


Article

# A Framework for Word Embedding Based Automatic Text Summarization and Evaluation

Tulu Tilahun Hailu <sup>1</sup> , Junqing Yu <sup>1,2,\*</sup> and Tessfu Geteye Fantaye <sup>1</sup>

<sup>1</sup> School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China; tutilcs@hust.edu.cn (T.T.H.); tessfug@hust.edu.cn (T.G.F.)

<sup>2</sup> Center of Network and Computation, Huazhong University of Science and Technology, Wuhan 430074, China

\* Correspondence: yjqing@hust.edu.cn

Received: 4 January 2020; Accepted: 28 January 2020; Published: 31 January 2020



**Abstract:** Text summarization is a process of producing a concise version of text (summary) from one or more information sources. If the generated summary preserves meaning of the original text, it will help the users to make fast and effective decision. However, how much meaning of the source text can be preserved is becoming harder to evaluate. The most commonly used automatic evaluation metrics like Recall-Oriented Understudy for Gisting Evaluation (ROUGE) strictly rely on the overlapping n-gram units between reference and candidate summaries, which are not suitable to measure the quality of abstractive summaries. Another major challenge to evaluate text summarization systems is lack of consistent ideal reference summaries. Studies show that human summarizers can produce variable reference summaries of the same source that can significantly affect automatic evaluation metrics scores of summarization systems. Humans are biased to certain situation while producing summary, even the same person perhaps produces substantially different summaries of the same source at different time. This paper proposes a word embedding based automatic text summarization and evaluation framework, which can successfully determine salient top-n sentences of a source text as a reference summary, and evaluate the quality of systems summaries against it. Extensive experimental results demonstrate that the proposed framework is effective and able to outperform several baseline methods with regard to both text summarization systems and automatic evaluation metrics when tested on a publicly available dataset.

**Keywords:** Automatic evaluation metrics; extrinsic evaluation; intrinsic evaluation; natural language processing; text summarization; word embedding

## 1. Introduction

With the tremendous and growing of smart phones and web technologies, the amount of text data on the web is increasing exponentially. Users should spend more time to find relevant information. This has inspired the development of automatic text summarization systems for producing a concise summary that preserves core idea of the original document [1]. In other words, any automatic text summarization system is intended for distilling most relevant information from the original document to create a shortened version. Further, Torres-Moreno [2] provided six reasons why automatic text summarization system is needed: (1) it creates summaries that reduce users reading time; (2) when researching documents, the generated summaries make the selection process easier; (3) it improves the effectiveness of indexing; (4) automatic summarization system is less biased than human summarizers; (5) the produced summaries are also useful in question-answering systems; and (6) using automatic summarization system enables commercial abstract services to increase the number of texts they are able to process.

Text summarization systems can be classified into several groups from different perspectives [3,4]. For instance, based on the input type, summarization can be classified into single and multi-document summaries. As the name suggests, single document summarization systems are designed to generate summary from a single document whereas multi-document summarization systems are intended to generate summary from multiple documents. Based on the purpose, text summarization can also be categorized into generic, domain specific and query-based summary. Generic summaries are produced with regardless of specific topic or domain whereas domain specific and query-based summaries are generated based on specific area and request made by the users respectively. In the literature, the most commonly mentioned types of summaries are extractive and abstractive types that are classified on the basis of output type.

Extractive text summarization involves identification, ranking and merging most important units from the source whereas an abstractive summarization involves selecting and compressing content of the source text to generate a summary of perhaps entirely new sentences. Further, it has been demonstrated that mixed summary can be produced by generating new words and copying fragments from the original text [5]. In order to identify salient sentences in the source text, numerous features have been considered for computing the relevance score of sentences. For instance, word/phrase frequency [6]; common lexical tokens, and location of the sentence [7]; and term frequency-inverse document frequency (TF-IDF) based cosine similarity, and longest common match [8], and different levels of text embeddings for computing cosine similarity [9]. Very recently, following sequence-to-sequence learning schemes in machine translation domain, several extractive [10–12], abstractive [13–15], and mixed [5,16,17] text summarization models have been proposed.

According to [18], extractive text summarization is easier and more successful text summarization approach when compared to abstractive summarization method. However, the reason behind the success of extractive text summarization seems that it has been favored by existing automatic evaluation metrics. For example, the most widely used automatic evaluation metrics like ROUGE intuitively disregard an abstractive summary that contains the same information as an extractive summary. Almost all existing automatic evaluation metrics including ROUGE have been designed based on ordinary co-selection measurements such as precision, recall and f-score. These measurements are used to complement techniques such as TF-IDF based latent semantic analysis, term frequency scheme, and n-gram matches between system and reference summaries [19]. We argue that evaluating abstractive summarization models based solely on lexical overlaps is completely unfair. Thus, there has been an awakening interest in the meaning of oriented automatic evaluation metrics.

Another possible challenge of automatic evaluation of text summarization systems is lack of consistent ideal reference summary. Studies have shown that there is low agreement among human summarizers in determining sentences for producing an ideal summary [20]. Even the same person perhaps produces drastically different summary of the same source text at different time. According to [20], there is more variability among the human summaries than among the systems summaries with large margin and very little agreement between human and system sentences selections.

In this paper, we propose a word embedding based automatic text summarization and evaluation framework. We used acronyms WETS and WEEM4TS for referring to our proposed word embedding based text summarization and word embedding based evaluation metric for text summarization respectively, and the code is available at GitHub (<https://github.com/TuluTilahun/Text-Summarization>). WETS and WEEM4TS are both simple yet surprisingly powerful text summarization and evaluation methods.

Our research questions are as follows:

RQ1: For salient top-n sentences determination, how can we leverage publicly available pre-trained word embedding models? In order to answer this research question, we develop a system called word embedding based text summarization (WETS for short), and compare with the baseline systems.

RQ2: Are publicly available pre-trained word embedding models useful for developing automatic evaluation metrics that are suitable to evaluate all kinds of system summaries? To answer this research

question, we design a word embedding based evaluation metric for text summarization (WEEM4TS), and compare it with the commonly used automatic evaluation metrics.

Our contributions compared to previous work are as follows:

- We propose an automatic evaluation metric called WEEM4TS for evaluating the performance of text summarization systems. WEEM4TS is designed to evaluate the quality of system summaries with regard to the preserved meaning of the original document. Hence, we believe it represents an appropriate evaluation metric for all types of systems summaries: extractive, abstractive, and mixed.
- We propose a method called WETS for determining the most important sentences of original document that can be used as a reference summary, which helps to address lack of ground truth summaries. This reference summary is produced carefully to be used by WEEM4TS for evaluating systems summaries against it.
- By comparing with six baseline text summarization systems, we validate the utility of summaries that generated by WETS. We also evaluate performance of WEEM4TS by correlating with human judgments. Further, we compare WEEM4TS with commonly used automatic evaluation metrics in the domain of text summarization and machine translation. The experimental results demonstrate that both WETS and WEEM4TS achieve promising performance.

The remainder of this paper is organized as follows: In Section 2, we discuss prior work on text summarization and evaluation of text summarization systems. In Section 3, we describe the proposed approaches. Experimental datasets, tools, baselines, results and discussion are presented in Section 4. Finally, the conclusions and some future research directions are discussed in Section 5.

## 2. Related Work

Our review focuses on two lines of literature most relevant to our work: text summarization and the evaluation of text summarization systems.

### 2.1. Text Summarization

Recently, numerous automatic text summarization approaches have been proposed exhaustively, and can be categorized based on the input type, purpose, and output type. Input type-based summarization can be classified into single- and multi-document summarization. Based on the purpose, text summarization can be divided into three main groups: generic, domain-specific, and query-based summarization. Based on the output type, it can also be categorized into extractive and abstractive summarization.

Extractive summarization approaches are intended to generate summaries by selecting important units of the original document. In contrast, abstractive summarization approaches are aimed at generating summary with new words or phrases. In order to preserve core idea of the original text, an abstract summarization approach requires advanced NLP tools such as ideal semantic parsers, and language generation systems. Therefore, pure extractive summarization methods are more applicable than abstract summarization methods [18]. On the other hand, as it requires extraction and preprocessing at the early stage, it is difficult to generate pure abstractive summaries. For instance, in order to generate an abstract summary from multiple documents, the model in [21] has been trained to extract relevant sentences first and then perform compression. Similarly, [22] used statistical methods to extract promising sentences to train a model that can mimic human summarizers in generating abstractive summary.

In the literature, most of the extractive summarization systems are purely heuristic. For instance, study in [7] encompasses several features such as sentence position and pragmatic words; picking the first three sentences of the source text as a summary [12,23]; for text ranking: the longest common substring and term frequency were used in [8]; and the overlapping of frequent words with the title, heading, and query words [6,24]. The presence of frequently occurring words in the sentence might

attribute to the relevance of the sentence [25]. In contrast, some common words (like stop words) have no significant contribution in conveying the importance of a sentence. To address this discrepancy, inverse document frequency has been employed in [26,27]. Further, to extract the most relevant sentences from source text, the fuzzy logic technique has been applied in [28].

Supervised and unsupervised learning approaches have also been considered to perform an extractive summary as classification [29] and clustering [30] tasks respectively. A trainable summarization model has also been proposed in [31]. The goal is to generate a summary by capturing several features: sentence position; positive and negative keywords in the sentence; sentence similarity with other sentences; sentence resemblance to the title; occurrence of proper noun in the sentence; sentence inclusion of numeric data; relative sentence length; bushy path of the sentence; summation of similarities for each sentence. The authors first investigated the effect of each sentence features on the summarization task. Then, to obtain a suitable combination of feature weights, they used all features score function to train a genetic algorithm and mathematical regression models, which learn relevant features from summaries produced by human expertise.

Perhaps humans often generate a summary by creating new sentences rather than simply extracting and concatenating fragments of source text. With regard to automatic summarizers, the problem of generating new condensed text from long source text is relatively difficult. In line with this, the automatic evaluation of abstractive summaries is a very challenging task. Consequently, abstractive text summarization has somehow received less attention. However, recently, several methods have been proposed to generate abstractive summary. For instance, study in [32] followed the statistical machine translation scheme to generate a short title. Similarly, the syntactic tree and hidden Markov model have been used to produce summary as headlines [33]. In some studies, manually crafted rules have been used to remove less important part of the sentences and retain the most relevant components for compression [22,34]. More recently, several methods or features have been considered to generate good quality summary: employing embedding models at different levels [9,35,36], query-based [37], attention mechanism oriented sequence learning [14,38–40], pointer-generator network was applied for hybrid text summarization in [5] and hierarchical or graph based summarization models [41–43]. Studies in [9,35,36] leveraged word embedding models for extractive text summarization that made them more related to ours. However, the way we used vector values of the words and computation of sentence relevance score are different. For instance, in [9,35,36], vector values of the words have been used to form sentence level embedding, and later used for calculating sentence relevance score, whereas sentence embedding is not required in our proposed text summarization system. To compare our work with these studies, the dataset used in each work is different and also not convenient for us to reproduce the result. For example, experimental datasets used in [9] are multi-document and multilingual, which are not appropriate in our case because our text summarization method is intended for single document and monolingual investigation.

Moreover, deep learning approaches have also been proposed generating summaries [44–46]. Noticeably, deep learning approaches require large dataset. Thus, to satisfy the demand, several studies have proposed ways of collecting large original-summary pairs. For instance, [47] employed deep learning approach to identify and prepare large-scale high quality document-summary pairs. Likewise, 1.3 million article-summary pairs high-quality dataset was prepared and made publicly available by [23]. Study in [48] also proposed a large scale corpus for Chinese text summarization task. These can help to address the lack of publicly available dataset for improving quality of text summarization models. Further, there are also few public and private datasets that have been used to train, validate, and test text summarization models. For example, CNN/Daily Mail dataset [49]; DUC 2004 Task [50]; Gigaword summarization dataset [40,51], NEWSROOM dataset to train Abs-N summarization model [23]; Webis-TLDR-17 Corpus [52]; and Google dataset [53] have been used to train summarizer models for generating different types of summaries: extractive [10–12], abstractive [13–15,53,54], and mixed [5,16,17].

## 2.2. Evaluation of Text Summarization Systems

Automatic evaluation metrics can be broadly classified into two categories: extrinsic and intrinsic methods [55].

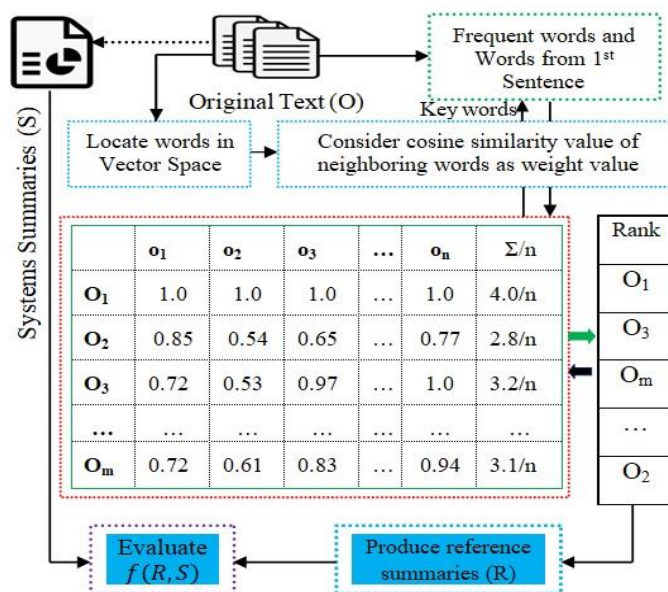
Intrinsic methods are intended to evaluate quality of system summaries in terms of ideal reference summaries [7] or by comparing with the important units of original document [56,57]. In contrast, extrinsic evaluation methods do not compare system and human reference summaries, and rather assess the impact of generated summaries on the performance of other NLP systems. For example, the quality of generated summaries can be determined by their suitability for surrogating original documents for categorization [58,59], information retrieval [60], and question-answering tasks [61]. Thus, in the literature, extrinsic evaluation methods are sometimes referred to as task-based evaluation methods.

For intrinsic based evaluation, the standard information retrieval measurements such as precision, recall, and f-score have been used in several automatic evaluation metrics. For instance, the most widely used automatic evaluation metric called ROUGE (Recall-Oriented Understudy for Gisting Evaluation) relies on these measurements [62]. Although ROUGE has been found to correlate well with human judgments [63], it is not capable to evaluate an abstractive type of system summaries. Further, as the extractive summarization sometimes dangles anaphoric references, these metrics are limited to assess the coherence of the system summaries. To tackle this problem, study in [64] engages two jurors to complement automatic coherence evaluation of generated summaries. Another semi-automatic method called pyramid involves manual judgments to quantify the relative importance of facts to be conveyed [65].

In order to improve performance of automatic evaluation metrics, series of text summarization evaluation campaigns have been organized since the late 1990 [3]. For example, SUMMAC (1996–1998) [66], DUC (the Document Understanding Conference, 2000–2007) [50], and more recently TAC (the Text Analysis Conference, 2008–present). However, still, the automatic evaluation of text summarization systems is a challenging issue. Lack of a standard evaluation metric has caused summary evaluation to be difficult [3]. Existing automatic evaluation metrics that have been developed based on co-selection measurements, n-gram matching, and term frequency based metrics are not appropriate for evaluating abstractive summaries. They all fail to provide equal or closer scores for system summaries that convey the same information with different words and phrases. According to [67], ordinary recall and precision are unsuitable measurements for text summary because a small difference in the system summary can noticeably affect the evaluation score. In order to address this problem, [68] proposed Roget's Thesaurus based sentence-ranking method. However, because of acknowledged challenges of thesaurus construction, it is difficult to get ideal thesaurus [69]. It would be better to employ other publicly available resources like the work similar to ours [70]. However, word weighting and final score computing techniques in our study is different from that of [70].

## 3. Methods

As depicted in Figure 1, the proposed framework consists of two major components: generation of reference summaries and evaluation of system summaries. We describe them in the following sections, and see example provided in Appendix A.



**Figure 1.** The general workflow architecture of proposed framework for automatic text summarization and evaluation. Given original sentences  $O = O_1, O_2, O_3 \dots O_m$  with their corresponding sequence of words  $o_{ij} = o_{i1}, o_{i2}, o_{i3} \dots o_{in}$ , assign the highest cosine similarity value to the words by comparing them with the keywords. Keywords are nonstop words from first sentence of the original document and top k number of frequent words from the document, (we experimented with values between 1 and 15, and found the optimal value to be  $k = 6$ ). Then sum up all weights ( $\Sigma$ ) and divide by the number of words ( $n$ ) in the corresponding sentence for ranking. Then top-y sentences are considered as a reference summary.

Although human generated summary has become a de facto ground truth, it is not available for most application domains. Another challenging issue in the domain of text summarization is lack of semantic oriented automatic evaluation metrics. Thus, in this study, we propose a word embedding based text summarization and evaluation framework. Word embedding is the most commonly used vector representation of words. Study in [71] introduced two efficient and high quality word representation model architectures that was successfully trained on millions of words. The authors have improved quality word vectors in their other work [72]. Word embedding has capability of preserving syntax and semantic regularities [73]. Further, sub-word level embedding was also introduced by [74], which helps to preserve morphological regularity. In this study, we use word embedding models that have been developed based on three different algorithms: Word2Vec [71], GloVe [75], and FastText [74].

### 3.1. Word Embedding Based Text Summarization

In order to answer the first research question (RQ1), we propose word embedding based text summarization (WETS) method for identifying, ranking and concatenating salient top-y sentences that can be used as a reference summary.

The most commonly used sentence relevance judgment method in the literature is checking the presence of frequent and pragmatic words. In view of that, a sentence that consisted of more of these words is considered most important. However, we argue that this technique has two major drawbacks: first, it encourages redundancy in the new summary; second, it fails to assign appropriate score for very important sentences consisted of other words. Thus, a redundancy handling mechanism and meaning oriented sentence relevance assessment technique is vital.

Accordingly, in this study, the preliminary task is removing irrelevant tokens like stop words from the original document. Then, we utilize words of the first sentence together with frequent words as keywords. The main reason we tend to focus on the words of the first sentence is that linguistic

literature show an explicit thesis statement mostly located at the beginning of the paragraph [76], which indicates that the important words might exist in the first sentence. In addition, relatively, at least a few words in the title might also exist in the first sentence. It is possible to conduct comparative analysis by considering words in the middle sentences or words in the last sentence. However, it is out of scope to deal with this here.

As depicted in Figure 1 and Algorithm 1, the cosine similarity between each word of each sentence in the document and the keywords is computed and value of the most similar key word is considered as a weight value of the target word. If the target word does not exist in the vocabulary of word embedding model, the word will be assigned with the weight value of 0. This weighting technique helps to allot fair score for the relevant sentences that comprised of words different from keywords. Further, in order to discourage redundancy, words of other sentences that also exist in the first sentence are ignored. Thus, the relevance score of the sentences composed of words in the first sentence become lower.

---

**Algorithm 1:** Word Embedding based Text Summarization
 

---

**Input:** Original\_Text, word\_embedding\_model, stop\_words  
**Output:** Top-y salient sentences

- 1: **for** i in range (len(Original\_Text)):
- 2:   sentences←Original\_Text[i].split(. )
- 3:   firstsentence ←Original\_Text [0].split(. )
- 4:   **for** sentence **in** sentences:
- 5:     tokenized←yield(gensim.utils.simple\_preprocess(str(sentence)))
- 6:     nonstopwords ←remove\_stopwords(tokenized)
- 7:     **for** m **in** range(len(firstsentence)):
- 8:       **if** firstsentence[m] is not **in** stop\_words:
- 9:         first\_keywords←firstsentence[m]
- 10:    second\_keywords←frequent\_words(Original\_Text)
- 11:    keywords←first\_keywords + second\_keywords
- 12:    weight←0, sentweight←0
- 13:    **for** n **in** range (len(nonstopwords)):
- 14:     weight←max(cosine\_similarity(tokenized[n], keywords))
- 15:     sentweight←sentweight + weight
- 16:    relevancescore←sentweight/len(nonstopwords)
- 17:    top-y←put\_sentence\_in\_order(relevancescore)
- 18: **return** top-y

---

The obtained cosine similarity values (weights) of all words in each sentence are added and then divided by the updated length of the corresponding sentence, as in Equation (1). Updated length means the length of the sentences after the removal of redundant and stop words. Based on the relevance scores, the sentences are ranked from top to bottom. Finally, concatenate top-y sentences as per the required length. It should be noted that although sentence relevance score is calculated on the basis of semantic similarity, WETS is an extractive text summarization system. In this study, length of reference summary is variable, i.e., it can be adjusted according to the required length of system summaries. This helps to compare system summaries with different lengths. For instance, assume the original documents comprised of 200 words; and if summary of text summarization systems: A and B comprised of 150 and 100 words respectively. Then length of reference summary can be adjusted to be 150 for system A and 100 for system B by picking the required number of words from top to bottom. However, a longer system summary might be favored. To control this, the length of system summary

should comply with the pre-determined threshold or compression ratio. In this study, system summary with longer length is indirectly penalized by modified bigram precision as in Equation (3).

$$relevance_{score}(O_i) = \frac{\sum_{j=1}^n w_{ij}}{|O_i|}, i = 1, 2, 3, \dots, m \quad (1)$$

where  $w_{ij}$  is the highest cosine similarity value between the word  $o_{ij}$  and the keywords, and  $|O_i|$  represents the updated length of sentence at  $i$ -th position in the original document.

### 3.2. Word Embedding Based Automatic Evaluation Metric for Text Summarization

To answer the second research question (RQ2), we propose a word embedding based evaluation metric for text summarization (WEEM4TS), see Algorithm 2. WEEM4TS consists of four different components: pre-processing, word weighting, computing modified unigram recall, and computing modified bigram precision.

---

#### Algorithm 2: Word Embedding based Evaluation Metric for Text Summarization

---

**Input:** Reference\_Text, System\_Summary, word\_embedding\_model, stop\_words  
**Output:** metric\_score

- 1: sentweight, unigramrecall, weight, countbigram ← 0.0
- 2: **for** i **in** range(len(Reference\_Text)):
- 3:     **for** n **in** range(len(System\_Summary)):
- 4:         **if** System\_Summary[i][n] **in** Reference\_Text[i]:
- 5:             weight ← 1.0
- 6:             sentweight ← sentweight + weight
- 7:         **else if** System\_Summary[i][n] **in** vocabulary(word\_embedding\_model):
- 8:             weight ← max(cosine\_similarity(System\_Summary[i][n], vocabulary))
- 9:             sentweight ← sentweight + weight
- 10:         **else:**
- 11:             weight ← 0.0
- 12:             sentweight ← sentweight + weight
- 13:     **if** len(Reference\_Text[i]) > 0:
- 14:         unigramrecall ← sentweight / len(Reference\_Text[i])
- 15:     **else:**
- 16:         unigramrecall ← 0.0
- 17:     bigramprecision ← (countbigram / (len(System\_Summary[i]) - 1)) \* 100
- 18:     unigramrecall ← unigramrecall \* 100
- 19:     WEEM4TSscore ← ( $\alpha$  \* unigramrecall) + ( $\beta$  \* bigramprecision)
- 20: **return** WEEM4TSscore

---

#### 3.2.1. Pre-processing

Remove irrelevant units such as stop words and characters from both system and reference summaries.

#### 3.2.2. Word weighting

By using word embedding model, we compute cosine similarity between words in the system and reference summaries. Accordingly, if a word is shared by system and reference summaries, it receives a + 1 score. If a target word in the system summary does not appear in the reference summary but exist in the word embedding vocabulary, a cosine similarity value between the word and the closest word in the vector space is considered as a weight value of the target word. If none of the two happen, a score of 0 is given to the target word.



### 3.2.3. Computing modified unigram recall

Based on weight values of the words in step 2 above, we compute a modified unigram recall, as in Equation (2). It should be noted that the term modified is used to indicate the recall used in this study is different from the commonly used recall. The standard recall is based on surface form matching whereas the modified version of it relies on both surface form and word embedding based matching. For example, in the system summary: [He is a decent boy.]; and reference summary: [He is a good boy], the words/tokens such as 'He', 'is', and 'a' are all ignored because they are stop words. On the other hand, the word 'boy' in the system summary exists also in the reference summary, thus it receives score of +1 whereas the word 'decent' in the system summary does not exist in the reference summary. Thus, the highest cosine similarity value between 'decent' and among nonstop words in the reference summary is considered as a weight value of the word. In other words, the cosine similarity value between 'decent' and 'good', which is via Word2Vec.

$$Unigram_{rec} = \frac{\sum_{i=1}^n w_i}{n'} * 100 \quad (2)$$

where  $n'$  is the number of words in the reference summary and  $w_i$  is a weight value of each word in the system summary. Based on the pre-determined compression ratio, the number of words in the system summary ( $n$ ) is expected to comply with the number of words in the reference summary, i.e.,  $n' \approx n$ .

### 3.2.4. Computing modified bigram precision

We also compute the sequence of bigram agreement between system and reference summaries. The bigrams are counted not only based on the exact matches, but also on the cosine similarity of the words. Thus, the term modified is used to indicate the bigram counting technique in this study is different from the conventional one. Accordingly, if the words next to shared word in the system and reference summaries are the most similar words then these words with their previous word can be considered as a bigram match that can be used for computing the modified bigram precision, as in as in Equation (3). For example, in the system summary: [Yesterday tornado hit the city]; and reference summary: [Yesterday cyclone caused great loss]; here, yesterday tornado can be counted as a bigram match with yesterday cyclone as the cyclone is perhaps the most similar word to tornado.

$$bigram_{pre} = \frac{\#bigram\_matches}{n - 1} * 100 \quad (3)$$

where  $n$  is number of words in the system summary.

In order to compute the final WEEM4TS score, the modified unigram recall and bigram precision are linearly combined with different importance levels, as in Equation(4).

$$WEEM4TS = \alpha(Unigram_{rec}) + \beta(Bigram_{pre}) \quad (4)$$

We experimented with nine combinations:  $\alpha = 0.1, 0.2 \dots 0.9$  and  $\beta = 0.9, 0.8, 0.7 \dots 0.1$ , where  $\alpha + \beta$  always 1, and found the optimal combination values to be  $\alpha = 0.8$  with  $\beta = 0.2$ . Further, we experimented with other modified measurements such as unigram precision and bigram recall, and found poor results. Thus, we chose to exclude them here.

## 4. Experiments

### 4.1. Dataset

We tested our proposed methods on NEWSROOM summarization dataset (<https://summaries/>) [23]. The corpus contains 1.3 million articles paired with summaries written by human expertise in newsrooms of 38 major publishers between 1998 and 2017. The articles and the corresponding

summaries were extracted from HTML body content and metadata of different domains respectively. These summaries exhibit diversity of summarization styles: extractive, abstractive, and mixed.

Extractive NEWSROOM summaries were produced by concatenating fragments of the original documents whereas abstractive summaries of this corpus comprised of novel words. The NEWSROOM summaries that contain both extractive and abstractive summaries are referred as mixed summaries. In order to compare our proposed text summarization method (WETS) with a baseline text summarization method called Lede-3, we randomly picked some of the dataset from the NEWSROOM test dataset described in Table 1. In addition, we compare summaries generated by WETS with six text summarization systems including Lede-3 on a dataset prepared for manual evaluation (<https://github.com/lil-lab/newsroom/tree/master/humaneval>) as shared by [23].

**Table 1.** Randomly selected paired article-summary dataset from NEWSROOM test data for comparing word embedding based text summarization (WETS) with Lede-3.

Summarization Style	Data Size
Extractive	346
Abstractive	405
Mixed	330

The shared manual evaluation dataset comprises the original text, system summaries, and human judgment scores. We used scores provided by human assessors for evaluating performance of our proposed automatic evaluation metric (WEEM4TS). In order to perform manual evaluation of system summaries, the authors disseminated 60 summaries of seven systems to three assessors along with the original text via Amazon Mechanical Turk. The assessors are required to evaluate summaries in terms of four criteria: informativeness, relevance, fluency, and coherence. Informativeness and relevance were used to assess semantic nature of the summaries with respect to original articles whereas fluency and coherence were intended for collecting syntactic judgments.

#### 4.2. Pre-Trained Word Embedding Models

For the purpose of developing both text summarization system and automatic evaluation metric, we adopted publicly available pre-trained word embedding models developed based on three different algorithms: Word2Vec [71], GloVe [75], and FastText [74], see Table 2.

**Table 2.** Statistic of publicly available pre-trained word embedding models with 300 dimensions.

Algorithm	Corpus	Vocabulary Size
Word2Vec	Google news	3 million
GloVe	Common crawl	2.2 million
FastText	Common crawl and Wikipedia	2 million

#### 4.3. Baselines

We compared our proposed text summarization method (WETS) with Lede-3 text summarization system on randomly selected NEWSROOM test dataset described in Table 1. We also compared WETS with six baseline text summarization systems including Lede-3 on human evaluation dataset shared by [23], see sample in Table 3. Likewise, we compared our proposed automatic evaluation metric (WEEM4TS) with the variants of three baseline automatic evaluation metrics. In the next subsections, we describe the baseline text summarization systems and automatic evaluation metrics consecutively.

**Table 3.** Sample dataset: original text, reference summary, baseline systems summaries and summary generated by our summarization system (WETS).

<b>Original Text: NEWSROOM Article</b>
Mortgage rates are still pretty cheap, even though they have risen a full percentage point since hitting record lows about a year ago. And with the stronger economy pulling housing along, this is a good time to get into the market Anika Khan, Wells Fargo Securities senior economist, told CNBC is Squawk Box; on Friday. But many first-time homebuyers are being left on the sidelines, watching all that cheap money inch higher because lending requirements remain tight. The average rate on a 30-year loan ticked up to 4.41 percent from 4.40 percent last week. Fifteen-year mortgages increased to 3.47 percent from 3.42 percent. In this video, Khan gives three reasons why it is still so hard for would-be buyers to purchase their first home.
<b>NEWSROOM Reference Summary</b>
Many first-time homebuyers are being left on the sidelines watching all that cheap money inch higher because among other reasons lending requirements remain tight.
<b>Lede-3</b>
Mortgage rates are still pretty cheap, even though they have risen a full percentage point since hitting record lows about a year ago. And with the stronger economy pulling housing along, this is a good time to get into the market; Anika Khan, Wells Fargo Securities senior economist, told CNBC is Squawk Box on Friday. But many first-time homebuyers are being left on the sidelines, watching all that cheap money inch higher because lending requirements remain tight.
<b>Abs-N</b>
device could help save lives
<b>TextRank</b>
Mortgage rates are still pretty cheap, even though they have risen a full percentage point since hitting record lows about a year ago. The average rate on a 30-year loan ticked up to 4.41 percent from 4.40 percent last week. Fifteen-year mortgages increased to 3.47 percent from 3.42 percent.
<b>Pointer-C</b>
Market, anika khan, wells fargo securities senior economist, told cnbc is “squawk box on friday. But many first-time homebuyers are being left on the 30-year loan ticked up to 4.41 percent from 4.40 percent last week. Fifteen-year mortgages increased to 3.47 percent from 3.42 percent. In this video, khan gives three reasons why it’s still so hard for would-be buyers to purchase their first home. The average rate on a 30-year loan ticked up to 4.41 percent from 4.40 risen a full percentage point since hitting record lows about a year ago. and with the
<b>Pointer-S</b>
mortgage rates are still pretty cheap, even though they’ve risen a full percentage point since hitting record lows about a year ago many reasons why the stronger economy pulling housing along, “this is a good time to get into the market, anika khan, wells fargo securities senior economist, told cnbc is” squawk box on friday many first-time homebuyers are being left on the sidelines, watching all that cheap money inch higher because lending requirements remain tight.
<b>Pointer-N</b>
mortgage rates are still pretty cheap—even though they have risen a full percentage point since hitting record lows about a year ago. and with the stronger economy pulling housing along
<b>WETS</b>
Mortgage rates are still pretty cheap, even though they have risen a full percentage point since hitting record lows about a year ago.

#### 4.3.1. Text Summarization Systems

Lede-3 used in [23] is similar with the Lead-3 employed in [12] and [5]. It was intended to consider the first three sentences of the original text as a summary. According to [23], though simple, Lede-3 is competitive extractive summarization with state-of-the-art systems. Thus, we used Lede-3 as a baseline to compare with our method in two dataset configurations: (1) we considered the first three sentences of randomly selected 1081 documents from NEWSROOM test data as a summary, and (2) we used 60 summaries generated based on Lede-3 as shared by [23].

TextRank is a graph based unsupervised sentence level extractive summarization system in which pragmatic words, title and heading words, and sentence locations have been considered to calculate sentence relevance score [40]. Study in [8] improved TextRank by considering three additional components: longest common substring, TF-IDF based cosine similarity, and another TF-IDF variant called BM25. For comparison, we used 60 summaries generated based on TextRank as shared by [23].

Abs-N stands for an abstractive summarization model trained on NEWSROOM dataset by [23]. For comparison, we used 60 Abs-N summaries generated based on this model that made publicly available by [23]. To train the model, the authors followed a TensorFlow implementation of study in [1]. For further information refer [23].

The Pointer model was trained to generate a mixed summary that consisted of new tokens and fragments from source text [5]. The model was developed based on pointer mechanism [77] and attention history [78]. In order to compare with our method, we used summaries generated based on the pointer models trained on three different datasets by [23]: Pointer-C was trained on the CNN/Daily Mail dataset; Pointer-N was trained on the NEWSROOM dataset; and Pointer-S was trained on a random subset of NEWSROOM training data. The authors used each model to generate 60 summaries and made the summaries publicly available. We utilized these summaries to compare with our method. For more information refer [5,77,78] as the authors in [23] followed these studies.

#### 4.3.2. Automatic Evaluation Metrics

ROUGE (<https://pypi.org/project/rouge/>) stands for recall-oriented understudy for gisting evaluation [62], see Equation (5). ROUGE is the most commonly used automatic evaluation metric in the domain of text summarization. It was intended to count the number of overlapping units. It has four major variants: ROUGE-N, ROUGE-L, ROUGE-W, and ROUGE-S. ROUGE-N measures an  $n$ -gram recall between system summaries and reference summaries. ROUGE-L was aimed to match the longest sequence of words between system and reference summaries for computing  $f$ -measure. It does not require consecutive match rather same order of words in both candidate and reference summaries. ROUGE-W was proposed to value the gap between words in the longest sequence match. Another variant of ROUGE is ROUGE-S that stands for skip-bigram co-occurrence statistics. This was intended for counting any overlapping pair of words by allowing arbitrary gaps. In this study, for comparison, we considered unigram (R-1), bigram (R-2), and the longest sequence match (R-L) variants.

$$\text{ROUGE-N} = \frac{\sum_{S \in \{\text{ReferenceSummaries}\}} \sum_{\text{gram}_n \in S} \text{Count}_{\text{match}}(\text{gram}_n)}{\sum_{S \in \{\text{ReferenceSummaries}\}} \sum_{\text{gram}_n \in S} \text{Count}(\text{gram}_n)} \quad (5)$$

where  $n$  stands for the length of the  $n$ -gram,  $\text{gram}_n$ , and  $\text{Count}_{\text{match}}(\text{gram}_n)$  is the maximum number of  $n$ -grams co-occurring in a candidate summary and a set of reference summaries.

BLEU ([https://www.nltk.org/\\_modules/nltk/translate/bleu\\_score.html](https://www.nltk.org/_modules/nltk/translate/bleu_score.html)), i.e., bilingual evaluation understudy (BLEU for short), was designed to evaluate the quality of machine translation systems' output in terms of reference translation [79], see Equation (6). The primary task of this metric is to count position independent  $n$ -gram matches between system and reference translations. In order to discourage repeated words in the candidate translation, the redundant words were clipped, and modified precision was computed by dividing clipped  $n$ -grams for the number of  $n$ -grams in the candidate translation. To calculate the final BLEU score, the modified precision scores are multiplied by the brevity penalty. Although it was proposed to evaluate system translations, the precision oriented BLEU is on-par with the recall oriented ROUGE in the evaluation of system summaries [80]. It is also

possible to compute BLEU-1, BLEU-2, BLEU-3, and BLEU-4 scores. However, BLEU-4 is the default and commonly referred to as BLEU. In our experiment, we used all variants.

$$BLEU = \min\left(1, \frac{\text{output length}}{\text{reference length}}\right) \left(\prod_{i=1}^4 \text{precision}_i\right)^{\frac{1}{4}} \quad (6)$$

CHRF ([https://www.nltk.org/\\_modules/nltk/translate/chrf\\_score.html](https://www.nltk.org/_modules/nltk/translate/chrf_score.html)) calculates a simple F-score that combines the recall and precision of character n-grams of maximum length 6 with different setting for the  $\beta$  parameter ( $\beta = 1, 2, \text{ or } 3$ ) [81], see Equation (7). Based on the  $\beta$  value, CHRF is referred to as CHRF1, CHRF2, and CHRF3 for  $\beta = 1, \beta = 2, \text{ and } \beta = 3$  respectively.  $\beta = 1$  implies equal weights for recall and precision whereas  $\beta = 3$  entails the recall has three times more weight. Variants of CHRF were the best in several translation directions among the involved automatic evaluation metrics in WMT16 [82] and WMT17 [83] metric shared task. Though it is not common for evaluating text summarization systems, based on its performance in the domain of machine translation, we used all variants of CHRF metric to compare with our proposed automatic evaluation metric.

$$CHRF = \left(1 + \beta^2 \frac{\text{ChrP} \cdot \text{ChrR}}{\beta^2 \cdot \text{ChrP} + \text{ChrR}}\right) \quad (7)$$

where ChrP and ChrR stand for character n-gram precision and recall arithmetically averaged over all n-grams.

#### 4.4. Results and Discussion

##### 4.4.1. Evaluation Results of Text Summarization Systems

In Table 4, we report the ROUGE scores of our proposed text summarization method (WETS) and baseline method (Lede-3) on randomly selected dataset from NEWSROOM test data. The dataset statistics is described in Table 1. We further compare our proposed text summarization method with several baseline systems including Lede-3 on human evaluation dataset (<https://github.com/lil-lab/newsroom/tree/master/humaneval>) that made publicly available by [23], and report the results in Table 5. The shared manual dataset is comprised of system summaries; the corresponding human judgment scores; and original article text. Among the seven systems, we selected six of them as baseline systems by ignoring extractive oracle fragments, because this system is favored as it has access to the reference summary. In the provided dataset, the reference summaries were not incorporated. Thus, in order to evaluate and compare system summaries, we fetch the corresponding reference summaries from NEWSROOM test data.

In order to produce a WETS summary, we used three different publicly available pre-trained word embedding models that had been trained on Word2Vec, GloVe, and FastText. However, we noticed that although their relevance scores are different, there are no significant differences among the WETS summaries that generated based on these models. Thus, we considered only Word2Vec based WETS summaries, and report the results in Table 4.

**Table 4.** ROUGE scores of WETS and Lede-3. For this experiment, we used pairs of article-summary dataset that described in Table 1. The articles are the source text from which Lede-3 and WETS summaries were generated while the paired summaries are used as a reference (ground truth) to evaluate both Lede-3 and WETS. Better results are highlighted bold.

	Extractive			Abstractive			Mixed		
	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L
Lede-3	<b>52.5</b>	<b>42.0</b>	44.0	10.5	1.2	5.2	21.5	10.2	12.2
WETS	49.0	37.8	<b>47.0</b>	<b>10.9</b>	<b>1.3</b>	<b>8.7</b>	<b>24.9</b>	<b>12.0</b>	<b>21.8</b>

**Table 5.** ROUGE variants (R-1, R-2 and R-L) and WEEM4TS variants ( $W_w$ ,  $W_g$  and  $W_f$ ) scores of WETS and six baseline text summarization systems. We used  $W_w$ ,  $W_g$ , and  $W_f$  that stand for Word2Vec, GloVe and FastText based variants of WEEM4TS respectively. The best results are in bold.

	R-1	R-2	R-L	$W_w$	$W_g$	$W_f$
Lede-3	<b>24.30</b>	<b>11.10</b>	17.60	26.84	38.81	33.93
TextRank	22.50	8.50	16.60	28.22	41.24	35.95
Abs-N	6.70	0.20	5.60	11.28	21.09	18.10
Pointer-C	13.90	3.50	9.90	24.34	39.20	32.66
Pointer-S	18.80	7.20	13.80	26.91	40.80	34.88
Pointer-N	20.10	8.50	16.30	28.87	42.13	36.56
WETS	21.51	9.98	<b>19.56</b>	<b>34.20</b>	<b>44.89</b>	<b>39.61</b>

It is apparent from Table 4 that, out of nine comparisons, WETS performs seven times better than Lede-3. Similarly, from results in Table 5, we can see that WETS was best four times out of six comparisons, which was beaten by Lede-3 only two times in R-1 and R-2. Moreover, closer inspection of Table 5 shows that results of WETS, Lede-3, and TextRank are relatively closer to each other. This could be due to all of them are extractive text summarization systems.

#### 4.4.2. Correlation Results of Automatic Evaluation Metrics

In order to evaluate performance of automatic evaluation metrics for text summarization, we used shared file that consisted of original text; system summaries; and human evaluation results [23]. The task organizers disseminated the same copy of 60 summaries of seven systems to three assessors along with the original text via Amazon Mechanical Turk. Based on the provided original text, the assessors rated each summary out of five in four dimensions: coherence, fluency, informativeness, and relevance. For the evaluation of each dimension, the annotators were provided question that help them to judge the summaries effectively. For instance, to rate informativeness level of the summary, the assessors were told to use the prompt: How well does the summary capture the key points of the article? Similarly, the annotators rated relevance of the summary based on the prompt: Are the details provided by the summary consistent with details in the article? For the fluency evaluation, the annotators rated the summary based on this question: Are the individual sentence of the summary well-written and grammatical? Likewise, for coherence judgment, the annotators were told to evaluate system summary based on this prompt: Do phrases and sentences of the summary fit together and make sense collectively?

Thus, as three annotators were involved in the evaluation process, for a single summary, three human evaluation scores were collected with regard to coherence, fluency, informativeness, and relevance. This means 1260 human evaluation scores were collected for a total of 1260 summaries: 60 summaries multiplied by three annotators and again multiplied by seven systems. In order to use human evaluation scores effectively, we deduplicated similar article-summary pairs by computing average score of each perspective of three annotators, so that number of rows decreased to 420 for 7 systems. We again computed a mean score of coherence, fluency, informativeness, and relevance scores of each summary of all systems to correlate with the automatic evaluation metrics scores. Finally, we ended up with the average score of each 60 summaries for each system.

In order to conduct correlation analysis between automatic evaluation metrics and human judgments, we computed scores of different automatic evaluation metrics of 60 summaries for each system. We computed scores of WEEM4TS via three publicly available pre-trained word embedding models that had been trained based on Word2Vec, GloVe, and FastText algorithms and we denote variants of WEEM4TS as  $WEEM4TS_w$  ( $W_w$ ),  $WEEM4TS_g$  ( $W_g$ ), and  $WEEM4TS_f$  ( $W_f$ ) respectively.

Consequently, using Pearson’s correlation measurement ( $r$ ) [84], we computed the correlation between automatic evaluation metrics and human judgments, as in Equation (8), and report the results in Table 6.

$$r = \frac{\sum_{i=1}^n (H_i - \bar{H})(M_i - \bar{M})}{\sqrt{(H_i - \bar{H})^2} \sqrt{(M_i - \bar{M})^2}} \quad (8)$$

where  $H_i$  is the human assessment score of each system summary and  $M_i$  is the corresponding score as predicted by an automatic evaluation metric, in which  $\bar{H}$  and  $\bar{M}$  are their means respectively.

**Table 6.** Pearson’s ( $r$ ) correlation between automatic evaluation metrics scores and human judgments. Automatic evaluation metrics scores are computed based on reference summaries generated by WETS. The asterisk (\*) shows significance is at  $p \leq 0.05$ . The best score in that system is highlighted in bold.

Systems	Lede-3	Fragments	TextRank	Abs-N	Pointer-C	Pointer-S	Pointer-N
#Summaries	<b>60</b>	<b>60</b>	<b>60</b>	<b>60</b>	<b>60</b>	<b>60</b>	<b>60</b>
Correlation	r	r	r	r	r	r	r
ROUGE-1	0.232	0.559 *	0.217	0.115	0.023	−0.162	−0.038
ROUGE-2	0.210	0.513 *	0.230	−0.077	−0.005	−0.138	−0.025
ROUGE-L	0.216	0.543 *	0.215	0.099	0.044	−0.131	−0.049
BLEU-1	−0.234	0.104	0.086	<b>0.172</b>	−0.227	−0.101	−0.020
BLEU-2	−0.142	0.241	0.128	0.160	−0.241	−0.098	−0.007
BLEU-3	−0.132	0.304 *	0.156	0.154	−0.228	−0.098	0.004
BLEU-4	−0.128	0.337 *	0.172	0.151	−0.215	−0.098	0.011
CHRF1	0.266 *	0.630 *	0.258 *	0.018	0.077	−0.081	−0.015
CHRF2	0.280 *	0.633 *	0.263 *	0.020	0.089	−0.081	−0.008
CHRF3	0.284 *	0.633 *	0.264 *	0.020	0.093	−0.081	−0.006
WEEM4TS <sub>w</sub>	<b>0.348 *</b>	<b>0.643 *</b>	<b>0.288 *</b>	0.150	<b>0.123</b>	−0.021	0.014
WEEM4TS <sub>g</sub>	0.333 *	0.633 *	0.268 *	0.022	0.118	<b>0.002</b>	<b>0.027</b>
WEEM4TS <sub>f</sub>	0.331 *	0.631 *	0.267 *	−0.033	0.106	−0.029	0.018

Table 6 presents the Pearson’s correlation results between variants of four automatic evaluation metrics scores and human judgments on the summaries of seven text summarization systems. As can be seen from the table, no significant correlation was found between automatic evaluation metrics and human judgments on the summaries generated by the systems depicted in the last four columns of this table. Further, correlation results of ROUGE-1, ROUGE-2, ROUGE-L, and BLEU are not significant on the summaries generated by Lede-3 and TextRank. What is interesting about the results in this table is that all variants of our proposed evaluation metric (WEEM4TS) are performing relatively better than all other metrics on almost all system summaries. Moreover, closer inspection of the table shows that, among variants of WEEM4TS, WEEM4TS<sub>w</sub> performs the best in five out of seven comparisons, whereas WEEM4TS<sub>g</sub> is the best in the remaining two comparisons.

#### 4.4.3. Discussion

Very little was found in the literature on leveraging publicly available pre-trained word embedding models for text summarization and evaluation. We exploit linguistic regularities in publicly available pre-trained word embedding models for extractive summarization, which is the focus of our first research question (RQ1).

RQ1: For salient top- $n$  sentences determination, how can we leverage publicly available pre-trained word embedding models?

In order to answer RQ1, we developed a word embedding based text summarization (WETS) system. After the removal of stop words from the original text, all words of the first sentence and top- $n$  frequent words are considered as keywords. The highest cosine similarity value between the word and the keywords is regarded as the weight value of the target word. Subsequently, sentence relevance score is computed by summing up the obtained weight values of all words in the sentence

and divide by the updated length of the sentence. Finally, based on the relevance score of the sentences, we concatenated the most important sentences up to the required length. It should be noted that words that are not available in the word embedding vocabulary and have not exact matches are assigned a weight value of zero. The cumulative effect of heuristic rules considered in our text summarization system (WETS) is reported in Tables 4 and 5.

As can be seen from Table 4, WETS performs better than its counterpart Lede-3 in all cases except in R-1 and R-2 of the extractive summarization style. A possible explanation for this might be that perhaps WETS could determine the most important sentences in the original document than the first three sentences considered in Lede-3. In a given long paragraph, topic sentence (aka focus sentence) might reside in the middle or/and at the end [85], which Lede-3 might fail to catch it whereas WETS is capable to identify this sentence as a salient sentence via keywords. Furthermore, results shown in Table 5 demonstrate superiority of the WETS.

From Table 4 and 5, it is possible to conclude publicly available pre-trained word embedding models are vital for developing text summarization systems. The results further confirm capability of pre-trained word embeddings for text summarization task [9,41]. WETS based summaries can therefore be considered as reference summaries for the automatic evaluation of system summaries. We believe this is a valuable alternative to reference summaries generated by human expertise for automatic evaluation task.

If we now turn to results of automatic evaluation metrics, for comparison purpose, we used variants of ROUGE, BLEU, and CHRF metrics. ROUGE is most commonly used to evaluate text summarization systems whereas BLUE and CHRF are used for evaluating machine translation systems. These metrics, however, are not suitable to evaluate quality of abstractive and mixed summaries, which is the focus of our second research question (RQ2).

RQ2: Are publicly available pre-trained word embedding models useful for developing automatic evaluation metrics that are suitable to evaluate all kinds of system summaries?

Thus, in order to answer RQ2, we propose automatic evaluation metric for text summarization, referred to as WEEM4TS, and compare it with the metrics identified as baselines. In WEEM4TS, we settled some similarity measurement criteria. Accordingly, if a word appears in the system and reference summaries, it receives a + 1 score. If a certain word in the system summary does not appear in the reference summary but exists in the word embedding vocabulary, the highest cosine similarity value of the words in the reference summary is considered as a weight value of that word. If neither of these two things happen, a weight value of 0 is given to that word. These weight values are used to compute the modified unigram recall and the modified bigram precision, and linearly combine them with different importance level to obtain the final WEEM4TS score. Consequently, we conduct a correlation analysis between all automatic evaluation metrics scores including WEEM4TS and human judgments as presented in Table 6.

Intuitively, we get a general observation that: (1) a correlation of WEEM4TS variants with human judgments is relatively best in most cases when compared with other automatic evaluation metrics; (2) when we compare WEEM4TS variants, WEEM4TS<sub>w</sub> is best five times out of seven comparisons, whereas WEEM4TS<sub>g</sub> wins two times; (3) Scores of the WEEM4TS<sub>f</sub> variant are also very close to the scores of the other best metrics in that system; (4) the correlation results clearly demonstrate the superiority of our proposed automatic evaluation metric. A possible explanation for this might be that the considered heuristic rules are empowered by the prominent word embedding models that can help to uncover meaning between system and reference summaries.

Further, from Table 6, it is interesting to note that there is a consistent correlation among WEEM4TS variants. However, according to their performance order, WEEM4TS<sub>w</sub> is the first followed by WEEM4TS<sub>g</sub>, and WEEM4TS<sub>f</sub>. It seems that the vocabulary size of the word embedding models positively influence the results of WEEM4TS variants. The results further support the idea that word embedding models are capable to bring similar words nearby each other [73,74].



Moreover, we examine the relationship among the automatic evaluation metrics in two ways: First, we picked the best scores among variants of each metric and compare them as shown in Figure 2. Accordingly, the best score of WEEM4TS variants is the highest in almost all cases, followed by the best scores of CHRF and ROUGE variants. Secondly, we use a heat map to show an association between automatic evaluation metrics in terms of correlation results of all system summaries (Figure 3). As shown in the graph (Figure 3), all variants of WEEM4TS are positively correlated with all variants of CHRF and ROUGE metrics. On the other hand, although BLEU is reported as an on-par metric with ROUGE in [80], according to the results in Figure 3, it has a weak correlation with all metrics considered in our experiment.

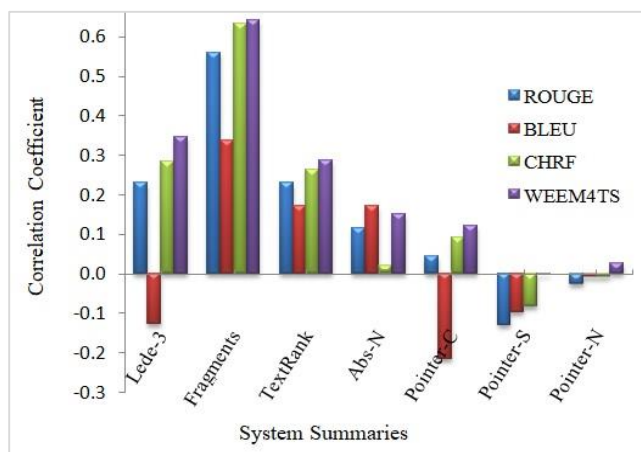


Figure 2. Performance comparison of individual best scores among variants of the metrics.

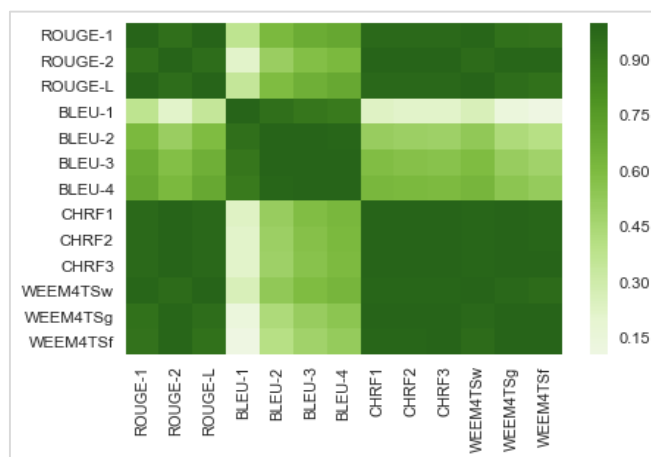


Figure 3. Pearson’s correlation heat map of automatic evaluation metrics. Deep green indicates strong positive correlation, light green is a medium whereas white is a weak correlation.

### 5. Conclusion and Future Work

With the emergence of smart phones and web technologies, the amount of text on the web is increasing enormously. As a result, text summarization has received more attention in several application domains. To satisfy the demand, several text summarization methods have been proposed. However, meaning oriented text summarization is a challenging task. This paper explores ways of using word embedding based sentence relevance scores for ranking top-n sentences as a summary, which can be used as a ground truth. Another aim of this study was to examine semantic based automatic evaluation metric for evaluating quality of systems summaries.

Our extensive experimental studies on different data configurations confirm our proposed text summarization (WETS) and automatic evaluation metric (WEEM4TS) can achieve a significant

performance when compared to baseline methods. However, we aware that our research may have three limitations: the first is as the focus of our proposed text summarization approach was on concatenating top-n sentences, there is a possibility that pronouns can refer back to incorrect nouns in the generated summary. Secondly, being limited to words of the first sentence and frequent words, the study did not explicitly evaluate the use of words in the middle or/and at the last sentence/s as keywords. In the future, alternative identification of relevant keywords and addressing issue of anaphora in text summarization would call for more research. Thirdly, although our proposed text summarization and automatic evaluation metric constitute promising meaning-oriented approaches, the performance of these methods was not evaluated on languages other than English. Thus, further research could be conducted to determine the effectiveness of the proposed methods on various languages.

**Author Contributions:** Conceptualization, T.T.H.; Data curation, T.T.H.; Formal analysis, T.T.H.; Funding acquisition, J.Y.; Investigation, T.T.H. and J.Y.; Methodology, T.T.H.; Resources, T.T.H., J.Y. and T.G.F.; Software, T.T.H.; Supervision, J.Y.; Validation, T.T.H.; Visualization, T.T.H.; Writing—original draft, T.T.H.; Writing—review & editing, T.T.H., J.Y. and T.G.F. All authors have read and agreed to the published version of the manuscript.

**Funding:** The APC was funded by National Natural Science foundation of China under grant numbers of 61572211, 61173114, and 61202300.

**Acknowledgments:** Thanks for Max Grusky et al. for making dataset publicly available. We are grateful to the editors and the reviewers for their time and constructive comments on our manuscript.

**Conflicts of Interest:** The authors declare no conflict of interest.

## Appendix A Demonstration of how word embedding models are used in both WETS and WEEM4TS

Assume we are given an original text to generate WETS summary that consisted of three sentences and evaluate it against the pre-determined reference text. Here, we would like to show the procedures of doing so in two phases: first generate WETS summary, and then evaluate the generated summary against reference text.

### Phase 1: Perform the following activities for generating WETS summary.

1. For the demonstration purpose, we used an original text and a reference text shown in the following box.

**Original Text:** A person who loves someone is surely loved in turn by the others {0}. Researchers show that the more a person loves people around him/her the better healthy life he/she has {1}. People who love others without any condition are mostly lead happy life {2}. Contrary, there are people who are ignorant and get satisfaction by hurting others {3}. Some of them develop this behavior from their childhood {4}. Adoring others will give you immense happiness and peace {5}.

**Reference Text:** A person who loves someone is unquestionably loved successively by the others. In fact, there are also some people who get satisfaction by hurting others. Adoring others will provide you with immense happiness and peace.

2. Using full stop as a delimiter, we split an original text into six sentences. We refer each sentence using index value enclosed by the green curly bracket.
3. Use nonstop words of the first sentence as keywords. Accordingly, 'person', 'loves', 'someone', 'surely', 'loved', 'turn', and 'others' are all considered as keywords.
4. Update the keywords identified in step 3 by adding relevant frequent words. In this particular example, all of the frequent words are also found in the first sentence except the word 'people' that occurs two times in the original text. So, we updated the keywords by adding the word 'people' to the list.
5. In order to compute relevance score of each sentence, we sum weight values of all words in that sentence and divide by the number of words in that sentence, Equation (1). Obviously, first sentence is favored to be a first top salient sentence. Before calculating relevance score of other

sentences, weigh value of each word in that sentence is determined based on the following rules: If the word in that sentence also exists in the first sentence, we temporarily remove that word rather than assigning a weight value to it. This is to discourage redundancy. If the word does not exist in the first sentence but does in the keywords, assign a weight value equal to +1. If the word does not exist in the keywords but does in the vocabulary of word embedding model then compute cosine similarity between the word and all words in the keywords to consider the highest cosine similarity value as a weight value of that word. For instance, the first word of the second sentence {1}, 'Researchers', exists neither in the first sentence nor in the list of keywords. Hence, to determine weight value for the word 'Researchers', we compute cosine similarity between 'Researchers' and all words in the keywords via Word2Vec in which the word 'people' is found the most similar word with the highest cosine similarity value equal to 0.104. Following the same process, we assign weight value for all words, which consequently used to compute relevance score. Based on the obtained relevance score, from the highest to lowest, the following sentences are identified as salient top-3, in order: {0}, {5}, and {3}. Accordingly, **WETS summary is:** *A person who loves someone is surely loved in turn by the others. Adoring others will give you immense happiness and peace. Contrary, there are people who are ignorant and get satisfaction by hurting others.*

### Phase 2: Evaluating WETS summary by using WEEM4TS.

1. Assume we are required to evaluate system generated summary, in this case WETS and Lede-3 summaries, against human generated reference summary.
2. We use WEEM4TS to evaluate both WETS and Lede-3 summaries against human generated reference summary. As described in Section 3.2, WEEM4TS score is calculated by a linear combination of modified unigram recall (Equation (2)) and modified bigram precision (Equation (3)). In modified unigram recall, each word in the WETS summary is assigned by the highest cosine similarity value among the words in the reference summary. Then sum of these values is divided by the number of words in the reference summary. In the modified bigram precision, counting the number of bigrams matches and then divided by the number of bigrams in the system summary. It should be noted that the matching is not only on based on the surface form but also on semantic similarity. This makes the standard recall and precision different from the recall and precision employed in this study, which we use the term modified to designate the differences. Accordingly, we compute WEEM4TS variants score for WETS summary and Lede-3 summary:

**Reference Text:** A person who loves someone is unquestionably loved successively by the others. In fact, there are also some people who get satisfaction by hurting others. Adoring others will provide you with immense happiness and peace.

**Lede-3 summary:** A person who loves someone is surely loved in turn by the others. Researchers show that the more a person loves people around him/her the better healthy life he/she has. People who love others without any condition are mostly lead happy life. WEEM4TS score: [WEEM4TS<sub>w</sub> = 53.81, WEEM4TS<sub>g</sub> = 67.08, and WEEM4TS<sub>f</sub> = 58.72]

**WETS summary:** A person who loves someone is surely loved in turn by the others. Adoring others will give you immense happiness and peace. Contrary, there are people who are ignorant and get satisfaction by hurting others.

WEEM4TS score: [WEEM4TS<sub>w</sub> = 84.19, WEEM4TS<sub>g</sub> = 85.03, and WEEM4TS<sub>f</sub> = 84.70]

### References

1. Rush, A.M.; Chopra, S.; Weston, J. A Neural Attention Model for Abstractive Sentence Summarization. *arXiv* **2015**, arXiv:1509.00685.
2. Torres-Moreno, J.-M. *Automatic Text Summarization*; ISTE Ltd. and John Wiley & Sons: London, UK; Hoboken, NJ, USA, 2014; pp. 1–320.

3. Saggion, H.; Poibeau, T. Automatic text summarization: Past, present and future. In *Multi-Source, Multilingual Information Extraction and Summarization*; Springer: Berlin/Heidelberg, Germany, 2013; pp. 3–21.
4. Kumar, Y.J.; Goh, O.S.; Basiron, H.; Choon, N.H.; C Suppiah, P. A review on automatic text summarization approaches. *J. Comput. Sci.* **2016**, *12*, 178–190. [[CrossRef](#)]
5. See, A.; Liu, P.J.; Manning, C.D. Get To The Point: Summarization with Pointer-Generator Networks. *arXiv* **2017**, arXiv:1704.04368.
6. Luhn, H.P. The Automatic Creation of Literature Abstracts \*. *IBM J.* **1958**, *2*, 159–165. [[CrossRef](#)]
7. Edmundson, H.P. New Methods in Automatic Extracting. *J. ACM.* **1969**, *16*, 264–285. [[CrossRef](#)]
8. Barrios, F.; Federico, L.; Argerich, L.; Wachenchauser, R. Variations of the Similarity Function of TextRank for Automated Summarization. *arXiv* **2016**, arXiv:1602.03606.
9. Rossiello, G.; Basile, P.; Semeraro, G. Centroid-based Text Summarization through Compositionality of Word Embeddings. In Proceedings of the MultiLing 2017 Workshop on Summarization and Summary Evaluation Across Source Types and Genres, Valencia, Spain, 3 April 2017; pp. 12–21.
10. Wu, Y.; Hu, B. Learning to Extract Coherent Summary via Deep Reinforcement Learning. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018; pp. 5602–5609.
11. Jadhav, A.; Rajan, V. Extractive Summarization with SWAP-NET: Sentences and Words from Alternating Pointer Networks. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, Melbourne, Australia, 15–20 July 2018; pp. 142–151.
12. Nallapati, R.; Zhai, F.; Zhou, B. SummaRuNNer: A Recurrent Neural Network based Sequence Model for Extractive Summarization of Documents. In Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, San Francisco, CA, USA, 4–9 February 2017; pp. 3075–3081.
13. Chen, Y.; Bansal, M. Fast Abstractive Summarization with Reinforce-Selected Sentence Rewriting. *arXiv* **2018**, arXiv:1805.11080.
14. Zhang, Y.; Li, D.; Wang, Y.; Fang, Y. Abstract Text Summarization with a Convolutional Seq2seq Model. *Appl. Sci.* **2019**, *9*, 1665. [[CrossRef](#)]
15. Bae, S.; Kim, T.; Kim, J.; Lee, S. Summary Level Training of Sentence Rewriting for Abstractive Summarization. *arXiv* **2019**, arXiv:1909.08752.
16. Li, C.; Xu, W.; Li, S.; Gao, S. Guiding Generation for Abstractive Text Summarization based on Key Information Guide Network. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, New Orleans, LA, USA, 1–6 June 2018; pp. 55–60.
17. Hsu, W.; Lin, C.; Lee, M.; Min, K.; Tang, J.; Sun, M. A Unified Model for Extractive and Abstractive Summarization using Inconsistency Loss. *arXiv* **2018**, arXiv:1805.06266.
18. Erkan, G.; Radev, D.R. LexRank: Graph-based Lexical Centrality as Saliency in Text Summarization. *J. Artif. Intell. Res.* **2004**, *22*, 457–479. [[CrossRef](#)]
19. Steinberger, J.; Ježek, K. Evaluation Measures for Text Summarization. *Comput. Inform.* **2009**, *28*, 1001–1025.
20. Rath, G.; Resnick, A.; Savage, T. The Formation of Abstracts By the Selection of Sentences. Part 1. Sentence Selection By Men and Machines. *Am. Doc.* **1961**, *12*, 139–141. [[CrossRef](#)]
21. Berg-Kirkpatrick, T.; Gillick, D.; Klein, D. Jointly Learning to Extract and Compress. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Portland, OR, USA, 19–24 June 2011; pp. 481–490.
22. Knight, K.; Marcu, D. Statistics-Based Summarization—Step One: Sentence Compression. *AAAI/IAAI* **2000**, *2000*, 703–710.
23. Grusky, M.; Naaman, M.; Artz, Y. Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies. *arXiv* **2018**, arXiv:1804.11283.
24. Radev, D.R.; Blair-Goldensohn, S.; Zhang, Z. Experiments in Single and Multi-Document Summarization Using MEAD. In Proceedings of the First Document Understanding Conference, New Orleans, LA, USA, 13–14 September 2001.
25. Luhn, H.P. A Statistical Approach to Mechanized Encoding and Searching of Literary Information. *IBM J. Res. Dev.* **1957**, *1*, 309–317. [[CrossRef](#)]
26. Jones, K.S. A statistical interpretation of term specificity and its application in retrieval. *J. Doc.* **1972**, *28*, 11–21. [[CrossRef](#)]

27. Salton, G.; Buckley, C. Term-weighting approaches in automatic text retrieval. *Inf. Process. Manag.* **1988**, *24*, 513–523. [[CrossRef](#)]
28. Kyoomarsi, F.; Khosravi, H.; Eslami, E.; Dehkordy, P.K.; Tajoddin, A. Optimizing Text Summarization Based on Fuzzy Logic. In *Seventh IEEE/ACIS International Conference on Computer and Information Science (Icis 2008)*; IEEE: Piscataway, NJ, USA, 2008; pp. 347–352.
29. Villatoro-Tello, E.; Villaseñor-Pineda, L.; Montes-y-Gómez, M. Using Word Sequences for Text Summarization. In *Proceedings of the International Conference on Text, Speech and Dialogue, Brno, Czech Republic, 11–15 September 2006*; pp. 293–294.
30. René Arnulfo, G.; Montiel, R.; Ledeneva, Y.; Rendón, E.; Gelbukh, A.; Cruz, R. Text Summarization by Sentence Extraction Using Unsupervised Learning \*. In *Proceedings of the Mexican International Conference on Artificial Intelligence, Atizapán de Zaragoza, Mexico, 27–31 October 2008*; pp. 133–143.
31. Fattah, M.A.; Ren, F. Automatic Text Summarization. *Int. J. Comput. Inf. Eng.* **2008**, *2*, 90–93.
32. Witbrock, M.J.; Mittal, V.O. Ultra-Summarization: A Statistical Approach to Generating Highly Condensed Non-Extractive Summaries. In *SIGIR 1999*, *9*, 1–14.
33. Zajic, D.; Dorr, B.J.; Lin, J.; Schwartz, R. Multi-candidate reduction: Sentence compression as a tool for document summarization tasks. *Inf. Process. Manag.* **2007**, *43*, 1549–1570. [[CrossRef](#)]
34. Jing, H.; McKeown, K.R. Cut and Paste Based Text Summarization. In *Proceedings of the 1st North American Chapter of the Association for Computational Linguistics Conference, Seattle, WA, USA, 29 April–4 May 2000*; pp. 178–185.
35. Mohd, M.; Jan, R.; Shah, M. Text document summarization using word embedding. *Expert Syst. Appl.* **2020**, *143*, 112958. [[CrossRef](#)]
36. Al-Sabahi, K.; Zuping, Z.; Kang, Y. Latent Semantic Analysis Approach for Document Summarization Based on Word Embeddings. *arXiv* **2018**, arXiv:1807.02748.
37. Nema, P.; Khapra, M.; Laha, A.; Ravindran, B. Diversity driven Attention Model for Query-based Abstractive Summarization. *arXiv* **2018**, arXiv:1704.08300.
38. Nallapati, R.; Zhou, B.; Gulcehre, C.; Xiang, B. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv* **2016**, arXiv:1602.06023.
39. Xiang, X.; Xu, G.; Fu, X.; Wei, Y.; Jin, L.; Wang, L. Skeleton to Abstraction: An Attentive Information Extraction Schema for Enhancing the Saliency of Text Summarization. *Inf.* **2018**, *9*, 217. [[CrossRef](#)]
40. Rush, A.M.; Chopra, S.; Weston, J. A Neural Attention Model for Abstractive Sentence Summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, 17–21 September 2015*.
41. Al-Sabahi, K.; Zuping, Z.; Nadher, M. A Hierarchical Structured Self-Attentive Model for Extractive Document Summarization (HSSAS). *IEEE Access* **2018**, *6*, 24205–24212. [[CrossRef](#)]
42. Yang, K.; Al-Sabahi, K.; Xiang, Y.; Zhang, Z. An Integrated Graph Model for Document Summarization. *Information* **2018**, *9*, 232. [[CrossRef](#)]
43. Ganesan, K.; Zhai, C.; Han, J. Opinosis: A graph based approach to abstractive summarization of highly redundant opinions. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), Beijing, China, 23–27 August 2010*; pp. 340–348.
44. Jing, H.; Mckeown, K.R. The Decomposition of Human-Written Summary Sentences. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Berkeley, CA, USA, 15–19 August 1999*; pp. 129–136.
45. Saggion, H. A classification algorithm for predicting the structure of summaries. In *Proceedings of the 2009 Workshop on Language Generation and Summarisation, Suntec, Singapore, 6 August 2009*; pp. 31–38.
46. Saggion, H. Learning Predicate Insertion Rules for Document Abstracting. In *International Conference on Intelligent Text Processing and Computational Linguistics*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 301–302.
47. Hou, Y.; Xiang, Y.; Tang, B.; Chen, Q.; Wang, X.; Zhu, F. Identifying High Quality Document–Summary Pairs through Text Matching. *Information* **2017**, *8*, 64.
48. Hu, B.; Chen, Q.; Zhu, F. LCSTS: A Large Scale Chinese Short Text Summarization Dataset. *arXiv* **2015**, arXiv:1506.05865.

49. Hermann, K.M.; Ko, T.; Grefenstette, E.; Espeholt, L.; Kay, W.; Suleyman, M.; Blunsom, P. Teaching Machines to Read and Comprehend. In Proceedings of the Advances in Neural Information Processing Systems, Montréal, QC, Canada, 11–15 December 2015; pp. 1693–1701.
50. Over, P.; Dang, H.; Harman, D. DUC in Context. *Inf. Process. Manag.* **2007**, *43*, 1506–1520. [[CrossRef](#)]
51. Chopra, S.; Auli, M.; Rush, A.M. Abstractive Sentence Summarization with Attentive Recurrent Neural Networks. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego, CA, USA, 12–17 June 2016.
52. Völske, M.; Potthast, M.; Syed, S.; Stein, B. TL; DR: Mining Reddit to Learn Automatic Summarization. In Proceedings of the Workshop on New Frontiers in Summarization, Copenhagen, Denmark, 7 September 2017; pp. 59–63.
53. Filippova, K.; Altun, Y. Overcoming the Lack of Parallel Data in Sentence Compression. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, Seattle, WA, USA, 18–21 October 2013; pp. 1481–1491.
54. Guo, Q.; Huang, J.; Xiong, N.; Wang, P. MS-Pointer Network: Abstractive Text Summary Based on Multi-Head Self-Attention. *IEEE Access.* **2019**, *7*, 138603–138613. [[CrossRef](#)]
55. Jones, K.S.; Galliers, J.R. Evaluating Natural Language Processing Systems: An Analysis and Review. *Comput. Linguist.* **1996**, *24*, 336–338.
56. Paice, C.D. Constructing literature abstracts by computer: techniques and prospects. *Inf. Process. Manag.* **1990**, *26*, 171–186. [[CrossRef](#)]
57. Radev, D.R.; Jing, H.; Budzikowska, M. Centroid-based summarization of multiple documents. *Inf. Proc. Manag.* **2004**, *40*, 919–938.
58. Mani, I.; House, D.; Klein, G.; Hirschman, L.; Firmin, T.; Sundheim, B.M. The TIPSTER SUMMAC text summarization evaluation. In Proceedings of the Ninth Conference of the European Chapter of the Association for Computational Linguistics, Bergen, Norway, 8–12 June 1999.
59. Hynek, J.; Jezek, K. Practical Approach to Automatic Text Summarization. In Proceedings of the 7th ICCCI/IFIP International Conference on Electronic Publishing, Minho, Portugal, 25–28 June 2003.
60. Radev, D.; Teufel, S.; Saggion, H.; Lam, W. Evaluation challenges in large-scale document summarization. In Proceedings of the 41st Annual Meeting on Association for Computational Linguistics, Sapporo, Japan, 7–12 July 2003; pp. 375–382.
61. Morris, A.H.; George, M.; Kasper, D.A.A. The Effects and Limitations of Automated Text Condensing on Reading Comprehension Performance. *Inf. Syst. Res.* **1992**, *3*, 17–35.
62. Lin, C. ROUGE: A Package for Automatic Evaluation of Summaries. In Proceedings of the Text Summarization Branches Out, Barcelona, Spain, 25–26 July 2004.
63. Lin, C.Y. Looking for a few good metrics: ROUGE and its evaluation. In Proceedings of the NTCIR Workshop, Tokyo, Japan, 2–4 June 2004.
64. Minel, J.-L.; Nugler, S.; Plat, G. How to appreciate the quality of automatic text summarization? Examples of FAN and MLUCE protocols and their results on SERAPHIN. In Proceedings of the Intelligent Scalable Text Summarization, Madrid, Spain, 11 July 1997.
65. Nenkova, A.; Passonneau, R. Evaluating Content Selection in Summarization: The Pyramid Method. In Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics, Boston, MA, USA, 2–7 May 2004.
66. Mani, I.; Klein, G.; House, D.; Hirschman, L.; Firmin, T.; Sundheim, B. SUMMAC: A text summarization evaluation. *Nat. Lang. Eng.* **2002**, *8*, 43–68. [[CrossRef](#)]
67. Jing, H.; Barzilay, R.; McKeow, K.; Elhadad, M. Summarization Evaluation Methods: Experiments and Analysis. In Proceedings of the AAAI Symposium on Intelligent Summarization, Madison, WI, USA, 26–30 July 1998.
68. Kennedy, A.; Szapkowicz, S. Evaluation of a Sentence Ranker for Text Summarization Based on Roget’s Thesaurus. *Int. Conf. Text, Speech Dialogue.* **2010**, *6231*, 101–108.
69. MILLER, U. Thesaurus construction: problems and their roots. *Inf. Process. Manag.* **1997**, *33*, 481–493. [[CrossRef](#)]
70. Ng, J.; Abrech, V. Better Summarization Evaluation with Word Embeddings for ROUGE. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, 17–21 September 2015.

71. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient Estimation of Word Representations in Vector Space. *arXiv* **2013**, arXiv:1301.3781.
72. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Distributed Representations of Words and Phrases and their Compositionality. In Proceedings of the Advances in Neural Information Processing Systems, Lake Tahoe, NV, USA, 5–10 December 2013.
73. Mikolov, T.; Yih, W.; Zweig, G. Linguistic Regularities in Continuous Space Word Representations. In Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Atlanta, GA, USA, 9–14 June 2013.
74. Bojanowski, P.; Grave, E.; Joulin, A.; Mikolov, T. Enriching Word Vectors with Subword Information. *arXiv* **2017**, arXiv:1607.04606. [[CrossRef](#)]
75. Pennington, J.; Socher, R.; Manning, C.D.. GloVe: Global Vectors for Word Representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014.
76. Husin, M.S.; Ariffin, K. The Rhetorical Organisation of English Argumentative Essays by Malay ESL Students: The Placement of Thesis Statement. *J. Asia TEFL*. **2012**, *9*, 147–169.
77. Vinyals, O.; Fortunato, M.; Jaitly, N. Pointer Networks. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 7–12 December 2015.
78. Tu, Z.; Lu, Z.; Liu, Y.; Liu, X. Modeling Coverage for Neural Machine Translation. *arXiv* **2016**, arXiv:1601.04811.
79. Papineni, K.; Roukos, S.; Ward, T.; Zhu, W.-J. BLEU: A Method for Automatic Evaluation of Machine Translation. In Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, Philadelphia, PA, USA, 7–12 July 2002.
80. Graham, Y. Re-evaluating Automatic Summarization with BLEU and 192 Shades of ROUGE. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, 17–21 September 2015.
81. Popović, M. CHRF: character n-gram F-score for automatic MT evaluation. In Proceedings of the Tenth Workshop on Statistical Machine Translation, Lisbon, Portugal, 17–18 September 2015.
82. Bojar, O.; Graham, Y.; Kamran, A.; Stanojević, M. Results of the WMT16 Metrics Shared Task. In Proceedings of the First Conference on Machine Translation, Berlin, Germany, 11–12 August 2016.
83. Bojar, O.; Graham, Y.; Kamran, A. Results of the WMT17 Metrics Shared Task. In Proceedings of the Second Conference on Machine Translation, Volume 2: Shared Task Papers, Copenhagen, Denmark, 7–11 September 2017.
84. Pearson, K. Note on regression and inheritance in the case of two parents. In Proceedings of the Royal Society of London, London, UK, 31 December 1895.
85. Villanueva, V. *Cross-Talk in Comp Theory: A Reader*, 2nd ed.; Revised and Updated; ERIC: Kern, CA, USA, 2003; p. 873.

