SUMMARY WORKBENCH
Unifying Application and Evaluation of Text Summarization Models

Shahbaz Syed **Dominik Schwabe** **Martin Potthast**

Leipzig University

<shahbaz.syed@uni-leipzig.de>

Abstract

This paper presents SUMMARY WORKBENCH, a new tool for developing and evaluating text summarization models. New models and evaluation measures can be easily integrated as Docker-based plugins, allowing to examine the quality of their summaries against any input and to evaluate them using various evaluation measures. Visual analyses combining multiple measures provide insights into the models' strengths and weaknesses. The tool is hosted at https://tldr.demo.webis.de and also supports local deployment for private resources.

1 Introduction

Automatic text summarization reduces a long text to its most important parts and generates a summary. Usually, a learning-based summarization model is developed in two basic steps: *model development* and *model evaluation*. Given a collection of documents accompanied by one or more human-written (reference) summaries, first a set of features representing the documents is manually created or automatically extracted through supervised learning. The resulting model is then used to generate one or more (candidate) summaries, which are analyzed manually and/or with evaluation measures for their similarity to the reference summaries. These steps are iterated, optimizing the model and its parameters using a validation set. The models that perform best in the validation are selected for evaluation on the test set. With standardized test sets for each document collection, comparisons with models created earlier are reported.

However, these steps are associated with comparatively tedious tasks: During model development, summaries of individual documents are often generated and immediately evaluated to identify deficiencies and improve the model, including comparisons to other models. The latter requires third-party models to be operational despite their heterogeneous software stacks. Such "on-the-fly evaluation" during development entails that candidate and reference summaries as well as source documents are analyzed manually or by automatic measures. This multi-text comparison is often not supported by visualization, although this leads to a better understanding of the content coverage and possible selection biases of a model (Vig et al., 2021; Syed et al., 2021b). The analysis of evaluation results for model selection also benefits from visual support (Tenney et al., 2020). Previous research in the field of automatic summarization has not yet resulted in a unified set of tools for these purposes which is the main goal of this paper.

With SUMMARY WORKBENCH, we introduce the first unified combination of application and visual evaluation environments for text summarization models. Currently, it integrates 15 well-known summarization models (26 variants in total) and 10 standard evaluation measures from the literature. With **FeatureSum**, it also includes a new feature-based extractive summarization model that implements features from the literature predating the deep learning era. Underlying all of the above is a specification and interface that allows easy integration of new models and measures to facilitate large-scale experiments and their reproducibility.

In what follows, Section 2 reviews related work on tools to assist summarization research and development. Section 3 overviews the key design principles of the SUMMARY WORKBENCH, and Section 4 provides a complete overview of all the models and measures hosted to date. Included are general-purpose models, guided models that accept user prompts to guide summary generation, and models tailored to argumentative language and to news articles. A wide range of commonly employed evaluation measures are included, covering both lexical as well as semantic overlap measures.[1]

[1]Source code is available at https://github.com/webis-de/summary-workbench.

Proceedings of the The 2022 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pages 232 - 241

2 Related Work

The development of tools for summarization research has gained momentum recently, and several tools have been presented for (sub)tasks of the two steps above: Tools such as HuggingFace (Wolf et al., 2020), FairSeq (Ott et al., 2019), SummerTime (Ni et al., 2021), TorchMetrics (Detlefsen et al., 2022), SacreROUGE(Deutsch and Roth, 2020), PyTorch Hub,[2] and TensorFlow Hub[3] focus on hosting several state-of-the-art text summarization models and automatic evaluation measures. These tools have significantly improved accessibility to working models. However, only some provide a very minimal interface for inference of summaries and their online/offline comparative analyses. Many authors also choose to share their models independently, be they on GitHub or elsewhere, as standalone repositories instead of integrating with any tools. To lower the bar of (latter) integration as much as possible, SUMMARY WORKBENCH simplifies model and measure integration as plugins (using Docker). In this way, models under development or private ones can be locally compared to others and can be archived together with all their dependencies for reproducibility. Similar efforts have been made in the information retrieval community via the Docker-based toolkit such as Anserini (Yang et al., 2018).

Tools such as LIT (Tenney et al., 2020), SummVis (Vig et al., 2021), and Summary Explorer (Syed et al., 2021b) focus on qualitative model evaluation by providing static visual analyses of the relation between the summary and its source document. SUMMARY WORKBENCH adapts some of their visualizations next to new ones, and complements them with interactive visual analytics for quantitative evaluation according to multiple measures. Users can explore the distribution of scores, select data points of interest and inspect them in relation to the source document to better understand the dataset.

The success of past summarization research and development has relied a lot on in-depth manual error analyses. This being one of the most laborious tasks in every natural language generation evaluation, we believe that visually comparing summaries from multiple models for many different texts, and contextualizing manual review with multiple measures is crucial to both scale up error analysis, and to better understand the capabilities and limitations of the technology. As this still requires juggling many different, incompatible tools, the unified approach of SUMMARY WORKBENCH aims at lowering the bar for scaling up interactive experimentation.

[2] https://pytorch.org/hub/

[3] https://www.tensorflow.org/hub/

3 An Interactive Visual Summarization Model Development & Evaluation Tool

SUMMARY WORKBENCH implements two interactive views corresponding to the two basic summarization model development steps: a *summarization view* and an *evaluation view*.

3.1 Summarization View

Figure 1 shows the summarization view, where multiple extractive/abstractive summarization models can be used to summarize texts, web pages, or scientific documents on demand, controlling for summary length. For scientific documents, relevant sections from a given paper to be summarized can be chosen. Explicit guidance signals for focused summarization can be provided as input to corresponding models (reviewed in Section 4).

Generated summaries (candidates) can be visually inspected for their lexical overlaps (highlighted on demand) with their source document or with another summary. This provides a quick overview of the models' effectiveness at capturing important content as well as any factual errors prevalent in abstractive summarization (Maynez et al., 2020). Additional functionalities include uploading multiple documents to be summarized via a single file, and command line access to all models.

3.2 Evaluation View

Figure 2 shows the evaluation view, where candidate summaries are compared with reference summaries using multiple lexical/semantic content overlap measures. Candidate summaries from multiple models, either generated using the summarization view or uploaded as a file where each example is encoded as $< \texttt{doc}, \texttt{ref}, \texttt{c}_1, \texttt{c}_2, ..., \texttt{c}_n >$ can be evaluated. Lexical/semantic overlap of candidate/reference summaries $\texttt{c}_i$/ref with the source document doc can also be visualized. Computed scores can be neatly exported as CSV or LaTeX tables.

Scores from the chosen evaluation measures can be further explored through an *interactive plotter*. Among other things, the plotter allows users to visually correlate different evaluation measures, identify outliers/challenging source documents or strongly abstractive summaries among the candi-

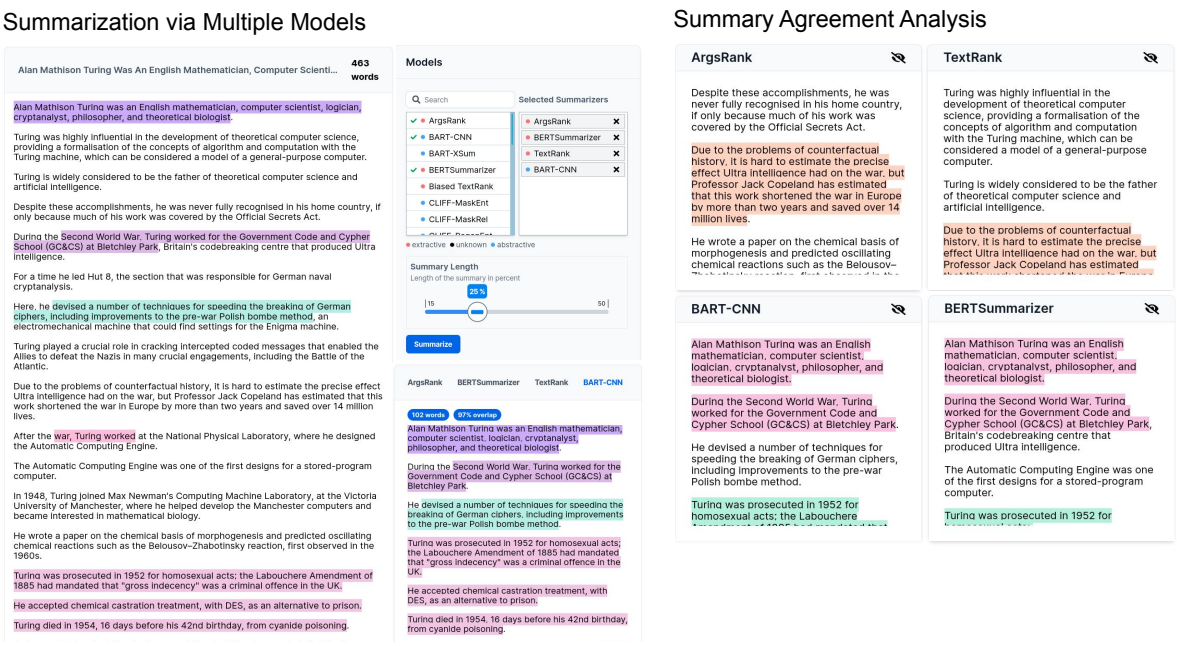

Figure 1: Two key components of the summarization view: On the left, an input text can be summarized via multiple extractive and abstractive summarization models; lexical overlap is highlighted on demand for each candidate summary and can be adjusted for varying n-gram lengths. On the right, content agreement among summaries from different models; any summary can be selected as the reference against which the others can be visually compared.

dates. This facilitates a deeper understanding of the quantitative performance of the models as well as an understanding of the evaluation datasets. Two more use cases of the interactive plotter are explained in Section 5.

3.3 Plugin Server

New summarization models and evaluation measures are integrated as container-based plugins. A model/measure plugin can either be a local directory or a remote Git repository containing specification of dependent software and data (checkpoints, embeddings, lexicons), and implementations of the interfaces SummarizerPlugin and MeasurePlugin. Model metadata such as name, type, version, source, citation, and other custom arguments are provided as YAML configurations. Each plugin runs inside a Docker container with its own server that handles API calls following the OpenAPI specification.[4] This setup allows users to safely self-host the entire application. Developed plugins can be easily shared with the community via DockerHub images or Git repositories. Examples are found in our tool's technical documentation.[5]

[4] https://www.openapis.org
[5] https://webis.de/summary-workbench/

4 Models and Measures

SUMMARY WORKBENCH hosts 15 extractive/abstractive summarization models and 10 lexical/semantic evaluation measures for English text. Each of these is configured as a Docker-based plugin that can be customized and instantiated accordingly. For details on model checkpoints, see Appendix A.

4.1 Summarization Models

We provide a diverse set of models applicable to multiple text domains such as news, argumentative texts, web pages, and product reviews. Model types include extractive, abstractive, supervised, unsupervised, and guided summarization, the latter requiring additional user input.

General-purpose Summarization

Models that work in an unsupervised fashion or leverage external knowledge via contextual embeddings are supposed to be capable of summarizing any kind of text. We provide the following models suitable for general-purpose text summarization.

FeatureSum is our new extractive summarization model which scores a sentence in the text based on a combination of standard features to identify key sentences (Luhn, 1958; Nenkova and McKeown, 2012): TF-IDF, content units (named entities, noun phrases, numbers), position in text, mean lexical

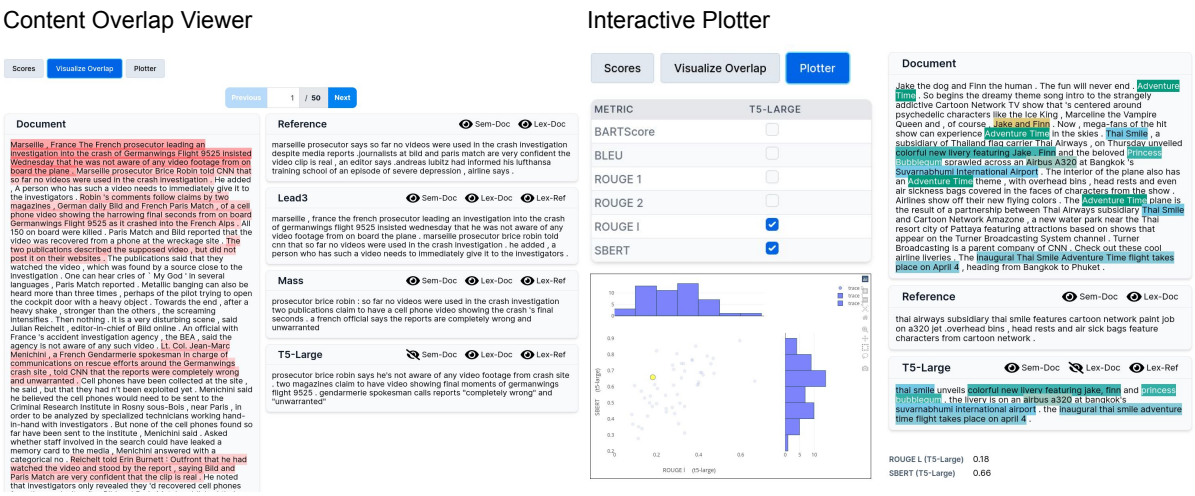

Figure 2: Two key components of the evaluation view: On the left, a text overlap viewer displays content coverage of the summaries in relation to the source document via lexical and semantic overlap (via Spacy embeddings). On the right, an interactive plotter allows selecting examples with specific scores for a combination of evaluation measures. Additionally, the distribution of scores is also shown.

connectivity (number of tokens shared with the remaining sentences), ratio of words that are not stop words, length (relative to the longest sentence in the text), and word overlap with the title. The final score of a sentence is the product of the individual feature values. Sentences are then ranked based on these scores to produce the final summary. Different combinations of these features can be chosen by simply toggling them in the interface. This also allows for dynamically reproducing existing models from the literature provided their specific feature sets are available.

TextRank (Mihalcea and Tarau, 2004) is a graph-based model which employs PageRank (Brin and Page, 1998) on the document graph consisting of sentences as nodes to compute the strength of their connections. Top-ranked sentences within a length budget are taken as the extractive summary. We also provide the two variants **PositionRank** and **TopicRank**, which consider the sentence position and its overlap with topic sentence (document's title or its first sentence) to compute the ranking via PyTextRank (Nathan, 2016).

BERTSum (Miller, 2019) employs contextual embeddings from BERT (Devlin et al., 2019) to extract key sentences in an unsupervised fashion by first clustering all sentence embeddings using k-means (Hartigan and Wong, 1979) and then retrieving those closest to the centroids as the summary.

PMISum (Padmakumar and He, 2021) is an unsupervised extractive model that includes measures to score the relevance and redundancy of the sentences of the source document. These measures are based on pointwise mutual information (PMI) computed by pre-trained language models. Summary sentences are selected via a greedy algorithm to maximize relevance and minimize redundancy.

LoBART (Manakul and Gales, 2021) addresses the input length limitations of transformers (Vaswani et al., 2017) that restrict capturing long-span dependencies in long document summarization. Local self-attention and explicit content selection modules are introduced to effectively summarize long documents such as podcast transcripts and scientific documents.

Longformer2Roberta effectively combines Longformer (Beltagy et al., 2020), developed for processing long documents, and RoBERTa (Liu et al., 2019), a robustly trained BERT model as the decoder, based on leveraging pre-trained checkpoints of large language models (Rothe et al., 2020).

Guided Summarization

The following models accept explicit inputs provided by users to guide the summarization process towards generating user-specific summaries.

Biased TextRank (Kazemi et al., 2020) is an extension of the TextRank model which takes an explicit user input as the "focus", represented via contextual embeddings to guide the ranking of the document sentences. Summary extraction is based on the semantic alignment between the document sentences and the provided focus signal.

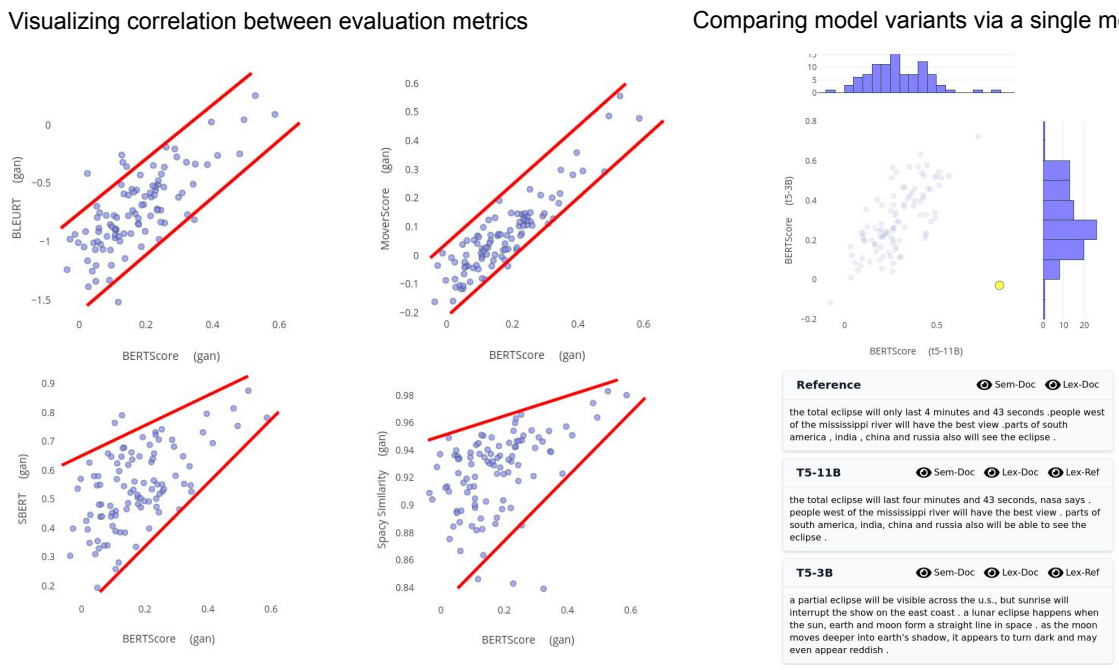

Figure 3: Two example use cases of interactive plotter of the evaluation view: On the left, correlations between pairs of evaluation measures are analyzed. On the right, abstractive summaries from two variants of the T5 model for the chosen data point (highlighted yellow) are shown.

GSum (Dou et al., 2021) is a guidance-based abstractive model that takes different types of external guidance signals: text inputs, highlighted sentences, keywords, or extractive oracle summaries derived from the training data. These signals along with the source text are used to generate focused and faithful abstractive summaries.

Argument Summarization

Summarizing argumentative texts (opinions, product reviews) requires that the model be able to identify high-quality, informative, and argumentative sentences from the text. We provide three models specifically developed for this task.

ArgsRank (Alshomary et al., 2020) is an extractive model for creating argument snippets. It augments TextRank with two new criteria: *centrality in context* and *argumentativeness* to help the model retrieve important and argumentative sentences.

ConcluGen (Syed et al., 2021a) is a transformer model for generating informative conclusions of argumentative texts by balancing the trade-off between abstractiveness and informativeness of the output. It was finetuned on the Conclugen corpus comprised of pairs of argumentative text and a human-written conclusion.

COOP (Iso et al., 2021) is an unsupervised opinion summarization model that employs latent vector aggregation by searching for optimal input combinations of sentence embeddings to address the summary vector degeneration problem caused by simple averaging. Specifically, it finds convex combinations that maximize the word overlap between the source document and its summary.

News Summarization

A majority of the existing summarization models are trained on news datasets, since news have been and are readily available. These models have shown strong performance in creating fluent abstractive summaries (Huang et al., 2020). We provide the following models for summarizing news.

BART (Lewis et al., 2020) is a transformer denoising autoencoder for pre-training sequence-to-sequence models. Its main objective is to reconstruct the source text corrupted by employing arbitrary noising functions (masking text spans, randomly shuffling sentences) which helps the model learn better representations of the source texts for text summarization (Huang et al., 2020).

T5 (Raffel et al., 2020) is a unified text-to-text transformer model that exploits the strengths of transfer learning on a variety of problems that can be modeled as text generation tasks. A task-specific

prefix is added to each input sequence (e.g., "summarize:<document>") that teaches the model to summarize accordingly.

Pegasus (Zhang et al., 2020a) is a transformer model pre-trained with a self-supervised summarization-specific training objective called "gap-sentences generation": important sentences are removed/masked from the source text and must be jointly generated as output from the remaining sentences, similar to an extractive summary.

CLIFF (Cao and Wang, 2021) leverages contrastive learning for generating abstractive summaries that are faithfully and factually consistent with the source texts. Reference summaries are used as positive examples while automatically generated erroneous summaries are used as the negative examples for training the model.

Newspaper3k is an open-source library for extracting news articles from the web which provides a module for extractive summarization that ranks sentences based on keywords and title words.[6]

4.2 Evaluation Measures

Evaluation measures for summarization typically quantify the lexical/semantic overlap of a candidate summary with a reference summary. We provide the following measures covering both.

Lexical Measures

Measures based on lexical overlap return precision, recall, or F1 scores on varying granularities of text between the candidate summary and one or more reference summaries.

BLEU (Papineni et al., 2002) is a standard measure for machine translation adapted for summarization. It includes a brevity penalty to account for length differences while computing n-gram overlap.

ROUGE (Lin, 2004) is the most common measure for summarization which computes precision, recall, and F1 scores based on n-gram overlap, where n-grams include unigrams, bigrams, and the longest common subsequence.

METEOR (Banerjee and Lavie, 2005) aligns a candidate with a set of references by mapping each unigram of a candidate to 0/1 unigrams of the reference based on exact, stem, synonym, and paraphrase matches. It then computes precision, recall, and F9 scores (i.e., weighted harmonic mean, strongly emphasizing recall) based on that.

[6] https://newspaper.readthedocs.io/en/latest/

CIDEr (Vedantam et al., 2015) is a consensus-based measure (originally for evaluating image captioning) which measures the similarity of a candidate against a set of references by counting the frequency of the common n-grams of a candidate.

Semantic Measures

Measures based on semantic overlap compute the semantic alignment between candidates and references at the token/word/sentence level based on their static/contextual embeddings.

Greedy Matching (Rus and Lintean, 2012) aligns a candidate and a reference by greedily matching each candidate word to a reference word based on their embeddings' cosine similarity. Average similarity over all candidate words aligned to reference words and vice versa are computed whose average is the final score.

MoverScore (Zhao et al., 2019) combines contextual embeddings from BERT using the word mover's distance (Kusner et al., 2015) to compare a candidate against a set of references by considering both the amount of shared content as well as the extent of deviation between them.

BERTScore (Zhang et al., 2020b) computes a similarity score for each candidate token with each reference token using contextual embeddings from BERT. The measure is also robust to adversarial modifications of the generated text.

BLEURT (Sellam et al., 2020) is a learned measure based on BERT that models human judgments with a few thousand biased training examples. The model is pre-trained using millions of synthetic examples created via scores from existing measures (BLEU, ROUGE, BERTScore), and textual entailment, for better generalization.

BARTScore (Yuan et al., 2021) uses the weighted log probability of generating one text given another to compute faithfulness (source $\rightarrow$ candidate), precision (reference $\rightarrow$ candidate), recall (candidate $\rightarrow$ reference), and the F1 score.

CosineSim includes two embedding-based cosine similarity measures using Spacy word vectors (Honnibal et al., 2020) and Sentence-BERT (Reimers and Gurevych, 2019).

5 Interaction Use Cases

Figure 3 shows two use cases of the interactive plotter. First, users can analyze any correlation

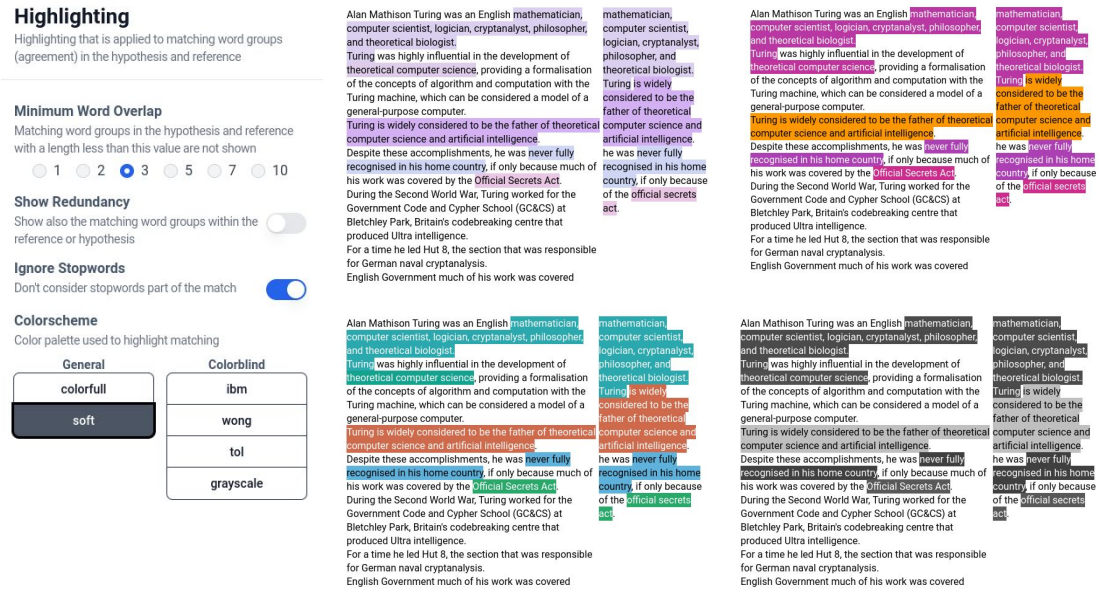

Figure 4: Customization options available for visualization of document and summary overlap. Users can select the minimum word overlap, preserve duplicate words, and ignore stop words to be visualized. Also, they can instantly preview each color scheme and set it as their default. The tool provides colorful, soft gradient-based, and grayscale schemes to account for color blindness.

between two measures of choice for a summarization model. Here, we find that MoverScore and BERTScore have strong correlation as they both employ contextual embeddings from BERT to compute the overlap between candidate and reference summaries. Likewise, we find that the static token embeddings from Spacy have a broader distribution of scores in comparison.

As a second use case, the interactive plotter allows comparing two variants of the same model architecture using any measure. Here, we inspect the T5 model (its 3B and 11B variants) using BERTScore to find that the larger variant generates a summary very similar to the reference while the smaller variant creates a summary that is topically related but not accurate in comparison to the reference.

6 Conclusion

In this paper we present SUMMARY WORKBENCH, a tool that unifies the application and evaluation of text summarization models. The tool supports integrating summarization models and evaluation measures of all kinds via a Docker-based plugin system that can also be locally deployed. This allows safe inspection and comparison of models on existing benchmarks and easy sharing with the research community in a software stack-agnostic manner. We have curated an initial set of 15 models (26 including all variants) and 10 evaluation measures and welcome contributions from the text summarization community. An extension of the tool's features to related text generation tasks such as paraphrasing and question answering is foreseen.

7 Ethical Statement and Limitations

Our tool builds on open source models and evaluation measures contributed by the corresponding authors. We expect all users of our tool to diligently cite the authors of all models and measures when they use them via our tool, instead of just citing our tool. The tool provides direct links to the relevant sources for each hosted summarization model and evaluation measure to facilitate this.

The models and measures may have intrinsic biases, which ideally, our tool may help to identify. However, our tool itself may have biases, especially with respect to its visualizations: Visualization in general is a difficult task, and visual analytics for data analysis in particular may lead to invalid conclusions if the underlying visualization itself is flawed. Although we did our best to avoid any non-standard visualizations and relied on widely used tools to plot them, we caution users of

possible errors from either the dependent libraries or their integration in our tool. Validating a visual analytics tool poses non-trivial research tasks of its own right, which we leave for future work. We do hope that the community will diligently report any errors they may encounter.

To account for color blindness, we strived to provide multiple (soft) gradient-based color schemes and grayscale colors (Figure 4) for all our visualizations. Instant previews of each color scheme are available to help users customize the tool's visuals.

References

Milad Alshomary, Nick Düsterhus, and Henning Wachsmuth. 2020. Extractive snippet generation for arguments. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, pages 1969–1972. ACM.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: an automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization@ACL 2005, Ann Arbor, Michigan, USA, June 29, 2005*, pages 65–72. Association for Computational Linguistics.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *CoRR*, abs/2004.05150.

Sergey Brin and Lawrence Page. 1998. The anatomy of a large-scale hypertextual web search engine. *Comput. Networks*, 30(1-7):107–117.

Shuyang Cao and Lu Wang. 2021. CLIFF: contrastive learning for improving faithfulness and factuality in abstractive summarization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 6633–6649. Association for Computational Linguistics.

Nicki Skafte Detlefsen, Jiri Borovec, Justus Schock, Ananya Harsh, Teddy Koker, Luca Di Liello, Daniel Stancl, Changsheng Quan, Maxim Grechkin, and William Falcon. 2022. TorchMetrics - Measuring Reproducibility in PyTorch.

Daniel Deutsch and Dan Roth. 2020. SacreROUGE: An open-source library for using and developing summarization evaluation metrics. In *Proceedings of Second Workshop for NLP Open Source Software (NLP-OSS)*, pages 120–125. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Zi-Yi Dou, Pengfei Liu, Hiroaki Hayashi, Zhengbao Jiang, and Graham Neubig. 2021. Gsum: A general framework for guided neural abstractive summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 4830–4842. Association for Computational Linguistics.

John A Hartigan and Manchek A Wong. 1979. Algorithm as 136: A k-means clustering algorithm. *Journal of the royal statistical society. series c (applied statistics)*, 28(1):100–108.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python.

Dandan Huang, Leyang Cui, Sen Yang, Guangsheng Bao, Kun Wang, Jun Xie, and Yue Zhang. 2020. What have we achieved on text summarization? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 446–469. Association for Computational Linguistics.

Hayate Iso, Xiaolan Wang, Yoshihiko Suhara, Stefanos Angelidis, and Wang-Chiew Tan. 2021. Convex aggregation for opinion summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pages 3885–3903. Association for Computational Linguistics.

Ashkan Kazemi, Verónica Pérez-Rosas, and Rada Mihalcea. 2020. Biased textrank: Unsupervised graph-based content extraction. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 1642–1652. International Committee on Computational Linguistics.

Matt J. Kusner, Yu Sun, Nicholas I. Kolkin, and Kilian Q. Weinberger. 2015. From word embeddings to document distances. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 957–966. JMLR.org.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training

for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7871–7880. Association for Computational Linguistics.

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.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Hans Peter Luhn. 1958. The automatic creation of literature abstracts. *IBM Journal of Reseach and Devopment*, 2(2):159–165.

Potsawee Manakul and Mark J. F. Gales. 2021. Long-span summarization via local attention and content selection. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 6026–6041. Association for Computational Linguistics.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan T. McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 1906–1919. Association for Computational Linguistics.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing , EMNLP 2004, A meeting of SIGDAT, a Special Interest Group of the ACL, held in conjunction with ACL 2004, 25-26 July 2004, Barcelona, Spain*, pages 404–411. ACL.

Derek Miller. 2019. Leveraging BERT for extractive text summarization on lectures. *CoRR*, abs/1906.04165.

Paco Nathan. 2016. PyTextRank, a Python implementation of TextRank for phrase extraction and summarization of text documents.

Ani Nenkova and Kathleen McKeown. 2012. A survey of text summarization techniques. In *Mining text data*, pages 43–76. Springer.

Ansong Ni, Zhangir Azerbayev, Mutethia Mutuma, Troy Feng, Yusen Zhang, Tao Yu, Ahmed Hassan Awadallah, and Dragomir R. Radev. 2021. Summertime: Text summarization toolkit for non-experts. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, EMNLP 2021, Online and Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 329–338. Association for Computational Linguistics.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Demonstrations*, pages 48–53. Association for Computational Linguistics.

Vishakh Padmakumar and He He. 2021. Unsupervised extractive summarization using pointwise mutual information. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pages 2505–2512. Association for Computational Linguistics.

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 of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA*, pages 311–318. ACL.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

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 and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3980–3990. Association for Computational Linguistics.

Sascha Rothe, Shashi Narayan, and Aliaksei Severyn. 2020. Leveraging pre-trained checkpoints for sequence generation tasks. *Trans. Assoc. Comput. Linguistics*, 8:264–280.

Vasile Rus and Mihai C. Lintean. 2012. A comparison of greedy and optimal assessment of natural language student input using word-to-word similarity metrics. In *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP, BEA@NAACL-HLT 2012, June 7, 2012, Montréal, Canada*, pages 157–162. The Association for Computer Linguistics.

Thibault Sellam, Dipanjan Das, and Ankur P. Parikh. 2020. BLEURT: learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7881–7892. Association for Computational Linguistics.

Shahbaz Syed, Khalid Al Khatib, Milad Alshomary, Henning Wachsmuth, and Martin Potthast. 2021a.

Generating informative conclusions for argumentative texts. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 3482–3493. Association for Computational Linguistics.

Shahbaz Syed, Tariq Yousef, Khalid Al Khatib, Stefan Jänicke, and Martin Potthast. 2021b. Summary explorer: Visualizing the state of the art in text summarization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, EMNLP 2021, Online and Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 185–194. Association for Computational Linguistics.

Ian Tenney, James Wexler, Jasmijn Bastings, Tolga Bolukbasi, Andy Coenen, Sebastian Gehrmann, Ellen Jiang, Mahima Pushkarna, Carey Radebaugh, Emily Reif, and Ann Yuan. 2020. The language interpretability tool: Extensible, interactive visualizations and analysis for NLP models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, EMNLP 2020 - Demos, Online, November 16-20, 2020*, pages 107–118. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 4566–4575. IEEE Computer Society.

Jesse Vig, Wojciech Kryscinski, Karan Goel, and Nazneen Fatema Rajani. 2021. Summvis: Interactive visual analysis of models, data, and evaluation for text summarization. *CoRR*, abs/2104.07605.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Peilin Yang, Hui Fang, and Jimmy Lin. 2018. Anserini: Reproducible ranking baselines using lucene. *ACM J. Data Inf. Qual.*, 10(4):16:1–16:20.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 27263–27277.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. 2020a. PEGASUS: pre-training with extracted gap-sentences for abstractive summarization. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 11328–11339. PMLR.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020b. Bertscore: Evaluating text generation with BERT. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M. Meyer, and Steffen Eger. 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 and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 563–578. Association for Computational Linguistics.

A Model Details

Summarizer	**Model**
BERTSummarizer	distilbert-base-uncased
LoBART	podcast_4K_ORC
Longformer2Roberta	patrickvonplaten/longformer2roberta-cnn_dailymail-fp16
ConcluGen	dbart
CLIFF	pegasus_cnndm
COOP	megagonlabs/bimeanvae
BART	facebook/bart-large
Pegasus	google/pegasus
T5-Base	huggingface.co/t5-base
Evaluator	**Model**
BARTScore	facebook/bart-large-cnn
Spacy Similarity	en_core_web_lg
SBERT	roberta-large-nli-stsb-mean-tokens
BLEURT	bleurt-base-128
BERTScore	roberta-large-mnli
Greedy Matching	glove.6B.300d
MoverScore	MoverScoreV1