Summarizing Legal Rulings: Comparative Experiments

Diego de Vargas Feijo
Institute of Informatics
Federal University of Rio Grande do Sul
http://www.inf.ufrgs.br/
dvfeijo@inf.ufrgs.br

Viviane Pereira Moreira
Institute of Informatics
Federal University of Rio Grande do Sul
http://www.inf.ufrgs.br/
viviane@inf.ufrgs.br

Abstract

In the context of text summarization, texts in the legal domain have peculiarities related to their length and to their specialized vocabulary. Recent neural network-based approaches can achieve high-quality scores for text summarization. However, these approaches have been used mostly for generating very short abstracts for news articles. Thus, their applicability to the legal domain remains an open issue. In this work, we experimented with ten extractive and four abstractive models in a real dataset of legal rulings. These models were compared with an extractive baseline based on heuristics to select the most relevant parts of the text. Our results show that abstractive approaches significantly outperform extractive methods in terms of ROUGE scores.

1 Introduction

Text summarization is the task of producing a condensed representation of an input text, keeping the most relevant information. A summary should be concise, fluent, and contains paraphrased versions of the input text with a reduced length.

There are two approaches to text summarization. The first, known as *extractive*, works by selecting entire sentences directly from the source text. This has been the most widely used solution for a number of years (Luhn, 1958; Edmundson, 1969; Erkan and Radev, 2004a; Mihalcea and Tarau, 2004). Extractive methods typically work by simply (i) scoring phrases or sentences to determine the most relevant; and (ii) selecting the top scoring sentences to compose the summary. Often, the sentences are arranged in the same order of occurrence in the original text in an attempt to preserve the ideas and meanings of the sentences. The scoring function should capture how well the selected sentences represent the text and cover the topics present in the text. The lack of connectives may cause the impression that the generated summary does not have a logical flow.

The second approach, known as *abstractive*, aims to extract the main concepts or ideas from the text and generate a new condensed version different from the original. In this case, the model must learn how to write sentences in a logical flow. Using this approach, it is possible to paraphrase the original and use words that did not occur in the source. This is much more similar to the way a human would create a summary. Most recent research has focused on this approach.

With the increasing availability of data, the need for summarization is felt in many areas. This is especially true in the legal area as the texts are usually lengthy. Law operators are expected to keep updated with important information ranging from news, jurisprudence changes, and rulings from many courts.

This important information amounts to huge volumes of data which cannot be processed by humans. Each ruling from the Brazilian Supreme Court typically has more than 2,000 tokens on average. Hence, it is necessary to focus on the essential portions of each topic and extract just the information that is necessary, leaving the details aside. With that in mind, Courts usually provide extracts, with about 200 tokens on average, of their most important decisions summarizing the main topics discussed and the final outcomes. This provides an easier way to find relevant information without needing to read the whole texts.

Currently, these legal summaries are generated by humans, in a process that is time-consuming and expensive. Human summarizers need to have a good knowledge of the subject to extract the

Proceedings of Recent Advances in Natural Language Processing, pages 313–322,
Varna, Bulgaria, Sep 2–4, 2019.
https://doi.org/10.26615/978-954-452-056-4_036

main topics that should appear in the summary. Another important issue with manually created summaries is the lack of standardization. Each specialist from each Court has their own writing style. A standardized way of writing is desirable as it would provide more homogeneous summaries. For these reasons, the use of abstractive approaches is especially appealing in this area.

The summarization of legal texts differs remarkably from mainstream work in text summarization, which is mostly devoted to summarizing news articles, headlines or tweets. Legal texts are generally lengthier and typically contain complex vocabulary and expressions. Also, the order of the words in some expressions can make a big difference in their meaning, *e.g.,"denied an appeal that had accepted"* is very different from *"accepted an appeal that had denied"*.

In this paper, we investigate the suitability of extractive and abstractive approaches in summarizing legal rulings. Given that the vast majority of works in this area focused solely on news datasets, we believe that testing summarization on a new domain is important given the different nature of the input documents. Fourteen approaches were tested over a real dataset containing 10K rulings from the Brazilian Supreme Court (Feijó and Moreira, 2018). Thus, another contribution is using a language that is not typically included in summarization experiments.

The remainder of this paper is organized as follows. Section 2 discusses text representations and introduces the problem of out of vocabulary tokens. Section 3 revises recent works on abstractive summarization. Section 4 describes the dataset and how it was prepared to be used in the experiments. Section 5 presents a simple heuristic-based extractive summarizer that we proposed to serve as a baseline. Sections 6 and 7 describes the algorithms and models that were investigated here. Section 8 discusses our results and findings. Finally, Section 9 concludes this paper and points out possibilities for future work.

2 Text Representation

Representing texts using a sequence of words is useful for exchanging information between humans. However, when using neural models, we need to convert the text into numbers that could be used as input into the neural network.

The usual way of generating a text representation is to split the text and generate a vocabulary from the training data keeping the most frequent tokens. Even considering that the vocabulary would be extracted from a single language, if we consider that texts have upper and lower case characters, numbers, dates, *etc.* the vocabulary usually becomes quite large, frequently with hundreds of thousands of different tokens.

A large vocabulary is a problem because the output layer of the neural network must have its size. This means that the probability of choosing the correct output diminishes as the vocabulary increases. Also, even if infrequent tokens are represented in the vocabulary, its infrequent use would not be sufficient for the model to learn when to use it correctly.

One approach to deal with the size of the vocabulary is to convert all characters to lowercase and replace numbers and dates with zero representations. With these modifications, the vocabulary becomes smaller, but the model will become less capable of generating outputs in the same way as humans would.

Even with the text simplifications, the problem with out of vocabulary words (OOV) remains because not all possible words will be represented. So, it is usual to represent any token that is not present in the vocabulary by a reserved OOV token. This token would be used for uncommon names, dates, and numbers.

Another approach is to use one token per character. This greatly reduces the vocabulary size only using lower and upper case characters, numbers, and symbols. Although the vocabulary is smaller, probably not more than few hundred tokens, it would require each word to be represented by several tokens, leading to very long document. This is problematic because the model will require a large memory to be able to generate the summary without repeating already generated tokens.

The alternative to mitigate the disadvantages of these two approaches is to use *sub-word units as the token representation* (Schuster and Nakajima, 2012; Chitnis and DeNero, 2015; Sennrich et al., 2015; Kudo, 2018). The idea is that a token would represent a common pattern seen in the training data rather than words or characters. This operates with a fixed vocabulary size and assigns a token for the most common patterns found. With this method, the OOV problem is reduced as one word now can be represented by a combination of

sub-words. The problem of longer sequences is also addressed because a token now can represent several characters.

3 Related Work

Neural Networks are models capable of learning very complex functions. In the last few years, there has been significant interest in applying them for natural language tasks such as automatic translation and summarization.

One of the first issues that needs to be addressed is that Neural Networks require that both inputs and outputs have a fixed a length, and that is not the case when dealing with text, because each document (or sentence) can have a different length. In order to overcome this limitation, Sutskever et al. (2014) introduced a general end-to-end approach capable of representing sequence-to-sequence models using LSTM (Long Short-Term Memory) cells. Nevertheless, both input and output must still have fixed lengths large enough to fit. The network is trained to output the end-of-sentence (EOS) token when the output is large enough. With this approach, both source input and generated sequence may logically have different lengths and do not require any type of alignment between input and output.

Bahdanau et al. (2014) proposed a model for learning to align the input with the generated output. They realized that the approach for encoding the whole text before starting to decode requires that the whole idea is represented by just one vector. So, they proposed to use auxiliary vectors to represent the alignment of the input in relation to the generated output. Their approach significantly improved the quality of the generated outputs and became known as Attention vectors.

Following the same ideas, Luong et al. (2015) explored different versions for the attention mechanism. They also noticed that as the length of the input increases, the attention vector has a lot more difficulty in learning the weights. So they evaluated the impact of using a local attention mechanism to look just for a smaller portion of the source at each time step.

The attention mechanism works well for short texts, but struggles to focus on relevant information when applied to long documents, common in the legal domain.

Neural Machine Translation (NMT) is a sequence-to-sequence task that is similar to summarization in the sense that both require some text comprehension before an output can be generated. When translating, a model does not have an exact alignment for each word read in the source to the word generated in the output. This happens because a source word may be represented by more (or fewer) words in the translation. Also, the order of the tokens can be different.

Wu et al. (2016) describe the architecture used for the Google Neural Machine Translation (GNMT) system. In that work, they used a LSTM network with 8 encoders and 8 decoders using attention and residual connections. In the beam search, they used length-normalization and applied a coverage penalty to favor an output that is able to cover more words from the source sequence.

Vaswani et al. (2017) improved the GNMT system replacing entirely the recurrence and convolutions by an attention-based model known as Transformer. The model is quite complex and relies on many training variables, but it has the advantage of allowing more parallel computation. With this modification, the model is able to use many GPUs and train a lot faster.

See et al. (2017) addressed two common problems found in the application of RNNs in the context of summarization. First, the problem of rare words that were not present in the vocabulary was solved by having hybrid pointer networks that are capable of using the source word as output when the attention weight is high enough. The second problem is using a coverage vector that represents the weighted sum of the attention vectors. Then, to increase coverage, their model is trained to penalize every time the attention vectors are high in the same regions, encouraging the model to better distribute the attention over the source input.

In our work, the problem of rare words has been diminished using sub-word encoding, as discussed in Section 2. The coverage mechanism, proposed to deal with repetition problem, is complex to train. It is used after the training to condition the decoder to avoid generating attention vector using the same positions that were already used. Paulus et al. (2017) proposed trigram avoidance during the beam search. This approach is much simpler and easy to apply but does not really fix the problem, as it still will happen, it only masks its effects for the evaluation.

There is a growing interest in using reinforce-

ment learning approaches (Paulus et al., 2017; Li et al., 2018; Celikyilmaz et al., 2018) to improve summarization performance. In general, reinforcement learning is employed when an non-differentiable operation is being used. These methods have the disadvantage of being hard to tune and generally slow to converge.

Chen and Bansal (2018) proposed a method for using both extracting and abstracting approaches. Their idea was to select salient sentences and rewrite them abstractively. Reinforcement learning was used to combine these two neural networks. Their idea seems to be a good approach for working with long documents. Although, they did not report results for long documents datasets. We intend to further explore the application of this technique to long documents in the legal domain.

4 Materials and Methods

Our summarization experiment in the legal area uses rulings written in Portuguese. There are some peculiarities when using a language different from English, *e.g.,* we need to check if the standard summarization evaluation (designed for English) can be directly applied.

4.1 Dataset

The dataset used in our experiments is RulingBR (Feijó and Moreira, 2018). It contains about 10K rulings from the Brazilian Supreme Court. These rulings were taken by a group of judges – the individual decision from each judge is known as their "vote", the final decision is made by the majority of the votes. Each ruling has four sections: (i) the *ementa*, which represents the summary; (ii) the *acórdão*, which has the final decision taken by the majority; (iii) the *relatório*, which is an extensive description reporting the request and the actions taken so far; and (iv) the *voto*, which contains the individual votes from all judges.

The dataset was randomly split into training, validation, and test sets. Training takes up 60% of the instances (6,373), whereas validation and test have 20% of the instances each (2,125). On average, each document has about 2,500 tokens.

We use the *ementa* as the human-generated ground truth summary. The other three sections together compose the input text, which is submitted to the summarization systems. On average, the *ementa* has a little less than 10% of the size of the source input.

4.2 Official Rouge Script

The standard evaluation metric for text summarization is called Recall-Oriented Understudy for Gisting Evaluation (ROUGE). The general idea of this metric is to count the number overlapping units between one or more reference summaries and the machine-generated summary. It is expected, that a high-quality summary should use the same words found in the reference summaries and preferably in the same order.

The results reported here include Precision, Recall, and F-measure for ROUGE-1, ROUGE-2, and ROUGE-L metrics. A confidence of 95% was adopted. We used the official ROUGE (Lin, 2004) 1.5.5 script in our experiments. Most parameters were set to their default values. The `pyrouge` package, which is a wrapper to the official script, was used to provide the required output text (in HTML) and the configuration file. The only change was to remove the call to the English Porter Stemmer, since our texts are in Portuguese.

4.3 Text Preprocessing

The official Rouge script treats any non-ASCII characters as word separators. Thus, we transformed all accented characters to their base form, *i.e.,* diacritics were removed. In addition, we changed all text to lowercase and isolated the standard punctuation symbols from the alphabetic characters to avoid them being interpreted as part of some word.

4.4 Vocabulary

Portuguese has a rich vocabulary, with words having lots of variant forms. Verbs, specially, can have dozens of different suffixes corresponding to the diverse conjugations. In the legal domain, it is common to reference existing laws, specific dates, and names. Thus, it is very unlikely that the vocabulary generated during training will contain all possible words that are present in the test set. To deal with problem of OOV words, we used the SentencePiece package (Google/SentencePiece, 2019), which implements the sub-word units with unigram representation. The combination of pieces is used to generate words even when they are not present in the training set.

5 A Simple Extractive Summarization Baseline

When generating summaries, the source input length and the desired summary length are required. Ideally, these two lengths should be automatically determined by the algorithm, but that is not how these standard extractive approaches work. In order to establish these lengths, we defined a heuristic-based baseline extractive method.

5.1 Heuristic for Sentence Selection

Through empirical observation, we found that the *relatório* (report) section from the source text usually contains most of the information that is typically present in the reference summary. So, after removing some boilerplate text that is usually present at the beginning of the documents, we extract a sequence of words until the desired summary length is reached.

5.2 Target Length

In the RulingBR dataset, the mean length for the test set is 190 tokens, with a minimum of 20 and maximum of 1,909 tokens. This represents a wide range of summary lengths. Since every summary generated by our baseline needed to have the same length, we experimented truncating the reference summaries at different points to observe the effect over the mean length of the test set.

Table 1 shows the effect of imposing length limits to reduce the standard deviation in favor of a more predictable summary length. The dilemma here is to balance between a lower limit and lower standard deviation (but risking losing important information) with a higher limit and higher deviation (but with a large error associated).

Another concern that arises when deciding about the target length is the fact that the summaries will be evaluated using the ROUGE metric. ROUGE relies both on the Precision and on the Recall. Shorter summaries are expected to have higher precision, while longer summaries tend to have higher recall. Our F-score results assign the same weight to precision and recall. Table 2 shows that truncating the source length has little effect over the F-measure of the ROUGE scores. In order to make a fair comparison between the algorithms, we aimed at generating summaries of similar lengths.

Limit	Mean	Min	Max	Std Dev
No	190	20	1,909	179.46
600	180	20	600	131.89
450	173	20	450	112.02
300	158	20	300	82.37
150	120	20	120	36.85

Table 1: Lengths of the reference summaries that compose our Test Set.

5.3 Source Length

Abstractive summarization models require a fixed size input. Different lengths will require padding, which in turn will have a negative impact on training. We used truncated parts of sections *relatório* and *voto* as they concentrate most of the important information. Table 2 shows the results of our experiment using *relatório* and *voto* with lengths of 150, 300, 450, or 600 tokens and trying to find summaries with this same length.

Because lengthier summaries would require more memory and would lead to a broader range of lengths, we adopted the 300+300 tokens as the input length limit in our experiments (*i.e.,* the input text is a concatenation of the 300 first tokens from *relatório* and the 300 first tokens from *voto*).

As shown in Table 1, truncating the target length to 300 tokens leads to summaries of 158 tokens on average. This will be used as the desired summary length.

6 Extractive Approaches

The ten extractive approaches used in our experiments are described in this section. We used the implementations provided by Sumy (Belica, 2018) and an improved version of TextRank provided by Gensim (Řehůřek and Sojka, 2010).

6.1 Luhn

Proposed by Luhn (1958), the seminal method for determining the importance of a sentence is calculated using Term Frequency (TF) and Inverse Document Frequency (IDF). Significant words are selected among the most frequent words found in the document. Then, the highest scoring sentences are selected to be part of the summary.

When calculating word frequencies, Luhn algorithm proposes a simple stemmer by matching words using their prefixes. A match happens when the number of non-matching letters is less than six.

Length	R1-F	R1-P	R1-R	R2-F	R2-P	R2-R	RL-F	RL-P	RL-R
600+600	33.99	33.46	**43.16**	**12.20**	12.34	**15.59**	19.44	19.10	**25.36**
450+450	34.07	34.40	41.32	12.06	**12.50**	14.72	19.48	19.64	24.24
300+300	**34.47**	**34.84**	40.25	12.08	12.43	14.19	19.74	19.88	23.58
150+150	**34.47**	34.32	37.46	11.88	11.84	13.01	**20.49**	**20.29**	22.61

Table 2: ROUGE scores for different source lengths. The results show that the F-measure is reasonably stable across different lengths.

This technique may be language specific, thus it is possible that a stemmer specificaly designed for the target language would have a different behavior. Here, we use the standard implementation as provided by the Sumy without stemming or stopword removal.

6.2 LexRank

This is a stochastic graph-based method for computing the relative importance of sentences (Erkan and Radev, 2004b). It assumes that the main idea of a text is often paraphrased. As a consequence, finding similar sentences would be the same as finding the important sentences. Also, the central sentence of a cluster would indicate that this sentence is the most similar among them and would probably capture more information.

6.3 TextRank

This is a graph-based ranking model of deciding the importance of a vertex within a graph (Mihalcea and Tarau, 2004). The basic idea is to have some form of votes every time one vertex is similar to another. The highest voted vertex would be the most important. Gensim uses a modified version (Barrios et al., 2016) in which the BM25 similarity function is used in place of just the number of common tokens as adopted by the original TextRank. Thus, TextRank appears twice in our results as we used both the Gensim and the Sumy implementations.

6.4 SumBasic

This algorithm is based on the fact that words present in the summary tend to be the most frequent in the text (Nenkova and Vanderwende, 2005). It computes the probability distribution over the words appearing in the input. Then the sentences containing the highest probability words are selected and for each word in these sentences, update their probabilities until the desired length is reached.

6.5 KLSum

Kullback-Leibler (KL) is a way to compare two probability distributions. It also computes the probability distribution of words in the text. Then, the problem of finding the summary can be stated as finding a set of summary sentences which the probability distribution closely matches the document distribution (Haghighi and Vanderwende, 2009).

6.6 LSA

The summarization using Latent Semantic Analysis (LSA) (Steinberger and Jezek, 2004; Gong and Liu, 2001) is done by constructing a sparse *token x sentences* matrix, applying Singular Value Decomposition (SVD), selecting the singular vector will retrieve the scores for each token, then selecting sentences with highest normalized scores.

6.7 Random

This is another baseline summarizer in which random scores are assigned to the sentences. The highest scoring sentences are selected to the summary.

7 Abstractive Approaches

We experimented with Neural Network models using the `OpenNMT-tf` package (Klein et al., 2017). The models evaluated here were NMTSmall, NMTMedium, Transformer, and TransformerAAN.

NMTSmall and NMTMedium are standard Recurrent Neural Network models. They use an encoder-decoder architecture. The decoder employs Luong et al. (2015) style attention model over the input. The network is trained to learn when to stop generating the summary. This is done appending an End-of-Document token to the instances during training. When the network generates this token, the output is truncated at this point. The model uses a beam search of size four when

decoding, and it is configured to ignore outputs that were shorter than the minimum length.

Both NMT and Transformer models use word embedder of size 512. Each model was evaluated until its training loss was no longer diminishing. We report ROUGE results with minimum decoding lengths of 100 and 120 tokens. Recall that we are using SentencePiece and each decoded word may be represented by more than one token. So, the generated output may contain fewer words than this minimum length. In all reported results, we show the mean length of the output considering generated tokens separated by spaces.

Two NMT configurations were used. NMT-Small uses 2-layers, unidirectional LSTM with 512 units, and it has converged in 15,000 steps. NMTMedium uses 4-layers, bidirectional LSTM, with 512 units and it has converged in 26,000 steps. Transformer model uses the configuration as originally proposed by Vaswani et al. (2017). TransformerAAN uses cumulative average attention network in the decoder as described in (Zhang et al., 2018). The objective is to reduce the required training and inference time.

8 Results and Discussion

The performance of the extractive algorithms shown in Table 3 was disappointing. With the exception of SumBasic, all other algorithms have performed worse than our simple Baseline by at least 0.6 points in ROUGE-L. In some cases, the performances were not far from the random baseline. A possible explanation for such poor results is the limitation of this approach of generating summaries using only complete sentences that were present in the source text. Looking at the generated summaries, most of them have selected just one very long sentence while others used a few random disconnected sentences.

Our experiments varying the lengths of the input method (Table 2) have shown that even with larger source inputs, which could contain more tokens that should be present in the output, the performance was decreasing.

The baseline results provided by Feijó and Moreira (2018) for the extractive methods ranged between 11 and 16 points in terms of ROUGE-L. Those results cannot be directly compared to the results in our experiments because they had removed stopwords and they reported results for the entire dataset. Since in this paper, we require a training phase for the neural network models, ROUGE results are reported only for the test set.

One advantage of extractive algorithms is that they do not require prior training, and they can be applied directly over the test data. On the other hand, after the time-consuming training, the abstractive approaches can create the summary a lot faster.

Table 4 shows reasonably good results for both NMT and Transformer models. There was a small advantage for the standard Transformer model when compared to its modified version with the Average Attention Network. They both have reached very similar results and have converged in about 40K steps.

Since the Transformer model has many variables, it requires a lot of memory to run. So, the batches need to be smaller. As a consequence, it needed more steps to converge. Despite that, we observed that it trains faster than standard RNNs. As we are using a concurrent environment, our measures of the time taken for training were not accurate, so we could not report them.

Summarization results in other datasets are not directly comparable to our results. Still, they may serve as reference. Zhang et al. (2019) reports that the current state-of-the-art for the CNN/Daily Mail dataset (Nallapati et al., 2016) reaches scores of ROUGE-1 41.71, ROUGE-2 19.49 and ROUGE-L 38.79.

The summaries generated by the abstractive approaches were promising. They look similar to those produced by humans. In most generated summaries, the main topic was correctly captured by the summarizer. Nevertheless, as shown in Figure 1 there are still some cases in which the summarizer barely captured any meaning of the text, generating summaries that had almost no relation with the expected output. In these cases, the extractive approach would probably have done better. In other cases, the general meaning was correctly captured, but the output had repeating expressions. We believe this may have been caused by the minimum length restriction.

Legal operators rely on summaries to their jobs, since it is impossible to read the full contents of each decision to find precedents for their cases. Missing or referring to an incorrect precedent may cause the petition to be denied and the case would be lost. Thus, considering the results seen so far, neither approach delivers results that could safely

Algorithm	R1-F	R1-P	R1-R	R2-F	R2-P	R2-R	RL-F	RL-P	RL-R
Random	31.52	34.42	34.81	10.55	11.81	11.49	17.88	19.67	19.99
Baseline	34.47	**34.84**	40.25	12.08	12.43	14.19	**19.74**	**19.88**	**23.58**
Luhn	33.16	33.17	39.08	11.06	11.25	13.09	18.77	18.67	22.65
LexRank	34.06	34.06	40.07	11.65	11.85	13.69	19.16	19.04	23.06
LSA	32.31	32.26	38.04	10.44	10.62	12.23	17.88	17.76	21.50
KLSum	31.96	32.42	37.14	11.45	11.74	13.30	18.24	18.38	21.66
SumBasic	**34.51**	34.41	**40.74**	**12.32**	**12.49**	**14.46**	18.76	18.69	22.43
TextRank[1]	33.09	33.07	38.99	10.85	11.07	12.73	18.78	18.67	22.60
TextRank[2]	33.66	34.14	39.10	12.00	12.31	13.97	19.16	19.24	22.82

Table 3: ROUGE scores using extractive algorithms

Model	Len	R1-F	R1-P	R1-R	R2-F	R2-P	R2-R	RL-F	RL-P	RL-R
NMTSmall	130	38.86	44.75	40.42	21.28	23.14	22.89	30.22	33.99	32.02
NMTMedium	130	43.25	49.25	44.80	25.41	27.60	27.05	33.91	37.78	35.69
Transformer	134	**44.27**	**49.38**	46.24	**26.50**	**28.36**	28.26	**35.27**	**38.52**	37.36
TransformerAAN	137	43.67	48.38	45.90	25.60	27.15	27.43	34.47	37.38	36.74
NMTSmall	141	38.37	42.48	41.54	20.77	21.77	23.29	29.55	31.92	32.68
NMTMedium	140	41.56	46.44	44.00	23.43	24.95	25.56	32.01	34.87	34.58
Transformer	145	43.91	47.34	**47.76**	25.95	26.93	**28.84**	34.55	36.48	**38.16**
TransformerAAN	147	43.39	46.46	47.37	25.23	25.93	28.13	33.90	35.51	37.60

Table 4: ROUGE scores using abstractive models. The mean length is shown because it affects the scores.

replace humans in this task. The current state is promising, but automatic systems are not always capable of generating good summaries. Hence, they could be used to prepare drafts which then need to be revised by humans.

Ground Truth: direito administrativo . lei nº 11.064/2002 . servico auxiliar voluntario . policial militar temporario . acrescimo de 1/3 , 13º salario , adicional de insalubridade e de local de exercicio . eventual violacao reflexa da constituicao da republica nao viabiliza o recurso extraordinario . recurso extraordinario interposto sob a egide do cpc/1973 . alegacao de ofensa aos arts . 2º, 5º, ii , e 37 , caput , ii e ix , da constituicao da republica . agravo manejado sob a vigencia do cpc/2015 . . .

Generated: direito administrativo . militar . promocao . ato de bravura . recurso extraordinario interposto sob a egide do cpc/2015 . eventual ofensa reflexa nao enseja recurso extraordinario . necessidade de interpretacao de legislacao local . aplicacao da sumula no 280/stf . agravo manejado sob a vigencia do cpc/2015 . . .

Figure 1: Summary generated by the Transformer model. The ground truth refers to a petition for compensation when made by a policeman. The generated summary was about a petition for benefits due to an act of bravery by military personnel.

9 Conclusion

This work presented a comparative investigation of ten extractive and four abstractive methods for text summarization using Portuguese language applied to the legal area. The data used here consist of a real-world legal domain dataset containing 10K rulings from the Brazilian Supreme Court. The results show that extractive methods provided weak performance being unable to generate useful summaries. On the other hand, abstractive models provided much better results, with summaries that were very similar to those produced by humans. However, they also presented severe problems with repeating expressions and the introduction of subjects that were not present in the source documents. We intend to further investigate the causes of the factual errors and address this problem in future work.

Acknowledgment: This work was partially supported by CNPq/Brazil and by CAPES Finance Code 001.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* .

Federico Barrios, Federico López, Luis Argerich, and Rosa Wachenchauzer. 2016. Variations of the similarity function of textrank for automated summarization. *arXiv preprint arXiv:1602.03606* .

Mišo Belica. 2018. sumy: Module for automatic summarization of text documents and HTML pages. https://github.com/miso-belica/sumy.

Asli Celikyilmaz, Antoine Bosselut, Xiaodong He, and Yejin Choi. 2018. Deep communicating agents for abstractive summarization. *arXiv preprint arXiv:1803.10357* .

Yen-Chun Chen and Mohit Bansal. 2018. Fast abstractive summarization with reinforce-selected sentence rewriting. *CoRR* abs/1805.11080. http://arxiv.org/abs/1805.11080.

Rohan Chitnis and John DeNero. 2015. Variable-length word encodings for neural translation models. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. pages 2088–2093.

H. P. Edmundson. 1969. New methods in automatic extracting. *J. ACM* 16(2):264–285. https://doi.org/10.1145/321510.321519.

Günes Erkan and Dragomir R. Radev. 2004a. Lexrank: Graph-based lexical centrality as salience in text summarization. *J. Artif. Int. Res.* 22(1):457–479. http://dl.acm.org/citation.cfm?id=1622487.1622501.

Günes Erkan and Dragomir R Radev. 2004b. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research* 22:457–479.

Diego de Vargas Feijó and Viviane Pereira Moreira. 2018. Rulingbr: A summarization dataset for legal texts. In Aline Villavicencio, Viviane Moreira, Alberto Abad, Helena Caseli, Pablo Gamallo, Carlos Ramisch, Hugo Gonçalo Oliveira, and Gustavo Henrique Paetzold, editors, *Computational Processing of the Portuguese Language*. Springer International Publishing, Cham, pages 255–264.

Yihong Gong and Xin Liu. 2001. Generic text summarization using relevance measure and latent semantic analysis. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, pages 19–25.

Google/SentencePiece. 2019. Unsupervised text tokenizer for neural network-based text generation. `https://github.com/google/sentencepiece`. Accessed: 2019-05-25.

Aria Haghighi and Lucy Vanderwende. 2009. Exploring content models for multi-document summarization. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 362–370.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander Rush. 2017. OpenNMT: Open-source toolkit for neural machine translation. In *Proceedings of ACL 2017, System Demonstrations*. Association for Computational Linguistics, Vancouver, Canada, pages 67–72. https://www.aclweb.org/anthology/P17-4012.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. *CoRR* abs/1804.10959. http://arxiv.org/abs/1804.10959.

Piji Li, Lidong Bing, and Wai Lam. 2018. Actor-critic based training framework for abstractive summarization. *arXiv preprint arXiv:1803.11070* .

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out* .

H. P. Luhn. 1958. The automatic creation of literature abstracts. *IBM J. Res. Dev.* 2(2):159–165. https://doi.org/10.1147/rd.22.0159.

Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. *CoRR* abs/1508.04025. http://arxiv.org/abs/1508.04025.

R. Mihalcea and P. Tarau. 2004. TextRank: Bringing order into texts. In *Proceedings of EMNLP-04and the 2004 Conference on Empirical Methods in Natural Language Processing*.

Ramesh Nallapati, Bing Xiang, and Bowen Zhou. 2016. Sequence-to-sequence rnns for text summarization. *CoRR* abs/1602.06023. http://arxiv.org/abs/1602.06023.

Ani Nenkova and Lucy Vanderwende. 2005. The impact of frequency on summarization. *Microsoft Research, Redmond, Washington, Tech. Rep. MSR-TR-2005* 101.

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304* .

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. ELRA, Valletta, Malta, pages 45–50.

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and korean voice search. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pages 5149–5152.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368* .

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909* .

Josef Steinberger and Karel Jezek. 2004. Using latent semantic analysis in text summarization and summary evaluation. *Proc. ISIM* 4:93–100.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. *CoRR* abs/1409.3215. http://arxiv.org/abs/1409.3215.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR* abs/1706.03762. http://arxiv.org/abs/1706.03762.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR* abs/1609.08144. http://arxiv.org/abs/1609.08144.

Biao Zhang, Deyi Xiong, and Jinsong Su. 2018. Accelerating neural transformer via an average attention network. *CoRR* abs/1805.00631. http://arxiv.org/abs/1805.00631.

Haoyu Zhang, Yeyun Gong, Yu Yan, Nan Duan, Jianjun Xu, Ji Wang, Ming Gong, and Ming Zhou. 2019. Pretraining-based natural language generation for text summarization. *arXiv preprint arXiv:1902.09243* .