Praktikum Final Report: Abstractive Summarization

Diego Miguel Lozano
Technical University Munich
diego.miguel@tum.de

Harun Eren
Technical University Munich
harun.eren@tum.de

Ralf Kinkel
Technical University Munich
ralf.kinkel@tum.com

Abstract

Creating good abstractive summaries remains a challenging problem in Natural Language Processing. Part of the challenge is that the evaluation of summaries is a hard problem in itself and metrics currently in use are insufficient to accurately reflect the quality of summaries. This makes navigating the space of potential architectures and hyperparameters for a summarization model even more difficult than usual in Machine Learning. Our contribution is a new approach for evaluating summaries that we hope will lead to better evaluation and consequentially faster exploration of such models. We additionally used our evaluation metric as a loss to train a summary generator in a novel GAN approach. This second research direction did not yield promising results, yet we hope that other researchers might either solve potential problems we overlooked or avoid the approach to avert fruitless endeavors.

1 Introduction

Humanity puts out exponentially more data each year. To handle this information overload it is critical to improving how we select and condense information by extracting the most important parts. While Information Retrieval (IR) algorithms and models like those used in search engines greatly help perform information selection, automatic information condensation is much less advanced. Creating concise summaries would help consume information in a textual form far more efficiently.

Commonly, there exist two types of summaries. Extractive summaries consist of subsets of sentences from the original text, while in abstractive summaries the meaning of the text is extracted first and then a summary is generated from an abstract representation (Lin and Ng, 2019). This summary often presents paraphrases to some extent, although an abstractive summary can take any form so that the space of extractive summaries is contained in the space of abstractive summaries. While abstractive summarization is often more challenging than extractive summarization, it better approximates human summaries. In recent years the endeavor of automating summarization has gained more traction as neural networks revolutionized Natural Language Processing (NLP). This also enabled a shift from extractive summaries to abstractive summaries (Lin and Ng, 2019).

Summaries can be evaluated either manually or automatically. Manual evaluations by human judges reflect the goal of creating summaries for humans perfectly but suffer from scalability problems. In addition, it also makes comparisons between methods harder, as different groups of judges vary in how they rate and what they value in a summary. On the other hand, automatic summarization suffers from the fact that it is very hard to define a general notion of what constitutes a good summary.

As a consequence, the automatic metrics currently in use suffer from a plethora of different problems (Fabbri et al., 2021a).

In this work, we propose a new evaluation tool for abstractive summaries that have the potential to reflect the quality of summaries better than the current metrics. We finetuned a model for sentence similarity to assign similar embeddings to match texts and summaries so that a similarity score can be used for evaluation[1]. Afterward, we tried to use this similarity score as a loss function to train a summary generator in a novel generative adversarial network (GAN) variant but did not yet succeed in creating a good generator. Reasons for this are discussed in Section 5.

[1]One could question how training a discriminator as evaluation tool could be useful, as it should not be able to surpass summary generators with similar architecture and training data in Natural Language Understanding. We argue that this approach could still be useful, as evaluation is much more tractable than a generation. This means that it should be easier to create strong discriminators than strong generators. The discriminator can then be used for evaluation to serve as a stepping stone for better summary generators.

Our code base can be found here.

2 Chronology and related work

We originally started our project with the idea of using a GAN where we would iteratively train a summary generator that would generate summaries from text and a discriminator that would rate these summaries.

After initializing the discriminator we noticed its potential as an evaluation tool for abstractive summaries and shifted some of our efforts into analyzing this potential and training variants of it.

2.1 Ideation stage: GANs for abstractive summarization

Generative adversarial networks (GANs) were proposed for generative tasks (Goodfellow et al., 2014) in Machine Learning and have enjoyed great success in Computer Vision, but face greater challenges in the deployment for NLP (Wang et al., 2019). We have considered using a GAN approach for our summarization task as it promises to give richer feedback for training than the typical next-word prediction approach. In this approach, only one token is treated as valid, while a discriminator used in a GAN approach would give feedback on the suitability of multiple tokens.

A huge part of the challenge of applying GANs for NLP is the following: In a typical GAN pipeline, the output of the generator is the input of the discriminator. The immediate output of our summary generator, however, is a probability distribution over a finite vocabulary and the discriminator expects a summary consisting of a discrete sequence of tokens. In language models, various sampling strategies can be used to go from this probability distribution to generated text. The problem is that, in general, this sampling step is non-differentiable.

2.1.1 Tackling non-differentiability

There are various mathematical approaches to tackle differentiability problems regarding sampling. A very well-known reparametrization trick is used in Variational Autoencoders (VAEs). Assume that a continuous or discrete input variable x is generated by a random process involving an unobserved continuous random variable z. We rewrite z by parameterized distribution and deterministic expression of $x|z$ so that we can approximate x by a differentiable parameterized function (Kingma and Welling, 2013). Similarly, the Gumbel-Softmax trick is used to represent discrete categorical distribution by a parameterized continuous differentiable distribution (Jang et al., 2016).

Another approach is to combine the GAN architecture with Reinforcement Learning. In one of these attempts, bi-directional LSTMs are used as the generator for predicting next token probabilities, a one-dimensional multi-layered CNN is used as a discriminator for binary text classification, and the reward policy of the RL mechanism is updated by the discriminator and also impacts the loss of the generator (Liu et al., 2017). There is a similar but more complex attempt using attention and an encoder-decoder architecture, that also employs RL within the GAN (Vo, 2021).

Reinforcement Learning is the method we saw most often for solving the non-differentiability problem in GANs. However, it presents the issue that it is very complex and computationally expensive, and the resulting generators currently lack state-of-the-art models. We introduce a new method where we sample the top-k next-token probabilities and pass them to the discriminator to get a score for each of them, which then we use to create a loss for the generator. Details are described in Section 3.2.1.

2.2 Execution stage: GANs for abstractive summarization

In recent years there have been vast advancements in NLP, most notably Deep Learning with the introduction of LSTMs (Hochreiter and Schmidhuber, 1997) has been a great step for the field and more recently transformers utilizing self-attention (Vaswani et al., 2017) show vast improvements across a variety of tasks, including abstractive summarization [2]. For this reason, we focused on transformer models for our discriminator. We first checked the performances of the available state-of-the-art models like OpenAI's GPT3 (Brown et al., 2020) and Google's T5 (Raffel et al., 2019) qualitatively with Wikipedia article inputs. While these show impressive results, we decided to start with smaller, less performant models to be able to iterate through ideas faster, especially since our work aims to explore a new direction in the field instead of achieving state-of-the-art in the chosen task. We decided to use transfer learning by employing pre-

[2] Benchmarks for abstractive summarization are dominated by transformer-related architectures, see for example https://paperswithcode.com/sota/abstractive-text-summarization-on-cnn-daily

trained models.

For the discriminator, we first started with "all-mpnet-base-v2"[3]. for sentence similarity, as we believed that it would be a good basis for learning similarity between texts and summaries, and the output was already given as embeddings. We first refined the model by applying contrastive training to it (see Section 3.1.1). After this initial training, we recognized the potential of the discriminator as a metric outside the GAN context and started with experimentation (see Section 4).

For the generator, we used a more efficient variant of BART, DistilBART ("xsum-9-6"). When trying to combine both in the GAN loop we ran into a problem with tokenizer mismatch explained in Section 3.1.3. We then switched discriminators to slightly modified BART-based models.

The quality and quantity of training data are crucial factors for the performance of machine learning models. We have chosen the Extreme Summarization (XSum) dataset, Wikipedia Summarization Dataset (WSD), and CNN-Daily Mail dataset as candidates for training and evaluating our model (Narayan et al., 2018; Fatima and Strube, 2021; Nallapati et al., 2016). We implemented data processing and data loading for these datasets, analyzed them, and later decided to continue working only with the CNN-Daily Mail dataset. The datasets, their analysis, and selection process are explained in detail in the Dataset section.

2.3 Related work: Metrics

Some of the proposed models use human-written feedback for either training or evaluation of their models. This is very costly and difficult to apply for large-scale data and problems. For evaluation, the widely used metrics are ROUGE scores which check the appearances of n-grams of the original text, ROUGE-1 checks for unigrams, ROUGE-2 checks for bigrams, etc. However, the family of ROUGE metrics are not entirely suited for abstractive summary evaluation, since these metrics do not consider semantic relationships. For example, if we replaced some words in a good summary with synonyms not appearing in the original text, the scores given by these metrics would drop even though the quality should remain the same. Therefore, there is ongoing research to find alternative metrics. In some studies, embeddings are utilized to capture semantics: ROUGE-WE is a rouge metric enhanced with word2word embeddings (Ng and Abrecht, 2015), BERTScore metric computes scores based on the similarities between BERT-based tokens (Zhang et al., 2019). The discriminator we trained in our architecture is able to distinguish the original ground-truth summary from others. Therefore if the discriminator model works as expected, it can be used as a metric for summary evaluation, capturing semantic relationships.

[3]Huggingface model available at https://huggingface.co/sentence-transformers/all-mpnet-base-v2.

3 Technical aspects

This section aims to give a deeper insight into the model architectures and training paradigms used.

3.1 Discriminator initial training

The general approach we took for training the discriminator is choosing one model D that outputs one embedding vector for each text or summary passed to it. The embedding vector is then L2-normalized. Those normalized embeddings can be multiplied and added to receive a similarity score in the range $1 \geq s \geq 1$. We then train the discriminator in a way that matching original texts and summaries have a high similarity score and mismatches have low similarity scores.

3.1.1 Discriminator training paradigm

We used the CNN-Dailymail dataset (Nallapati et al., 2016) containing text-summary pairs for contrastive training in a similar approach to the one used in (Radford et al., 2021).

Here a batch with size b of text-summary pairs is passed through the discriminator and a vector of embeddings with $shape = (b, e)$ is formed for the original texts and the summaries each. Those vectors are multiplied and summed across the embedding axis so that a similarity matrix of $shape = (b, b)$ is formed. A symmetrical cross-entropy loss is then applied with $label = 1$ for matching pairs that are at the diagonal of the similarity matrix. An illustration of the process is shown in Figure 1.

Contrastive training is usually used with GPU clusters enabling large batch sizes to fit in the GPU memory. This way there is a high likelihood of more similar non-matching texts and summaries in the same batch, which is more challenging for the discriminator. The increased difficulty translates into a better discriminator after training (Chen et al., 2020). Because our computational resources

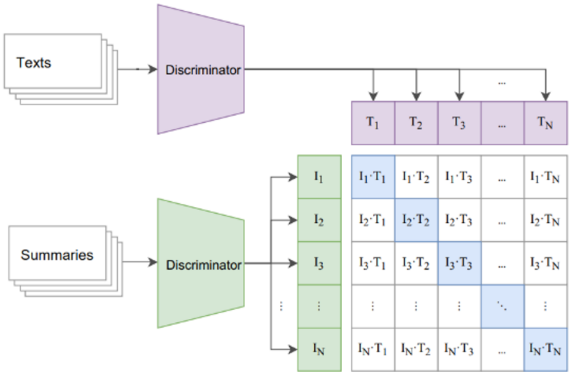

Figure 1: Contrastive training of the discriminator: A batch of texts and a corresponding batch of summaries are passed through the discriminator to get embeddings. The embeddings are L2-normalized, then multiplied and summed across the embedding axis to gain a similarity matrix. The diagonal is shown in light blue representing matching texts and summaries. The figure is adapted from (Radford et al., 2021).

are more limited, our batch sizes were always in the single digits. To still provide an appropriate challenge for the discriminator, we extended our paradigm with data augmentation to include more difficult mismatches.

3.1.2 Discriminator data augmentation

The data augmentation aims to create additional non-matching summaries using similar words and phrases as the original summary and text. We used 3 different augmentations to create these.

- Random: Create a new summary by iterating through words in the original text and adding them to the new summary based on $chance = length(text)/length(summary)$.
- Shuffle: Shuffle the original summary randomly.
- Replace: Create a new summary by replacing every third word in the original summary with a random word from the original text.

The new summaries are added to the matrix changing it to $shape = (b, b + 3)$. In this setting, we also change the symmetrical cross-entropy loss to a regular cross-entropy loss.

3.1.3 Discriminator models

We used 3 different model architectures for the discriminator, we started with a pre-trained model for each architecture.

We first chose a model for sentence similarity as the most similar task we could find, specifically "all-mpnet-base-v2" (Unknown, 2021). While the model yielded good results, we ran into two problems. Firstly, it is pretty big which leads us to only being able to train with $batchsize = 3$ without data augmentation and $batchsize = 1$ with data augmentation and also complicated generator training because of tokenizer mismatch between discriminator and generator. This mismatch could lead to tokens being generated not having correspondence with the discriminator and is difficult to compensate.

We then switched to BART models for the discriminator to have matching tokenizers with the generator. We used BART-base (Lewis et al., 2019) for the second discriminator model and used the encoder from "DistilBART-xsum-9-6" (Shleifer, 2021) as our third discriminator model. All models were first trained without data augmentation. With data augmentation, the GPU memory requirements increased a lot leading us to only train the smallest distilbart encoder-based discriminator. The results can be seen in Section 4.

3.2 GAN Pipeline

The idea of the pipeline is to iterate between training the summary generator ("the summarizer") with a fixed discriminator and training the discriminator with a fixed summarizer. In the first step, the summarizer learns to output summaries the discriminator ranks highly and in the second step, the discriminator learns to differentiate between generated summaries and ground-truth summaries and assign lower scores to the generated ones. The goal is to have a min-max game between the two models where both get better each iteration. The principle is based on (Goodfellow et al., 2014).

3.2.1 Training of the Generator

The process of training the summarizer is illustrated in Figure 2. The summarizer takes the original text and all previously generated tokens as input and outputs the next token probabilities. When first run, the generated summary will only contain the BOS (Beginning of Sequence) token. Then, at each step, the similarity between the discriminator embeddings of the original text and the generated summary is used as a loss to train the generator.

As mentioned in Chronology and related work, this loss cannot be used directly, as the sampling process that is applied on the outputs of the gen-

erator, before being passed to the discriminator, is non-differentiable. We circumvent this problem by passing the discriminator a batch of possible next-token candidates, sampled with top-k, and letting it rank them according to the similarity of each of the k embeddings from the resulting summaries with the embedding of the original text. With these rankings we can create a new objective for the summarizer output that is then used in a cross-entropy loss, bypassing the non-differentiable sampling step.

The way the new objective is created is by re-assigning the probability weight on the sampled tokens according to the similarity scores while making the objective equal to the summarizer output for all non-sampled tokens. This way a signal for change is only given for those tokens the discriminator could score.

As base model we used "sshleifer/distilbart-xsum-9-6" [4].

3.2.2 Discriminator training

Once the generator has been trained, we perform the last step in the GAN loop, which is the training of the discriminator. Here we keep the summarizer fixed and try to minimize the discriminator embedding similarity between the original text and the generated summary and maximize the similarity between the original text and the ground-truth summary. This way, the discriminator will get better at discerning between original and generated summaries.

Figure 3 offers a visual representation of the process.

3.3 Dataset

The quality and quantity of data are crucial for performance in machine learning models in NLP. This section aims to analyze the summarization datasets and our selection criteria.

3.3.1 Wikipedia Summarization Dataset

Our original goal was to improve the abstractive summarization of Wikipedia articles specifically. Because of this, we were interested in datasets that had a similar distribution to Wikipedia articles. For that reason, we decided to use the "Wikipedia Summarization Dataset"[5]. It was originally developed for monolingual and English-German cross-lingual summarization. The dataset uses the "abstract" sections of Wikipedia articles as summaries and the rest of the articles as texts. After analyzing the dataset in more detail we noticed that these abstracts were not good proxies for summaries as they often do not correspond to the text in a meaningful way. We use one example article from the dataset to illustrate the problem. The article about the mathematician Paul Stäckel has two sections: the abstract and his works. The abstract explains the mathematician's life and career and the other section is a list of names of his books and papers. In the dataset, the list of his papers and books is the original text, and his biography is the summary. Summary and original text are therefore almost completely unrelated. There are other examples, such as an article about a tool, where the abstract explains everything about the tool, and the rest of the article explains its applications. Because these types of data samples are no rare exceptions but actually widespread in the dataset, we decided to switch our focus to other datasets.

3.3.2 XSum Dataset

The Extreme Summarization dataset is one of the biggest summarization datasets with 204,045 text-summary samples[6]. It consists of BBC articles and their summaries. The length of texts and summaries appears to be inconsistent, especially for shorter texts. Because XSum and CNN/DailyMail are very similar, we decided to finally focus on the larger but more consistent CNN/DailyMail dataset.

3.3.3 CNN-Daily Mail Dataset

The CNN-DailyMail dataset has about 313,000 samples of articles and human-written summaries from CNN and DailyMail newspapers[7]. It has been one of the most important datasets for training and benchmarking summarization models in recent years. We utilized this dataset heavily in training and evaluation since it is big, turned out to be relatively consistent after some preliminary analysis, and provided an opportunity to compare with other models.

We preprocessed the dataset before training by removing samples from the dataset based on a minimum threshold of word counts and a model-dependent maximum threshold of the token length.

[4] Huggingface model available at https://huggingface.co/sshleifer/distilbart-xsum-9-6.

[5] The Wikipedia Summarization Dataset is accessible at https://github.com/mehwishfatimah/wsd.

[6] XSum dataset is accessible at https://huggingface.co/datasets/xsum.

[7] CNN-DailyMail dataset is accessible at https://huggingface.co/datasets/cnn_dailymail.

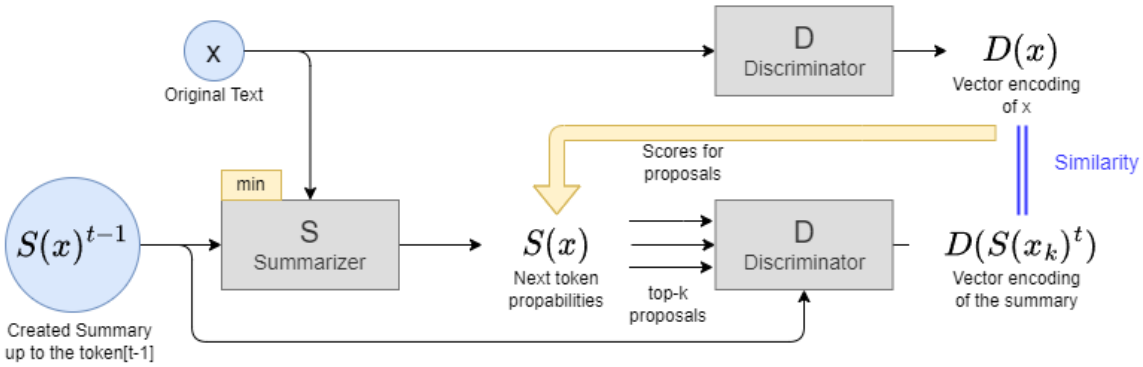

Figure 2: Overview of the generator training. At each step, the discriminator receives the generated summary up to that step, as well as the list of top-k next-token candidates. These tokens are then ranked according to the discriminator criteria. In order to improve efficiency, the generated summary is cached so only the evaluation of the candidate tokens has to be run multiple times.

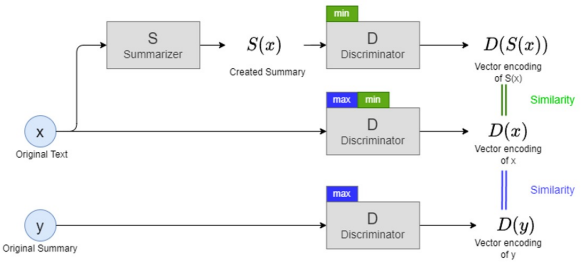

Figure 3: GAN Pipeline training overview.

The word count removal is done to throw out some outliers like samples where the summaries are longer than the texts. The token length threshold is to avoid complications by cutoffs at maximum sequence length for the transformers we used. For example, a text with 2000 tokens could have much of its most relevant content in the second half and a perfect corresponding summary. A discriminator with a max sequence length of 1024 could only look at the first half of the text, with the summary referring to parts of the text the discriminator does not see. This way the discriminator would get very misleading training signals.

4 Results

Our results can be divided into two categories. On the one hand, we measured how well our discriminator performs as an evaluation metric for abstractive summaries. On the other hand, we also analyzed how well the GAN pipeline – Discriminator + Generator –, works as an abstractive summarization tool.

4.1 Discriminator

We evaluated our discriminator on the CNN/Daily Mail Dataset and compared the results with several flavors of the ROUGE score, namely Rouge-1, Rouge-2, and Rouge-L. Table 1 collects the averaged results over 3,000 samples of the test split, as well as the standard deviation. Note that lower scores are not always bad here, as summaries given in the CNN/Daily Mail dataset are no perfect ground-truth.

Furthermore, we also manually annotated a toy dataset (5 samples) with the goal of testing our evaluator in a more comprehensive way, by creating a series of summaries with emphasis on certain characteristics. First, we came up with two categories of correct summaries:

- Ground truth: A correct summary with no special modifications.
- Synonyms: The ground-truth summary replacing some words with synonyms.

Nevertheless, special focus was placed on incorrect summaries, creating a greater number of categories for it[8]:

- Unrelated: A summary that doesn't correspond with the original text at all.
- Nonsensical: A summary that is related to the original text but makes no sense semantically.

[8]This also reflects the fact that there are many more bad summaries than good summaries, as good summaries occupy a very small space in all possible word sequences.

	MPNet v2	Bart base	DB enc.	DB enc.+aug.	Rouge-1	Rouge-2	Rouge-L
Mean	0.93	0.91	0.87	0.59	0.87	0.47	0.63
Std. Dev.	0.07	0.09	0.08	0.15	0.08	0.17	0.13

Table 1: Averaged results over 3,000 individual evaluations on the CNN/Daily Mail dataset. "DB enc." stands for "Distilbart Encoder", and "aug." refers to using augmentation.

- Keywords: A summary that includes a lot of words from the original text, but misses the point of it.
- Bad grammar: A summary that could be appropriate but contains several grammar mistakes.
- Negations: A correct summary but negated, so that it conveys the opposite meaning.

The results are presented in table 2. Our discriminator models perform very well with correct summaries, capturing semantics better than the Rouge metrics, which suffer a score drop on the synonym summaries. However, some of the categories (e.g., bad grammar or negations) seems to be especially conflicting for our models, reporting high scores incorrectly. We believe this stems from the fact that pre-trained models are not well suited to detect dissimilarity between opposite sentences, most likely due to the lack of this kind of sentence in their training sets. To alleviate this we could further fine-tune our models to detect these nuances. The first step in this direction was done with our data augmentation. The model trained with it improved on the one trained without it on nonsensical and bad grammar summaries without trading in performance in other areas. Contrastive training can also be used in conjunction with this data augmentation.

4.2 GAN Pipeline

As we mentioned previously, the two goals of our pipeline were training a good evaluator model (the discriminator), but also being able to train an abstractive summarizer (the generator). Regarding the latter, we were able to integrate the discriminator and generator together into a unified pipeline.

Nonetheless, after several training sessions with many variations of hyperparameters, we have not been able to improve on the pre-trained baseline, having encountered two main problems. The first of them is related to time constraints. On a 16 GB GPU, running a single loop takes a day, even without many epochs. The second – the generator training is very unstable, even after a few hundred samples, the generator collapses to a state where it repeatedly outputs meaningless word sequences, no matter the sampling strategy.

5 Discussion

The main parts we discuss here are how to improve on our work in further research and the relevancy of our results.

5.1 Relevancy

We think that the results of the discriminator look very promising, especially considering the many constraints on resources and data we faced that could be corrected in future work. With this step we hope that the gap in evaluation quality between automatic evaluation metrics and manual evaluation can be closed, to allow researchers working on abstractive summarization to iterate through ideas fast without needing to spend hours of work or thousands of dollars on manual evaluation of results. Different avenue for improvements are discussed in the next section.

We hope that the negative results we had on our GAN approach still help researchers to focus their effort on either avoiding this avenue or using the code we provide to fix potential issues we might have overlooked.

5.2 Future research

The discriminator could be improved by training with larger models, as we only used very small ones in the context of NLP, for that purpose more computing and training time would be helpful. One limitation to using the discriminator as a metric is reasonable inference time, which should be considered before choosing to train on very large models.

Training on a wider variety of high-quality datasets[9] should improve performance and make

[9]Even CNN/DailyMail has many flawed ground-truth summaries, lacking some notable points from the original text, or directly including completely irrelevant information. This has been discussed in (Fabbri et al., 2021b), where researchers showed that summaries generated by BART outperform the ground-truth summaries in terms of quality. Higher quality datasets would of course not only help our efforts but the whole field.

		MPNet v2	Bart base	DB enc.	DB enc.+aug.	Rouge-1	Rouge-2	Rouge-L
Good Summ.	Ground Truth	0.96	0.95	0.93	0.95	0.88	0.64	0.78
	Synonyms	0.96	0.92	0.94	0.95	0.70	0.38	0.56
Bad Summ.	Unrelated	0.10	0.21	0.35	0.38	0.26	0.01	0.18
	Nonsensical	0.85	0.82	0.89	0.51	0.84	0.21	0.51
	Keywords	0.41	0.63	0.63	0.61	0.68	0.24	0.58
	Bad grammar	0.96	0.91	0.93	0.69	0.82	0.48	0.69
	Negations	0.94	0.92	0.91	0.91	0.68	0.41	0.59

Table 2: Results of the different metrics on our dataset.

the discriminator better able to generalize. We also only looked into data augmentation very shortly and there are definitely many useful data transforms we have not employed.

One particularly interesting avenue could be a comparison of a discriminator-based metric with several other metrics, also including metrics with slightly more similar approaches to ours like BERTScore and Rouge-WE metrics that utilize BERT-based embedding and word2word embeddings and proposed to tackle the same issue of ROUGE metrics (Zhang et al., 2019; Ng and Abrecht, 2015). By looking at the correlation of all these metrics with more ambiguous metrics humans use to rate summaries. One such set of metrics are "Coherence", "Consistency", "Fluency" and "Relevance" as used in (Fabbri et al., 2021a). Going one step further a model could be trained to get a text and a corresponding summary as input and be trained to predict these 4 human-centric metrics.

6 Conclusion

In this work, we presented a discriminator trained with contrastive learning for the evaluation of abstractive summaries and we presented a new approach for bypassing the non-differentiability problem in training GANs for NLP.

While we have not met success with the second endeavor, we could show some first promising results utilizing it as an evaluation tool and we would be excited to see some of the avenues to improve it explored. A good evaluation tool would help in the development of better generators and might even be used for automatic grading of the summaries written by students further down the road.

References

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

Alexander R Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2021a. Summeval: Re-evaluating summarization evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409.

Alexander R. Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2021b. SummEval: Re-evaluating Summarization Evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409.

Mehwish Fatima and Michael Strube. 2021. A novel Wikipedia based dataset for monolingual and cross-lingual summarization. In *Proceedings of the Third Workshop on New Frontiers in Summarization*, pages 39–50, Online and in Dominican Republic. Association for Computational Linguistics.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. *Advances in neural information processing systems*, 27.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with gumbel-softmax.

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Hui Lin and Vincent Ng. 2019. Abstractive summarization: A survey of the state of the art. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 9815–9822.

Linqing Liu, Yao Lu, Min Yang, Qiang Qu, Jia Zhu, and Hongyan Li. 2017. Generative adversarial network for abstractive text summarization. *CoRR*, abs/1711.09357.

Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. *CoRR*, abs/1808.08745.

Jun-Ping Ng and Viktoria Abrecht. 2015. Better summarization evaluation with word embeddings for ROUGE. *CoRR*, abs/1508.06034.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer.

Sam Shleifer. 2021. Distilbart. `https://huggingface.co/sshleifer/distilbart-xsum-9-6`. Accessed: 2022-08-20.

Author Unknown. 2021. Sentence transformer: all-mpnet-base-v2. `https://huggingface.co/sentence-transformers/all-mpnet-base-v2`. Accessed: 2022-08-20.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Tham Vo. 2021. Sgan4absum: A semantic-enhanced generative adversarial network for abstractive text summarization.

William Yang Wang, Sameer Singh, and Jiwei Li. 2019. Deep adversarial learning for NLP. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Tutorials*, pages 1–5, Minneapolis, Minnesota. Association for Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert.