Hybrid Ensemble Predictor as Quality Metric for German Text Summarization: Fraunhofer IAIS at GermEval 2020 Task 3

David Biesner[*†‡], **Eduardo Brito**[*†§], **Lars Patrick Hillebrand**[*†‡], **Rafet Sifa**[†]
[†] Fraunhofer IAIS, Schloss Birlinghoven, 53757 Sankt Augustin, Germany
[§] Fraunhofer Center for Machine Learning, Germany
[‡] B-IT, University of Bonn, Endenicher Allee 19a, 53115 Bonn, Germany

Abstract

We propose an alternative quality metric to evaluate automatically generated texts based on an ensemble of different scores, combining simple rule-based metrics with more complex models of very different nature, including ROUGE, tf-idf, neural sentence embeddings, and a matrix factorization method. Our approach achieved one of the top scores on the second German Text Summarization Challenge.

1 Introduction

In our previous work on automatic text summarization (Brito et al., 2019), we concluded criticizing the suitability of ROUGE scores (Lin, 2004) for overall evaluation purposes. These and other common quality metrics found in the automatic text summarization literature like BLEU (Papineni et al., 2002) or METEOR (Banerjee and Lavie, 2005) are far from being optimal since they only focus on the lexical overlap as a proxy for assessing content selection. They do not only penalize certain abstractions (e.g. when the original sentences are heavily reformulated or when synonyms are applied) but they also ignore other aspects that are usually considered desirable in good summaries, including grammatical correctness and compactness.

The second German Text Summarization Challenge aims to address this issue by releasing a text corpus with several summaries per text[1]. Its participants were asked to rate these summaries with new ideas and solutions regarding an automatic quality assessment of German text summarizations. We propose to combine the advantages of neural approaches that excel at encoding semantic textual similarity (and are thus suitable to predict content) with statistical and rule-based metrics that can evaluate other important summarization aspects such as compactness and abstractiveness.

In our approach, we employ an ensemble of 7 statistically significant predictors (p-value $< 15\%$) in a linear regression model (see Table 2). Comparing our predictions to the competition host's own non-public annotations we achieved a score (i.e. loss) of 33.72, one of the lowest and therefore best scores of participating teams.

In the following sections, we detail the different metrics that we considered and how we optimized its combination.

2 Experimental Setup

This section describes our experimental setup, namely the underlying dataset and the methodological approach.

2.1 Data

The shared task organizers released a corpus consisting of 216 texts with a corresponding reference summary and a generated summary, each of them rated with a value between 0 (bad) to 1 (excellent).

In order to evaluate the methods we manually annotated all summaries in the dataset with a score from 0 to 1. We independently rated a part of the corpus each, such that different human biases can be compensated to a certain extent. A submission of these annotations to the competition received a high score, indicating a large similarity to the gold standard annotations set by the organizers. Additionally, we expanded the dataset by considering the given reference summaries as perfect generated summaries with an automatic score of 1.

[*]These co-first authors contributed equally to this work.

[1]https://swisstext-and-konvens-2020.org/2nd-german-text-summarization-challenge.

This results in a dataset of 248 summary texts with their corresponding score, which is used to evaluate the unsupervised methods described below.

2.2 Methodology

We address this challenge as a metric learning problem, where we define a set of unsupervised predictors covering one or several features that answer the required properties of a good summary (content relevancy, compactness, abstractiveness and grammatical correctness). After calculating all predictor scores (unsupervised) for each document we apply min-max normalization to assure all scores lay in the closed 0-1 interval. In a final step, we ensemble these predictors in a capped linear regression model (output between 0 and 1), which is trained via ordinary least squares on our manual summary annotations (see Section 2.1). We iteratively remove non-significant predictors, p-value $\geq 15\%$, and re-run the regression model until all predictors yield significant t-statistics, namely their coefficients lay within the two-sided 85% confidence interval. Due to the limited amount of documents and the loss of interpretability, we refrain from including non-linearities (e.g. multiple layers, non-linear activation functions, interaction terms of different polynomial degrees, etc.) into the regression model. Also, by using a simple linear ensemble model, we reduce the likelihood of overfitting on our annotations, especially since no validation set for parameter tuning is available.

The following subsections lay the focus on our predictors and describe their functionality. We start presenting three content predictors, which all determine the most important words in the original text and compute the fraction of how many of these words occur in the generated summaries. We assume that the most important words in a document capture the essence of the text and thus, function as proxy for contentual relevance. We continue with neural language model driven predictors which primarily focus on contentual relevance and grammatical correctness. We also include the standard quality metrics for automatic summary evaluation, ROUGE, BLEU, and METEOR, which all aim to measure contentual relevance, as well. The remaining predictors are mainly rule-based and refer largely to compactness and abstractiveness.

2.2.1 Tf-Idf content predictor

A very popular text vectorization method is tf–idf (Term frequency – Inverse document frequency). It is a frequency-based statistic, which intends to reflect how important a word is to a “document” in a corpus.

Given that our entire corpus contains N documents and the vocabulary of our corpus is of size K, we can collect the individual tf–idf scores in some matrix $\boldsymbol{M} \in \mathbb{R}^{N \times K}$. Each row vector in this matrix corresponds to a document embedding. We find the top 10 important words per document by decreasingly sorting the tf-idf scores within each embedding.

We utilize the `sklearn` [2] implementation of the `TfidfVectorizer` and restrict our vocabulary to words with a document frequency below 0.9. Before vectorization, we apply lower-casing, punctuation and stop word removal, and stemming to the entire text corpus, which helps to better capture meaning and content in the text's vector representation.

2.2.2 NMF content predictor

NMF (Nonnegative Matrix Factorization) (Paatero and Tapper, 1994; Lee and Seung, 2001) is a common matrix factorization technique frequently used for topic modeling. In previous work, we find that NMF achieves good results in clustering document words to a predefined number of latent topics. Assuming that a good summary should cover all main topics in a text, we apply NMF on each document, and determine the top 5 important words per latent topic dimension. In particular, we factorize the document's symmetrical co-occurrence matrix[3] $\boldsymbol{S} \in \mathbb{R}^{N \times N}$ into a nonnegative loading matrix $\boldsymbol{W} \in \mathbb{R}^{N \times M}$ and a nonnegative affinity matrix $\boldsymbol{H} \in \mathbb{R}^{M \times N}$,

$$\boldsymbol{S} = \boldsymbol{W}\boldsymbol{H} + \boldsymbol{\mathcal{E}}, \tag{1}$$

where N is the vocabulary size of the document at question, $M = 10$ is the number of latent topics and $\boldsymbol{\mathcal{E}} \in \mathbb{R}^{N \times N}$ is the error matrix, whose elements approach zero for a perfect decomposition.

[2] `https://github.com/scikit-learn/scikit-learn`.

[3] We apply the same document preprocessing as in Section 2.2.1 before calculating the co-occurrence matrix. Also, we choose a window size of 5 and each context word j contributes $1/d$ to the total word pair count, given it is d words apart from the base word i.

For both, $\boldsymbol{W}$ and $\boldsymbol{H}^T$ we assign each word (row vector) to the latent topic dimension with the highest value. Next, we decreasingly sort the assigned words per topic, so that the most distinct topic words are ranked on top. Finally, we get the important words per document by removing all duplicates from the selected topic words of $\boldsymbol{W}$ and $\boldsymbol{H}$.

2.2.3 Flair NER content predictor

Flair (Akbik et al., 2018) is a specific contextual string embedding architecture. The backbone of the flair framework is a pretrained character-based language model (based on an LSTM[4]-RNN), which is bidirectionally trained on a huge independent text corpus for different languages, including German.

Build on top of this language model, the framework provides a German named entity tagger, which is pretrained on the Conll-03 dataset (Sang and De Meulder, 2003). First, raw and unprocessed text is fed sequentially into the encoding part of the bidirectional language model. Second, we retrieve for each word i a contextual embedding by concatenating the forward model's hidden state after word i and the backward model's hidden state before word i. This word embedding is then passed into a vanilla BiLSTM-CRF[5] sequence labeler.

We apply this sequence tagger on our raw input documents and consider all predicted named entities as the document's important words.

2.2.4 Flair grammar predictor

In order to evaluate grammatical correctness, we again leverage the aforementioned flair language model, which was trained as an auto-encoder to correctly predict the next character in a text. For a grammatically correct text we would expect the model to mostly guess the next character correctly. A text with grammatical errors however would not match the expectations of the model, thus creating a larger reconstruction error on the characters that do not fit grammatically. To assess grammatical correctness we feed the summary text through the model and score the summary based on the accumulated reconstruction error.

2.2.5 Sentence-BERT predictor

We explore how sentence embeddings can be used to measure "how similar" (semantically) a summary is compared to its original text. In particular, we infer sentence embeddings with the pretrained *bert-base-german-uncased* BERT model from the HuggingFace's transformers library (Wolf et al., 2019) in the fashion proposed with the Sentence-BERT architecture (Reimers and Gurevych, 2019). The output of the BERT model is max-pooled to obtain a fixed-size vector for each processed piece of text. This way, we can obtain embeddings for both the original text and each of the summaries. The resulting predictor score is thus the cosine similarity of the summary vector with the original text vector.

2.2.6 ROUGE predictor

The ROUGE score is a classic metric for assessing the quality of summaries. Even though it alone is not sufficient to evaluate summaries it can give useful insight when applied in an ensemble setting. We calculate the rouge-1, rouge-2 and rouge-L scores between the summary and both the full original text and the reference summary. While rouge-1 and rouge-2 calculates the overlap of unigrams and bigrams (i.e. single words and adjacent word pairs) between reference text and summary, rouge-L evaluates the longest common subsequence between reference and summary.

2.2.7 BLEU predictor

BLEU is a metric that calculates an n-gram precision between one or multiple reference texts and a summary hypothesis, in which n-gram counts in the summary are compared to their maximum count in one of the references.

2.2.8 METEOR predictor

METEOR is a metric that calculates a harmonic mean between the recall and precision of an n-gram matching which considers word order between a reference text and a summary.

2.2.9 Compactness predictor

We calculate the compactness score as the compression rate with respect to the original text, where the text length is measured by the number of characters.

2.2.10 Number matching predictor

A good summary should be factually correct. While there might be some ambiguity from different word choices between original text and summary, there usually is only one way to display exact numbers like dates. We thus expect every number in the summary to also appear in the original text.

[4]Long Short Term Memory.

[5]Bi-directional Long Short-Term Memory Conditional Random Field.

To assess factual correctness regarding numbers, we count how many of the numbers in the summary are also present in the text.

2.2.11 Sentence copying predictor

At times, one can generate a usable summary by simply extracting the first sentences of the original text, since they often provide an introduction and therefore a mini-summary of the remaining text. However, the goal of our evaluation is finding abstractive and novel summaries. We therefore perform a binary check on whether the summary exactly matches the first sentences of the original text and assign a 1 if they are extracted from the original text and a 0 if they are more abstracted.

3 Evaluation

In this section, we report and analyze our results of employing a capped linear regression model to ensemble the significant subset of our predictors to generate a representative summarization quality metric. We start by fitting a capped linear regression model to the full set of predictors, including an intercept, and consider the p-values of each predictor. We iteratively remove the most insignificant predictor (largest p-value) and re-run the linear regression. We stop once all predictors are statistically significant to the 15% level.

The final regression model on the remaining 7 significant predictors is described in Table 1.

	coef	**std_err**	**P> |t|**
constant	0.072	0.095	0.447
tfidf_content	0.535	0.107	0.000
flair_grammar	0.226	0.109	0.038
sbert	0.169	0.106	0.110
sentence_copying	−0.168	0.064	0.009
rouge-1	2.560	0.571	0.000
rouge-2	−1.531	0.340	0.000
rouge-L	−1.329	0.646	0.041

Table 1: Regression coefficients, standard errors and p-values for final predictor set.

The columns show the estimated coefficients, standard errors and p-values of each predictor. Since all predictors have been normalized (min-max normalization) prior to the regression, their regression coefficients are directly comparable in magnitude. It can be seen that the rouge-1 predictor has the highest coefficient and thus, is most important for predicting the summary evaluation score. However, the other predictors also contribute significantly to the prediction outcome, which gets evident when comparing the final ensemble error of 33.72 (see Table 2) to the individual rouge-1 error of 35.99 (see Table 3).

Further, the coefficients of the sentence copying, rouge-2 and rouge-L predictors imply a negative correlation to the annotated summary scores. This is expected because all three predictors yield high scores, when entire sentences, bigrams or common subsequences of the original documents get copied to or make up the generated summaries. Yet, our annotations favor abstractive summaries which is why a higher score of one of the above predictors indicates a worse summary when taking abstractiveness as a quality indicator into account.

Table 2 shows the final error values obtained by different predictor ensembles in the shared task public ranking. Despite of more predictors increasing the likelihood of overfitting on our manual annotations and thereby lowering our final error score, one can observe the opposite. Removing insignificant predictors actually yields the best performing model and puts us among the top participating teams.

Ensemble	**Error**	**Predictors**
7 predictors	33.72	constant, tfidf_content, flair_grammar, sentence_copying sbert, rouge-1, rouge-2, rouge-L
10 predictors	33.90	+ nmf_content, bleu, meteor
13 predictors	33.82	+ flair_ner_content, compression, number_matching

Table 2: Error values obtained in the shared task public ranking by different predictor ensembles. A lower value means better performance.

4 Comparison with standard metrics

In order to show the validity of our approach and its improvement over previously established methods, we take a look at the performance of BLEU, METEOR and ROUGE as single predictors.

We implement each metric using the standard definition and further employ min-max normalization as described above in order to receive a metric that assigns a score between 0 (bad) and 1 (good) so that both extremes appear in the dataset. This approach is developed entirely without manual annotations. The scores received on the challenge task are depicted in the middle column of Table 3.

Furthermore, we use our manual annotations to adjust the predictors to the available dataset, fitting a linear regression of a single predictor to the annotated summary scores. These scores are depicted in the right column of Table 3.

As already signified, we see that using these metrics out-of-the-box results in significantly worse performance than both the fitted algorithm and our ensemble approach. While the fitted metrics score is considerably higher than their original counterpart, we still see a distinct improvement when employing an ensemble of different predictors.

Predictor	Error (original)	Error (fitted)
rouge-1	44.26	35.99
rouge-2	52.50	36.08
rouge-L	44.27	36.12
bleu	64.16	36.11
meteor	53.05	36.06

Table 3: Error values obtained by some of the common evaluation metrics for automatic text summarization after uploading their scores to the shared task public ranking. A lower value means better performance. The middle column represents the errors for the min-max normalized predictor scores. The right column shows the final errors for the normalized predictor scores being fitted via linear regression to our manual summary annotations.

5 Conclusion and Future Work

We showed that a hybrid combination of rule-based, statistical and deep-learning techniques outperforms other alternatives for automatic evaluation of automatically generated German text summarization given the provided shared task dataset.

Although the text corpus covers a wide range of topics, the text style is quite homogeneous. Mostly, it consists of generally grammatically perfect descriptive texts. It would be interesting to test if our approach also works for more informal noisy texts. Furthermore, it would be also interesting to evaluate different state-of-the-art summarization approaches with our new metric.

Acknowledgments

The authors of this work were supported in parts by the Fraunhofer Research Center for Machine Learning (RCML) within the Fraunhofer Cluster of Excellence Cognitive Internet Technologies (CCIT) and by the Competence Center for Machine Learning Rhine Ruhr (ML2R) which is funded by the Federal Ministry of Education and Research of Germany (grant no. 01—S18038B). We gratefully acknowledge this support.

References

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Eduardo Brito, Max Lübbering, David Biesner, Lars Patrick Hillebrand, and Christian Bauckhage. 2019. Towards supervised extractive text summarization via RNN-based sequence classification. *arXiv preprint arXiv:1911.06121*.

Daniel D Lee and H Sebastian Seung. 2001. Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, pages 556–562.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Pentti Paatero and Unto Tapper. 1994. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5(2):111–126.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Erik F Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *arXiv preprint cs/0306050*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.