Proceedings of the 13th Conference on Language Resources and Evaluation (LREC 2022), pages 6009–6017
Marseille, 20-25 June 2022

Sentence Pair Embeddings Based Evaluation Metric for Abstractive and Extractive Summarization

Ramya Akula, Ivan Garibay
University of Central Florida
Orlando, Florida, U.S.A
ramya.akula@knights.ucf.edu, igaribay@ucf.edu

Abstract

The development of an automatic evaluation metric remains an open problem in text generation. Widely used evaluation metrics, like ROUGE and BLEU, are based on exact word matching and fail to capture semantic similarity. Recent works, such as BERTScore, MoverScore and, Sentence Mover's Similarity, are an improvement over these standard metrics as they use contextualized word or sentence embeddings to capture semantic similarity. We in this work, propose a novel evaluation metric, Sentence Pair EmbEDdings (*SPEED*) Score, for text generation which is based on semantic similarity between sentence pairs as opposed to earlier approaches. To find semantic similarity between a pair of sentences, we obtain sentence-level embeddings from multiple transformer models pre-trained specifically on various sentence pair tasks such as Paraphrase Detection (PD), Semantic Text Similarity (STS), and Natural Language Inference (NLI). As these sentence pair tasks involve capturing the semantic similarity between a pair of input texts, we leverage these models in our metric computation. Our proposed evaluation metric shows impressive performance in evaluating both abstractive and extractive summarization models and achieves state-of-the-art results on the SummEval dataset, demonstrating the effectiveness of our approach. Also, we perform the run-time analysis to show that our proposed metric is faster than the current state-of-the-art.

Keywords: Evaluation Metric, Abstractive Summarization, Extractive Summarization, Semantic Similarity

1. Introduction

Developing an efficient automatic evaluation metrics for text summarization will have a two-fold impact on the advancement of algorithms for text generation systems. First, it eliminates the need for time-consuming and expensive human evaluations, second, it helps for a better comparison and assessment of the developed systems. With this goal, various evaluation metrics have been proposed in the literature to automatically evaluate summarization methods. Widely used metrics: ROUGE (Lin, 2004) and BLEU (Papineni et al., 2002), rely on word/token overlap between reference and system summary.

Recent works such as BERTScore (Zhang et al., 2019), MoverScore (Zhao et al., 2019) are improvements over earlier methods as they use word/token embeddings from pre-trained language models instead of using exact word/token matches. However, a drawback of these metrics is that they do not capture the performance of the models in terms of semantic consistency of the generated summary to its reference (Clark et al., 2019). This shortcoming becomes more prominent as the length of the summaries increase. To overcome these issues and to better evaluate summarization models, there is a need for a metric that can find semantic similarity between the ground truth and system summary at a level beyond the individual words/tokens.

In this work, we propose the idea of using sentence-pair embeddings to compare multi-sentence summaries for automatic evaluation of text summarization models. To extract the sentence-pair embeddings, we propose the use of existing deep-learning models trained on various tasks which involve learning semantic similarity between sentences. More concretely, we present Sentence Pair EmbEDding (*SPEED*) Score, an evaluation metric which uses Transformer (Vaswani et al., 2017) models, trained on sentence-pair tasks such as Paraphrase Detection, Semantic Text Similarity (STS), Natural Language Inference (NLI), and Passage Ranking. We hypothesize that, embeddings from these models better capture the semantic similarities between sentences and thus aid in improving the performance of the evaluation metric.

To validate this hypothesis, we propose a simple evaluation metric that uses models trained on these sentence-pair tasks to evaluate generated summaries. To compute our metric, we used publicly available transformer models, pre-trained on sentence-pair tasks. Using our proposed approach for computing the evaluation metric, we achieve comparable results with the state-of-the-art metrics on the recent SummEval dataset by (Fabbri et al., 2021). Main contributions of our work are:

- We propose a novel and simple evaluation metric for text summarization, based only on sentence embeddings.
- We propose the use of Transformer models, trained on sentence-pair tasks to extract sentence embeddings our metric computation.
- We compare our proposed metric with current state-of-the-art on the CNN/DailyMail dataset and show improved results.

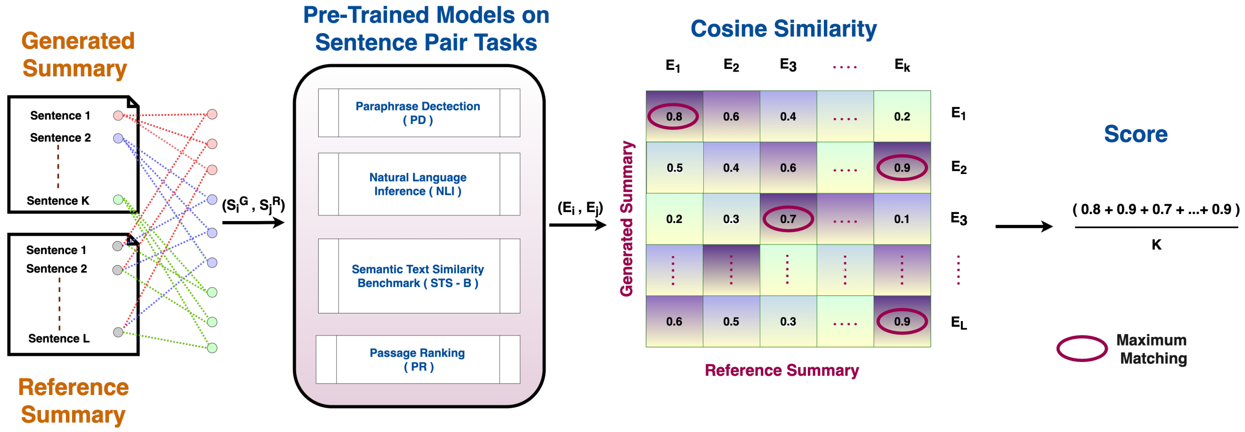

Generated Summary
Sentence 1
Sentence 2
Sentence K
Sentence 1
Sentence 2
Sentence L
Reference Summary
(S_i^G, S_j^R)
Pre-Trained Models on Sentence Pair Tasks
Paraphrase Dectection (PD)
Natural Language Inference (NLI)
Semantic Text Similarity Benchmark (STS - B)
Passage Ranking (PR)
(E_i, E_j)
Cosine Similarity
E_1 E_2 E_3 .... E_k
0.8 0.6 0.4 .... 0.2 E_1
0.5 0.4 0.6 .... 0.9 E_2
0.2 0.3 0.7 .... 0.1 E_3
0.6 0.5 0.3 .... 0.9 E_L
Generated Summary
Reference Summary
Score
(0.8 + 0.9 + 0.7 + ...+ 0.9)
K
Maximum Matching

Figure 1: Overview of steps involved in computation of our proposed *SPEED-Score* metric: (i) Our metric employs multiple models trained on different sentence-pair tasks to extract sentence embeddings for sentences. (ii) Cosine similarity is computed for every pair of sentences from the system generated and reference summaries. (iii) Sentences in the generated summary are mapped to the sentences in the reference summary with highest similarity score. (iv) Final metric is the average of similarity scores for all the sentences in the system generated summary.

2. Related Work

In this section, we provide a brief overview of existing automatic evaluation metrics for text summarization. Widely used evaluation metrics for summarization are BLEU (Papineni et al., 2002), ROUGE(Lin, 2004), and METEOR(Lavie and Agarwal, 2007), which measures the extent of lexical overlap between ground truth and system-generated summaries. These metrics are mainly developed for the task of machine translation and are based on stemming, matching synonyms, and weighted sub-sequences. Recent evaluation metrics such as S^3(Peyrard et al., 2017), BERTScore (Zhang et al., 2019), MoverScore (Zhao et al., 2019), and SMS (Clark et al., 2019) are deep-learning based and use word embeddings to compute token similarity, instead of using exact matches like in n-gram based metrics.

In regards to evaluating text quality, standard metrics, BLUE (Papineni et al., 2002) and ROUGE (Lin, 2004), can neither capture contextual information nor consider the semantic similarity between system generated summary and the reference summary. These metrics also fail in presence of word re-ordering or paraphrasing (Kilickaya et al., 2017). Recent word embedding based metrics, like BERTScore (Zhang et al., 2019), MoverScore (Zhao et al., 2019), and S^3 (Peyrard et al., 2017) use embeddings from language models to capture contextual information but still suffer from the critical issue of capturing factual correctness and faithfulness (Maynez et al., 2020) of system generated text towards reference text. All the metrics discussed above operate with words or word representations and fail to capture similarity between semantic concepts expressed using a group of words or phrases. This problem is alleviated as the length of the summaries being compared increases. We in our proposed metric overcome this drawback by using sentence embeddings which are higher level representations of text.

Recent work, Sentence Mover's Similarity (SMS) (Clark et al., 2019) uses sentence level embeddings along with word embedding and is a based on Word Mover's Distance (Kusner et al., 2015). In (Gao et al., 2020), an evaluation metric is proposed for multi-document summarization where pre-trained language models are used to extract sentence embeddings for computing the similarity between summaries. In both the works mentioned above, sentence embeddings are extracted from language models trained on individual sentences. These embeddings fail capture semantic relationship between sentence pairs. We in our metric computation, use pre-trained transformer models trained on sentence-pair tasks to extract sentence embedding which capture the semantic similarity.

3. Proposed Approach

In this section, we present our proposed metric for evaluating the similarity between the summary generated by a text summarization system and the reference summary provided by a human. In our metric computation, we employ sentence pair embeddings, instead of sentence/word embeddings or token matching. Given a system generated summary $G = \{S_i^G | 1 \leq i \leq K\}$ with a set of sentences S$_i^G$ and a reference summary $R = \{S_j^R | 1 \leq j \leq L\}$ with a set of sentences S$_j^R$, we : (i) Use sentence embeddings from pre-trained sentence-pair models to represent each sentence pair (S$_i^G$, S$_j^R$) in G and R. (ii) Compute semantic similarity between each pair of sentences from G and R using Cosine distance between sentence embeddings. (iii) Match each sentence in the generated summary to a sentence in reference summary with maximum similarity. (iv) Compute average score across all the sentences in the generated summary as the semantic similarity score between G and R; these steps are also illustrated in Figure 1.

3.1. Sentence Embeddings

To represent sentences in a summary, we extract embeddings for each sentence from a transformer model trained on a sentence-pair task. As the sentence-pair tasks involve finding semantic similarity between a pair of sentences, we leverage models trained on these tasks to extract embeddings for our metric computation. These models, take a pairs of sentences as input and provide D-dimensional embeddings for each sentence. As the goal of our metric is to compute semantic similarity between system generated summary and the reference summary, we pass the sentence pairs (S$_i^G$, S$_j^R$), where S$_i^G \in G$ and S$_j^R \in R$ to extract the sentence embeddings (E$_i$, E$_j$). As there are multiple pre-trained models available on these sentence-pair tasks, we use the sentence embeddings (E_i^n, E_j^n), from each of the $N \in [1, 2, ..., n]$ models in the computation.

3.2. Pre-train Tasks

For extracting sentence embeddings, we use the following sentence-pair tasks.

- **Paraphrase Detection**: This task involves detecting if a pair of sentences are paraphrases of each other i.e., if they both have the same semantic meaning. As this sentence-pair task requires finding semantic similarity between a pair of sentences, it is an ideal task for our metric computation.
- **Semantic Text Similarity:** STS task deals with determining similarity between a pair of texts. This task involves assigning a score between 0 to 5, denoting the level of similarity/entailment.
- **Natural Language Inference:** Given two sentences, *Premise* and *Hypothesis*, the task is to identify if the *Premise* agrees/contradicts/is-neutral to the *Hypothesis*.
- **Passage Ranking:** This task involves ranking a set of passages in order of relevance to a given query. As each passage contain multiple sentences, this task requires identifying semantic similarity between the sentences in the passage with the query sentence.

All the pre-trained models used in our metric computation are implemented in PyTorch (Paszke et al., 2017) and are publicly available for download[1] (Reimers and Gurevych, 2019).

3.3. Similarity Score

Representing each sentence as an embedding allows for computing a soft similarity score rather than exact token matching or heuristic matching. In our metric computation, we use cosine distance between the embeddings to compute the similarity between two sentences. Given $S_i^G, i \in [1, K]$ and $S_j^R, j \in [1, L]$ are sentences from G and R respectively, the semantic similarity (Sim_{ij}) between S_i^G and S_j^R is given by Equation 1. Here, E_i and E_j are D-dimensional embeddings for sentences S_i^G and S_j^R from one of the N pre-trained models, trained on different sentence-pair tasks.

$$Sim_{ij} = \frac{1}{N} \times \sum_{n=1}^{N} 1 - CosineDistance(E_i^n, E_j^n) \tag{1}$$

3.4. Metric Computation

To compute the semantic similarity between two summaries (G and R), we consider the similarity between all sentence pairs and match each sentence in G to a sentence in R. We use a greedy matching approach, where a sentence in G is matched to a sentence in R with highest similarity score. The final metric is the average of the similarity scores for all the sentences in G. We call this metric, *SPEED-Score*, as the models used in this computation are transformer models trained on sentence-pair tasks.

4. Experiments

4.1. Datasets

Text Summarization To evaluate our proposed metric at the text summarization task, we use the SummEval dataset (**?**). It consists of 1600 summaries generated by 16 state-of-the-art text summarization models for 100 news articles from CNN/Daily Mail dataset(Hermann et al., 2015). Each of these summaries is annotated by 3 independent experts and 5 independent crowd-sourced Amazon Turkers, amounting to a total of 12,800 summary level annotations. Each annotation is a score between 0 and 5, rating the quality of the generated summary along four dimensions as in (Kryściński et al., 2020). Inter-annotator agreement for the collected crowd-sourced and expert annotations is evaluated by computing the Krippendorff's alpha coefficient.

For each news article, in addition to the golden summary from the CNN/Daily Mail dataset, 10 additional human written reference summaries are provided in this dataset. All 16 summarization models in this dataset, are trained on the CNN/DailyMail News corpus and the collected summaries are generated using the test split of the dataset. There are a total of 4 extractive and 12 abstractive summarization models in this dataset which allows for independent evaluation of the proposed metric on both classes of summarization models. Datasets mentioned below are used for training sentence-pair transformer models we employ in our metric computation. Description of these datasets are included for completeness.

Paraphrase Detection: For this task, we use models trained on Quora Question Pairs (QQP) (Iyer et al., 2017) and Microsoft Research Paraphrase Corpus (MRPC) (Dolan and Brockett, 2005), which are two

[1] Sentence-Transformers

widely used datasets. QQP dataset contains more than 400K question pairs from Quora, the online question-answering site, with the annotations indicating if the questions are duplicates of each other. MRPC contains sentence pairs extracted from online news sources. For each sentence pair, annotations by two human judges are presented, indicating whether the sentences describe the same news event.

Semantic Text Similarity: STS-Benchmark (Cera et al.,) is the dataset used for this task and it is a collection of sentence pairs from news headlines and other online sources. Each sentence pair in this dataset is annotated with a score between 1 and 5 denoting how similar the two sentences are with respect to their semantic meaning.

Natural Language Inference: This task is also known as Recognizing Textual Entailment (RTE). Stanford Natural Language Inference (SNLI) corpus (Bowman et al., 2015), Multi-Genre Natural Language Inference (Multi-NLI) corpus (Williams et al., 2018) are two datasets on which models are trained. These datasets contain sentence pairs manually labeled for classification with labels entailment, contradiction and neutral for the NLI task.

Passage Ranking: MS-MARCO Passage Ranking dataset (Nguyen et al., 2016) is used to train the models for this task. Apart from passage ranking, this dataset is also used for machine comprehension, question answering, and key phrase extraction. This dataset contain 1 million queries from the search engine Bing, and 8.8 million passages extracted from the web in response to those queries.

4.2. Evaluation Measure

To evaluate our proposed metric, we compute Pearson Correlation Coefficient between the evaluation score by our automatic metric and the human evaluation score, along 4 dimensions. For each dimensions, human evaluation score is a value between 0 and 5.

- **Coherence:** Checks if generated summary is well-structured, and well-organized. The sentences in the generated summary should be related on each other and build a coherent narrative about a topic.
- **Consistency:** Looks at factual alignment between the generated and source summary. Sentences in the generated summary should be related to the source summary.
- **Fluency:** Considers quality of individual sentences with respect to formatting, capitalization errors, and ungrammatical constructs. Generated summary should be easy to read.
- **Relevance:** Checks if important content from the source is included in generated summary. Generated summary should neither be too long nor contain redundant information.

4.3. Implementation Details

All the pre-trained models used in our metric computation are implemented in PyTorch (Paszke et al., 2017) and are publicly available for download[2] (Reimers and Gurevych, 2019). For each sentence-pair task, we pick two pre-trained models based on their performance on their corresponding benchmark datasets. Thus, in our metric computation we use an ensemble of $N = 8$ models in total, with 2 models from each of the 4 pre-train tasks.

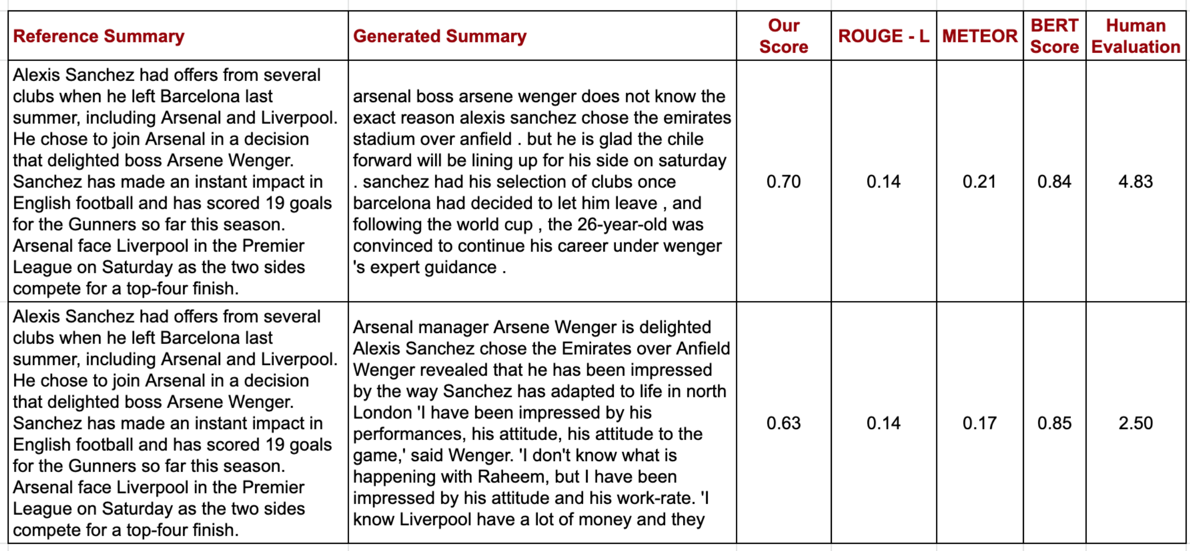
4.4. Results

To evaluate our proposed metric for each generated summary, we compute the Pearson Correlation between the evaluation score by our metric and the human evaluation score, for each of the reference summaries provided by (**?**) as well as the single golden reference summary. Table 1 and, 2, show that our proposed metric achieves better results than the current state-of-the-art validating our hypothesis that models pre-trained on sentence-pair tasks capture the semantic similarity between a pair of sentences and help in better evaluating text summarization methods. To compute our final metric, we use an ensemble of 4 pre-trained models, one from each sentence-pair task, and average the similarity scores from all 4 models for each sentence-pair. More details on the ensemble strategy and the performance of individual models in the ensemble are presented in the next section. Refer to Figure 2, for a sample qualitative result comparison with other metrics. Also, refer to the supplementary material for more results including performance of our metric on individual summarization models and Pearson correlation with single golden reference summary for abstractive and extractive models.

4.4.1. Baselines

Our metric is based on models trained specifically on sentence-pair tasks. To show the importance of pre-training on sentence-pair tasks, we present the results of our metric using models trained on other language modeling tasks such as Masked Language Modeling (MLM) and Named Entity Recognition (NER). These tasks are word/token-level tasks which involve masked token prediction (MLM) or token classification (NER). For a fair comparison, the models we use here as baseline differ only on the pre-training task but not in model complexity; results are presented in Table 3. These results show that none of the models trained on word/token-level tasks achieve comparable performance to the models trained on sentence-pair tasks. This shows the inability of models trained on word/token-level tasks in capturing the semantic similarity between sentences and highlight the importance of pre-training on sentence-pair tasks.

[2] Sentence-Transformers



Reference Summary	Generated Summary	Our Score	ROUGE - L	METEOR	BERT Score	Human Evaluation
Alexis Sanchez had offers from several clubs when he left Barcelona last summer, including Arsenal and Liverpool. He chose to join Arsenal in a decision that delighted boss Arsene Wenger. Sanchez has made an instant impact in English football and has scored 19 goals for the Gunners so far this season. Arsenal face Liverpool in the Premier League on Saturday as the two sides compete for a top-four finish.	arsenal boss arsene wenger does not know the exact reason alexis sanchez chose the emirates stadium over anfield . but he is glad the chile forward will be lining up for his side on saturday . sanchez had his selection of clubs once barcelona had decided to let him leave , and following the world cup , the 26-year-old was convinced to continue his career under wenger 's expert guidance .	0.70	0.14	0.21	0.84	4.83
Alexis Sanchez had offers from several clubs when he left Barcelona last summer, including Arsenal and Liverpool. He chose to join Arsenal in a decision that delighted boss Arsene Wenger. Sanchez has made an instant impact in English football and has scored 19 goals for the Gunners so far this season. Arsenal face Liverpool in the Premier League on Saturday as the two sides compete for a top-four finish.	Arsenal manager Arsene Wenger is delighted Alexis Sanchez chose the Emirates over Anfield Wenger revealed that he has been impressed by the way Sanchez has adapted to life in north London 'I have been impressed by his performances, his attitude, his attitude to the game,' said Wenger. 'I don't know what is happening with Raheem, but I have been impressed by his attitude and his work-rate. 'I know Liverpool have a lot of money and they	0.63	0.14	0.17	0.85	2.50

Figure 2: Qualitative result comparison for an example summary. In this example, our metric score correlates better with human evaluation score when compared to other metrics. With drop in human evaluation score, our metric score also drop, while other metric scores remain the same or increase. Human evaluation scores range between 0-5, and the evaluation metric scores range between 0-1.

Metrics	Coherence	Consistency	Fluency	Relevance	Average
BERTScore(p/r/f)	0.14/0.17/0.18	0.15/0.21/0.20	0.20/0.16/0.21	0.19/**0.35**/0.32	0.17/0.22/0.22
BLEU	0.22	0.17	0.14	**0.35**	0.22
METEOR	0.03	0.03	0.07	-0.01	0.03
MoverScore	0.21	0.19	0.20	0.31	0.22
ROUGE-1/2/3/	0.20/0.15/0.16	0.18/0.16/0.16	0.15/0.10/0.09	**0.35**/0.27/0.26	0.22/0.17/0.16
ROUGE-we (1/2/3)	0.20/0.15/0.13	0.18/0.13/0.10	0.15/0.09/0.05	**0.35**/0.29/0.26	0.22/0.19/0.13
S3 (pyr/resp)	0.16/0.16	0.16/0.16	0.08/0.08	0.34/0.32	0.18/0.18
SMS	0.18	0.18	0.17	0.27	0.20
Our SPEED-Score	**0.23**	**0.25**	**0.27**	0.32	**0.27**

Table 1: Pearson correlation between expert annotations and various evaluation metrics, along 4 quality dimensions for all 16 summarization models using all reference summaries.

Metrics	Coherence	Consistency	Fluency	Relevance	Average
BertScore (p/r/f)	0.11/0.17/0.15	0.11/0.18/0.16	0.15/0.14/0.16	0.21/0.33/0.30	0.14/0.20/0.19
BLEU	0.03	0.05	0.06	0.04	0.04
METEOR	0.15	0.18	0.11	0.33	0.19
MoverScore	0.14	0.16	0.13	0.28	0.17
ROUGE-1/2/3	**0.19**/0.14/0.13	0.18/0.14/0.12	0.13/0.10/0.08	**0.33**/0.24/0.21	0.20/0.15/0.13
ROUGE-we (1/2/3)	0.18/0.14/0.13	0.18/0.12/0.10	0.13/0.10/0.08	**0.33**/0.26/0.24	0.20/0.15/0.13
S3 (pyr/resp)	0.15/0.15	0.17/0.16	0.10/0.09	0.32/0.29	0.18/0.17
SMS	0.16	0.17	0.14	0.24	0.17
Our SPEED-Score	0.17	**0.24**	**0.26**	0.29	**0.24**

Table 2: Pearson correlation between expert annotations and various evaluation metrics, along 4 quality dimensions for all 16 summarization models using single golden reference summary.

4.4.2. Ablations

Pre-train Tasks: We use ensemble of models trained on different sentence-pair tasks in our metric computation. To evaluate the contribution of each model, we compute our metric using individual models and present the results in Table 6. As shown in this table, we evaluate a total of 8 models, 2 for each of the 4 pre-train tasks. These results show that the performance

Model Name	Pre-train Task	Coherence	Consistency	Fluency	Relevance
bert_base_uncased	MLM	0.02	0.11	0.15	-0.02
roberta_base	MLM	0.01	0.05	0.19	0.07
roberta_large	MLM	-0.05	0.03	0.06	0.01
distilroberta_base	MLM	-0.01	-0.06	0.12	-0.01
xlm_roberta_base	MLM	0.06	-0.04	-0.01	0.03
bert_base_NER_uncased	NER	-0.01	0.08	0.13	0.04

Table 3: Performance of our metric using different models trained on token level language modeling tasks. MLM - Mask Language Modeling task involves masked token prediction, NER - Named Entity Recognition task involves classification of tokens.

of the individual models vary from one dimension to another and no single model outperform others on all dimensions.

In Table 6, we also show results using ensemble of pre-trained models for each sentence-pair task. From these results we can infer that ensembling of models trained on the same task help improve the overall results across all the dimensions. Also, these results show the significance of the STS and NLI tasks in capturing the semantic similarity between sentence pairs. The second row from the last, show the results using all the 8 models in the ensemble and we observe that ensemble of models improve the performance of our metric compared to using individual models. Our final results are presented in the last row of Table 6, using best models from each of the pre-train tasks in metric computation.

Ensemble - Average vs Max-Voting: To compute the semantic similarity between a pair of sentences, we compute cosine similarity between their embeddings as shown in Equation 1. As shown in this equation, we average the cosine similarity scores for the sentence pair across all the N pre-trained models used in metric computation. An alternative approach to *Averaging* is *Max-Voting*, where the maximum similarity score across different pre-trained models is used instead of the mean as shown in Equation 2. Results using this ensemble approach is presented in Table 7 and, *Averaging* outperforms *Max-Voting* in 3 out of 4 dimensions. This shows that using embeddings from all models for each sentence lead to better performance.

$$Sim_{ij} = max(\{1 - Cos.Dis.(E_i^n, E_j^n) : n = 1, .., N\}) \tag{2}$$

Maximum vs Optimal Matching: After computing the similarity scores for each pair of sentences from the system generated summary and the reference summary, we match every sentence in the generated summary to a sentence in the reference summary. For this matching, we use *Maximum* similarity score as the matching criterion. In this setup, multiple sentences in the generated summary can be mapped to a single sentence in the reference and not all sentences in the reference are mapped. An alternative to *Maximum* matching approach would be find an *Optimal* matching, where every sentence in the generated summary is matched to a sentence in the reference summary and vice-versa. The criterion for this matching is to maximize the sum of similarity scores for the sentences in the generated summary. We present the results of our metric with these matching strategy in Table 8. Results show that *Maximum* matching outperforms *Optimal* matching along all the dimensions, except *Relevance*. This is expected as the dimension *Relevance* checks if all the content from the reference is included in the generated summary and *Optimal* matching allows for matching of every sentence in the reference sentence to a sentence in the generated summary.

Extractive vs Abstractive Summarization: As the SummEval dataset contain system generated summaries from both abstractive and extractive summarization models, we analyze the performance of our metric on both classes of text summarization models. We present the results using only abstractive models in Table 4, and using only extractive models in Table 5. In both these tables, we also present results from existing evaluation metrics on these models for a fair comparison. As shown in the Tables 4 & 5, our metric outperforms existing metrics in 3 out of 4 dimensions and achieve better overall performance. It is also worth noting that we achieve consistent results on both abstractive, extractive summarization models.

5. Discussion

Computing semantic similarity between two texts is at the core of our metric computation. We do so by leveraging language models pre-trained on sentence-pair tasks which involve finding semantic similarity between sentence pairs. Using our proposed metric, we achieve better results than the current state-of-the-art on both abstractive and extractive summarization models. Results presented above show that our proposed metric achieves improvement along the dimensions *Consistency* and *Fluency* on both abstractive and extractive models. *Consistency* deals with factual alignment between generated summary and the reference summary. An improvement over the state-of-the-art along this dimension validates the claim that our proposed metric better captures the semantic similarity between generated and reference summaries. *Flu-*

Metrics	Coherence	Consistency	Fluency	Relevance	Average
BERTScore(p/r/f)	0.21/0.13/0.20	0.22/0.19/0.23	**0.27**/0.14/0.24	0.24/0.34/0.33	0.23/0.20/0.25
BLEU	0.22	0.17	0.13	0.36	0.22
METEOR	0.08	0.10	0.14	0.05	0.09
MoverScore	0.23	0.23	0.24	0.34	0.26
ROUGE-1/2/3/	0.21/0.16/0.17	0.19/0.16/0.16	0.15/0.10/0.09	**0.37**/0.27/0.26	0.23/0.17/0.17
ROUGE-we (1/2/3)	0.21/0.15/0.13	0.19/0.13/0.10	0.16/0.08/0.05	**0.37**/0.29/0.26	0.23/0.16/0.13
S3 (pyr/resp)	0.12/0.12	0.12/0.12	0.02/0.03	0.33/0.31	0.16/0.14
SMS	0.20	0.21	0.20	0.30	0.22
Our SPEAD-Score	**0.24**	**0.25**	**0.27**	0.31	**0.27**

Table 4: Performance of metrics on abstractive summarization models. Pearson correlation between various automatic evaluation metrics with expert annotations along 4 quality dimensions using all reference summaries.

Metrics	Coherence	Consistency	Fluency	Relevance	Average
BERTScore(p/r/f)	-0.01/**0.25**/0.10	0.04/0.00/0.03	0.02/0.06/0.02	0.17/0.25/0.25	0.06/0.14/0.1
BLEU	0.16	-0.01	0.07	0.24	0.12
METEOR	-0.07	-0.01	-0.07	-0.10	0.06
MoverScore	0.15	0.04	0.04	0.24	0.11
ROUGE-1/2/3/	0.14/0.09/0.09	-0.03/-0.07/-0.07	0.03/-0.01/-0.02	0.23/0.18/0.19	0.10/0.08/0.09
ROUGE-we (1/2/3)	0.14/0.12/0.08	-0.03/-0.06/-0.09	0.02/0.00/-0.01	0.23/0.22/0.20	0.10/0.10/0.09
S3 (pyr/resp)	0.20/0.18	-0.05/-0.05	0.07/0.05	0.28/0.25	0.15/0.13
SMS	0.10	0.00	0.02	0.19	0.07
Our SPEED-Score	0.12	**0.10**	**0.19**	**0.30**	**0.18**

Table 5: Performance of metrics on extractive summarization models. Pearson correlation between various automatic evaluation metrics with expert annotations along 4 quality dimensions using all reference summaries.

ency, on the other hand, is the dimension dealing with quality of the generated summary and an improvement along this dimension is attributed to the use of sentence embeddings in metric computation. Also using sentence embeddings help our metric improve performance along the dimensions *Coherence* and *Relevance* on abstractive and extractive models respectively. Our metric shows consistent performance on both abstractive and extractive summarization models, while most of the existing metrics favor one class of models over the other. This is an important advantages of our metric which makes it more applicable when compared to other metrics.

5.1. Runtime Analysis

In this section, we evaluate the run-time of our proposed metric and compare it with other well-know metrics. The results of this comparison are shown in Figure3, which is a scatter plot showing the performance versus run-time of the metrics. Metrics which are faster to compute and have high performance lie closer to the top-left corner of the plot, as is the case with our proposed metric. As our proposed metric uses multiple pre-trained models to compute the evaluation score, the computation time using a naive implementation is higher than the existing metrics. However, a significant boost in speed is achieved by optimizing the implementation to enable parallel processing of inputs by the pre-trained models.

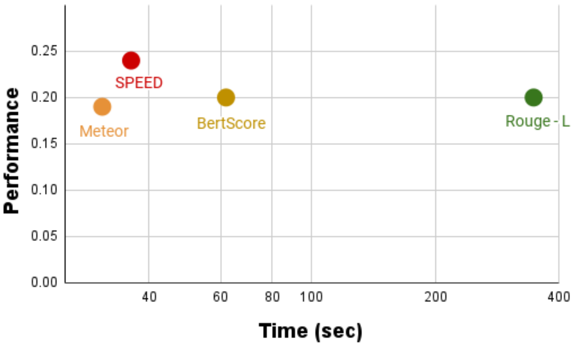


Figure 3: Runtime vs Performance plot showing a comparison between metrics. Here we show the total computation time to process all the 1600 summaries from the SummEval dataset.

For this analysis, we compute the total time taken by each metric to process the 1600 generated summaries in the SummEval dataset. To compute the runtime for the metrics Rouge-L, BertScore, and Meteor we use the publicly available implementation provided by Hugging Face[3]. The performance scores, y-axis in the plot, are the Pearson correlation coefficients from Table 2, computed using the single golden reference summary.

[3] https://huggingface.co/metrics

Model Name	Pre-train Task	Coherence	Consistency	Fluency	Relevance
paraphrase-xlm-r-multilingual	PD	0.15	0.16	0.12	0.23
paraphrase-distilroberta-base	PD	0.16	0.15	0.12	0.23
ce-roberta-large-stsb	STS	0.12	0.18	0.35	0.17
ce-distilroberta-base-stsb	STS	0.10	0.15	0.25	0.16
nli-roberta-large	NLI	0.23	0.20	0.15	0.30
bert-base-nli-max-tokens	NLI	0.22	0.18	0.12	0.30
ce-ms-marco-TinyBERT-L-2	PR	0.15	0.21	0.22	0.23
msmarco-roberta-base-v2	PR	0.11	0.14	0.12	0.17
Ensemble-PD	PD	0.16	0.16	0.12	0.23
Ensemble-STS	STS	0.12	0.19	**0.35**	0.19
Ensemble-NLI	NLI	**0.24**	0.19	0.14	**0.31**
Ensemble-PR	PR	0.12	0.17	0.14	0.19
Ensemble-ALL	PD, STS, NLI, PR	0.21	0.23	0.22	0.30
Ensemble-BEST	PD, STS, NLI, PR	0.23	**0.25**	0.27	**0.31**

Table 6: Performance of our metric using different models trained on different sentence-pair tasks. PD - Paraphrase Detection, STS - Semantic Text Similarity, NLI - Natural Language Inference, PR - Passage Retrieval, ALL - All 8 models, BEST - One best model from each pre-train task. The model names highlighted in bold, are the models we select to compute our metric for comparison with state-of-the-art.

Ensemble	1	2	3	4
Max-Voting	0.15	0.22	**0.36**	0.21
Averaging	**0.23**	**0.25**	0.27	**0.31**

Table 7: Performance of our metric with Averaging and Max-Voting ensembling. 1: Coherence, 2: Consistency, 3: Fluency, 4: Relevance.

Matching	1	2	3	4
Optimal	0.22	0.20	0.23	**0.34**
Maximum	**0.23**	**0.25**	**0.27**	0.31

Table 8: Performance of our metric with Maximum and Optimal matching techniques. 1: Coherence, 2: Consistency, 3: Fluency, 4: Relevance.

6. Conclusion

We in this work, propose a simple evaluation metric, *SPEED-Score*, for text summarization which captures the semantic similarity between system generated summary and the reference summary. Our metric uses sentence-pair embeddings in contrast to existing metrics which use sentence or word/token embeddings. We propose the use of pre-trained models, trained specifically on sentence-pair tasks, to extract sentence embeddings as these models better capture the semantic similarity between sentences.

Our proposed metric is effective, simple and easy to use. It achieves better results than the current state-of-the-art on the SummEval dataset, demonstrating it's effectiveness. It is computed using simple sentence matching and is based on sentence embeddings from pre-trained models. Our experimental results show that applicability of our metric to both abstractive and extractive summarization models. We perform baseline experiments to show the importance of using models trained on sentence-pair tasks for extracting sentence embeddings. Also, we perform multiple ablations to support our design decision in each step of our metric computation. Finally, we compare the runtime of our metric with other well-know metrics and show that our metric is comparably faster while achieving the best performance.

7. References

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

Cera, D., Diabb, M., Agirrec, E., Lopez-Gazpioc, I., Speciad, L., and Donostia, B. C.). Semeval-2017 task 1: Semantic textual similarity multilingual and cross-lingual focused evaluation.

Clark, E., Celikyilmaz, A., and Smith, N. A. (2019). Sentence mover's similarity: Automatic evaluation for multi-sentence texts. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2748–2760.

Dolan, W. B. and Brockett, C. (2005). Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Fabbri, A. R., Kryściński, W., McCann, B., Xiong, C., Socher, R., and Radev, D. (2021). Summeval: Re-evaluating summarization evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409.

Gao, Y., Zhao, W., and Eger, S. (2020). SUPERT:

Towards new frontiers in unsupervised evaluation metrics for multi-document summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1347–1354, Online, July. Association for Computational Linguistics.

Hermann, K. M., Kocisky, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., and Blunsom, P. (2015). Teaching machines to read and comprehend. In *Advances in neural information processing systems*, pages 1693–1701.

Iyer, S., Dandekar, N., and Csernai, K. (2017). First quora dataset release: Question pairs. *data. quora. com.*

Kilickaya, M., Erdem, A., Ikizler-Cinbis, N., and Erdem, E. (2017). Re-evaluating automatic metrics for image captioning. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 199–209.

Kryściński, W., McCann, B., Xiong, C., and Socher, R. (2020). Evaluating the factual consistency of abstractive text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9332–9346.

Kusner, M., Sun, Y., Kolkin, N., and Weinberger, K. (2015). From word embeddings to document distances. In *International conference on machine learning*, pages 957–966.

Lavie, A. and Agarwal, A. (2007). Meteor: An automatic metric for mt evaluation with high levels of correlation with human judgments. In *Proceedings of the second workshop on statistical machine translation*, pages 228–231.

Lin, C.-Y. (2004). Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Maynez, J., Narayan, S., Bohnet, B., and McDonald, R. (2020). On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919.

Nguyen, T., Rosenberg, M., Song, X., Gao, J., Tiwary, S., Majumder, R., and Deng, L. (2016). Ms marco: A human generated machine reading comprehension dataset. In *CoCo@ NIPS*.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. (2017). Automatic differentiation in pytorch.

Peyrard, M., Botschen, T., and Gurevych, I. (2017). Learning to score system summaries for better content selection evaluation. In *Proceedings of the Workshop on New Frontiers in Summarization*, pages 74–84.

Reimers, N. and Gurevych, I. (2019). Sentence-bert: Sentence embeddings using siamese bert-networks. 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)*, pages 3982–3992.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Williams, A., Nangia, N., and Bowman, S. (2018). A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

Zhang, T., Kishore, V., Wu, F., Weinberger, K. Q., and Artzi, Y. (2019). Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Zhao, W., Peyrard, M., Liu, F., Gao, Y., Meyer, C. M., and Eger, S. (2019). Moverscore: Text generation evaluating with contextualized embeddings and earth mover distance. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP-IJCNLP)*, pages 563–578.