detected multi-scale SUs in short texts and introduced new information using pre-trained word-level embeddings. Furthermore, the emerging trends of big data provide a massive amount of short text. Earlier, the data was accumulated manually from power sources. However, Liu et al. [21] proposed a convolutional neural network (CNN) and LDA to mine and understand the global features from the short text automatically. Moreover, many traditional algorithms of machine learning affect the generalization ability of short text. Due to this fact, Hu et al. [22] presented a CNN with a support vector machine convolutional neural network (SVMCNN) to improve short text classification. Meanwhile, the researchers trained their model on TensorFlow by utilizing the Twitter social platform.

Table 7. Summary of convolutional neural network (CNN)-based studies in STSS.

Ref.	Model	Purpose	Category	Feature Set	Experimentation
Wang et al. (2021) [1]	Convolutional neural network (CNN) and semantic extension (SECNN)	To capture the sentence-level and word-level similarity	Short text classification	Twitter, AG news	Python 3.7, Intel(R)i7-7700, 3.60 GHz processor 16 GB memory
Shih et al. (2019) [16]	Convolutional neural network (CNN) and Word2Vec	To grade the student's short answers and questions	Short answer questions (Chines Language)	SCiEntBank, Chines Wikipedia	Ubuntu 18, Python 3.6, PyTorch 10.0, NumPy 1.12.1, genism 2.0, Intel(R)i7-6700
Xu et al. (2020) [17]	Dual embeddings convolutional neural networks DE-CNN	To understand the conceptual information from short text	Short text classification	Knowledge base Probase, Movie Review. AG corpus of news	Python 3.6, PyTorch10.0

Table 7. *Cont.*

Ref.	Model	Purpose	Category	Feature Set	Experimentation
Perera et al. (2020) [18]	Convolutional neural network (CNN)	To identify irrelevant answers from a web-based community	Short answers	Web-based forum (Bing Query longs, WikiQA)	Scikit learn library Regression R-CNN
Surya et al. (2019) [19]	Character-level convolutional neural network (CNN)	The model learns to anticipate the target information for short answers.	Short answers	Short answer scoring by Hewlett foundation	NVIDIA 940mx GPU Max pooling, min pooling
Wang et al. (2017) [20]	Semantic clustering and convolutional neural network (CNN)	Developed a brand new semantic hierarchical paradigm for categorizing short texts.	Short text categorization	Google snippets	K-max pooling SVM parser
Liu et al. (2022) [21]	Convolutional neural networks (CNN)	To mine and understand the global features from the short text automatically.	Short text classification	1000 defect text data from a power company in a northwestern country	LDAvis toolkit Confusion matrix Python 3.7
Hu et al. (2018) [22]	CNN+SVM	To understand short text information	Short text classification	Twitter	TensorFlow

4.2. Recurrent Neural Network (RNN)

With the emergence of short text semantic similarity, the recurrent neural network (RNN), along with various other models, give tremendous performance on STSS. Agarwal et al. [23] proposed the recurrent neural network (RNN) with the convolutional neural network (CNN). The combined approach performed the semantic matching between various words; however, the model further used semantic representations between sentences to find the similarity. The study further achieved an F1 score of 0.751, and the precision rate was 0.760. The most common limitation that occurs while utilizing the recurrent neural network for short text semantic similarity is the gradient vanishing problem. In order to overcome the problem, Yao et al. [24] presented the model known as long short-term memory (LSTM), which is often known as the variant of RNN. The model employed cosine similarity to calculate the distance and similarity between two short texts and also used backward propagation after the normalization process. Moreover, Dwivedi et al. [25] applied the RNN and CNN to the semantic features of the gender classification, and they included short text. In order to do this, they combined the semantic features for the classification process. Li et al. [26] presented multiscale CNN–RNN to represent the short text. The model was able to produce the word as level as character-level representations. Moreover, Edo-Osagie et al. [27] proposed the gated recurrent neural network (ABRNN) that automatically filters short tweets. The model extracted only those tweets that were relevant to a syndrome, such as asthma/difficulty breathing. Furthermore, Hassan et al. [28] presented various methods and limitations for short text similarity. They researched that recurrent neural networks (RNN) can utilize Tf-IDf vectors to better understand the similarity among short sentences. Lee et al. [29] generated vector representations of short text through a recurrent neural network, and, later, they evaluated the model on a classification dataset known as DSTC, which is abbreviated from the dialogue state tracking challenge. Table 8 further illustrates the summary of RNN-based studies.

Table 8. Summary of recurrent neural networks (RNN).

Ref.	Model	Category	Feature Set
Agarwal et al. (2018) [23]	RNN	Short text	Semantic matching
Yao et al. (2018) [24]	RNN & LSTM	Short text	SIM
Dwivedi (2017) [25]	RNN & CNN	Short text	Gender classification SIM
Li et al. (2019) [26]	RNN & CNN	Short text	Character level representations
Edo-Osagie et al. (2019) [27]	ABRNN	Short tweet	Asthma syndrome SIM
Hassan Amur et al. (2022) [28]	RNN	Short text	SIM
Lee et al. (2017) [29]	RNN	Short text	Vector representations

4.3. Transformer Learning Models

Currently, in deep learning techniques, the bi-directional encoder representation from the transformer (BERT) model displayed excellent achievement on many natural language processing (NLP) tasks [30]. In order to do this, Mozafari et al. [31] proposed the BAS (BERT-answer selection model) to understand the syntactic and semantic information from short question answers. The study further used CNN, RNN, and BOW as classifiers to detect the correct answers. Wijaya et al. [32] proposed a BERT model to automatically grade short answers. The model was trained by using the Indonesian language. However, they use Cohen's Kappa to check the inter-rater reliability among student answers. Moreover, Luo et al. also [33] also proposed the BERT model to grade the student's short answers. He further trained the model on a short answer scoring V2.0 dataset and used the regression task function to check the linearity between answers. Furthermore, Alammary [34] presented the BERT model for the Arabic short text classification. The study synthesized the different Arabic BERT versions for text classification. Moreover, the study provides a comparison between ENGLISH and ARABIC text classification models. Haider et al. [35] used domain-independent subjects, such as biology and geography, to grade the Indonesian short answers. Furthermore, they implemented the BERT model to detect the word embeddings from sentences and analyzed the contextual information. In relation to this, Gaddipati et al. [36] mentioned the difference between transformer learning models BERT, GPT, GPT2, and ELMO. BERT uses a transformer mechanism and takes the benefits of both ELMO and GPT to extract the contextual embeddings bi-directionally. It is trained on BookCorpus and Wikipedia datasets; the size of the dataset is 800 M and 2500 M words. Moreover, GPT and GPT2 also use a stacked transformer. GPT comprises the 800 M dataset word size and is pre-trained on BookCorpus. In contrast, GPT2 is pre-trained on WebText and utilizes the stacked transformer. In order to compare ELMO to these three transformers (BERT, GPT, and GPT2), ELMO uses the bi-LSTM architecture and a benchmark dataset with a word count of 1 billion in size. Furthermore, Garj et al. [37] proposed a BERT regressor model to grade short answers. The model further used domain-specific datasets forkey-valuee pairs. Since automatic short answer grading (ASAG) gained a lot of attention in recent years, a number of methods have been proposed in this domain. In order to do this, Zhu et al. [38] also proposed a BERT-based framework to grade short answers. The method further used CNN, capsule, and triple hot loss strategy to encode the short key sentences. Moreover, Burrow et al. [39] researched 35 ASAG systems from 1996 to 2015 and classified them into five categories. The authors further presented various limitations inside the ASAG systems. However, Mohler et al. [40] used lexical semantic similarity to grade short answers. They further explained that deep learning techniques are more

useful to grade short answers. Wang et al. [41] proposed the ml-BERT model to grade short answers. They targeted unlabeled data on the domain-specific dataset. They further used meta-learning to initialize the model parameters. Sung et al. [42] improved the BERT model in the ASAGs domain. They used fine-tuning techniques to use multi-domain resources. Khodeir et al. [43] used a combined approach and added a multilayer recurrent unit, along with the BERT model, to develop the classification model. Moreover, Camus et al. [44] compared the BERT model with ALBERT and Roberta and used the pretraining method to grade short answers. They also used optimization techniques to improve the BERT model performance. Various variants of BERT, such as SBERT, DistilBERT, KeyBERT, and Roberta, actively work on short-text semantic similarity tasks, which include, short text classification, short keyword extraction, sequence classification, semantic matching, and so on [45–50]. Ye et al. [51] used the BERT model for the context-sensitive representations. They further utilized the GCN model for the classification of short text. Hu et al. [52] surveyed transformer models for short text similarity, and they dictated that the BERT model has the ability to understand the contextual meaning of words inside the sentences, and, based on context, can extract the similarity between two tasks. They further applied semantic matching to the short text. Moreover, Xiao et al. [53] utilized ELMO embeddings to understand the contextual embeddings from a short text. They further used the BERT model architecture to understand the structure of sentences on ATS and SNIPS datasets. Moreover, Wan et al. [54] utilized the ELMO model to understand the level of sentences through their context. In the study, the ELMO model adopted the architecture of the Bi-LSTM model for entire sentences and mapped the sentences into the sequence of vectors. Further, they highlighted various entities from the text that represented conceptual embeddings. In the domain of ASAGs, the short text suffers from task-specific architectures, and extracting the data from multiple domains was difficult to encode. The NLP community developed a number of pre-trained and finetuned models on domain-specific and domain-independent datasets. These models can be transformed and utilized for many NLP tasks. Recently, the BERT model outperformed other pre-trained algorithms. It can learn the context of any sentence, left-to-right or right-to-left, simultaneously [55]. Moreover, various other models, such as BioBERT for biomedical tasks and SciBERT for science domain applications, have also been introduced to understand domain-specific language and tasks [56]. The following Table 9 depicts the summary of transformer learning models that exploit the similarity in the domain of short text applications.

Table 9. Summary of transformer learning models.

Study	Architecture	Trained	Category	Features	Dataset	Size
Mozafari et al. (2019) [31]	Stacked transformer with attention mechanism	Fine-tuned (BERT)	Questions and answers	Short answers (SIM)	WikiQA	2351 QA
Wijaya et al. (2021) [32]	Stacked transformer	Pre-trained (BERT)	Questions and answers	Short answers (SIM)	QABank Indonesian language	100 QA
Luo et al. (2021) [33]	Stacked Transformer	Pre-trained BERT	Questions and answers	Short answers (SIM)	V2.0	2440 QA
Haidir et al. (2020) [35]	Stacked Transformer	Fine-tuned BERT	Questions and answers	Short answers (aontextual embeddings)	Mohler	7605
Garg et al. (2022) [37]	Stacked Transformer	Fine-tuned BERT regressor	Grading	Short answers (SIM)	Domain-specific	2273 pairs

Table 9. *Cont.*

Study	Architecture	Trained	Category	Features	Dataset	Size
Zhu et al. (2022) [38]	Stacked transformer and Bi-LSTM	Pre-trained BERT	Grading	Short answers (SIM)	Mohler SemEval 2013	2273 Pairs & 2100 QA
Wang et al. (2019) [41]	Stacked transformer	Pre-trained ml-BERT	Grading (meta-learning)	Short answers (SIM)	Biology textbook	-
Sung et al. (2019) [42]	Stacked transformer	Pre-trained BERT	Grading	Short answers (SIM)	Multi-domain	1.3 Million words
Camus et al. (2020) [44]	Stacked transformer	Fine-tuned BERT, Roberta	Grading	Short answers	SemEval 2013	2100 QA
Xiao et al. (2020) [53]	Bi-LSTM transformer BERT Transformer (Combined approach)	Fine-tuned ELMO	Short sentences	Contextual embeddings (SIM)	ATS and SNIPS	-

5. Which Existing Deep Learning Techniques Are Most Appropriate for Generating High-Level Contextual Representations from Sentences in Order to Improve the Similarity?

To answer this question, we have included approx. 76 research articles to review various techniques for understanding contextual information.

Numerous machine learning algorithms in deep learning modified the representations of word vectors. These representations include many NLP applications, such as question–answering systems (QAS), sentiment analysis, textual entailment, and named entity recognition [57,58]. Traditional methods, such as statistical and graph-based methods, extract the words from sentences based on their frequencies and co-occurrences. To complete this section, we focus on words in the context of sentences. Contextual embeddings are used to extract words and phrases, and a number of techniques, including BERT (Bidirectional Encoder Representations from Transformers), ELMO (Embeddings from Language Model), and GPT-2 (Generative pre-trained transformer-2) are frequently employed [59–62].

5.1. BERT Model for Contextual Word Embeddings (CWE)

Word embeddings are a form of keyword representation. Many studies used to refer to the keyword or key phrase, which helps to identify the meaning of content from a document or sentences. Keywords refer to a unigram, while key phrase is an N-gram, which is usually concatenated with one or two sub-words. For example, the computer is a keyword, and the computer system is a key phrase [61–63]. BERT has shown many promising results for capturing contextual embeddings from long or short sentences. The model employed transformer architecture and the attention mechanism to extract the embeddings. In order to do this, Kovaleva et al. [64] proposed a self-attention mechanism in the BERT model to select the linguistic features and conduct various experiments to identify how these features co-related with one another. Moreover, the study used the GLUE benchmark and SQuAd dataset for the feature selection task and improved the model by an absolute gain of 3.2%. Khan et al. [65] proposed the KeyBERT model for the impact analysis with RAKE, YAKE, and TF-IDF approaches. The model produced contextual word embeddings with the authors' provided keywords. The average similarity rate of the proposed model was 51%, which is higher than other baseline methods. Moreover, Tang et al. [66] proposed a multilayer attention BERT model to extract the contextual

embeddings of clinical data from the EHR system. They improved the model by employing an additional layer of the BiLSTM model. This resulted in an improved accuracy of 97.6%, which is higher than the fine-tuned BERT model. Furthermore, Lyu et al. [67] introduced a linguistic knowledge-enhanced graph transformer to extract the ambiguity from short chine sentences. They further proposed the BERT model, along with the graph transformer model, to extract Chinese characters. However, they conducted the experiments on the Chinese dataset BQ and LCQMC; the model showed 88.38% accuracy. Eke et al. [68] proposed a BERT model with feature fusion techniques to identify the sarcastic contextual-based features. The features were found from e-commerce and social media sites, and, for experimentation, they used the Twitter benchmark and achieved 98.0% precision. With an account of this, Wiedemann et al. [69] used a pre-trained BERT model to identify the polysemy words that provide sense embeddings in their contextual space. The KNN and POS were further utilized to capture the grammatical structure of sentences. They used the SenseEval dataset, and the F1 results of the model showed a value of 83.32%. Zhou et al. [70] used the pre-trained BERT model for conversational topic classification. The model captured the salient features from the representations. For experimentation, the authors used a five-class conversational corpus. The model achieved a 91.5% score in precision, recall, and F1 measures. Many studies used the BERT pre-trained model. The model uses a vast amount of unlabeled data for the general domain. However, multiple types of domains can be utilized for pre-trained models. The more diverse benchmark creates higher complexity for Ml algorithms. Thus, for every new specific domain, pre-trained versions are worth utilizing [71–74]. Zhang et al. [75] proposed the SEMBERT model to understand the contextual semantics. The model further used reading comprehension and language inference tasks. The GLUE benchmark was utilized for experimentation. The model showed an 83.6% score in the F1 measure and presented an accuracy of 91.42%. Moreover, the BERT model used input embeddings; these embeddings are the combinations of three other embeddings, such as:

Token Embeddings: These embeddings use word-piece vocabulary to split the sentences into words.

Sentence Embeddings: The BERT model uses next-sentence prediction (NSP), which determines the next sentence and predicts if the words belong to sentence A or sentence B [76].

Position Embeddings: These embeddings encode and identify the position of words inside the given sentence.

In the base form, the BERT model includes 12 layers, which are often known as transformer encoders. These encoders are further composed of two layers, the multi-head layer and the self-attention layer, which divide the sentence to learn contextual embeddings. These embeddings create subwords to arrange the sentence in a more retained form. However, the model uses 12 attention heads and presents 110 M trainable parameters to understand the embeddings.

5.2. ELMO Model for Contextual Word Embeddings (CWE)

The ELMO (embeddings from language model) was introduced by Peter et al. [77] for the purpose of extracting contextual and morphological representations. The embeddings of each word can change based on their syntactical and contextual structures. Gupta et al. [78] proposed the ELMO model for text summarizers. The model further represents the documents into a vector that includes syntax and the contextual dependent information of words. The model further used Kaggle datasets, and experimentation was completed by using Python programming. Liu et al. [79] used the ELMO model to gather information from patients with schizophrenia. The authors used small statements, and, later, they extracted the features which included predicate information of patients. The model achieved 80% accuracy by using cross-validation. Rezaii et al. [80] reported that it is difficult to extract the contextual information through the ELMO model for those words that contain very little information, such as 'such as, somehow, some, well, etc'.

Naseem et al. [81] use the LSTM as an input to name entity recognition with the ELMO model. They used a pre-trained version of the ELMO model on the corpus of medicine. Their results showed that the ELMO model is good for domain-specific tasks.

Furthermore, the studies [82–86] suggest using convolutional layers with the ELMO model to improve its performance for extracting text features. These layers used max-pooling to represent the fixed length of the entire word. Similar to the convolutional layers, the ELMO model can utilize the layers of Fasttext to improve its performance in character-level representations. To improve the process for the input layers, these representations employ a two-layer network [87]. Additionally, the Bi-LSTM network layers are used by the ELMO language model to understand the previous word. In relation to this, Al-Bataineh et al. [88] used the ELMO model for ARABIC short Q2Q similarities. The representation of the model detects the high contextual embeddings for similarity. The model used character-level embeddings to overcome the morphological nature of answers. The model scored 71.30% for the Pearson correlation coefficient. The ELMO model uses a diverse approach to predict model similarity. The following are the three categories that the model utilizes for the contextual word and sentence representations [88–90]:

Word Embeddings: Inside these embeddings, tokens usually convert into embeddings, and words from sentences represent as vectors.

Sentence Representations: These embeddings are used to generate the Sentence2Vec and build on top of the word embeddings.

Prediction layer: The ELMO model uses this layer to predict the similarity.

Traditional methods, including Bag-of-words [91], Sentence2Vec [92], and Word2Vec [93] models, developed the dictionary for source text. The available dictionary includes many words that are used as a vocabulary for the text. These models organize the text and developed the vectors for each sentence to capture the similarity. Furthermore, these models use documents, long, and short sentences for similarity tasks and are often known as context-independent models. The context-dependent model, such as ELMO, uses characters to understand the hidden meaning. In order to do this, Laskar et al. [94] state that, to extract the context meaning of words, the pertained version of the ELMO model is the most suitable. It employs the token embeddings with position embeddings and sends them to the transformer encoder. These encoders follow the self-attention mechanism, which uses feed word and pooling layers. As a result, these layers generate condensed vector representations. However, Reimers et al. [95] dictated that the result of these layers fused to domain-specific neural architectures. This integration is not straightforward for the pre-trained ELMO model. A number of authors [96–98] suggested many simplified ways to use ELMO contextual embeddings with domain-specific tasks. Some researchers used the final ELMO layer to predict the score, while others fused vectors, and many other researchers assessed the average of all layers.

5.3. GPT-2 Model for Contextual Word Embeddings (CWE)

The generative pre-trained model-2 was developed by GoogleAI in February 2019 [99]. The model is recognized as one of the deep neural language models, which uses fine-tuning and pre-training for the majority of NLP downstream tasks. Contextualized word representations are internal word representations. These representations are used as a function for the whole input sentence. Similar to the BERT model, GPT-2 is a uni-directional as well as a bi-directional language-based transformer model. For each input sentence, the GPT-2 utilizes the 12 layers as part of the tokenization [100]. In comparison to the ELMO, the GPT-2 uses multi-encoder transformer layers to capture the features from sentences; however, the ELMO uses Bi-LSTM architecture to encode the sentences. Another key difference is that the ELMO utilizes an unsupervised approach for the selection of features from the text, while the GPT-2 uses a fine-tune approach for all end-tasks. The objective of the GPT-2 is quite similar to that of the BERT model: to predict the word from a set of all possible words based on context. The model is context-dependent, as is the case with other transformer models. Ethayarajh et al. [101] compared the performance of the BERT, ELMO, and GPT-2 models for the purpose of contextualized word embeddings.

They further dictated that the representations of the GPT-2 model in comparison with the BERT and ELMO are more context-specific. Furthermore, Han et al. [102] provided a brief history of pre-trained transformer models (PTMs). Models like the BERT, ELMO, and GPT-2 develop a rich context in terms of efficiency and predicting various interpretations. Schneider et al. [103] proposed the GPT-2 to identify a Portuguese biomedical text. They used a fine-tuning approach for transfer learning; however, they manually annotated the public dataset for classification tasks. Unlike the BERT model, the GPT-2 is based on stacked decoder blocks, whereas the BERT uses encoder blocks. The GPT-2 receives the input information in the form of word vectors and then finalizes the prediction probability for the next words. However, the BERT model is not autoregressive. It takes the entire context all at once. As for the attention mechanism, the BERT and ELMO use a self-attention mechanism for contextualization, but the GPT-2 utilizes the masked attention mechanism for next-word predictions [104–106].

5.4. Model Selection for Contextual Word Embeddings (CWE)

To achieve the second objective, we studied three models for contextual word embeddings (CWE). These models (BERT, ELMO, GPT-2) present the text features from long as well as short sentences. Traditional methods such as Glove, Word2Vec, FastText, count vector, TF-IDF, and co-occurrence matrix are context-independent [107]. Unlike these methods, transformer learning models generate similarity based on the context. As we have noted in Figure 7, the ELMO captures the context bidirectionally. The model is more suitable for domain-specific tasks. However, the GPT-2 is a domain-independent model, also referred to as a task-agnostic model. It identifies the context unidirectionally [108]. The BERT predicts the context bidirectionally; it requires additional changes (addition of different model architectures such as LSTM, CNN, semantic fusion, etc.) for the improvement of many NLP tasks [109]. The BERT model has the ability to represent the word tokens in both directions. When it comes to supervised learning, the BERT is quite similar to the GPT-2. First, the added output layer of the BERT model uses the representations with additional changes to predict the words or sequence of words. Second, the model fine-tunes the parameters for specific tasks. For the newly customized dataset, the model needs to train the parameters from scratch [110]. Moreover, the BERT model achieved tremendous performance on the majority of NLP tasks, such as question–answering systems (QAS), short-text similarity (STS), single-text classification, text tagging, and so on. Moreover, Table 10 illustrates the summary of all three models. Based on the current state-of-the-art models, we have noted that the BERT model produces maximum similarity. The model has vocabulary for the English language by default, but it outperformed in various other languages as well.

Table 10. Summary of BERT, ELMO, and GPT-2 Language Models.

Model	Layers	Context-Independent/ Dependent	Encoder/ Decoder	Representation	Sentence/Words	Suitable Domain	Masked Language Modeling/Next Sentence Prediction
BERTBase	12	Context-dependent	Multi-Encoder	Vector representation, polysemy, context	The sequence of words/subwords	Domain-specific and domain-independent	Masked language modeling and next-sentence prediction
ELMo	2	Context-dependent	Encoder, decoder	Morphological	Character level	Task-specific	Masked language modeling
GPT-2	12	Context-dependent	Multi-layer transformer decoder	Context	words	Domain-specific and independent	Masked language modeling

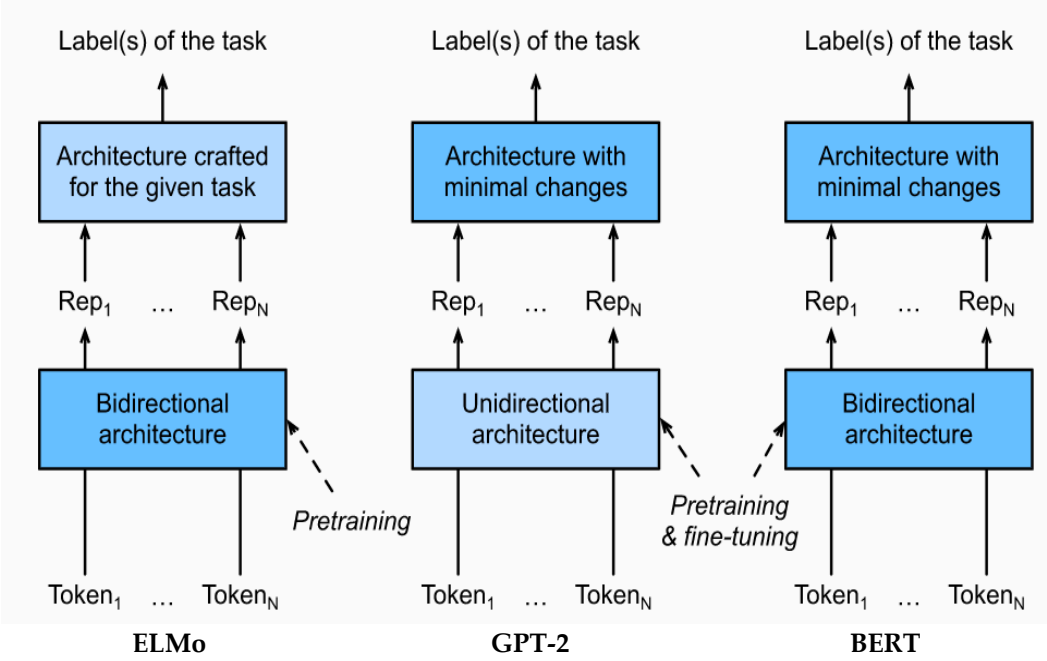

Figure 7. Architectures of ELMO, GPT-2, and BERT model for NLP tasks.

5.5. Performance Measurements of BERT, ELMO, and GPT-2

Every model in the deep learning pipeline was assessed based on performance metrics. The model initially trains or fine-tunes the text dataset for sentence similarity, and, after testing, it validates the similarity results. There are three main functions of any similarity model: create contextualized relationships, predict relevant embeddings, and make appropriate decisions. In this section, we have identified the most used '✔' and unused '*x*' measurements used by the researchers for short-sentence similarity tasks presented in Table 11. We targeted three models: the BERT, ELMO, and GPT-2. Approx. 24 studies were included. In the performance measurements, we noted that the parameters of confusion matrices are still widely used by many researchers.

Table 11. Performance Measurements of BERT, ELMo, and GPT-2.

Model and Study	ACC	Precision	Recall	F1	Train Loss	Validation Loss	Cosine SIM	Jaccard SIM	EM	Specificity	MAP	MRR	RMSE	MAE	Pearson *r*	QWK	Cohen Kappa
BERTBase (12 layers) Kovaleva et al. (2019) [64]	✔	*x*	*x*	✔	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*
KeyBERT Khan et al. (2022) [65]	*x*	*x*	*x*	*x*	*x*	*x*	✔	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*
BERTBase AKE et al. (2021) [68]	✔	✔	✔	✔	✔	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*
Pre-Trained BERT Wiedemann et al. (2019) [69]	*x*	*x*	*x*	✔	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*
Pre-Trained BERT Zhou et al. (2019) [70]	*x*	✔	✔	✔	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*
SEMBERT Zhang et al. (2020) [75]	✔	*x*	*x*	✔	*x*	*x*	*x*	*x*	✔	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*

Table 11. *Cont.*

Model and Study	ACC	Precision	Recall	F1	Train Loss	Validation Loss	Cosine SIM	Jaccard SIM	EM	Specificity	MAP	MRR	RMSE	MAE	Pearson *r*	QWK	Cohen Kappa
BERT Surya et al. (2019) [20]	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	✔	x
BERT Mozafari et al. (2019) [31]	x	x	x	x	x	x	✔	x	x	x	✔	✔	x	x	x	x	x
BERT Wijaya (2021) [32]	✔	✔	✔	✔	x	x	x	x	x	✔	x	x	x	x	x	x	✔
BERT Haidar et al. (2020) [35]	x	x	x	x	x	x	✔	x	x	x	x	x	✔	✔	x	x	x
ELMo Liu (2022) [79]	✔	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
BERT & ELMO Lasker (2020) [94]	x	x	x	x	x	x	x	x	x	x	✔	✔	x	x	x	x	x
ELMo Reimers et al. (2019) [98]	x	x	x	x	✔	x	x	x	x	x	x	x	x	x	x	x	x
GPT-2 Radford et al. (2019) [99]	x	x	x	✔	x	x	x	x	x	x	x	x	x	x	x	x	x
GTP-2 Ethayarajh et al. (2019) [101]	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
GPT-2 Schneider et al. (2021) [103]	x	x	x	✔	✔	x	x	x	x	x	x	x	x	x	x	x	x
BERT Lee et al. (2020) [106]	✔	✔	✔	✔	x	x	x	x	x	x	x	x	x	x	x	x	x
GPT-2 Lee et al. (2020) [108]	x	x	x	x	✔	x	x	x	x	x	x	x	x	x	x	x	x
BERTBase Li et al. (2020) [109]	✔	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
BERT Mallikarjuna et al. (2022) [111]	✔	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
BERT Li et al. (2020) [112]	✔	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
Multilingual BERT Alammary et al. (2022) [34]	x	x	x	✔	x	x	x	x	x	x	x	x	x	x	x	x	x
BERT Garg et al. (2022) [37]	x	x	x	x	x	x	x	x	x	x	x	x	✔	✔	✔	x	✔

6. What Are the Available Datasets Used for Short-text Semantic Similarity?

To answer this question, we reviewed approx. 34 studies for the selection of suitable datasets specific to short-text semantic similarity.

Movie Review (MR): The MR dataset is an open-source dataset, which generates positive and negative movie reviews. This dataset contains 10,662 pairs. The average length of each sentence is 20 words [16]. Due to this account, Mitra et al. [113] used the MR dataset for sentiment analysis. They further imported the NLTK features in Python and focused them on the built-in classifier to compare the accuracy with other datasets. The dataset is suitable for detecting user reviews, recommender systems, text classification, and

sentiment analysis. Many researchers applied a number of algorithms to this dataset and achieved maximum accuracy rates [114–118].

TREC: The TREC dataset is used for question–answering systems (QAS). The total size of QA pairs is 6452 [111,119]. The dataset is further partitioned for the training and testing phases. For the training phase, it includes 5952 QA pairs, and, for testing, the dataset includes 500 test items [120]. Furthermore, it also includes six categories for the selection of answers. However, the responses can be positive or negative. TRECQA raw and TRECQA clean are the recent versions of the dataset; the average length of each sentence is 20 words. As for the selection of correct answers, researchers imported the dataset by using Python 3.6 with the PyTorch library and utilized the Colab platform [1,121].

AG news: This dataset is an English news article dataset [112,122,123] that includes a 127,600 data size. The dataset is additionally split into a training phase and a testing phase. The testing phase includes 7600 data sizes, whereas the training phase includes 120,000 [16]. The length of the sentence is seven words, which is suitable for short text. However, the dataset is divided into three categories for experimentation. The dataset is suitable for data mining and information retrieval techniques. However, Sitikhu et al. [122] used the NLTK library for English stop-words removal, and, for the dataset cleaning and preprocessing, they employed the WordNet lemmatizer.

Mohler dataset: Mohler et al. [40] created a dataset that has been widely used for automated short-answer grading systems (ASAGs). The dataset is the domain-specific dataset. They collected the data from an introductory computer science class at the University of North Texas. From ten assignments, two exams were conducted. The dataset, however, includes 2273 student responses to the 80 questions. Meanwhile, most of the questions are factoids and descriptive. The average length of reference answer is 15–20 words. The student answers were scored by two teachers manually on a scale of 0–5. The dataset has two categories for grading purposes: (i) correct and (ii) incorrect. Moreover, to import the dataset, many studies used Python 3.6 with the NLTK library. The Mohler dataset is the benchmark for grading short answers and is widely evaluated by many studies [36–38,85,124]. However, the Adam optimizer, different regression techniques, and models have been used to grade the student answers.

SST-2: The SST-2 is the Stanford Sentiment Treebank (SST) dataset, which is one of the most popular movie review datasets [125]. The SST-2 is the recent extension of the movie review dataset. Constituency parsers and tree structures are suitable for the SST-2. Moreover, the SST-2 is a collection of 9613 sentence pairs. The average length of each sentence is 19 words. The dataset is divided into five categories: very negative (labeled 0), negative (labeled 1), neutral (labeled 2), positive (labeled 3), and very positive (labeled 4) [126]. The dataset is a binary dataset if it considers only positive and negative categories. For experimentation, the universal sentence encoder, Python 3.6, NLTK library, and GPU-2 have been widely used, and a number of studies used this benchmark for sentiment analysis, text classification, data augmentation, customer feedback systems, and information aggregation [127–130].

SciEntsBank: In automated short-answer grading (ASAGs), the SciEntsBank is used as a corpus of questions and answers. This corpus lies under SemEval-2013 (Semantic evaluation) dataset and is known as a domain-specific corpus, which includes science-domain questions and answers [38]. The average sentence length is 15–20 words. Approx. 10,000 responses from 197 assignments in 15 different science fields are found in the SciEntsBanks [131]. The corpus further includes three categories to identify and grade the short answers (i) correct and incorrect (labeled two-way); (ii) correct, incorrect, and contradictory (labeled three-way); (iii) correct, partially correct, contradictory, irrelevant, and non-domain (labeled five-way). The corpus is suitable for multi-perspective evaluation [132]. Several studies import datasets using Google Colab, Python 3.6, and GPU TeslaK80 with 12 GB RAM. The SciEntsBank further includes unseen answers (UA), unseen questions (UQ), and an unseen domain (UD) [133–136]. Moreover, Table 12 illustrates the summary of the datasets.

Table 12. Summary of available datasets used for short-text similarity.

Dataset	Type	Category	Size	Average Length of Sentences	Experimentation	Studies
Movie Review (MR)	User reviews, recommender systems, text classification, sentiment analysis	Positive, negative	10,662 pairs	20 Words	NLTK Python 3.6	Wang et al. (2021) [1], Mitra et al. (2020) [113], Rehman et al. (2019) [114], Hassan et al. (2017) [115], Khadim et al. (2019) [116], Khan et al. (2020) [117] Van et al. (2017) [118]
TREC	Question–answering systems (QAS), Six categories of questions and answers	Positive, negative	6452 pairs	20 words	PyTorch, Python 3.6, Google Colab platform	Wang et al. (2021) [1], Mallikarjuna et al. (2022) [111], Li et al. (2018) [119], Madabushi et al. (2019) [120], Perevalov et al. (2021) [121]
AG News	English news articles	Positive, negative, neutral	127,600 pairs	7 words	Python 3.6, WordNet Lemmatizer, NLTK	Wang et al. (2021) [1], Sachan et al. (2019) [122], Sitikhu et al. (2019) [123], Li et al. (2020) [112]
Mohler	Automated short-answer grading; domain-Specific	Correct and incorrect	2273 QA pairs	15–20 words	Adam optimizer, Python 3.7, NLTK	Gaddipati et al. (2020) [36], Garg et al. (2022) [37], Zhu et al. (2022) [38], Mohler et al. (2011) [40], Saha et al. (2018) [85], Tulu et al. (2021) [124]
SST-2	Sentiment analysis	Very negative, negative, neutral, positive, very positive	9613 pairs	19 words	Universal sentence encoder, GPU-2 Python 3.6, NLTK	Munikar et al. [125], Quteineh et al. (2020) [126], Feng et al. (2021) [127], Srivastava et al. (2020) [128], Gong et al. (2018) [129], Shen et al. (2020) [130]
SciEntsBank	Automated short-answer grading	2-way, 3-way, 5-way	10,000	15–20 words	Google Colab, Python 3.6, GPU tesla k80, 12 GB RAM	Zhu et al. (2022) [38], Marvaniya et al. (2018) [131], Thakkar et al. (2021) [132], Haller et al. (2022) [133], Pandey et al. (2022) [134], Filighera, et al. (2022) [135], Sawatzki et al. (2022) [136]

7. What Are the Current Challenges and Suggested Improvements in Short Question–Answering Systems (SQAS)?

This section discusses the challenges and suggested improvements in short question–answering systems (QAS). In order to do that, we reviewed approx. 25 studies.

SQAS Challenge 01: In a short question–answering system, the student's answers are relatively very short, and most of the words in answers do not contribute rich semantics when mapped to the model answers. Secondly, the textual similarity is yet unable to perceive the syntactic semantics from student responses [1,4]. Consider the phrase "develop a taste' and 'taste development'. The cosine value between phrases is 0.911, which is close to 1.000 [4]. Both of these phrases are incorrect from a grammatical standpoint, and a human judge can clearly tell the distinctions between them.

Suggested Improvements: Language models such as the BERT, Roberta, ELMO, and GPT-2 understand sentences with extremely deep semantics [101]. The models extract the similarity based on context. Secondly, the language models use an attention mechanism. Reif et al. [137] discovered evidence of grammatical representations from attention metrics. They further presented the syntactic subspaces that represent semantic information.

SQAS Challenge 02: According to Huang et al. [138], the majority of background knowledge and facts are ignored when matching the question–answer pairs. This is due to the fact that the question attributes are excluded from the knowledge base.

Suggested Improvements: The self-attention mechanism can be used to understand context-dependent and context-independent information. Additionally, other kinds of extraneous knowledge, such as simple texts, can be useful for enhancing knowledge representations [138].

SQAS Challenge 03: In a question–answering systems, the majority of studies focused on grading the answers, providing the scoring rubric, and comparing the human-assigned score with the machine-provided score [83]. But more attention still has to be paid to separating the word embedding characteristics from student and reference answers based on their hyperparameters.

Suggested Improvements. Window size, the location of the context window weightings, learning rate, activation rate, batch size, and dropout rate are examples of hyperparameters [76]. The best techniques for maximizing these parameters are Bayesian optimization, grid search, and random search [139,140].

SQAS Challenge 04: Question–answering systems lack consistent support by using general domain question answers. Existing datasets are biased towards specific domains. Secondly, reference answers have more than one correct answer, and there is no golden standard way to grade and score such answers [133,141].

Suggested Improvements: The mBERT model has the ability to understand multilingual datasets. Khan, L et al. [142] introduced a multi-class Urdu dataset for sentiment analysis. They further fine-tuned the mBERT model and identified the positive, negative, and neutral reviews. Secondly, scoring rubrics and tools such as c-rater can be used to score and grade the answers.

SQAS Challenge 05: Every question has a different effect on the intent type of answers, including questions such as list type, descriptive, factoid, open-ended, and so on. Such types of questions require answers in different forms. List-type questions require one or more than two words to answer the questions. However, essay-type answers usually come from descriptive-type questions. Factoid questions are based on factual answers, such as "what is the capital city of Pakistan?", and the answer can be a single statement or one word—"Islamabad". Moreover, open-ended questions expect answers in open thoughts [19]. There is no specific answer limit for such types of questions, which makes it more challenging for the deep learning models to identify the correct or relevant answers [29,82].

Suggested Improvements: Deep learning models are able to adjust the length of sentences, and capture the weights accordingly. The variants of the BERT (SBERT, BioBERT, Roberta, KeyBERT) models are suitable for short-length sentences. However, these models use default vocabulary and workpiece tokenizers for language understanding. Other models, such as CNN, RNN, and Bi-LSTM, also use various preprocessing techniques to limit and normalize the sentences [16,20,46,49].

SQAS challenge 06: In a question–answering systems, there is still room to explore the specific length of a short sentence. Some studies use 10–20 words [143], some studies recommend less than 20 words [144], and some recommend less than 15 words [132]. Hence, there is no set of standards that explains the length of short sentences for machine learning algorithms.

Suggested Improvements: Computational linguistics can be used in order to identify the length of a short sentence.

SQAS Challenge 07: Various methods perform poorly on a number of QA datasets, as the datasets are not cleaned up. Moreover, applying the aligning techniques to answers can remove various relevant words, which can affect the sentence embeddings. [36,145].

Suggested Improvements: Various techniques, such as the universal sentence encoder [146], can be used to improve sentence-level embeddings. Furthermore, preprocessing techniques can be utilized to clean the dataset.

SQAS Challenge 08: The number of studies working on two-way, three-way, and five-way answer categories. Many machine learning algorithms still have difficulty dealing with answers that are semantically ambiguous [10,147] and answers that match the main reference answer in multiple senses.

Suggested improvements: Lyu et al. [67] proposed HowNet with a knowledge-enhanced graph transformer to deal with ambiguity and create multiple senses from sentences. Other deep learning models, such as the BERT, ELMO, and GPT-2, can sort the polysemy and ambiguity from natural language responses.

8. Discussion

This systematic literature review presents the recent trends in the domain of short-text semantic similarity. Semantic similarity is the most widely used area in natural language processing tasks. Understanding the semantics of sentences is still challenging for machine learning algorithms, as there are various sentence drawbacks, such as anomalies, sparsity, semantic ambiguity, polysemy, and the number of sentence structures, such as word order, punctuation, spelling errors, and so on. These drawbacks make the standard methods more complicated to identify the hidden semantics of sentences.

We included approximately 42 deep learning-based research articles to answer the first research question (RQ). We have noted that from the year 2017 to 2021, a number of studies used convolutional neural networks (CNN) with traditional methods, such as the support vector machine and Word2Vec, for different STSS techniques, such as short text classification, short answers, short text categorization, short tweets, and many more. Many research studies, however, made extensive use of Twitter, AG news, Chinese Wikipedia, knowledge base, Probase, Google snippets, and the Hewlett Foundation for short answer scoring. In addition, the recurrent neural network (RNN) has a significant impact on short-text classification and short-sentence similarity. For sentence similarity tasks, they used semantic matching techniques, as well as character-level representations. From the year 2018 until today, language models like the BERT, ELMO, KeyBERT, Roberta, SciBERT, BioBERT, and distilBERT achieved the highest performance in understanding the syntax and semantics of a short sentence. These models employ pre-training and fine-tuning techniques to train on domain-specific and domain-independent datasets. These studies, however, heavily relied on domain-specific and multi-domain datasets to determine similarity.

To answer the second research question, we reviewed approx. 76 research studies to understand the contextual information of the short text. Earlier models, such as convolutional neural networks and recurrent neural networks, used Word2vec, WordNet, support vector machine, and bag-of-word models to understand the background information of text; however, these models are context independent in nature. It means they create the same embeddings for the same word, which is used in different contexts. To that end, in this section, we included the three most wieldy used transformer models, such as the BERT, ELMO, and GPT-2. These models are context-dependent; in order to do this, these models create embeddings based on the given context. Moreover, the BERT model is suitable for vector representation, for extracting ambiguous terms such as polysemy, and for context information from sentences by using a stacked transformer and attention mechanism. The model uses a multi-encoder to understand the sentence in the sequence of words or sub-words, referred to as keywords or key phrases. The BERT model also employed masked language modeling and next-word prediction for domain-specific and multi-domain sentence structures. There are two versions of the BERT model: the BERT Base with 12 layers and the BERT large with 24 layers. The model required huge corpora to

understand the information. However, ELMO uses the encoder and decoder to understand the sentences. The ElMO is suitable for morphological and character-level representations, and the model also uses masked language modeling for the task-specific domains. Furthermore, GPT-2 is a multi-layer transformer decoder. It is a unidirectional stacked transformer that understands the context of the given sentence and extracts the similarity. The model split sentences into words and, as with the BERT model, it is suitable for domain-specific and domain-independent datasets. The model also uses the masked language modeling technique. In terms of performance metrics, these models heavily rely on confusion metrics. There is still room to improve the performance of these models for other available metrics, as is mentioned in Table 11.

Moreover, a number of 34 studies have been included in the selection of the dataset. We added six datasets, such as movie review, TREC, AG news, Mohler, SST-2, and SciEntsBank. These datasets are suitable for short answers, movie reviews, and short text classification. The average length of these datasets is 19–20 words, and the category for responses used includes positive, negative, neutral, contradictory, three-way, and five-way.

Furthermore, we selected approx. 25 studies for reviewing the challenges that are specific to short questions and answers. We have identified the number of challenges in teacher and student-provided answers. The student's answers are short, sometimes irrelevant, and unclear to understand. There is no standard rule to grade and score such answers. However, the suggested changes can also be used to address problems and broaden the research area for future studies.

9. Conclusions

The current study reviewed noteworthy publications on short-text semantic similarity. We discussed some open challenges and drawbacks related to the structure of sentences. We also identified deep learning techniques that work under the domain of short text similarity, short text classification, and short question–answering systems. The study further specified the three most used models for learning the context and background information of sentences. The performance metrics employed by deep learning algorithms were also studied. We also reviewed state-of-the-art datasets that are particularly effective and readily available for short text similarity. These datasets have been tested by a variety of tools. For future recommendations, we discussed the challenges and provided suggested improvements for future research directions.

Author Contributions: Abstract contemplation, Z.H.A., Y.K.H. and H.B.; investigation, Z.H.A. and K.D.; methodology, Z.H.A., Y.K.H. and K.D.; formal analysis, H.B.; data curation, Z.H.A., G.M.S. and H.B.; interpretation of data, Z.H.A., Y.K.H. and H.B.; review, writing and editing, Z.H.A., Y.K.H. and K.D.; validation, H.B. and K.D.; supervision, Y.K.H.; project administration, Z.H.A. and Y.K.H.; writing—editing, original draft, Z.H.A. and Y.K.H.; funding procurement, Y.K.H. All authors have read and agreed to the published version of the manuscript.

Funding: Yayasan UTP Pre-commercialization grant (YUTP-PRG) 015PBC-005.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: Appreciation goes to the Yayasan UTP Pre-commercialization grant (YUTP-PRG) 015PBC-005 and the Computer and Information Science Department of Universiti Teknologi PETRONAS for supporting this work.

Conflicts of Interest: Authors have no conflict of interest.

References

1. Wang, H.; Tian, K.; Wu, Z.; Wang, L. A short text classification method based on convolutional neural network and semantic extension. *Int. J. Comput. Intell. Syst.* **2021**, *14*, 367–375. [CrossRef]
2. Zhao, H.; Hu, G.; Jiao, C. Short Text Similarity Calculation Using Semantic Information. In Proceedings of the 2017 3rd International Conference on Big Data Computing and Communications (BIGCOM), Chengdu, China, 10–11 August 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 144–150.
3. Mohammad, A.-S.; Jaradat, Z.; Mahmoud, A.-A.; Jararweh, Y.J.I.P. Management, Paraphrase identification and semantic text similarity analysis in Arabic news tweets using lexical, syntactic, and semantic features. *Inf. Process. Manag.* **2017**, *53*, 640–652.
4. Olowolayemo, A.; Nawi, S.D.; Mantoro, T. Short, answer scoring in English grammar using text similarity measurement. In Proceedings of the 2018 International Conference on Computing, Engineering and Design (ICCED), Bangkok, Thailand, 6–8 September 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 131–136.
5. Zhang, Y.; Tuo, M.; Yin, Q.; Qi, L.; Wang, X.; Liu, T. Keywords extraction with the deep neural network model. *Neurocomputing* **2020**, *383*, 113–121. [CrossRef]
6. Hua, W.; Wang, Z.; Wang, H.; Zheng, K.; Zhou, X. Short text understanding through lexical-semantic analysis. In Proceedings of the 2015 IEEE 31st International Conference on Data Engineering, Seoul, Republic of Korea, 13–17 April 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 495–506.
7. Han, M.; Zhang, X.; Yuan, X.; Jiang, J.; Yun, W.; Gao, C.J. A survey on the techniques, applications and performance of short text semantic similarity. *Concurr. Comput. Pract. Exp.* **2021**, *33*, e5971. [CrossRef]
8. Hasanah, U.; Permanasari, A.E.; Kusumawardani, S.S.; Pribadi, F. A scoring rubric for automatic short answer grading system. *Telkomnika* **2019**, *17*, 763–770. [CrossRef]
9. Hu, Y.; Ding, J.; Dou, Z.; Chang, H.J.C. Neuroscience, Short-Text Classification Detector: A Bert-Based Mental Approach. *Comput. Intell. Neurosci.* **2022**, *2022*, 8660828.
10. Huang, P.-S.; Chiu, P.-S.; Chang, J.-W.; Huang, Y.-M.; Lee, M. A study of using syntactic cues in the short-text similarity measure. *J. Internet Technol.* **2019**, *20*, 839–850.
11. Alsalami, A.I. Challenges of Short Sentence Writing Encountered by First-Year Saudi EFL Undergraduate Students. *Arab World Engl. J.* **2022**, *13*, 534–549. [CrossRef]
12. Gaddipati, S.K. Automatic Formative Assessment for Students' Short Text Answers through Feature Extraction. Ph.D. Thesis, Hochschule Bonn-Rhein-Sieg, Sankt Augustin, Germany, 2021.
13. Rehman, A.; Hassan, M.F.; Yew, K.H.; Paputungan, I.; Tran, D. State-of-the-art IoV trust management a meta-synthesis systematic literature review (SLR). *PeerJ Comput. Sci.* **2020**, *6*, e334. [CrossRef]
14. Moustaka, V.; Vakali, A.; Anthopoulos, L. A systematic review for smart city data analytics. *ACM Comput. Surv.* **2018**, *51*, 1–41. [CrossRef]
15. Kitchenham, B.J.K. *Procedures for Performing Systematic Reviews*; Keele University: Keele, UK, 2004; Volume 33, pp. 1–26.
16. Shih, S.-H.; Yeh, C. A Short Answer Grading System in Chinese by CNN. In Proceedings of the 10th International Conference on Awareness Science and Technology (iCAST), Morioka, Japan, 23–25 October 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1–5.
17. Xu, J.; Cai, Y.; Wu, X. Incorporating context-relevant concepts into convolutional neural networks for short text classification. *Neurocomputing* **2020**, *386*, 42–53. [CrossRef]
18. Perera, N.; Priyankara, C.; Jayasekara, D. Identifying Irrelevant Answers in Web Based Question Answering Systems. In Proceedings of the 20th International Conference on Advances in ICT for Emerging Regions (ICTer), Colombo, Sri Lanka, 4–7 November 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 11–16.
19. Surya, K.; Gayakwad, E.; Nallakaruppan, M. Deep learning for short answer scoring. *Int. J. Recent. Technol. Eng.* **2019**, *7*, 1712–1715.
20. Wang, P.; Xu, J.; Xu, B. Semantic clustering and convolutional neural network for short text categorization. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*; Association for Computational Linguistics: Toronto, ON, Canada, 2017; pp. 352–357.
21. Liu, J.; Ma, H.; Xie, X.; Cheng, J.J.E. Short Text Classification for Faults Information of Secondary Equipment Based on Convolutional Neural Networks. *Energies* **2022**, *15*, 2400. [CrossRef]
22. Hu, Y.; Li, Y.; Yang, T.; Pan, Q. Short text classification with a convolutional neural networks based method. In Proceedings of the 15th International Conference on Control, Automation, Robotics and Vision (ICARCV), Singapore, 18–21 November 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 1432–1435.
23. Agarwal, B.; Ramampiaro, H.; Langseth, H. Management, A deep network model for paraphrase detection in short text messages. *Inf. Process. Manag.* **2018**, *54*, 922–937. [CrossRef]
24. Yao, L.; Pan, Z.; Ning, H.J.I.A. Unlabeled short text similarity with LSTM encoder. *IEEE Access* **2018**, *7*, 3430–3437. [CrossRef]
25. Dwivedi, V.P.; Singh, D.K.; Jha, S. Gender classification of blog authors: With feature engineering and deep learning using LSTM networks. In Proceedings of the 9th International Conference on Advanced Computing (ICoAC), Chennai, India, 14–16 December 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 142–148.
26. Li, Q.; Wu, Q.; Zhu, C.; Zhang, J. Bi-level masked multi-scale CNN-RNN networks for short text representation. In Proceedings of the International Conference on Document Analysis and Recognition (ICDAR), Sydney, NSW, Australia, 20–25 September 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 888–893.

27. Edo-Osagie, O.; Lake, I.L. Attention-based recurrent neural networks (RNNs) for short text classification: An application in public health monitoring. In Proceedings of the 15th International Work-Conference on Artificial Neural Networks, IWANN 2019, Gran Canaria, Spain, 12–14 June 2019; Springer: Berlin/Heidelberg, Germany, 2019; pp. 895–911.
28. Hassan Amur, Z.; Kwang Hooi, Y. State-of-the-Art: Assessing Semantic Similarity in Automated Short-Answer Grading Systems. *Inf. Sci. Lett.* **2022**, *11*, 40.
29. Lee, J.Y.; Dernoncourt, F. Sequential short-text classification with recurrent and convolutional neural networks. *arXiv* **2016**, arXiv:1603.03827.
30. Liu, P.; Yuan, W.; Fu, J. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Comput. Surv.* **2023**, *55*, 1–35. [CrossRef]
31. Mozafari, J.; Fatemi, A. BAS: An answer selection method using BERT language model. *arXiv* **2019**, arXiv:1911.01528.
32. Wijaya, M.C. Automatic Short Answer Grading System in Indonesian Language Using BERT Machine Learning. *Rev. D'intelligence Artif.* **2021**, *35*, 503–509. [CrossRef]
33. Luo, J. Automatic Short Answer Grading Using Deep Learning. Ph.D. Thesis, Illinois State University, Normal, IL, USA, 2021.
34. Alammary, A.S. BERT Models for Arabic Text Classification: A Systematic Review. *Appl. Sci.* **2022**, *12*, 5720. [CrossRef]
35. Haidir, M.H.; Purwarianti, A. Short answer grading using contextual word embedding and linear regression. *J. Linguist. Komputasional* **2020**, *3*, 54–61.
36. Gaddipati, S.K. Comparative evaluation of pretrained transfer learning models on automatic short answer grading. *arXiv* **2020**, arXiv:2009.01303.
37. Garg, J.; Papreja, J.; Apurva, K.; Jain, G. Domain-Specific Hybrid BERT based System for Automatic Short Answer Grading. In Proceedings of the 2nd International Conference on Intelligent Technologies (CONIT), Hubli, India, 24–26 June 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 1–6.
38. Zhu, X.; Wu, H.; Zhang, L. Automatic Short-Answer Grading via BERT-Based Deep Neural Networks. *IEEE Trans. Learn. Technol.* **2022**, *15*, 364–375. [CrossRef]
39. Burrows, S.; Gurevych, I.; Stein, B.J. The eras and trends of automatic short answer grading. *Int. J. Artif. Intell. Educ.* **2015**, *25*, 60–117. [CrossRef]
40. Mohler, M. Learning to grade short answer questions using semantic similarity measures and dependency graph alignments. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Portland, OR, USA, 19–24 June 2011; pp. 752–762.
41. Wang, Z.; Lan, A.S.; Waters, A. *A Meta-Learning Augmented Bidirectional Transformer Model for Automatic Short Answer Grading*; EDM: Munich, Germany, 2019.
42. Sung, C. Pre-training BERT on domain resources for short answer grading. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*; Association for Computational Linguistics: Toronto, ON, Canada, 2019; pp. 6071–6075.
43. Khodeir, N.A. Bi-GRU Urgent Classification for MOOC Discussion Forums Based on BERT. *IEEE Access* **2021**, *9*, 58243–58255. [CrossRef]
44. Camus, L.; Filighera, A. Investigating transformers for automatic short answer grading. In Proceedings of the International Conference on Artificial Intelligence in Education, Ifrane, Morocco, 6–10 July 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 43–48.
45. Sung, C.; Dhamecha, T.I.; Mukhi, N. Improving short answer grading using transformer-based pre-training. In Proceedings of the International Conference on Artificial Intelligence in Education, Chicago, IL, USA, 25–29 June 2019; Springer: Berlin/Heidelberg, Germany, 2019; pp. 469–481.
46. Mayfield, E.; Black, A.W. Should you fine-tune BERT for automated essay scoring? In Proceedings of the 15th Workshop on Innovative Use of NLP for Building Educational Applications, Seattle, WA, USA, 9 July 2020; pp. 151–162.
47. Nie, F.; Zhou, S.; Liu, J.; Wang, J. Aggregated semantic matching for short text entity linking. In Proceedings of the 22nd Conference on Computational Natural Language Learning, Brussels, Belgium, 31 October–1 November 2018; pp. 476–485.
48. De Boom, C. Learning semantic similarity for very short texts. In Proceedings of the International Conference on Data Mining Workshop (ICDMW), Atlantic City, NJ, USA, 14–17 November 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 1229–1234.
49. Prakoso, D.W.; Abdi, A.; Amrit, C.J.S.C. Short text similarity measurement methods: A review. *Soft Comput.* **2021**, *25*, 4699–4723. [CrossRef]
50. Yang, J.; Li, Y.; Gao, C.; Zhang, Y. Measuring the short text similarity based on semantic and syntactic information. *Futur. Gener. Comput. Syst.* **2021**, *114*, 169–180. [CrossRef]
51. Ye, Z.; Jiang, G. Document and word representations generated by graph convolutional network and bert for short text classification. In *ECAI 2020*; IOS Press: Amsterdam, The Netherlands, 2020; pp. 2275–2281.
52. Hu, W.; Dang, A.; Tan, Y. A survey of state-of-the-art short text matching algorithms. In Proceedings of the International Conference on Data Mining and Big Data, Chiang Mai, Thailand, 26–30 July 2019; Springer: Berlin/Heidelberg, Germany, 2019; pp. 211–219.
53. Xiao, M.; Yao, M.; Li, Y. Short-text intention recognition based on multi-dimensional dynamic word vectors. *J. Phys.* **2020**, *1678*, 012080. [CrossRef]

54. Wan, Q.; Liu, J. Engineering, A self-attention based neural architecture for Chinese medical named entity recognition. *Math. Biosci. Eng.* **2020**, *17*, 3498–3511. [CrossRef]
55. Lin, X.; Xiong, G.; Gou, G.; Li, Z. Et-bert: A contextualized datagram representation with pre-training transformers for encrypted traffic classification. In *Proceedings of the ACM Web Conference 2022*; Association for Computing Machinery: New York, NY, USA, 2022; pp. 633–642.
56. Beltagy, I.; Lo, K.; Cohan, A.J. SciBERT: A pre-trained language model for scientific text. *arXiv* **2019**, arXiv:1903.10676.
57. Devlin, J.; Chang, M.-W.; Lee, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.
58. Bojanowski, P.; Grave, E.; Joulin, A. Enriching word vectors with subword information. *Trans. Assoc. Comput. Linguist.* **2017**, *5*, 135–146. [CrossRef]
59. Sedoc, J.; Ungar, L. The role of protected class word lists in bias identification of contextualized word representations. In *Proceedings of the First Workshop on Gender Bias in Natural Language Processing*; Association for Computational Linguistics: Toronto, ON, Canada, 2019; pp. 55–61.
60. Sarzynska-Wawer, J.; Wawer, A.; Pawlak, A. Detecting formal thought disorder by deep contextualized word representations. *Psychiatry Res.* **2021**, *304*, 114135. [CrossRef]
61. Fernandez, N.; Ghosh, A.; Liu, N.; Wang, Z.; Choffin, B.; Baraniuk, R.; Lan, A.J. Automated Scoring for Reading Comprehension via In-context BERT Tuning. In Proceedings of the Artificial Intelligence in Education: 23rd International Conference, AIED 2022, Durham, UK, 27–31 July 2022; Springer: Cham, Switzerland, 2022.
62. Li, Y.; Yang, Y.; Hu, Q.; Chen, C. An Argument Extraction Decoder in Open Information Extraction. In Proceedings of the Advances in Information Retrieval: 43rd European Conference on IR Research, ECIR 2021, Virtual Event, 28 March–1 April 2021; Springer: Berlin/Heidelberg, Germany, 2021; pp. 313–326.
63. Yin, X.; Huang, Y.; Zhou, B.; Li, A. Deep entity linking via eliminating semantic ambiguity with BERT. *EEE Access* **2019**, *7*, 169434–169445. [CrossRef]
64. Kovaleva, O.; Romanov, A. Revealing the dark secrets of BERT. *arXiv* **2019**, arXiv:1908.08593.
65. Khan, M.Q.; Shahid, A.; Uddin, M.I. Impact analysis of keyword extraction using contextual word embedding. *PeerJ Comput. Sci.* **2022**, *8*, e967. [CrossRef]
66. Tang, M.; Gandhi, P.; Kabir, M. Progress notes classification and keyword extraction using attention-based deep learning models with BERT. *arXiv* **2019**, arXiv:1910.05786.
67. Lyu, B.; Chen, L. Let: Linguistic knowledge enhanced graph transformer for chinese short text matching. In Proceedings of the AAAI Conference on Artificial Intelligence, Virtual Event, 2–9 February 2021; pp. 13498–13506.
68. Eke, C.I.; Norman, A.A.; Shuib, L.J.I.A. Context-based feature technique for sarcasm identification in benchmark datasets using deep learning and BERT model. *IEEE Access* **2021**, *9*, 48501–48518. [CrossRef]
69. Wiedemann, G.; Remus, S. Does BERT make any sense? Interpretable word sense disambiguation with contextualized embeddings. *arXiv* **2019**, arXiv:1909.10430.
70. Zhou, Y.; Li, C.; He, S.; Wang, X.; Qiu, Y. Pre-trained contextualized representation for Chinese conversation topic classification. In Proceedings of the 2019 IEEE International Conference on Intelligence and Security Informatics (ISI), Shenzhen, China, 1–3 July 2019; IEEE: Pisctaway, NJ, USA, 2019; pp. 122–127.
71. Vaswani, A.; Shazeer, N.; Parmar, N. Attention is all you need. In Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA, 4–9 December 2017.
72. Heidari, M.; Jones, J.H.; Uzuner, O. Deep contextualized word embedding for text-based online user profiling to detect social bots on Twitter. In Proceedings of the 2020 International Conference on Data Mining Workshops (ICDMW), Sorrento, Italy, 17–20 November 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 480–487.
73. Amur, Z.H.; Hooi, Y.K.; Soomro, G.M. Automatic Short Answer Grading (ASAG) using Attention-Based Deep Learning MODEL. In *2022 International Conference on Digital Transformation and Intelligence (ICDI)*; IEEE: Pisctaway, NJ, USA, 2022; pp. 1–7.
74. Mu, J. All-but-the-top: Simple and effective postprocessing for word representations. *arXiv* **2017**, arXiv:1702.01417.
75. Zhang, Z.; Wu, Y. Semantics-aware BERT for language understanding. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; pp. 9628–9635.
76. Chiu, B.; Baker, S.J. Word embeddings for biomedical natural language processing: A survey. *Lang. Linguist. Compass* **2020**, *14*, e12402. [CrossRef]
77. Peters, M.E.; Neumann, M.; Gardner, M.; Clark, C.; Lee, K.; Zettlemoyer, L. Deep contextualized word representations. *arXiv* **2018**, arXiv:1802.05365.
78. Gupta, H.; Patel, M. Study of extractive text summarizer using the Elmo embedding. In Proceedings of the Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), Palladam, India, 7–9 October 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 829–834.
79. Liu, C.; Gao, Y.; Sun, L.; Feng, J.; Yang, H.; Ao, X. In User Behavior Pre-training for Online Fraud Detection. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*; Association for Computing Machinery: New York, NY, USA, 2022; pp. 3357–3365.
80. Rezaii, N.; Walker, E. A machine learning approach to predicting psychosis using semantic density and latent content analysis. *Schizophrenia* **2019**, *5*, 9. [CrossRef] [PubMed]

81. Naseem, U.; Musial, K.; Eklund, P.; Prasad, M. Biomedical named-entity recognition by hierarchically fusing biobert representations and deep contextual-level word-embedding. In Proceedings of the 2020 International Joint Conference on Neural Networks (IJCNN), Glasgow, UK, 19–24 July 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1–8.
82. Amur, Z.H.; Hooi, Y. State-of-the Art: Short Text Semantic Similarity (STSS) Techniques in Question Answering Systems (QAS). In Proceedings of the International Conference on Artificial Intelligence for Smart Community, Seri Iskandar, Malaysia, 17–18 December 2022; Springer: Berlin/Heidelberg, Germany, 2022; pp. 1033–1044.
83. Galhardi, L.B.; Brancher, J.D. Machine learning approach for automatic short answer grading: A systematic review. In Proceedings of the Advances in Artificial Intelligence-IBERAMIA 2018: 16th Ibero-American Conference on AI, Trujillo, Peru, 13–16 November 2018; Springer: Berlin/Heidelberg, Germany, 2018; pp. 380–391.
84. Zhang, Y.; Shah, R. *Deep Learning + Student Modeling + Clustering: A Recipe for Effective Automatic Short Answer Grading*; Institute of Education Sciences: Washington, DC, USA, 2016.
85. Saha, S.; Dhamecha, T.I.; Marvaniya, S.; Sindhgatta, R.; Sengupta, B. Sentence level or token level features for automatic short answer grading? Use both. In Proceedings of the Artificial Intelligence in Education: 19th International Conference, AIED 2018, London, UK, 27–30 June 2018; Springer: Berlin/Heidelberg, Germany, 2018; pp. 503–517.
86. Li, Z.; Tomar, Y.; Passonneau, R.J. A Semantic Feature-Wise Transformation Relation Network for Automatic Short Answer Grading. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, Punta Cana, Dominican Republic, 7–11 November 2021; pp. 6030–6040.
87. Hassan, S.; Fahmy, A. Applications, Automatic short answer scoring based on paragraph embeddings. *Int. J. Adv. Comput. Sci. Appl.* **2018**, *9*, 397–402.
88. Al-Bataineh, H.; Farhan, W. Deep contextualized pairwise semantic similarity for Arabic language questions. In Proceedings of the 2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI), Portland, OR, USA, 4–6 November 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1586–1591.
89. Yang, Y.; Yuan, S.; Cer, D. Learning semantic textual similarity from conversations. *arXiv* **2018**, arXiv:1804.07754.
90. Soliman, A.B.; Eissa, K. A set of Arabic word embedding models for use in Arabic NLP. *Procedia Comput. Sci.* **2017**, *117*, 256–265. [CrossRef]
91. Neelakantan, A.; Shankar, J. Efficient non-parametric estimation of multiple embeddings per word in vector space. *arXiv* **2015**, arXiv:1504.06654.
92. Church, K.W. Word2Vec. *Nat. Lang. Eng.* **2017**, *23*, 155–162. [CrossRef]
93. Wieting, J.; Bansal, M.; Gimpel, K. Charagram: Embedding words and sentences via character n-grams. *arXiv* **2016**, arXiv:1607.02789.
94. Laskar, M.T.R.; Huang, X.; Hoque, E. Contextualized embeddings based transformer encoder for sentence similarity modeling in answer selection task. In Proceedings of the 12th Language Resources and Evaluation Conference, Marseille, France, 11–16 May 2020; pp. 5505–5514.
95. Reimers, N.; Gurevych, I. Alternative weighting schemes for elmo embeddings. *arXiv* **2019**, arXiv:1904.02954.
96. Liu, L.; Ren, X.; Shang, J.; Peng, J. Efficient contextualized representation: Language model pruning for sequence labeling. *arXiv* **2018**, arXiv:1804.07827.
97. Walker Orr, J.; Tadepalli, P. Event Detection with Neural Networks: A Rigorous Empirical Evaluation. *arXiv* **2018**, arXiv:1808.08504.
98. Reimers, N.; Gurevych, I. Why comparing single performance scores does not allow to draw conclusions about machine learning approaches. *arXiv* **2018**, arXiv:1803.09578.
99. Radford, A.; Wu, J. Language models are unsupervised multitask learners. *OpenAI Blog* **2019**, *1*, 9.
100. Vig, J.; Belinkov, Y. Analyzing the structure of attention in a transformer language model. *arXiv* **2019**, arXiv:1906.04284.
101. Ethayarajh, K. How contextual are contextualized word representations? comparing the geometry of BERT, ELMo and GPT-2 embeddings. *arXiv* **2019**, arXiv:1909.00512.
102. Han, X.; Zhang, Z. Pre-trained models: Past, present and future. *AI Open* **2021**, *2*, 225–250. [CrossRef]
103. Schneider, E.T.R.; de Souza, J. A GPT-2 Language Model for Biomedical Texts in Portuguese. In Proceedings of the 34th International Symposium on Computer-Based Medical Systems (CBMS), Aveiro, Portugal, 7–9 June 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 474–479.
104. Zhao, Z.; Wallace, E.; Feng, S. Calibrate before use: Improving few-shot performance of language models. In Proceedings of the International Conference on Machine Learning, Virtual Event, 18–24 July 2021; pp. 12697–12706.
105. Carlini, N.; Tramer, F.; Wallace, E.U. Extracting training data from large language models. In Proceedings of the 30th USENIX Security Symposium (USENIX Security 21), Virtual Event, 11–13 August 2021; pp. 2633–2650.
106. Lee, J.-S.; Hsiang, J. Patent classification by fine-tuning BERT language model. *World Pat. Inf.* **2020**, *61*, 101965. [CrossRef]
107. Birunda, S.S.; Devi, R.K. A review on word embedding techniques for text classification. In Proceedings of the Innovative Data Communication Technologies and Application, Coimbatore, India, 20–21 August 2021; pp. 267–281.
108. Lee, J.-S.; Hsiang, J.J.W.P.I. Patent claim generation by fine-tuning OpenAI GPT-2. *World Pat. Inf.* **2020**, *62*, 101983. [CrossRef]
109. Li, B.; Zhou, H.; He, J. On the sentence embeddings from pre-trained language models. *arXiv* **2020**, arXiv:2011.05864.
110. Su, J.; Cao, J. Whitening sentence representations for better semantics and faster retrieval. *arXiv* **2021**, arXiv:2103.15316.
111. Mallikarjuna, C.; Sivanesan, S. Question classification using limited labeled data. *Inf. Process. Manag.* **2022**, *59*, 103094. [CrossRef]

112. Li, D.; Zhang, Y.; Peng, H. Contextualized perturbation for textual adversarial attack. *arXiv* **2020**, arXiv:2009.07502.
113. Mitra, A. Sentiment analysis using machine learning approaches (Lexicon based on movie review dataset). *J. Ubiquitous Comput. Commun. Technol.* **2020**, *2*, 145–152.
114. Rehman, A.U.; Malik, A. Applications, A hybrid CNN-LSTM model for improving accuracy of movie reviews sentiment analysis. *Multimed. Tools Appl.* **2019**, *78*, 26597–26613. [CrossRef]
115. Hassan, A.; Mahmood, A. Deep learning approach for sentiment analysis of short texts. In Proceedings of the 3rd International Conference on Control, Automation and Robotics (ICCAR), Nagoya, Japan, 24–26 April 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 705–710.
116. Kadhim, A.I. Survey on supervised machine learning techniques for automatic text classification. *Artif. Intell. Rev.* **2019**, *52*, 273–292. [CrossRef]
117. Khan, A.; Gul, M.A. Summarizing online movie reviews: A machine learning approach to big data analytics. *Sci. Program.* **2020**, *2020*, 5812715. [CrossRef]
118. Van-Tu, N. Technology, Improving question classification by feature extraction and selection. *Indian J. Sci. Technol.* **2017**, *9*, 1–8.
119. Li, D. Representation learning for question classification via topic sparse autoencoder and entity embedding. In Proceedings of the International Conference on Big Data (Big Data), Seattle, WA, USA, 10–13 December 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 126–133.
120. Madabushi, H.T.; Lee, M. Integrating question classification and deep learning for improved answer selection. In Proceedings of the 27th International Conference on Computational Linguistics, Santa Fe, NM, USA, 10–26 August 2018; pp. 3283–3294.
121. Perevalov, A. Improving answer type classification quality through combined question answering datasets. In Proceedings of the International Conference on Knowledge Science, Engineering and Management, Tokyo, Japan, 14–16 August 2021; Springer: Berlin/Heidelberg, Germany, 2021; pp. 191–204.
122. Sachan, D.S. Revisiting LSTM networks for semi-supervised text classification via mixed objective function. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019.
123. Sitikhu, P.; Pahi, K.; Thapa, P.; Shakya, S. A comparison of semantic similarity methods for maximum human interpretability. In Proceedings of the Artificial Intelligence for Transforming Business and Society (AITB), Kathmandu, Nepal, 5 November 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1–4.
124. Tulu, C.N. Automatic short answer grading with semspace sense vectors and malstm. *IEEE Access* **2021**, *9*, 19270–19280. [CrossRef]
125. Munikar, M.; Shakya, S.; Shrestha, A. Fine-grained sentiment classification using BERT. In Proceedings of the Artificial Intelligence for Transforming Business and Society (AITB), Kathmandu, Nepal, 5 November 2019; IEEE: Piscatway, NJ, USA, 2019; pp. 1–5.
126. Quteineh, H.; Samothrakis, S.; Sutcliffe, R. Textual data augmentation for efficient active learning on tiny datasets. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*; Association for Computational Linguistics: Toronto, ON, Canada, 2020; pp. 7400–7410.
127. Feng, L.; Yu, J.; Cai, D.; Liu, S.; Zheng, H.; Wang, Y.J. ASR-GLUE: A new multi-task benchmark for asr-robust natural language understanding. *arXiv* **2021**, arXiv:2108.13048.
128. Srivastava, A.; Makhija, P.; Gupta, A. Noisy text data: Achilles' heel of BERT. In Proceedings of the Sixth Workshop on Noisy User-Generated Text (W-NUT 2020), Gyeongju, Republic of Korea, 12–17 October 2020; pp. 16–21.
129. Gong, J.; Qiu, X.; Wang, S. Information aggregation via dynamic routing for sequence encoding. *arXiv* **2018**, arXiv:1806.01501.
130. Shen, S.; Dong, Z.; Ye, J.; Mahoney, M.W.; Keutzer, K. Q-bert: Hessian based ultra-low precision quantization of bert. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 12–17 February 2020.
131. Marvaniya, S.; Saha, S.; Dhamecha, T.I.; Foltz, P.; Sindhgatta, R.; Sengupta, B. Creating scoring rubric from representative student answers for improved short answer grading. In Proceedings of the 27th ACM International Conference on Information and Knowledge Management, Torino, Italy, 22–26 October 2018; pp. 993–1002.
132. Thakkar, M.J. Finetuning Transformer Models to Build ASAG System. *arXiv* **2021**, arXiv:2109.12300.
133. Haller, S. Survey on Automated Short Answer Grading with Deep Learning: From Word Embeddings to Transformers. *arXiv* **2022**, arXiv:2204.03503.
134. Pandey, S.J. Modelling Alignment and Key Information for Automatic Grading. Ph.D. Thesis, The Open University, Milton Keynes, UK, 2022.
135. Filighera, A.; Ochs, S.; Steuer, T.; Tregel, T.J. Cheating Automatic Short Answer Grading: On the Adversarial Usage of Adjectives and Adverbs. *arXiv* **2022**, arXiv:2201.08318.
136. Sawatzki, J. Deep Learning Techniques for Automatic Short Answer Grading: Predicting Scores for English and German Answers. In *Artificial Intelligence in Education: Emerging Technologies, Models and Applications*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 65–75.
137. Reif, E.; Yuan, A.; Wattenberg, M. Visualizing and measuring the geometry of BERT. *arXiv* **2019**, arXiv:1906.02715.
138. Huang, W.; Qu, Q. Applications, Interactive knowledge-enhanced attention network for answer selection. *Neural Comput. Appl.* **2020**, *32*, 11343–11359. [CrossRef]
139. Wu, J.; Chen, X.-Y.; Zhang, H.; Xiong, L.-D. Hyperparameter optimization for machine learning models based on Bayesian optimization. *J. Electron. Sci. Technol.* **2019**, *17*, 26–40.

140. Saha, A.; Ganesan, B. Short Text Clustering in Continuous Time Using Stacked Dirichlet-Hawkes Process with Inverse Cluster Frequency Prior. In Proceedings of the 6th Joint International Conference on Data Science & Management of Data (10th ACM IKDD CODS and 28th COMAD), Mumbai, India, 4–7 January 2023; pp. 118–122.
141. Iparraguirre-Villanueva, O.; Guevara-Ponce, V. Text prediction recurrent neural networks using long short-term memory-dropout. *Indones. J. Electr. Eng. Comput. Sci.* **2023**, *29*, 1758–1768. [CrossRef]
142. Khan, L.; Amjad, A.; Ashraf, N. Multi-class sentiment analysis of urdu text using multilingual BERT. *Sci. Rep.* **2022**, *12*, 5436. [CrossRef]
143. Nguyen, H.T.; Duong, P.H. Learning short-text semantic similarity with word embeddings and external knowledge sources. *Knowl.-Based Syst.* **2019**, *182*, 104842. [CrossRef]
144. Kadayat, B.B. Impact of sentence length on the readability of web for screen reader users. In Proceedings of the International Conference on Human-Computer Interaction, Copenhagen, Denmark, 19–24 July 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 261–271.
145. Koponen, I.T.; Södervik, I.; Nousiainen, M. Lexical networks constructed to correspond to students' short written responses: A quantum semantic approach. In Proceedings of the International Conference on Complex Networks and Their Applications, Paris, France, 13–14 April 2023; Springer: Cham, Switzerland, 2023; pp. 137–149.
146. Cer, D.; Yang, Y.; Kong, S.-Y.; Hua, N. Universal sentence encoder. *arXiv* **2018**, arXiv:1803.11175.
147. Hussain, M.J.; Bai, H.; Wasti, S.H.; Huang, G.; Jiang, Y. Evaluating semantic similarity and relatedness between concepts by combining taxonomic and non-taxonomic semantic features of WordNet and Wikipedia. *Inf. Sci.* **2023**, *625*, 673–699. [CrossRef]