UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

DIEGO DE VARGAS FEIJO

Summarizing Legal Rulings

Thesis presented in partial fulfillment of the requirements for the degree of Doctor of Computer Science

Advisor: Prof. Dr. Viviane Pereira Moreira

Porto Alegre
September 2021

CIP — CATALOGING-IN-PUBLICATION

Feijo, Diego de Vargas

Summarizing Legal Rulings / Diego de Vargas Feijo. – Porto Alegre: PPGC da UFRGS, 2021.

101 f.: il.

Thesis (Ph.D.) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2021. Advisor: Viviane Pereira Moreira.

1. Legal ruling summarization. 2. Abstractive summarizer. 3. Content digest. 4. Legal case brief. 5. Summary writing. 6. Abstract generator. 7. Automatic text summary. 8. Textual entailment. 9. Fact checking. I. Moreira, Viviane Pereira. II. Título.

Sumarização de Decisões Judiciais

RESUMO

Com o incremento de disponibilidade de dados, a sumarização de textos tornou-se uma necessidade para encontrar informações relevantes rapidamente. Na área jurídica, documentos possuem peculiaridades relacionadas ao comprimento, jargão especializado e vocabulário. Estas características tornam difícil a utilização de técnicas convencionais para geração de resumos. Avanços recentes nas abordagens usando redes neurais obtiveram elevados resultados em termos de qualidade. Contudo, estas abordagens vêm sendo usadas majoritariamente para criação de sumários curtos e no contexto jornalístico. Portanto, a sua aplicação no domínio jurídico segue como um problema em aberto. Neste trabalho, propomos LegalSumm, um método de sumarização para decisões judiciais baseado em Tranformers e implicação textual (*textual entailment*). Nossa abordagem cria oito pedaços de texto a partir da decisão judicial e gera um sumário-candidato para cada. A seguir, avaliamos a implicação textual entre a decisão e o sumário, e selecionamos o candidato que obtiver a maior pontuação. Comparamos nosso método com linhas de base extrativas e abstrativas e coletamos as opiniões de especialistas na área jurídica. Os resultados demonstram que LegalSumm melhora a qualidade dos resumos gerados.

Palavras-chave: sumarização de decisões jurídicas, sumarização abstrativa, sumarizador de conteúdo, ementa de processo judicial, escrita de sumário, gerador de resumo, resumo automático de texto, implicação textual, verificação de fatos.

ABSTRACT

With the increasing availability of data, text summarization becomes helpful to find relevant information quickly. In the legal domain, documents have peculiarities related to their length, specialized use, and vocabulary. These characteristics are challenging for standard summarization techniques. Recent neural network-based approaches can generate high-quality summaries. However, these approaches have been used mostly for creating concise abstracts for news articles. Thus, their applicability to the legal domain remains an open issue. In this work, we propose LegalSumm, a method for summarizing legal texts based on Transformers and textual entailment. Our approach builds eight chunks of text from the ruling and generates one candidate summary for each. Then, we compute the entailment score from the ruling to each candidate and select the candidate with the highest score. We compared our method to strong extractive and abstractive summarization baselines and collected the opinion of legal experts. The results show that LegalSumm improves the quality of the generated summaries.

Keywords: Legal ruling summarization. abstractive summarizer. content digest. legal case brief. summary writing. abstract generator. automatic text summary. textual entailment. fact checking.

LIST OF ABBREVIATIONS AND ACRONYMS

BERT	Bidirectional Encoder Representations from Transformers
CNN	Convolutional Neural Network
FNN	Feedforward Neural Network
GPU	Graphics Processing Unit
GRU	Gated Recurrent Unit
JSON	JavaScript Object Notation
LCS	Longest Common Subsequence
LSH	Locality Sensitive Hashing
LSTM	Long Short-Term Memory
NLP	Natural Language Processing
NMT	Neural Machine Translator
OOV	Out-of-Vocabulary
POS	Part-of-Speech
RNN	Recurrent Neural Network
RTE	Recognizing Textual Entailment
TFIDF	Term Frequency times Inverse Document Frequency
TPU	Tensor Processing Unit

LIST OF FIGURES

LIST OF TABLES

CONTENTS

1 INTRODUCTION

With the increasing availability of data, many areas need text summarization, especially in the legal field, where the texts are usually lengthy. Legal practitioners are expected to keep updated with relevant information ranging from news, jurisprudence changes, and rulings from many courts. When researching, a legal practitioner must often seek through many precedents looking for those that fit specific requirements. Each of these precedents may have dozens of pages with details specific to that case. As an example, each ruling from the Brazilian Supreme Court typically has more than 2,000 tokens on average (FEIJO; MOREIRA, 2018). In these situations, details are not essential, and the legal practitioner needs just an outlook of the general rules applicable to that case. Hence, it is necessary to focus on the essential portions of each topic and extract just the core information, 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 outcomes. These summaries provide a faster way to find the required information without needing to read the whole text.

Currently, these legal summaries are generated by humans in a process that is time-consuming and expensive. Human summarizers need to know the subject sufficiently to select the main topics to compose the summary. Another critical issue with manually created summaries is the lack of standardization. Each specialist from each court has a writing style. A standardized way of writing is desirable as it would provide more homogeneous summaries (GUIMARÃES, 2011). A summary should be concise, fluent, and contain paraphrased versions of the input text with a reduced length.

Automatic text summarization is the process of using computer programs to mimic the summaries generated by humans. There are two conventional approaches: extractive and abstractive. Extractive refers to the process of generating the summary by selecting the most significant sentences from the source. On the other hand, abstractive is the process of building a contextual representation of the main ideas from the source and generate a complete new summary writing it word by word.

With the summary of legal rulings being the context for the proposed thesis, we detail the research problem in the next section. Then, an overview of our proposed solution, LegalSumm, is presented.

1.1 Research Problem

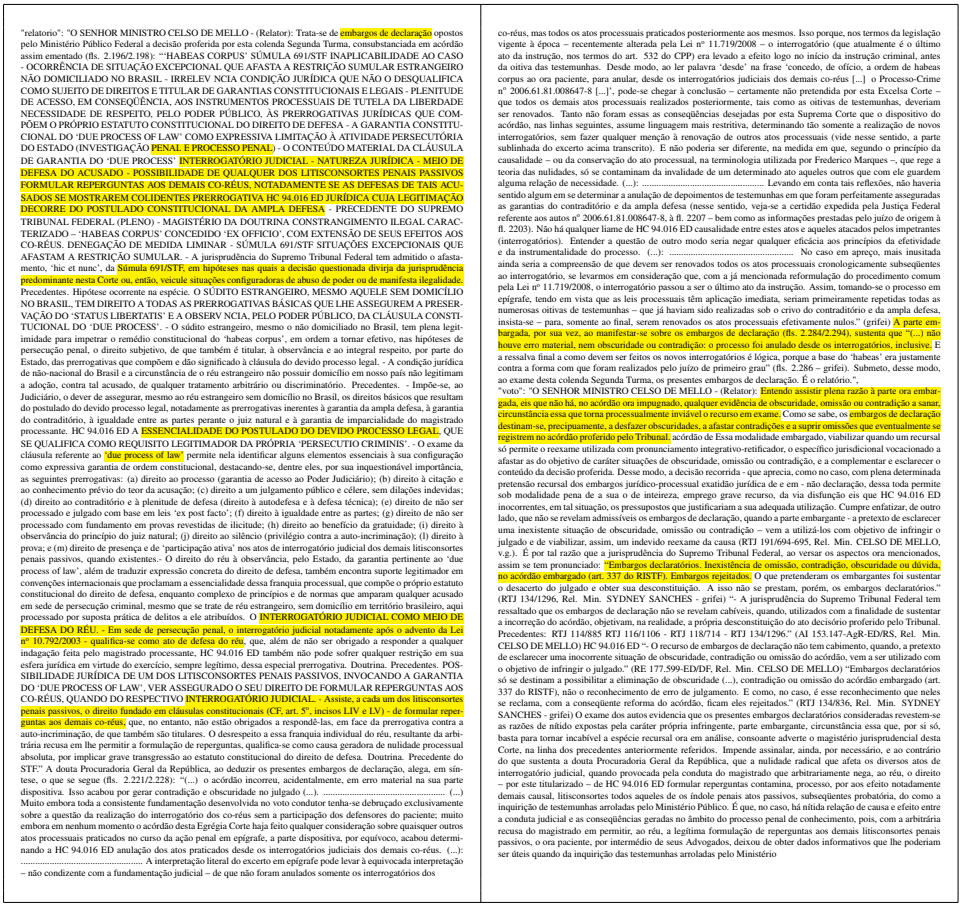
This work aims to propose and evaluate a suitable method for generating summaries of legal texts. The summarization of legal texts differs remarkably from mainstream text summarization, which is mainly devoted to summarizing news articles, headlines, or tweets. According to Turtle (1995), legal documents have some distinguishing characteristics compared to newspapers articles or scientific papers, namely (i) *size*, they tend to be longer; (ii) *structure*, they present an internal structure; (iii) *vocabulary*, many of technical terms are often used; (iv) *ambiguity*, ambiguous terms may lead to different meanings for the same words; and (v) *citations*, play a prominent role in the legal domain more than in other areas. News articles often start with some catch sentence called "lede" that summarizes the entire article, making the summarization task considerably easier. Legal texts do not follow this pattern; they are generally lengthier and typically contain sophisticated vocabulary and expressions. To illustrate information spread, Figure 1.1 shows an example of a (shortened) legal ruling and how the critical information that should be included in the summary is scattered through the text.

Another concern when generating the summary is to preserve the original meaning of the sentences. The same words in a different order 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"*.

There are multiple possible ways of writing a summary. Each person has a style, and they may disagree on the essential aspects that the summary must contain. To have more standardization, the model could mimic one particular style or develop its method based on an average of several samples. Following just one particular style would restrict the subjects covered by the summarizer and reduce the availability of samples. The state-of-the-art in automatic text summarization uses deep learning models. As these models require many samples to train, and a broad topic summarizer is desirable, the training procedure might use all available samples.

Summing up, the objective of this work is to propose and evaluate a model capable of generating summaries for legal rulings using deep learning methods. LegalSumm should fulfill some design goals: (i) abstractive text generation - mimic human behavior of reading and summarizing the key points using their own words; (ii) neutral bias - combining writing styles from many judges would lead to a style not biased by just one judge; and (iii) recognize entailment - the source ruling must entail the generated summary.

Figure 1.1 – Example of Legal Ruling

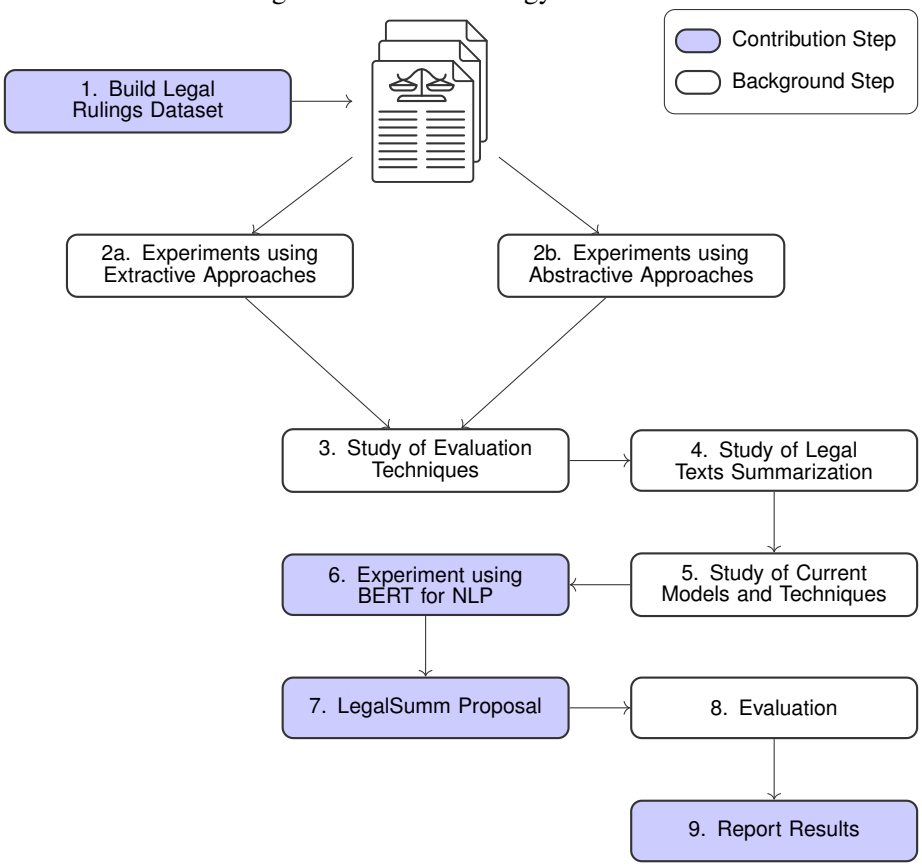


"relatorio": "O SENHOR MINISTRO CELSO DE MELLO - (Relator): Trata-se de embargos de declaração opostos pelo Ministério Público Federal a decisão proferida por esta colenda Segunda Turma, consubstanciada em acórdão assim ementado (fls. 2.196/2.198): ""'HABEAS CORPUS' SÚMULA 691/STF INAPLICABILIDADE AO CASO - OCORRÊNCIA DE SITUAÇÃO EXCEPCIONAL QUE AFASTA A RESTRIÇÃO SUMULAR ESTRANGEIRO NÃO DOMICILIADO NO BRASIL - IRRELEV NCIA CONDIÇÃO JURÍDICA QUE NÃO O DESQUALIFICA COMO SUJEITO DE DIREITOS E TITULAR DE GARANTIAS CONSTITUCIONAIS E LEGAIS - PLENITUDE DE ACESSO, EM CONSEQÜÊNCIA, AOS INSTRUMENTOS PROCESSUAIS DE TUTELA DA LIBERDADE NECESSIDADE DE RESPEITO, PELO PODER PÚBLICO, ÀS PRERROGATIVAS JURÍDICAS QUE COMPÕEM O PRÓPRIO ESTATUTO CONSTITUCIONAL DO DIREITO DE DEFESA - A GARANTIA CONSTITUCIONAL DO 'DUE PROCESS OF LAW' COMO EXPRESSIVA LIMITAÇÃO À ATIVIDADE PERSECUTÓRIA DO ESTADO (INVESTIGAÇÃO PENAL E PROCESSO PENAL) - O CONTEÚDO MATERIAL DA CLÁUSULA DE GARANTIA DO 'DUE PROCESS' INTERROGATÓRIO JUDICIAL - NATUREZA JURÍDICA - MEIO DE DEFESA DO ACUSADO - POSSIBILIDADE DE QUALQUER DOS LITISCONSORTES PENAIS PASSIVOS FORMULAR REPERGUNTAS AOS DEMAIS CO-RÉUS, NOTADAMENTE SE AS DEFESAS DE TAIS ACUSADOS SE MOSTRAREM COLIDENTES PRERROGATIVA HC 94.016 ED JURÍDICA CUJA LEGITIMAÇÃO DECORRE DO POSTULADO CONSTITUCIONAL DA AMPLA DEFESA - PRECEDENTE DO SUPREMO TRIBUNAL FEDERAL (PLENO) - MAGISTÉRIO DA DOUTRINA CONSTRANGIMENTO ILEGAL CARACTERIZADO – 'HABEAS CORPUS' CONCEDIDO 'EX OFFICIO', COM EXTENSÃO DE SEUS EFEITOS AOS CO-RÉUS. DENEGAÇÃO DE MEDIDA LIMINAR - SÚMULA 691/STF SITUAÇÕES EXCEPCIONAIS QUE AFASTAM A RESTRIÇÃO SUMULAR. - A jurisprudência do Supremo Tribunal Federal tem admitido o afastamento, 'hic et nunc', da Súmula 691/STF, em hipóteses nas quais a decisão questionada divirja da jurisprudência predominante nesta Corte ou, então, veicule situações configuradoras de abuso de poder ou de manifesta ilegalidade. Precedentes. Hipótese ocorrente na espécie. O SÚDITO ESTRANGEIRO, MESMO AQUELE SEM DOMICÍLIO NO BRASIL, TEM DIREITO A TODAS AS PRERROGATIVAS BÁSICAS QUE LHE ASSEGUREM A PRESERVAÇÃO DO 'STATUS LIBERTATIS' E A OBSERV NCIA, PELO PODER PÚBLICO, DA CLÁUSULA CONSTITUCIONAL DO 'DUE PROCESS'. - O súdito estrangeiro, mesmo o não domiciliado no Brasil, tem plena legitimidade para impetrar o remédio constitucional do 'habeas corpus', em ordem a tornar efetivo, nas hipóteses de persecução penal, o direito subjetivo, de que também é titular, à observância e ao integral respeito, por parte do Estado, das prerrogativas que compõem e dão significado à cláusula do devido processo legal. - A condição jurídica de não-nacional do Brasil e a circunstância de o réu estrangeiro não possuir domicílio em nosso país não legitimam a adoção, contra tal acusado, de qualquer tratamento arbitrário ou discriminatório. Precedentes. - Impõe-se, ao Judiciário, o dever de assegurar, mesmo ao réu estrangeiro sem domicílio no Brasil, os direitos básicos que resultam do postulado do devido processo legal, notadamente as prerrogativas inerentes à garantia da ampla defesa, à garantia do contraditório, à igualdade entre as partes perante o juiz natural e à garantia de imparcialidade do magistrado processante. HC 94.016 ED A ESSENCIALIDADE DO POSTULADO DO DEVIDO PROCESSO LEGAL, QUE SE QUALIFICA COMO REQUISITO LEGITIMADOR DA PRÓPRIA 'PERSECUTIO CRIMINIS'. - O exame da cláusula referente ao 'due process of law' permite nela identificar alguns elementos essenciais à sua configuração como expressiva garantia de ordem constitucional, destacando-se, dentre eles, por sua inquestionável importância, as seguintes prerrogativas: (a) direito ao processo (garantia de acesso ao Poder Judiciário); (b) direito à citação e ao conhecimento prévio do teor da acusação; (c) direito a um julgamento público e célere, sem dilações indevidas; (d) direito ao contraditório e à plenitude de defesa (direito à autodefesa e à defesa técnica); (e) direito de não ser processado e julgado com base em leis 'ex post facto'; (f) direito à igualdade entre as partes; (g) direito de não ser processado com fundamento em provas revestidas de ilicitude; (h) direito ao benefício da gratuidade; (i) direito à observância do princípio do juiz natural; (j) direito ao silêncio (privilégio contra a auto-incriminação); (l) direito à prova; e (m) direito de presença e de 'participação ativa' nos atos de interrogatório judicial dos demais litisconsortes penais passivos, quando existentes.- O direito do réu à observância, pelo Estado, da garantia pertinente ao 'due process of law', além de traduzir expressão concreta do direito de defesa, também encontra suporte legitimador em convenções internacionais que proclamam a essencialidade dessa franquia processual, que compõe o próprio estatuto constitucional do direito de defesa, enquanto complexo de princípios e de normas que amparam qualquer acusado em sede de persecução criminal, mesmo que se trate de réu estrangeiro, sem domicílio em território brasileiro, aqui processado por suposta prática de delitos a ele atribuídos. O INTERROGATÓRIO JUDICIAL COMO MEIO DE DEFESA DO RÉU. - Em sede de persecução penal, o interrogatório judicial notadamente após o advento da Lei nº 10.792/2003 - qualifica-se como ato de defesa do réu, que, além de não ser obrigado a responder a qualquer indagação feita pelo magistrado processante, HC 94.016 ED também não pode sofrer qualquer restrição em sua esfera jurídica em virtude do exercício, sempre legítimo, dessa especial prerrogativa. Doutrina. Precedentes. POSSIBILIDADE JURÍDICA DE UM DOS LITISCONSORTES PENAIS PASSIVOS, INVOCANDO A GARANTIA DO 'DUE PROCESS OF LAW', VER ASSEGURADO O SEU DIREITO DE FORMULAR REPERGUNTAS AOS CO-RÉUS, QUANDO DO RESPECTIVO INTERROGATÓRIO JUDICIAL. - Assiste, a cada um dos litisconsortes penais passivos, o direito fundado em cláusulas constitucionais (CF, art. 5º, incisos LIV e LV) - de formular reperguntas aos demais co-réus, que, no entanto, não estão obrigados a respondê-las, em face da prerrogativa contra a auto-incriminação, de que também são titulares. O desrespeito a essa franquia individual do réu, resultante da arbitrária recusa em lhe permitir a formulação de reperguntas, qualifica-se como causa geradora de nulidade processual absoluta, por implicar grave transgressão ao estatuto constitucional do direito de defesa. Doutrina. Precedente do STF." A douta Procuradoria Geral da República, ao deduzir os presentes embargos de declaração, alega, em síntese, o que se segue (fls. 2.221/2.228): "(...) o acórdão incorreu, acidentalmente, em erro material na sua parte dispositiva. Isso acabou por gerar contradição e obscuridade no julgado (...). ..... (...) Muito embora toda a consistente fundamentação desenvolvida no voto condutor tenha-se debruçado exclusivamente sobre a questão da realização do interrogatório dos co-réus sem a participação dos defensores do paciente; muito embora em nenhum momento o acórdão desta Egrégia Corte haja feito qualquer consideração sobre quaisquer outros atos processuais praticados no curso da ação penal em epígrafe, a parte dispositiva, por equívoco, acabou determinando a HC 94.016 ED anulação dos atos praticados desde os interrogatórios judiciais dos demais co-réus. (...): ..... A interpretação literal do excerto em epígrafe pode levar à equivocada interpretação – não condizente com a fundamentação judicial – de que não foram anulados somente os interrogatórios dos co-réus, mas todos os atos processuais praticados posteriormente aos mesmos. Isso porque, nos termos da legislação vigente à época – recentemente alterada pela Lei nº 11.719/2008 – o interrogatório (que atualmente é o último ato da instrução, nos termos do art. 532 do CPP) era levado a efeito logo no início da instrução criminal, antes da oitiva das testemunhas. Desde modo, ao ler palavra 'desde' na frase 'concedo, de ofício, a ordem de habeas corpus ao ora paciente, para anular, desde os interrogatórios judiciais dos demais co-réus [...] o Processo-Crime nº 2006.61.81.008647-8 [...]', pode-se chegar à conclusão – certamente não pretendida por esta Excelsa Corte – que todos os demais atos processuais realizados posteriormente, tais como as oitivas de testemunhas, deveriam ser renovados. Tanto não foram essas as conseqüências desejadas por esta Suprema Corte que o dispositivo do acórdão, nas linhas seguintes, assume linguagem mais restritiva, determinando tão somente a realização de novos interrogatórios, sem fazer qualquer menção à renovação de outros atos processuais (vide nesse sentido, a parte sublinhada do excerto acima transcrito). E não poderia ser diferente, na medida em que, segundo o princípio da causalidade – ou da conservação do ato processual, na terminologia utilizada por Frederico Marques –, que rege a teoria das nulidades, só se contaminam da invalidade de um determinado ato aqueles outros que com ele guardem alguma relação de necessidade. (...): ..... Levando em conta tais reflexões, não haveria sentido algum em se determinar a anulação de depoimentos de testemunhas em que foram perfeitamente asseguradas as garantias do contraditório e da ampla defesa (nesse sentido, veja-se a certidão expedida pela Justiça Federal referente aos autos nº 2006.61.81.008647-8, à fl. 2207 – bem como as informações prestadas pelo juízo de origem à fl. 2203). Não há qualquer liame de HC 94.016 ED causalidade entre estes atos e aqueles atacados pelos impetrantes (interrogatórios). Entender a questão de outro modo seria negar qualquer eficácia aos princípios da efetividade e da instrumentalidade do processo. (...): ..... No caso em apreço, mais inusitada ainda seria a compreensão de que devem ser renovados todos os atos processuais cronologicamente subseqüentes ao interrogatório, se levarmos em consideração que, com a já mencionada reformulação do procedimento comum pela Lei nº 11.719/2008, o interrogatório passou a ser o último ato da instrução. Assim, tomando-se o processo em epígrafe, tendo em vista que as leis processuais têm aplicação imediata, seriam primeiramente repetidas todas as numerosas oitivas de testemunhas – que já haviam sido realizadas sob o crivo do contraditório e da ampla defesa, insista-se – para, somente ao final, serem renovados os atos processuais efetivamente nulos." (grifei) A parte embargada, por sua vez, ao manifestar-se sobre os embargos de declaração (fls. 2.284/2.294), sustenta que "(...) não houve erro material, nem obscuridade ou contradição: o processo foi anulado desde os interrogatórios, inclusive. E a ressalva final a como devem ser feitos os novos interrogatórios é lógica, porque a base do 'habeas' era justamente contra a forma com que foram realizados pelo juízo de primeiro grau" (fls. 2.286 – grifei). Submeto, desse modo, ao exame desta colenda Segunda Turma, os presentes embargos de declaração. É o relatório.",

"voto": "O SENHOR MINISTRO CELSO DE MELLO - (Relator): Entendo assistir plena razão à parte ora embargada, eis que não há, no acórdão ora impugnado, qualquer evidência de obscuridade, omissão ou contradição a sanar, circunstância essa que torna processualmente inviável o recurso em exame. Como se sabe, os embargos de declaração destinam-se, precipuamente, a desfazer obscuridades, a afastar contradições e a suprir omissões que eventualmente se registrem no acórdão proferido pelo Tribunal, acórdão de Esa modalidade embargado, viabilizar quando um recursal só permite o reexame utilizada com pronunciamento integrativo-retificador, o específico jurisdicional vocacionado a afastar as do objetivo de caráter situações de obscuridade, omissão ou contradição, e a complementar e esclarecer o conteúdo da decisão proferida. Desse modo, a decisão recorrida - que aprecia, como no caso, com plena determinada pretensão recursal dos embargos jurídico-processual exatidão jurídica de e em - não declaração, dessa toda permite sob modalidade pena de a sua o de inteireza, emprego grave recurso, da via disfunção eis que HC 94.016 ED inocorrentes, em tal situação, os pressupostos que justificariam a sua adequada utilização. Cumpre enfatizar, de outro lado, que não se revelam admissíveis os embargos de declaração, quando a parte embargante - a pretexto de esclarecer uma inexistente situação de obscuridade, omissão ou contradição – vem a utilizá-los com objetivo de infringir o julgado e de viabilizar, assim, um indevido reexame da causa (RTJ 191/694-695, Rel. Min. CELSO DE MELLO, v.g.). É por tal razão que a jurisprudência do Supremo Tribunal Federal, ao versar os aspectos ora mencionados, assim se tem pronunciado: "Embargos declaratórios. Inexistência de omissão, contradição, obscuridade ou dúvida, no acórdão embargado (art. 337 do RISTF). Embargos rejeitados. O que pretenderam os embargantes foi sustentar o desacerto do julgado e obter sua desconstituição. A isso não se prestam, porém, os embargos declaratórios." (RTJ 134/1296, Rel. Min. SYDNEY SANCHES - grifei) "- A jurisprudência do Supremo Tribunal Federal tem ressaltado que os embargos de declaração não se revelam cabíveis, quando, utilizados com a finalidade de sustentar a incorreção do acórdão, objetivam, na realidade, a própria desconstituição do ato decisório proferido pelo Tribunal. Precedentes: RTJ 114/885 RTJ 116/1106 - RTJ 118/714 - RTJ 134/1296." (AI 153.147-AgR-ED/RS, Rel. Min. CELSO DE MELLO) HC 94.016 ED "- O recurso de embargos de declaração não tem cabimento, quando, a pretexto de esclarecer uma inocorrente situação de obscuridade, contradição ou omissão do acórdão, vem a ser utilizado com o objetivo de infringir o julgado." (RE 177.599-ED/DF, Rel. Min. CELSO DE MELLO) "Embargos declaratórios só se destinam a possibilitar a eliminação de obscuridade (...), contradição ou omissão do acórdão embargado (art. 337 do RISTF), não o reconhecimento de erro de julgamento. E como, no caso, é esse reconhecimento que neles se reclama, com a conseqüente reforma do acórdão, ficam eles rejeitados." (RTJ 134/836, Rel. Min. SYDNEY SANCHES - grifei) O exame dos autos evidencia que os presentes embargos declaratórios consideradas revestem-se as razões de nítido expostas pela caráter própria infringente, parte embargante, circunstância essa que, por si só, basta para tornar incabível a espécie recursal ora em análise, consoante adverte o magistério jurisprudencial desta Corte, na linha dos precedentes anteriormente referidos. Impende assinalar, ainda, por necessário, e ao contrário do que sustenta a douta Procuradoria Geral da República, que a nulidade radical que afeta os diversos atos de interrogatório judicial, quando provocada pela conduta do magistrado que arbitrariamente nega, ao réu, o direito – por este titularizado – de HC 94.016 ED formular reperguntas contamina, processo, por aos efeito notadamente demais causal, litisconsortes todos aqueles de os índole penais atos passivos, subseqüentes probatória, do como a inquirição de testemunhas arroladas pelo Ministério Público. É que, no caso, há nítida relação de causa e efeito entre a conduta judicial e as conseqüências geradas no âmbito do processo penal de conhecimento, pois, com a arbitrária recusa do magistrado em permitir, ao réu, a legítima formulação de reperguntas aos demais litisconsortes penais passivos, o ora paciente, por intermédio de seus Advogados, deixou de obter dados informativos que lhe poderiam ser úteis quando da inquirição das testemunhas arroladas pelo Ministério

Note: Sample of legal ruling highlighting the spread of the critical information that should be covered by the summary.
Source: The Author

Figure 1.2 – Methodology Overview



Source: The Author

1.2 Methodology

Figure 1.2 depicts our research methodology. The blue rectangles denote the tasks in which original contributions are proposed, while the blank boxes reflect methods from background work. Our first step was to create a dataset containing court rulings (step 1). Each instance is composed of a legal ruling and its corresponding reference summary. These summaries are used both for training and evaluation, as they are required to calculate the quality metrics for the automatically generated summaries. The creation of the dataset is discussed in detail in Chapter 5. A paper describing the dataset was presented at the PROPOR 2018 conference (FEIJO; MOREIRA, 2018).

The tasks where the input is a sequence and the output is also a sequence are called sequence-to-sequence. Text summarization and translation are examples of these tasks. They require building an understanding of the context before generating the summary. Recurrent Neural Networks (RNN) (RUMELHART; HINTON; WILLIAMS, 1986) were the most common type of architecture for sequence-to-sequence tasks (SUTSKEVER; VINYALS; LE, 2014; BAHDANAU; CHO; BENGIO, 2014; CHO et al., 2014b) until the

proposal of the Transformer model (VASWANI et al., 2017). The problem with RNNs is that they are built using recursion, so each step is processed sequentially. The Transformer model no longer depends on a recursion allowing it to take advantage of the parallel processing power provided by Graphical Processing Units (GPU) and Tensor Processing Units (TPU).

We designed an experiment to investigate the suitability of the Transformer model applied to the legal domain. In this experiment, the Transformer model achieved the best results compared to attention RNNs and extractive approaches. These experiments are steps 2a and 2b in Figure 1.2. The experimental setup and the results are discussed in Section 7.1. This experimental evaluation was reported in a paper presented at the RANLP 2019 conference (FEIJO; MOREIRA, 2019).

Step 3 presents the conventional metrics for evaluation of sequence-to-sequence tasks (as discussed in Section 2.8). Steps 4 and 5 describe the text summarization task applied to legal texts and general texts.

Bidirectional Encoder Representations from Transformers (BERT) (DEVLIN et al., 2018) was an improvement over fixed word embedding representations. The idea is to train an unsupervised model using large volumes of text and fine-tune it to some desired task. With this technique, BERT achieved state-of-the-art results in eleven Natural Language Processing (NLP) tasks, such as Question Answering, Recognizing Textual Entailment, Paraphrasing, Semantic Textual Similarity, among others.

Although it cannot be directly applied to text summarization, BERT's good results motivated us to analyze its suitability for improving text comprehension in summarization. Training a BERT model is an expensive task and requires considerable processing power and memory. A significant improvement in terms of efficiency was made by a model called A Lite BERT (ALBERT) (LAN et al., 2019). This model reduced the size requirements without jeopardizing its language modeling capabilities.

Unfortunately, the authors of both BERT and ALBERT did not release a model using only Portuguese. The models released were in English, Chinese, and a Multilingual model, trained using Wikipedia articles in 104 languages (including Portuguese). Thus, we pre-trained both BERT and ALBERT using Wikipedia and a few other Portuguese corpora. The goal was to compare our pre-trained Portuguese models with the available BERT multilingual, evaluating them in seven language understanding tasks. This analysis showed that the fine-tuning technique could reach state-of-the-art results in Portuguese, even when using small datasets, in many different kinds of tasks. This step is represented

by the number 6 in Figure 1.2, and the details of this analysis are discussed in Chapter 3.

The proposal of LegalSumm is depicted in step 7. Then, in steps 8 and 9, the proposed technique is applied to summarization, and the results are analyzed.

1.3 Contributions

The intended contributions of this thesis are:

- LegalSumm, a method for improving text summarization applied to the legal domain.
- RulingBR[1], a public dataset for text summarization containing court rulings in Portuguese and their reference summaries.
- BertPT and AlbertPT[2], language models trained in Portuguese.

1.4 Structure of the Text

The remainder of this work is organized as follows. Chapter 2 reviews alternatives for text representation, vocabulary requirements, and the problem with Out-Of-Vocabulary tokens. It also introduces the main standard extractive approaches and the techniques for generating abstractive summaries. Finally, it discusses the main evaluation metrics used in the summarization context. Chapter 3 describes our analysis using BERT for seven different NLP tasks in Portuguese, including textual entailment required for legal text summarization. Chapter 4 revises different approaches that were applied to summarization in the legal domain. Chapter 5 describes the RulingBR dataset, the collection process, its main characteristics, and some statistics the show the difficulties that may be faced while generating a summary. Chapter 6 discusses the current text summarization issues when applied to the legal domain and proposes LegalSumm to deal with these problems. Chapter 7 describes the experiments both using extractive and abstractive approaches for summarization using the RulingBR dataset. Finally, Chapter 8 summarizes the work done in this research and points out opportunities for future work.

[1] <https://github.com/diego-feijo/rulingbr/>
[2] <https://github.com/diego-feijo/bertpt/>

2 BACKGROUND

There are two main approaches to text summarization. The first, known as *extractive*, works by selecting entire sentences directly from the source text. This approach has been the most widely used solution for several years (LUHN, 1958; EDMUNDSON, 1969; ERKAN; RADEV, 2004; MIHALCEA; 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 to preserve the ideas and meanings of the sentences. The scoring function should capture how well the selected sentences represent the text and cover its topics. 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. It is possible to paraphrase the original and use words that did not occur in the source document. This method is more similar to the way a human would create a summary. Most recent research has focused on this approach.

Before diving into these approaches, one needs to understand how to represent the text for the model and how this representation may interfere with the results produced by a model. The following section addresses this topic.

2.1 Vocabulary

Representing texts using a sequence of words helps exchange information between humans. However, for most NLP models, it is helpful to create a vocabulary with which its corresponding index represents text tokens. This representation often leads to a lower memory requirement.

The usual way of creating a vocabulary is by splitting the text into representation units, keeping the most frequent units found. Even the vocabulary for 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.

Therefore, the construction of the vocabulary requires the definition of this unit

of representation. Each definition has its advantages and drawbacks that will be briefly discussed below. Nevertheless first, one should analyze a problem when the vocabulary cannot represent the text.

2.1.1 Out-of-Vocabulary

The term *out-of-vocabulary* (OOV) designates the problem that happens when the vocabulary has no unit to represent some piece of text. For instance, suppose that a single language text, say English, contains an unusual Chinese expression composed of Chinese characters. In this case, even if the vocabulary were composed of all the English characters, it would not have how to represent the Chinese characters. Chances are, these Chinese characters were not frequent enough to be chosen to be in the vocabulary.

The cause of this problem is that the vocabulary construction must rely on a finite number of possibilities. It is designated to build a representation for the most common units found in the training data. Using the entire Unicode may lead to having representation for any possible unit in the text. Now every unit can be represented, but it is possible that for most units, there is not a single example in the training data for the model to build any inference of its meaning.

Therefore, it is not enough to have the token represented, but the model must have access to sufficient samples to build a meaningful representation. Underrepresented units are not useful for the model because if they are never seen or occur only a few times, the model cannot learn when to use them in a context. So, the model may have to choose to ignore these characters or to replace them with a specially reserved token (often represented by the *UNK* acronym).

2.1.2 Character Representation

The most straightforward way of creating the vocabulary is by using the alphabet. However, the letters are not enough. Both lower and uppercase letters, diacritics, numerical digits, and special characters are also needed. With this representation, there are no predefined words, just sequences of characters.

The advantage of this representation is in its size. A few hundred characters will probably be enough for representing most of any single language text. As shown in previ-

ous section, even with this configuration, a text may contain characters not present in the training data. Despite being alleviated, the problem with OOV remains.

The drawback of this representation is in the length of sequences and the meaning associated with each unit. When letters represent the sentences, they will become longer. This is problematic because the model will require lots of memory to keep track of what has been read. For instance, suppose a particular language/domain has an average length for characters per token of μ and a sentence length of n, now each sentence will be $length = n * \mu$. Another drawback is the superposition of meanings over the same unit. While the model is learning the meaning of the sequence, it may also be learning some meaning for each unit. The meaning of the characters used too often will be probably be jeopardized by their common usage.

2.1.3 Word Representation

The most common form of creating a vocabulary is splitting text into words/tokens. It is usual to employ tokens to refer to these units because they can be numbers, dates, names, acronyms, etc. For western languages, it is common to use the space character as a separator. Many other characters may be used as separators like parentheses, comma, full stop, colon, etc.

After splitting, a frequency table counts how often each token is found in the training data is created. The top $vocab_{size}$ frequent tokens are used as vocabulary. That means that the least represented tokens will become OOV and need to be described as a specially reserved UNK token. The vocabulary needs to be huge to reduce the number of OOV. An extensive vocabulary is also problematic because the neural network output layer must have its size. This means that the probability of choosing the correct output is reduced as the vocabulary increases. Also, even if infrequent tokens are represented in the vocabulary, their occasional use is insufficient for the model to learn when to use them correctly.

It is possible to reduce the size of the vocabulary by converting all characters to lowercase and replacing numbers and dates with zero representations, removing diacritics (ç => c, ñ => n, á => a). Also, for tasks that do not require language modeling (text generation), it is possible to use stemming for reducing word variants their roots. Stemming the words "*connecting*" and "*connects*", removes their suffixes leading to a common representation "*connect*". Even with the text simplifications, the problem with OOV re-

mains because not all possible words are represented. Misspelled words, proper nouns, and many other situations still would lead to OOV.

The advantages of using word/token representation are that it is possible to represent long texts and create or use pre-trained word embeddings from Word2Vec (MIKOLOV et al., 2013), FastText (BOJANOWSKI et al., 2017), Wang2Vec (LING et al., 2015), or Glove (PENNINGTON; SOCHER; MANNING, 2014). Using these representations, the model will often improve its performance.

2.1.4 Subword Representation

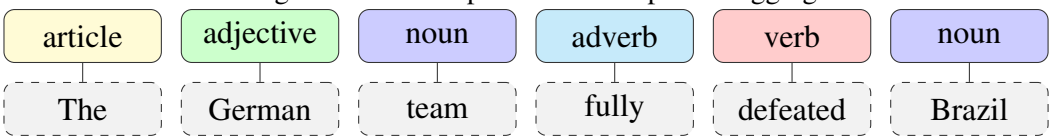
Using character representation reduces the OOV problem, but it is memory expensive; word representation is more memory efficient, but the issue of OOV increases. The alternative to mitigate the disadvantages of these two approaches is to use *subword units* (SCHUSTER; NAKAJIMA, 2012; CHITNIS; DENERO, 2015; SENNRICH; HADDOW; BIRCH, 2015; KUDO, 2018) as the token representation. The idea is that a token would represent a typical pattern seen in the training data rather than words or characters. This operates with a fixed vocabulary size and assigns a token to the most common patterns found.

With this method, the OOV problem is reduced as a combination of subwords can represent one word. The problem of longer sequences is also addressed because a unit may represent several characters. This kind of representation is being used by many of the most recent NLP models (LAN et al., 2019; DEVLIN et al., 2018)

2.2 Term Weighting

Term weighting aims at quantifying the importance of terms in documents. The most widely known term weighting scheme is TFIDF. Term Frequency (TF) designates the number of occurrences of some term for each document. Inverse Document Frequency (IDF) is the inverse of the number of occurrences of some term in the collection. The product between these two is often referred to as (TFIDF) and reflects the weight (specificity) of this term for each document (JONES, 1972).

Figure 2.1 – Example of Part-of-Speech Tagging



Note: POS tagging requires semantic disambiguation.
Source: The Author

2.3 Part-of-Speech Tagging

Part-of-speech (POS) tagging is the process of labeling each word with its grammatical function in the sentence. The role is not fixed for each term and may change according to the context. So, this task requires some comprehension of the text and the relation among the words.

Figure 2.1 shows an example of this process. The tagger must determine the grammatical function of each word in the sentence. There are nine possible word classes: noun, pronoun, verb, article, adjective, adverb, conjunction, preposition, and interjection. Modern linguistic annotation frameworks further refine these word classes, adding some classes for punctuation and other symbols. For example, Universal Dependencies framework (MARNEFFE et al., 2021) distinguishes 17 coarse-grained classes of words and other elements.

2.4 Extractive Approaches

In this section, we analyze the standard extractive techniques found in the literature. The general technique is to break the text into sentences, apply some scoring function over these sentences, and keep as many sentences as needed until the desired summary length is reached. Despite this simple concept, several difficulties arise when implementing this technique.

First, splitting the text into sentences is not a trivial task. When creating a scoring function, most methods need to build term frequency counts. Also, computing similarities between sentences can be challenging. Frequent solutions include removing stopwords and using stemmers to build more meaningful term frequencies. However, the algorithm becomes language-dependent when using these techniques.

2.4.1 Luhn

Proposed by Luhn (1958), the seminal method for determining the importance of a sentence is calculated using just TF. Significant words are selected among the most frequent words found in the document. The highest-scoring sentences are selected to be part of the summary.

When calculating word frequencies, Luhn's 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. This technique is language-specific. Thus a stemmer designed explicitly for the target language would have different behavior.

2.4.2 LexRank

LexRank is a stochastic graph-based method for computing the relative importance of sentences (ERKAN; RADEV, 2004). It assumes that the main idea of a text is often paraphrased and shares many common words. As a consequence, finding similar sentences would be the same as finding the crucial sentences. Also, the central sentence of a cluster indicates that this sentence is the most similar among them and would probably capture more information.

2.4.3 TextRank

TextRank is also a graph-based ranking model of deciding the importance of a vertex within a graph (MIHALCEA; 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. This original version of the algorithm is implemented in Sumy Library (BELICA, 2018) that will be used for experiments.

One improved version was implemented in the Gensim Library (ŘEHŮŘEK; SOJKA, 2010), using 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. Further, both versions will be used in our experiments.

2.4.4 SumBasic

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

2.4.5 KLSum

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

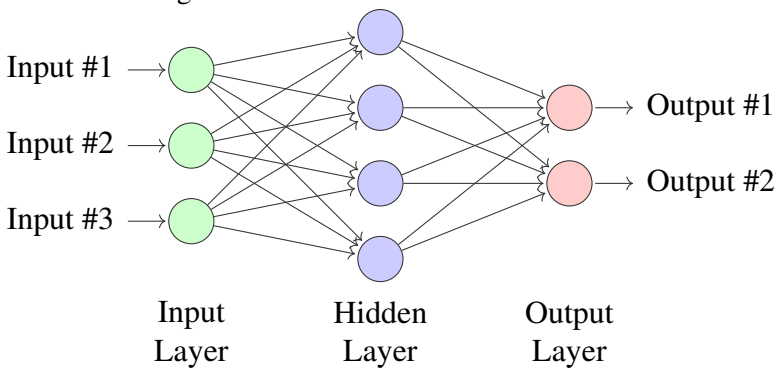
2.4.6 Latent Semantic Analysis

Latent Semantic Analysis (LSA) (STEINBERGER; JEZEK, 2004; GONG; LIU, 2001) generates a sparse *token x sentences* matrix, applies Singular Value Decomposition (SVD) and selects the singular vector that will define the scores for each token. Sentences with the highest normalized scores are chosen to compose the summary.

2.5 Abstractive Approaches

When humans write summaries, they usually select the main subjects and rewrite the text using their own words in a process known as *paraphrase*. Extractive approaches are limited to the words and the sentences that were seen in the original. This is a big issue because it does not correctly mimic human behavior when writing a summary. Abstractive approaches aim to address this drawback. They can read entire blocks of texts and generate summaries using sentences and words that were not present in the original text. Many NLP models are suitable to be used in this abstractive approach. The standard neural network architecture for use with sequences is recurrent networks, described in the

Figure 2.2 – Feedforward Neural Network



Note: This network has fixed size both for input and output.
Source: Adapted from <http://www.texample.net/tikz/examples/neural-network/>

next section.

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

2.5.1 Recurrent Neural Networks

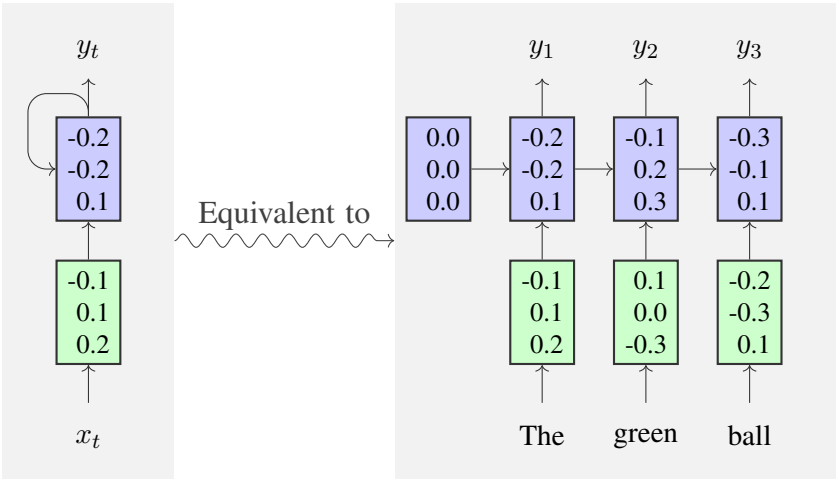
The Feedforward Neural Network (FNN) is the simplest form of neural network. A sample of this architecture can be seen in Figure 2.2. Its architecture has a fixed size input, and it also generates a fixed size output.

A text is a sequence of tokens. An FNN is not adequate to deal with sequences since it requires the entire text input to be encoded and fit into the input size. In the same way, the entire summary output would be generated directly. One better way of doing this is by using the output of an FNN as an additional input for the FNN, creating a recursion. This is the idea behind the Recurrent Neural Network (RNN) (RUMELHART; HINTON; WILLIAMS, 1986).

With this modification, the network can use its previous outputs to decide the next. Each output now is also called a state, and the previous state is appended to the current input to decide the next output/state. Figure 2.3 shows a sample representation of this architecture.

Theoretically, this architecture would allow any previous state to be considered, and its weights are adjusted using backpropagation. However, when the backpropagation computes the gradients for adjusting the weights in the network, two common problems are found: vanishing and exploding gradients. These problems prevent the network from

Figure 2.3 – Recurrent Neural Network



(a) Rolled RNN (b) Unrolled RNN

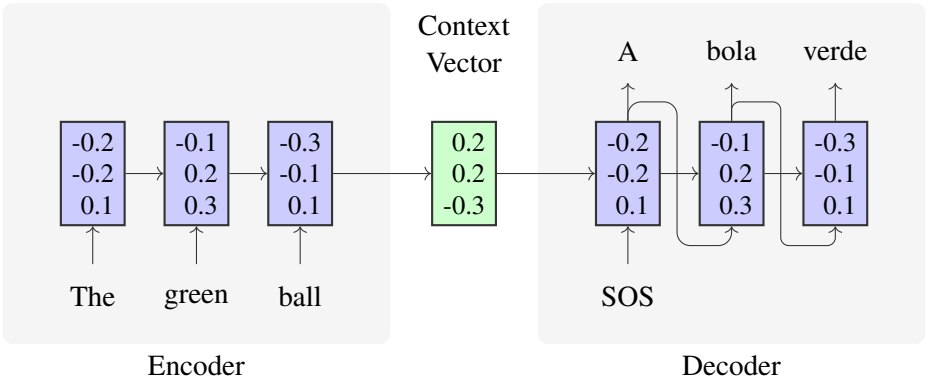
Note: Green: embedding layer; blue: hidden layer. The output from the hidden layer is fed as additional input for the next step. (a) Shows the loop in the hidden layer. (b) Shows the unrolled version where the output from the previous step is fed as input to the current step.

Source: The Author

learning long-range dependencies. For instance, assuming the current sentence is encoded as an input, solving its meaning requires using a state processed several steps before. The two most prominent solutions to deal with these problems are the use of Long Short-Term Memory (LSTM) (HOCHREITER; SCHMIDHUBER, 1997) and Gated Recurrent Unit (GRU) (CHO et al., 2014a). Both work by adding direct connections that allow the input to be wholly used, combined, or forgotten when generating the output/state. With these changes, the network improves its ability to handle long-range dependencies.

One issue that needs to be addressed is that Neural Networks require that both inputs and outputs have a fixed length, and that is not the case when dealing with text because each document (or sentence) can have a different size. 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 production 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.

Figure 2.4 – Encoder-Decoder Architecture



Source: The Author

2.5.2 Encoder-Decoder Architecture

RNNs can also learn a compressed vector representation that another RNN could later use to generate another text. This idea was originally called sequence-to-sequence (SUTSKEVER; VINYALS; LE, 2014) architecture. This same architecture was later called encoder-decoder because the first network is responsible for generating this encoded representation. The second one is trained to use this context to create the output.

For some NLP problems like summarization and translation, the network cannot start creating the output directly because it lacks enough context. In these cases, the network (encoder) needs to read the entire input, create an internal representation of this input, and generate the output.

Figure 2.4 illustrates the encoder-decoder architecture. It is composed of two RNNs. The first one is the encoder, and the second is the decoder. The last state of the encoder is the context vector. In the first step of the decoder, the context vector and a reserved start of sentence (SOS) token are combined to generate the first output. For the following steps., the input is formed from the previously made output combined with the previous state. The context vector should capture all the content read in the encoder. This should have enough representation to understand both syntax and semantics.

Word embeddings use a similar concept (MIKOLOV et al., 2013; LING et al., 2015; BOJANOWSKI et al., 2017; PENNINGTON; SOCHER; MANNING, 2014). They share the idea of representing both syntax and semantics from words using a vector. Each number that composes the vector can be thought of as a dimension representing some feature as to whether it is singular or plural, its gender, and many others. These representations have some interesting properties, like symmetry, which allow some arithmetic

operations. The most widely known example is using the vector from the word “king”, subtracting the vector from “man”, and adding the vector from the “woman”, which results in a vector whose closest neighbor is the vector representation of the word “queen” (MIKOLOV et al., 2013).

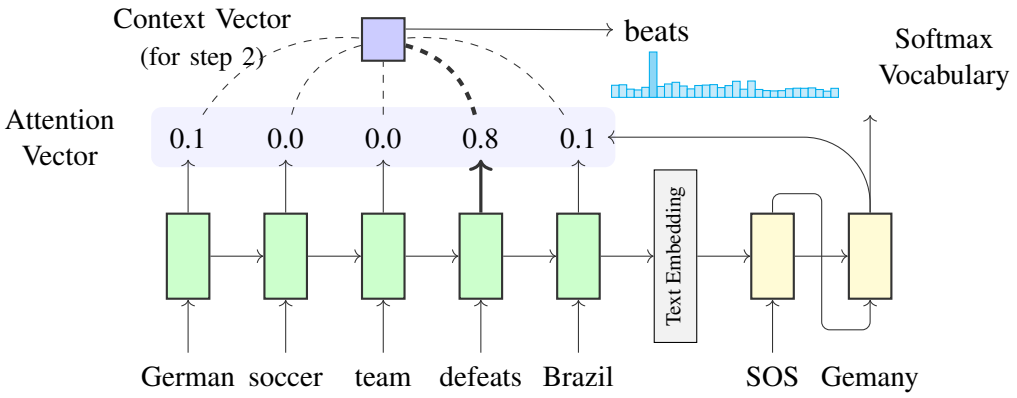
Representing multiple meanings of words using vectors seems possible. The amount of information encoded in a vector is significant and probably enough to represent the most relevant features. However, considering their order in the sentences, positions, and previous and further contexts, several possibilities arise when words are combined. The words interact with each other changing their meanings. So, despite their success in representing words, using just a vector to represent all features of an entire sentence or document can be too difficult.

2.5.3 Attention Networks

Another solution to deal with long-range dependencies is to train the network to identify relevant data from the source input when generating the output (BAHDANAU; CHO; BENGIO, 2014). The intuition behind this approach is that when the decoder is generating the output, it could have more information than just the context vector. This additional information could provide direct information from the source to the decoder when it was needed. This technique was initially called *alignment*, and it was used for translation. In reality, alignment and attention should be considered as different things. In a Neural Machine Translation (NMT) context, the alignment would represent the mapping from source to target language. On the other hand, *attention* would represent which context should be considered while choosing the target word, so it would probably be often blurred. 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, the order of the tokens can be different, and several words can influence others.

To illustrate this process, Figure 2.5 represents a summarization model. It follows the general idea of the encoder-decoder network. The encoder is represented in green. For each step in the decoder, represented in yellow, a new attention vector is calculated over the entire input. The context vector is the weighted sum of the hidden states of the encoder using the attention vector. Finally, the decoder context vector and hidden state are combined to generate the final output for the second step. In the illustration, the attention

Figure 2.5 – Attention Network



Note: A new attention vector and context information is calculated for each step in the decoder.
Source: The Author

vector increases the attention over the encoder hidden state of the word “defeats” while the decoder generates the second word “beats”.

2.5.4 Computing Attention

Attention can be seen as a similarity search using some *queries* (the current decoder context) looking for similar *keys* (the hidden state from the input sequence) to retrieve the *values* associated with these keys. Eventually, queries, keys, or values may be represented by the same vectors according to some particular usage. For instance, it is common to represent the encoder hidden state both as key and value. Notwithstanding, most works have been using this distinction to keep it in the general standard form.

Computing attention scores is an essential part of this process. In other words, how the query vector will be combined with the key vector. One of the simplest ways of computing these scores is using a dot product (matrix multiplication) and applying the softmax function over it. Softmax is a function that normalizes values from a raw distribution into a probability distribution. All values range from 0 to 1, and the sum of all values is equal to 1.

Despite its possible use, the mere dot product without a normalized vector is not a suitable similarity measure because it is biased by the vector magnitude. This means that the dot product of *queries* = $[[1, 1, 2]]$ with *keys* = $[[1, 1, 1], [4, 4, 4]]$ will results in greater value for the second key, which is not the expected. Also, *queries* and *keys* must have the same feature dimension for the matrix multiplication.

Table 2.1 – Attention Score Functions

Name	Equation
Dot Product	QK^T
General Dot Product[1]	QWK^T
Bilinear	$QW_qW_kK^T$
Additive	$V_a^T\tanh(W_qQ + W_kK)$
Multiplicative	$V_a^T\tanh(W_q[Q;K])$
Scaled Dot Product[2]	$\frac{QK^T}{\sqrt{d_k}}$
Masked Multi-Head Attention[3]	$\frac{(QK^T+M)}{\sqrt{d_k}}$

Note: The output of the score is given applying a softmax over the logits.

Table 2.1 summarizes the most frequent score functions found in the literature. The general dot product improves the dot product by adding trainable weights W. This means that these weights can be adjusted to fit a function to help mitigate any magnitude issue and allow different feature dimensions between *queries* and *keys*. Bilinear has trainable weights W_q and W_k, allowing more complex mapping functions.

Additive (BAHDANAU; CHO; BENGIO, 2014) was proposed for learning to align the input with the generated output. Instead of o product between *queries* and *keys*, it uses the product with trainable weights W_q and W_k and combines them with an addition. It uses a nonlinear function *tanh* to normalize values between $(-1; 1)$ and a final trainable values V_a to compute the scores.

Multiplicative (LUONG; PHAM; MANNING, 2015) was made by just concatenating *queries* and *keys* (though requiring that they share a common dimension). This function uses just two trainable vectors instead of the three required by the additive form.

The Transformer model (VASWANI et al., 2017) replaced the recurrence and convolutions with an attention-based model. This architecture enables parallel processing of the whole sequence taking advantage of GPU's parallel computation. Figure 2.6 illustrates the architecture of the Transformer model. It is composed of a stacked number of Encoders and Decoders. As the whole sequence is processed together, a causality mask is needed to forbid the model to peek at the next word in the series (because it uses teacher forcing when training).

This architecture introduces the scaled dot product, which is the dot product normalized by the size of the vectors (it requires both *queries* and *keys* to have the same dimension). This same model also introduces the masked multi-head attention. This

[1] where W is a matrix used to map to a common space *queries* and *keys*.
[2] where d_k is the size of the *keys*.
[3] where M is a mask of 0's or $-\infty$'s

function splits both *queries* and *keys* into eight small vectors allowing them to compute in parallel. Another contribution is introducing the masking necessary when the model is decoding and cannot use information from future steps.

Attention is excellent in detecting and attending to some desired feature from the source. However, when just one attention model is responsible for finding relevant information in the source, it may have a task that becomes too difficult. The Transformer created several attention heads, and each one may acquire some specialty, working as a feature detector. For instance, one head may be specialized in detecting actions, while another is good at detecting subjects or objects.

2.5.5 Local Attention

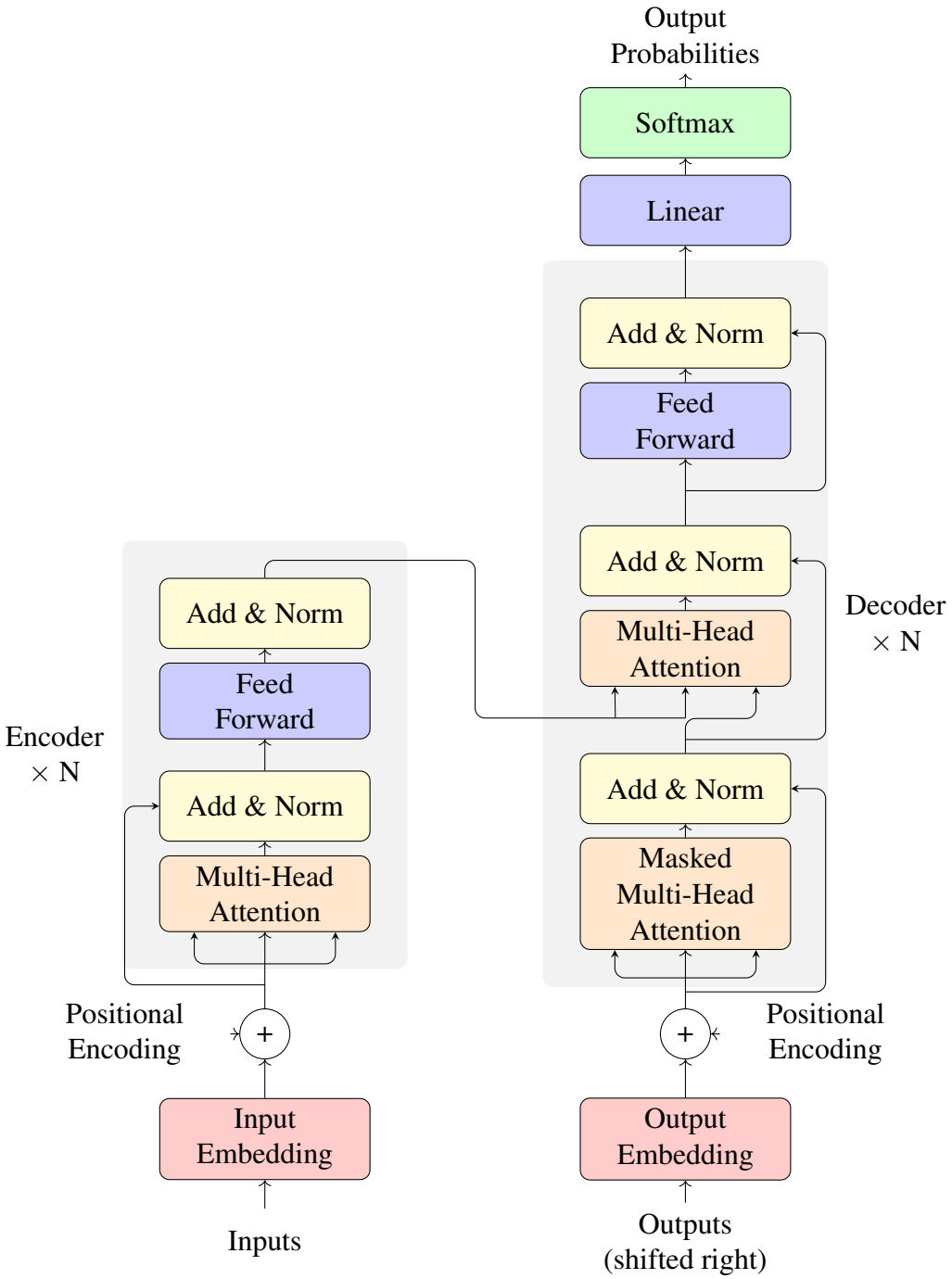
The traditional attention look for information using the whole input sequence. As the length of the input increases, the attention vector has more difficulty in targeting specific parts of the sequence (LUONG; PHAM; MANNING, 2015). The attention mechanism works well for short texts but struggles to focus on relevant information when applied to long documents.

In fact, despite its general form, global attention does not take advantage of some kind of positional alignment. Even for translation, where alignment is relatively common, global attention does not take advantage of this characteristic. That was the motivation behind the development of local attention, incorporating some structural biases.

Luong, Pham and Manning (2015) proposed restricting attention range to local attention by constraining attention to a window span whose hidden states were averaged. Cohn et al. (2016) examined adding absolute positional information, Markov condition over alignment, fertility, and symmetry. They showed that incorporating positional information for translation almost always is beneficial. They followed the empirical intuition that the alignment is prone to have a diagonal shape.

Beltagy, Peters and Cohan (2020) used the standard sliding window, a dilated sliding window, and global attention with a sliding window. The standard sliding window uses just n left and right neighbors from each position. The dilated sliding window uses n interspersed left and right positions, leaving some isolated spots between them. The best results were obtained using global attention with a sliding window. It allows full global attention for specially reserved tokens “CLS” and “SEP", and keeps the standard sliding window for everything else.

Figure 2.6 – Transformer Architecture



Source: Adapted from (VASWANI et al., 2017)

2.5.6 Self-Attention

These attention mechanisms were from the decoder hidden state being applied over the encoder hidden states. When RNNs are used, the current state of the model encodes some information from previous processed states. However, even LSTM or GRU cells suffer from vanishing or exploding gradients for long sequences, making it harder for the model to remember long-range dependencies. The Transformer (VASWANI et al., 2017) model replaced the RNN with six layers of self-attention. As the encoder is no longer recursive, the input is wholly processed in parallel, and there are several layers of self-attention in which the model can generate different relations. Also, as this model does not use recursion, and all words are processed in parallel, the model would be just a bag of words, not considering the positional information. The introduction of a sinusoidal positional encoding summed over the word embeddings input addressed this issue.

2.5.7 Locality Sensitive Hashing-Based Attention

When dealing with long sequences, global attention suffers from high spreading. Local attention enforces the focus using some heuristic over the positions. For some NLP tasks such as text summarization, where the sequences are long and the attention does not necessarily have symmetry, both these solutions do not seem to be good enough. For the summarization task, the model is not particularly focused on some words but more on a cluster that represents some topic that the summary should cover. One possible way of grouping topics with similar concepts is using the Locality Sensitive Hashing (LSH) algorithm.

This was used by the Reformer Model (KITAEV; KAISER; LEVSKAYA, 2020) replaced Transformer's Full Attention (scaled dot-product attention) by LSH-based attention allowing similar concepts to be grouped. It uses a hashing function to create buckets of positions where the attention is restricted to places in the same bucket. With this modification, the model uses a fraction of the memory required by the standard Transformer while having similar performance.

2.6 Dealing with Long Texts

Legal rulings are usually long texts with several thousand tokens. RNNs with LSTM and GRU cells (as described in Section 2.5.1) can propagate the context for just a few dozen steps. After this number of steps, the network starts to suffer from the vanishing or exploding gradient problem. This may not be a problem for short text translation, where each sentence can be translated soon after it was read. However, in a summarization context, the whole input should be processed before starting to output the summary; this is a significant drawback.

The Transformer model (VASWANI et al., 2017) was proposed to address some of these issues. The number of layers of attention used by the model is small when compared to the number of steps required by RNNs that is proportional to the length of the inputs. With this small number of layers, the problem of the vanishing or exploding gradient is alleviated. However, transformer-based models also cannot work with long texts. The transformer problem with long texts is caused by the large amount of memory required for the attention heads. The memory requirement has a complexity of $O(n^2)$ because, for each token in the text, one attention vector of the same size must be computed. In practice, this restricts the transformers-based models to use just a few hundred tokens, which is not good enough.

Recently, some works have addressed this memory requirement, reducing the size of the networks, sharing weights tensors among the layers (LAN et al., 2019), and using half-precision floating points. Even with these improvements, the models are unable to reach a few thousand tokens.

BERT also has this 512 sequence length limit. It managed to work with longer documents by using a sliding window approach. Following the example mentioned in BERT source code[4], where maximum sequence length was 5, and stride was 2:

Table 2.2 – Example of Sliding Window Approach

Original:	the man went to the store and bought a gallon of
Span 1:	the man went to the
Span 2:	to the store and bought
Span 3:	and bought a gallon of

For this document, each text span is fed as an example in the same batch. The stride is used to keep some context at the start and the end of each sequence. Later, the

[4] <https://github.com/google-research/bert/blob/master/run_squad.py>

weights from the final output layer are combined, avoiding counting the tokens belonging to the stride. Finally, tokens from these examples are combined for the specific objective of the task.

Liu et al. (2018) proposed using a standard extractive approach (TFIDF, TextRank, SumBasic) to reduce the input length and use the Transformer Decoder to generate the summary abstractively. The disadvantage of this approach is that these extractive approaches cannot be trained. The advantage is that it mimics the human behavior of first highlighting some sentences and then generating an abstractive version for the summary.

The Reformer model (KITAEV; KAISER; LEVSKAYA, 2020) managed to work with several thousand tokens. It adapted the Transformer model and replaced the full attention with a hash-based attention model. Locality Sensitive Hashing (LSH) is an algorithm used to group similar shingles into buckets and restrict the attention to query just the similar buckets. Despite these impressive features, this model is unstable, and we have not seen any work successfully training it for abstractive summarization.

The LongFormer model (BELTAGY; PETERS; COHAN, 2020) was also able to work with a few thousand tokens. They use a sliding window for standard tokens and global attention only for special tokens “CLS” and “SEP". The disadvantage is that far distant dependency cannot be resolved for standard tokens. The advantage is the reduced memory requirement allowing processing longer sequences. Also, nearby dependencies are more common than distant ones. Despite handling memory more efficiently than BERT, this model still requires GPUs with more memory than we have access to.

2.7 Fact Checking

While generating a novel summary, a language model may include some facts that were not present in the source text. Up to 30% of summaries (CAO et al., 2018) generated by abstractive models contain some inconsistency. The language model used to generate jokes or fantasy stories may have this creative behavior, but the model should keep the original intent of the ruling for the juridical domain. *Textual entailment* is an NLP task that evaluates whether a reference text entails a hypothesis text. BERT (DEVLIN et al., 2018) achieves state-of-the-art by fine-tuning in a number of NLPs tasks including *text entailment*. However, this task is usually based on a sentence-pair comparison and is insufficient to detect most problems.

Some false facts happen when infrequent tokens are used, for instance, proper

nouns or numbers. One possible solution is using pointer networks (VINYALS; FORTUNATO; JAITLY, 2015) for the source position. The approach of pointing to the source while decoding has already been applied using RNNs when the attention is not confident, or the token frequency is rare (SEE; LIU; MANNING, 2017).

Goodrich et al. (2019) proposed using a Part-Of-Speech (POS) tagger to build a pattern SUBJ-ACTION-OBJ and try to remove patterns that were not present in the source text. Kryściński et al. (2020) argued that this approach is not enough. They proposed a training data generation using paraphrasing, sentence negation, pronoun swap, entity swap, number swap, and noise injection. The idea is to teach the model to both detect inconsistency and point where the error is. This approach would filter the number of errors introduced by the decoder. The required set of rules is shown in Table 2.3.

Table 2.3 – Example of Factual Change Generation

Transformation	Original	Modified
Paraphrasing	the cat sat on the mat	on the mat the cat sat
Sentence negation	the cat was on the mat	the cat was not on the mat
Pronoun swap	the cat sat on its mat	the cat sat on your mat
Number swap	8 cats were sat on the mat	2 cats were sat on the mat
Entity swap	Barack Obama was president	Michael Jackson was president
Noise Injection	the cat sat on the mat	the ~~cat~~ sat on garden the mat

Unfortunately, most of these transformations impose challenges when applying them to Portuguese. The translation required for paraphrasing has to deal with legal terms that are often misunderstood. Sentence negation and pronoun swap require POS information; entity swap requires NER information; both tools need to be available trained in the target language. Using non-reliable methods in these transformations might generate unintelligible versions or even too subtle changes that the model would not distinguish the introduced error when compared to the original.

2.7.1 Coverage

Ideally, a good summary should cover the most relevant aspects of the input text. However, while the model is computing the attention, it is common to evaluate the same positions as relevant several times. One possible way of dealing with this problem is using a coverage vector (WU et al., 2016; SEE; LIU; MANNING, 2017) representing the

weighted sum of the attention vectors. Their model is trained to penalize the attention vectors in the same regions to increase coverage, encouraging the model to distribute the attention over the source input.

Another way of improving the coverage is restricting the attention to focus to some area using Local Attention, as seen before. In this case, the model may penalize when attention is being widely spread (more than one word) when the current position is far from the previous one (penalize large jumps) (COHN et al., 2016). As seen in Local Attention, while these techniques favor generating a symmetric attention distribution, at the same time, they yield high coverage and follow the same order of occurrence as in the input.

2.7.2 Recognizing Textual Entailment

Recognizing Textual Entailment (RTE) involves determining whether the meaning of one hypothesis text H is entailed (can be inferred) from a source text S (DAGAN et al., 2013). The main component of this definition is the inference process. The objective is to determine whether a consequence hypothesis can be derived from a set of given premises. This objective can be defined as a one-directional relation $S \implies H$.

Comparing this task to *text similarity* and *paraphrasing* gives a better understanding of the difference between them. Although these two tasks lack a precise definition, they have a bidirectional relation, where $S \iff H$.

The RTE task is helpful for question answering, unsupervised information extraction, machine translation, intelligent tutoring, and text summarization.

2.7.3 General Problems while Decoding

After the creation of the context representation, the Decoder model can start generating the output. The decoding is represented by a loop where the Decoder model receives the context and the previously generated token.

Beam Search

For each step, the decoder selects the token with the maximum estimated probability. If the previous step was a wrong choice, it would jeopardize the next steps.

Instead of keeping the output with the maximum probability, the decoder could track n best possible outputs. This alternative would create a tree, with each new output creating n new branches at each step. The Beam Search algorithm can avoid the exponential growth of this tree. For each step in the sequence, n new branches are appended for each of the previous n branches. A heuristic function is then used to sort and keep only n the best candidates among these n^2 alternatives.

Trigram Avoidance

Another common problem with abstractive approaches is the decoder getting stuck, generating the same few words several times. Paulus, Xiong and Socher (2017) proposed trigram avoidance during beam search. This approach is more straightforward to apply than using coverage mechanisms, as seen before. However, it does not fix the problem, as it still happens; it only masks its effects for the evaluation.

Loss versus Evaluation Function

Another problem observed is the use of an evaluation method for training and a different process for evaluation. While training, the model usually is adjusted to minimize the cross-entropy between the expected and the generated output. However, while evaluating, a completely different method will be applied (as studied in the next section). It is not immediately possible to use the same method because the evaluation methods are often non-differentiable. One way of dealing with this disadvantage is to use reinforcement learning approaches (PAULUS; XIONG; SOCHER, 2017; LI; BING; LAM, 2018; CELIKYILMAZ et al., 2018; CHEN; BANSAL, 2018) to improve summarization performance. However, these methods have the disadvantage of being hard to tune and generally slow to converge.

2.8 Evaluation

The evaluation of an automatic or even a human-generated summary is a challenging task. People may disagree on the critical content that the summary should contain. Thus, there are multiple possible ways of writing valuable summaries. It would be desirable for a thorough evaluation that many other reference summaries were available. These would allow a fairer evaluation to check how far the automatic summary is from any pos-

sible solutions. However, it is rare to have more than one reference summary available for each input text in real-world applications. Chances are, for a source text, just one summary, if any, will be available.

2.8.1 BLEU

In the context of machine translation, the Bilingual Evaluation Understudy (BLEU) (PAPINENI et al., 2002) is the most widely adopted. Machine translation is similar to the summarization task because both should produce an output based on an input text. From the same text, more than one reference translation would be acceptable.

For this metric, a high-quality translation resembles one or more references provided by humans. Although multiple translations are possible, usually high-quality translations are the ones that share many words and phrases with the references (PAPINENI et al., 2002).

This metric uses a modified n-gram precision. The goal is to penalize overgenerated terms that would match words in the reference translations. So, a reference word is considered exhausted after a matching candidate is found.

$$p_n = \frac{\sum_{c \in cand} \sum_{n\text{-}gram \in c} count_{clip}(n\text{-}gram)}{\sum_{c \in cand} \sum_{n\text{-}gram \in c} count(n\text{-}gram)}$$

The example provided by Papineni et al. (2002) is helpful to understand. To keep the explanation more straightforward, we will only be examining the unigram case, where each word is considered independently.

Candidate:	the the the the the the the	
Reference 1:	the cat is on the mat	$p_1 = \frac{2}{7}$
Reference 2:	there is a cat is on the mat	$p_1 = \frac{1}{7}$

Despite being an awful candidate translation, all its words are present in the references. So, the standard precision from the candidate to each reference is 1 because each word in the candidate is in the reference. Nevertheless, the modified unigram precision limits the counting by the number of occurrences in the references. So, for Reference 1, the word "the" can be used at most twice, implying modified unigram precision of $p_1 = \frac{2}{7}$.

The precision metric punishes verbosity because when more words are used, more

substantial will be the denominator. Shorter candidates are addressed by using brevity penalty factor BP as it can be seen in the BLEU equation:

$$\text{BLEU} = BP \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right)$$

$$BP = \begin{cases} 1 & \text{if} \quad c > r \\ e^{(1-r/c)} & \text{if} \quad c \leq r \end{cases}$$

where: p_n = modified n-gram precision
w_n = weight for each kind of n-gram
c = candidate length
r = reference length

When bigrams, trigrams, or any n-gram is considered, they are combined in the formula using the w_n. For instance, suppose it is desired to combine unigram and bigram but to weigh more bigrams than unigrams, the weights could be $w_1 = 0.33$ for unigram and $w_2 = 0.67$ for bigram.

2.8.2 METEOR

The Metric of Evaluation of Translation with Explicit Ordering (METEOR) (BANERJEE; LAVIE, 2005) is based on the harmonic mean of unigram precision and recall. This metric was designed to fix some issues observed in the BLEU metric.

The first issue addressed is combining precision and recall by using a harmonic mean where the recall is weighted nine times more than the precision. Both recall and precision are also computed with the same restriction used in BLEU, where each word in both reference and candidate can be used only once.

$$\text{METEOR} = F_{mean} \cdot (1 - P_n)$$

$$F_{mean} = \frac{10PR}{R + 9P}$$

$$p = 0.5 \left(\frac{c}{u_m}\right)^3$$

where: P = modified n-gram precision
R = modified n-gram recall
p = penalty
c = number of chunks
u_m = number of unigram matches

Each chunk is a group of consecutive matches. When more chunks are necessary to map between the reference and candidate, the translation is not following the same order of words, and it is more fragmented.

The second issue is related to the use of similar words. BLEU requires an exact match. For example, words "color", "colour", "colors", or "animal" are equally evaluated as different, even though the last word is a far distant translation than the others. To address this issue, METEOR proposes using both stemming and synonym matching. Stemming is used for reducing the inflection of the words to a stem. With this technique, the words "color" and "colors" can match. Synonym matching is composed of a list of words that should be considered equally. This technique makes it possible to find a match between "color" and "colour". Despite these benefits, both stemming and synonyms are language-dependent. The introduction of these features makes the score less prone to comparison because it is required that any future model use the same frozen stemmer and synonyms list.

2.8.3 Pyramid Method

This method considers that there is no single best summary for a text. Humans may select different parts or use different expressions to represent the significant parts of the text. To address this problem, the pyramid method (NENKOVA; PASSONNEAU, 2004) evaluates the observed distributions of content over a set of human summaries to assign weights according to the frequency of semantic content units (SCUs).

To illustrate the general idea, consider the example presented by Nenkova and Passonneau (2004):

A1 In 1998 two Libyans indicted in 1991 for the Lockerbie bombing were still in Libya.

B1 Two Libyans were indicted in 1991 for blowing up a Pan Am jumbo jet over Lockerbie, Scotland in 1988.

C1 Two Libyans, accused by the United States and Britain of bombing a New York bound Pan Am jet over Lockerbie, Scotland in 1988, killing 270 people, for 10 years were harbored by Libya who claimed the suspects could not get a fair trial in America or Britain.

D2 Two Libyan suspects were indicted in 1991.

Similar sentences like the underlined ones above are identified. Then, the method extracts the following SCUs:

$SCU_1(w = 4)$: two Libyans were officially accused of the Lockerbie bombing

A1 [two Libyans] [indicted]

B1 [Two Libyans were indicted]

C1 [Two Libyans,] [accused]

D2 [Two Libyan suspects were indicted]

The weight four is associated because similar expressions were found in all four references. The associated "description" of the SCU is not used. Another example of SCU that is present only in three references:

$SCU_2(w = 3)$: the indictment of the two Lockerbie suspects was in 1991

A1 [in 1991]

B1 [in 1991]

D2 [in 1991.]

The idea is that each shared expression with a similar meaning should be weighted according to the number of times they appear among the set of reference summaries. Although this method seems robust for summary evaluation, it requires at least a few reference summaries to be applied. This requirement makes it difficult to apply it to real data. It is hard to find a plethora of texts with several distinct reference summaries in the legal domain, which makes it impractical to use this method.

2.8.4 ROUGE

The standard evaluation metric for text summarization is called Recall-Oriented Understudy for Gisting Evaluation (ROUGE) (LIN, 2004). The general idea of this metric

is to count the number of 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.

There are five variants of ROUGE metrics: ROUGE-N, ROUGE-L, ROUGE-W, ROUGE-S and ROUGE-SU. ROUGE-N measures n-gram units between a candidate and a collection of reference summaries. This metric is recall-oriented because the denominator in Equation 2.1 is the sum of n-grams in the reference.

$$\text{ROUGE-N} = \frac{\sum_{S \in RefSums} \sum_{gram_n \in S} count_{match}(gram_n)}{\sum_{S \in RefSums} \sum_{gram_n \in S} count(gram_n)} \tag{2.1}$$

Where n stands for the length of the n-gram, $gram_n$, and $count_{match}(gram_n)$ is the maximum number of n-grams co-occurring in a candidate summary and a set of reference summaries. When multiple references are used, the measure retrieves the one with the maximum score.

ROUGE-L measures the Longest Common Subsequence (LCS). Considering a sentence as a sequence of words, the task is to find a sub-sequence (an ordered subset) match between the candidate and a reference. Even if other words separate them, words occurring in the same order might indicate a good match between candidate and reference summaries.

$$\begin{aligned} R_{lcs} &= \frac{LCS(X,Y)}{m} \\ P_{lcs} &= \frac{LCS(X,Y)}{n} \\ F_{lcs} &= \frac{(1+\beta^2) R_{lcs} P_{lcs}}{R_{lcs} + \beta^2 P_{lcs}} \end{aligned} \tag{2.2}$$

where: X = reference summary of length m
Y = candidate summary of length n
R_{lcs} = recall
P_{lcs} = precision
β = weight of recall over precision
F_{lcs} = ROUGE-L

Equation 2.2 shows how ROUGE-L is calculated. First, the LCS between a reference X and a candidate Y is computed. The ROUGE-L is the F-measure, *i.e.,* the

harmonic mean between recall and precision. One critical distinction between ROUGE-N and ROUGE-L is that the first is recall-based, while the second is F-measure-based. ROUGE-N is not affected by a lengthy candidate. On the other hand, ROUGE-L accounts for both recall and precision and is affected by the length of the candidate.

ROUGE-W also measures the LCS, but it weights in favor of consecutive word matches. For example, a "$reference = [A, B, C, X, X]$" and "$cand_1 = [A, B, C, Y, Y]$" and "$cand_2 = [A, Y, B, Y, C]$", both $cand_1$ and $cand_2$ will have the same ROUGE-L, but it may be desirable to weight in favor of $cand_1$ because the matching subsequence is consecutive.

ROUGE-S measures skip-bigram co-occurrence; this looks for any pair of words in the same order, allowing arbitrary gaps.

ROUGE-SU is an extension of ROUGE-S. Besides counting skip-gram, it also adds unigrams as counting units. With this change, it gives some credit to the sentence that has matching words, but do not share any skip bigram (for instance, a complete reversal "$reference = [A, B, C]$" and "$candidate_1 = [C, B, A]$").

2.8.5 Manual Evaluation

Human preferences for summaries vary in writing style and content depth. Despite being impossible and unnatural for all users to agree about the quality of individual summaries, manual averaged scales for readability over many documents give a closer perspective of the summary quality by the end-user.

One concern for manual evaluation is defining the criteria and instructing the assessors to follow these guidelines. For instance, NIST 2005 (MARNEFFE et al., 2021) described that the manual assessment of readability should be measured for grammaticality, non-redundancy, referential clarity, focus, and structure and coherence. These criteria are too general. They require an adaptation to conform to the requirements of a particular field. In the Brazilian legal system, the guidelines for non-redundancy, focus, structure and coherence needed to be adapted to follow the field practice (GUIMARÃES, 2011).

The advantages of this kind of evaluation are that it can deal with abstractness and evaluate linguistic quality. The disadvantages are the high cost and the low number of summaries that an expert can rate.

2.9 Summary

This chapter introduced the main concepts related to the generation of text summaries. We covered the alternatives to represent text using characters, words, and subword representation. For each alternative, the main benefits and drawbacks were explored. We also discussed the OOV problem and the difficulty of deciding the vocabulary size.

The conventional algorithms for generating extractive and abstractive summaries were presented. The transition from RNNs to the state-of-the-art Transformer architecture was covered. Finally, we covered the evaluation metrics used in sequence-to-sequence tasks. In the next chapter, we will examine the suitability of using BERT for NLP tasks in Portuguese.

3 APPLICABILITY OF BERT-BASED MODELS FOR NLP TASKS IN PORTUGUESE

BERT and ALBERT are pre-trained language models that can be fine-tuned for various Natural Language Understanding tasks. These methods were applied to several such tasks (mainly in English), achieving results that outperform the state-of-the-art.

The authors of both models released trained monolingual models in English and Chinese. However, for the other languages, only BERT released a multilingual model trained on Wikipedias for 104 languages (PIRES; SCHLINGER; GARRETTE, 2019). Portuguese is among the languages that compose the multilingual model. However, it is unclear how the multilingual model performs in single language tasks compared to a model trained exclusively for that language. We believe that a comparison between the mono and multilingual models applied to various single language tasks would help researchers decide whether investing in training a model on a target language or using a multilingual model is enough.

We evaluated our pre-trained models on seven natural language understanding tasks. Our performance was compared to baselines published in the literature and to the multilingual BERT model. The results show that our Portuguese Monolingual models can outperform the baselines in most cases. Concerning the multilingual BERT model, the results were within 5% proportional differences in most tests.

For this experiment, we called our trained models in Portuguese as BertPT and AlbertPT[1]. The compilation of this experiment resulted in a pre-print published on arXiv (FEIJO; MOREIRA, 2020).

3.1 Pre-Training

BERT and ALBERT frameworks are designed to be used in two steps: pre-training and fine-tuning. In the first step, the unsupervised model is pre-trained using a large corpus. At this stage, the model learns the language features using two training objectives. BERT uses Masked Language Model (MLM) and Next Sentence Prediction (NSP), while ALBERT changes the NSP objective to a Sentence Ordering objective (SO). In MLM, at random, 15% of the tokens are replaced by a `[MASK]` token, and the model is supposed to guess which was the best token to be put in its place. NSP requires that, during pre-training, the input is composed of two sentences, and the model should learn if the second

[1] <https://github.com/diego-feijo/bertpt/>

sentence correctly follows the first. ALBERT's authors showed that the NSP objective is straightforward, so they changed it to a SO objective to force the model to learn more in-depth features. The combination of these objectives forces the model to learn many language features.

Preparation. Although pre-training is an unsupervised task (*i.e.,* it just needs a raw document corpus), it requires each training instance to be in a specific format. The documents are split into sentences, and the sentences are tokenized. BERT uses WordPiece (WU et al., 2016), while ALBERT uses SentencePiece (KUDO; RICHARDSON, 2018). We used the vocabulary size of 30,000, following the same size used by both BERT and ALBERT when training their single language English model.

Before building the vocabulary, pre-processing choices need to be made. Convert all words to lowercase, and striping diacritics are usually made to normalize the texts. For example, in a corpus distributed as an HTML source, one could remove TAGs as they would not contribute to building the understanding of the language. Any formatting information is removed to keep just the text content. While numbers, dates, e-mails, and URLs are not usually helpful when building the vocabulary, we opted for keeping them. Portuguese uses diacritics and, while the text can be understood without them, removing them introduces noise as some discriminating features are lost – *e.g.,* the distinction between baby (*bebê*) and s/he drinks (*bebe*) is on the diacritical mark. Diacritics removal for pre-training could allow the model to interpret better informal texts that generally do not use, like tweets or short message services (SMS). Nevertheless, we decided to keep diacritics and the original casing. The goal was to maintain discriminating features that could be useful in some tasks.

Ideally, the pre-trained model should be exposed to texts in the format that will later be fine-tuned. In other words, the same pre-processing steps should be applied both when training and evaluating.

We used 4.8GB of text (992 million tokens) from different kinds of sources. Because each model has a different vocabulary and tokenizer, pre-training data should be generated for either BertPT and AlbertPT.

Corpora. Pre-training is unsupervised, and it requires a large corpus. To allow for the model to encompass different text styles (formal and informal), Brazilian (BP), and European (EP) variants, we used corpora formed using various sources.

- **Wikipedia-PT**[2], with 8M sentences (after pre-processing) in formal writing, casing, and its contents include all regional variations of Portuguese.
- The **Open Subtitles** corpus[3] in BP was used as a source of informal language (as it represents spoken language containing slangs and curses). Sentences are short, frequently having fewer than five words.
- **News articles** from two corpora: (i) the CHAVE corpus[4], which contains full news articles from the Portuguese Público[5] and the Brazilian Folha de São Paulo[6]. This corpus was assembled for the CLEF campaign[7] and contains 210K news articles from 1994 and 1995. Combined, they have a total of 106M tokens; and (ii) a news corpus from Kaggle[8] containing 167K news articles from *Folha de São Paulo* published between January 2015 and September 2017.
- The **EuroParl** corpus extracted from the proceedings of the European Parliament[9]. The Portuguese sub-corpus contains about 75M tokens, and it is available at the Open Parallel Corpus site[10]. Texts are formally written in EP.
- **Research abstracts** from 23K MSc theses and Ph.D. dissertations from several areas. The sources were taken from the Brazilian website *Domínio Publico* [11].

Parameters. During pre-training, we used whole word masking to avoid that a token being masked in the middle. The default base configurations were used – 12 layers, 768 hidden size, and 12 heads of attention. With this configuration, the model used for BertPT has a total of 110M parameters. As for AlbertPT, there are only 12M parameters. This configuration was the same as the authors used for pre-training their base models, including the multilingual version.

Following the recommended pre-training procedures, BertPT was trained for 1M steps and AlbertPT, 175K steps. Due to the high number of parameters, BERT takes longer than ALBERT, and training with longer sequences becomes quite expensive. As the complexity is quadratic to the length of the sequence, our models were pre-trained with sequences of lengths 128 and 512. Their training took 33 and 17 hours, respectively,

[2]<https://dumps.wikimedia.org/backup-index.html>
[3]<http://opus.nlpl.eu/OpenSubtitles-v2016.php>
[4]<https://www.linguateca.pt/CHAVE/>
[5]<http://www.publico.pt/>
[6]<https://www.folha.uol.com.br/>
[7]<http://www.clef-campaign.org/>
[8]<https://www.kaggle.com/marlesson/news-of-the-site-folhauol>
[9]<https://www.europarl.europa.eu/>
[10]<http://opus.nlpl.eu/>
[11]<http://www.dominiopublico.gov.br/>

on one cloud TPU v2.

3.2 Evaluation

In this section, we report on experiments that fine-tune the pre-trained modes BertPT and AlbertPT. The goal was to compare our trained monolingual models to the provided multilingual and the current state-of-the-art when applied to several different Natural Language Understanding tasks. To focus on the capacity of the models in extracting features from texts, our models have simple architectures. They are composed of the standard base model. The pooled output from the last layer is used as input for a classifier or regression layer. The new parameters introduced are just the weights from this last layer. This pooled output is a vector that should capture the desired features to solve the task. We avoided using more elaborate features to keep the model as simple as possible since the goal was to check if our Portuguese training had been successful. Fine-tuning to each specific task took around five minutes of training using one cloud TPU v2.

We report on our experiments on the following tasks: (i) semantic textual similarity, (ii) recognizing textual entailment, (iii) identifying offensive texts, (iv) detecting fake news, (v) categorizing news, (vi) classifying sentiment polarity, and (vii) identifying emotions.

In all tasks, BertPT and AlbertPT were compared to the Multilingual BERT model released by Google Research and published results for each dataset. Each model has its tokenizer – BERT uses WordPiece, and ALBERT uses SentencePiece. Also, BERT multilingual was trained on a different corpus. Because of that, the models have different vocabularies.

To keep the results comparable, in all experiments, we used the batch size of 32; the maximum sequence length is the minimum value between 512 and the lengthiest input in the training set; the learning rate for BERT models was 2×10^{-5}, and for ALBERT it was 4×10^{-5}. It is worth pointing out that it is possible to achieve better results if the parameters (such as sequence length, learning rate, batch size, and the option for case folding) were tuned for each specific task.

3.2.1 Recognizing Textual Entailment (RTE)

The ASSIN dataset is used for semantic textual similarity and recognizing textual entailment. It was first used in a shared task in the PROPOR 2016 Conference[12]. It has 10K pairs of sentences, equally split between Brazilian (BP) and European Portuguese (EP). Each language variant is composed of three files: training (2,5K pairs), development (500 pairs), and testing (2K pairs). The sentences were relatively short. As we said before, each tokenizer split tokens differently. The maximum length in this dataset was 78 tokens.

This three-class classification problem requires assigning a label (None, Entailment, or Paraphrase) to the given pair of sentences. These classes are highly imbalanced, with the *None* class having three times more examples than *Entailment* and nine times more instances than *Paraphrase*. The evaluation is done using Accuracy and Macro F1. Macro F1 evaluates the F-measure for each class independently and takes the average. The models were trained using both training and development sets. Ten-fold stratified cross-validation was applied over the training data. Table 3.1 shows results using just EP, just BP, and a concatenation of both EP+BP training data. There were no published baselines for the evaluation of only BP.

Table 3.1 – Recognizing Textual Entailment Results

Data	Model	Acc	F1-M
EP	Baseline (ROCHA; CARDOSO, 2018)	0.83	0.73
	BertPT	0.86	0.76
	AlbertPT	0.87	**0.80**
	Multilingual	**0.88**	0.79
BP	BertPT	0.85	0.52
	AlbertPT	0.85	0.51
	Multilingual	**0.86**	**0.57**
EP+BP	Baseline (ROCHA; CARDOSO, 2018)	0.82	0.70
	BertPT	0.87	0.75
	AlbertPT	0.88	0.79
	Multilingual	**0.90**	**0.80**

Note: Evaluated using 10-fold cross-validation.

For this task, all models were superior to previously reported baselines. In this dataset, sentences are well-written, contain punctuation and diacritics. The models were not harmed for having a few missing tokens in the vocabulary so that they could build excellent representations of the sentences. Also, these short sentences (up to 78 tokens)

[12]<http://propor2016.di.fc.ul.pt/>

that do not mix different subjects allow the models to generate more specific representations for the output classifier. Longer sentences would require representing all subjects covered. Consequently, the vector representation of the sentences would be more diluted.

We applied our fine-tuned model to the entire training set and ran the evaluation over the test set. The results are on Table 3.2. The first observation is that the BP setting is difficult for all models. The EP+BP** in the last rows indicates that the model was trained using EP and BP but evaluated only using the EP test set. The results of this evaluation can be compared to the first rows, in which the models were trained and evaluated only on the EP test set. Although it makes sense to think that more training data would help the model generalize, we reached the same conclusion as (FIALHO et al., 2016) that the improvement in this dataset, if any, is meaningless. We could not find a published baseline for the case where the model was trained using EP+BP and evaluated using both test sets. Again, as expected, BERT models were superior to previous baselines for all combinations. Analyzing the 10-fold evaluation and using only the test set, we conclude that BERT-based models were superior to previous baselines.

3.2.2 Semantic Textual Similarity (STS)

The goal of Semantic Textual Similarity is to quantify the degree of similarity between two sentences. In the ASSIN dataset, the similarity ranges from 1 (no resemblance) to 5 (high similarity). This is a regression problem as the output is a similarity score. The evaluation measures how far the predicted score is from the reference using two metrics: (i) the Pearson Correlation, which measures the correlation with the reference, so the higher, the better, and (ii) and the Mean Squared Error, which measures the square of the difference between the prediction and the reference, so the lower, the better. Following the same procedure of Textual Entailment, we used 3,000 pairs for training. The results are in Table 3.2. Since STS is related to RTE, again, all models obtained better results than the baselines. Also, the multilingual model was again superior to our pre-trained Portuguese models.

Table 3.2 – Recognizing Entailment and Semantic Similarity Results

Data	Model	RTE		STS	
		Acc	F1-M	Pearson	MSE
EP	Baseline (FIALHO et al., 2016)	0.84	0.73	0.74	0.60
	BertPT	0.84	0.69	0.79	0.54
	AlbertPT	0.88	0.78	0.80	0.47
	Multilingual	**0.89**	**0.81**	**0.84**	**0.43**
BP	Baseline (FIALHO et al., 2016)	0.86	0.64	0.73	0.36
	BertPT	0.86	0.53	0.76	0.32
	AlbertPT	0.87	**0.65**	0.79	0.30
	Multilingual	**0.88**	0.55	**0.81**	**0.28**
EP+BP	BertPT	0.87	0.72	0.79	0.39
	AlbertPT	0.89	0.76	0.78	0.39
	Multilingual	**0.90**	**0.82**	**0.83**	**0.33**
EP+BP**	Baseline (FIALHO et al., 2016)	0.83	0.72	-	-
	BertPT	0.86	0.66	-	-
	AlbertPT	0.88	0.79	-	-
	Multilingual	**0.91**	**0.83**	-	-

Note: EP = European Portuguese; BP = Brazilian Portuguese. Evaluated on test sets.

3.2.3 Offensive Comment Identification

OffComBR-2 and OffComBR-3 (PELLE; MOREIRA, 2017) are variations of a dataset containing comments that readers posted about published news in a Brazilian news portal. The comments are annotated with *yes* or *no*, meaning whether the comment was considered offensive by human judges. In the OffComBR-2 variation, at least two judges found the comment offensive. In the OffComBR-3 variation, all three judges agreed that the comment was offensive. Intuitively, OffComBR-3 should be less prone to judge bias and thus be more stable to classify. This dataset imposes a challenge because the comments are written very informally. They contain slangs, profanities, emoticons, and abbreviations. There is no punctuation, no diacritics, and the casing is almost random.

Maybe, in this task, the model that uses a vocabulary trained using diacritics and formal writing might suffer from having many unknown tokens and may be unable to build a good sentence representation. In this case, a model pre-processed to remove diacritics and case folding may be more successful because the comments and the vocabulary will be more similar. The length of the comments varies significantly. In some cases, the offensive part is just one or two tokens, and it may not be identified as a vocabulary word. Also, authors of the offensive comments typically try to obfuscate profane language (*e.g.,*

by inserting spaces or other symbols among the characters of the offensive word) to make the task of identifying these comments more difficult. The WordPiece and SentencePiece tokenizers can represent these misspelled tokens using a sequence of single characters, which is advantageous over traditional vocabulary-based representations. While a sequence of letters may not be enough for the model to build a good feature representation, in some cases, this may suffice.

The results in Table 3.3 indicate that this task is challenging for BERT models. Deep learning models were expected to have a significant advantage over the baseline, using only shallow methods such as SVM and Naïve Bayes over unigrams. We believe that the models could not achieve a significant advantage because, during pre-training, they were never exposed to offensive language.

Table 3.3 – Offensive Comment Identification Results

Data	Model	Acc	F1-W
BR-2	Baseline (PELLE; MOREIRA, 2017)	-	**0.77**
	BertPT	**0.78**	**0.77**
	AlbertPT	0.76	0.76
	Multilingual	0.76	0.74
BR-3	Baseline (PELLE; MOREIRA, 2017)	-	0.82
	BertPT	**0.84**	**0.83**
	AlbertPT	**0.84**	0.81
	Multilingual	0.83	0.81

Note: Evaluated using 10-fold cross-validation.

3.2.4 Fake News Detection

The Fake.Br Corpus[13] (MONTEIRO et al., 2018) contains real examples of fake news written in Brazilian Portuguese. Each instance was manually fact-checked and is annotated, indicating whether it is fake or accurate. There are 7,200 instances evenly split between the classes. All sentences are well-written, use punctuation and diacritics according to the grammar.

We ran the evaluation using 5-fold cross-validation over the training data. The results are in Table 3.4. The baseline reported achieving up to 0.89 in F-measure when combining several classification models. BERT models have done an impressive job in this dataset – they outperformed the baseline and achieved almost perfect results. Since

[13]<https://github.com/roneysco/Fake.br-Corpus>

the results of BertPT and AlbertPT were so high, we experimented using a simple network with just one fully connected layer. With such a small architecture, the model reached 89% of accuracy, but any further improvements require the model to capture some more elaborated features. BERT models can capture such sophisticated features. Three reasons may explain such good results. The sentences were well-written, so the model was able to build a good context representation. Despite the maximum length of the texts being more than 512, most of the samples have similar lengths. We believe that this similarity allowed the model to detect any variations that indicate that the news was genuine or fake. Finally, we attribute the excellent performance mainly to the fact that this dataset is reasonably large compared to the others evaluated here. The larger volume of training data allowed the model to learn enough features to distinguish fake from real.

Table 3.4 – Fake News Detection and Sentiment Polarity Results

Model	Fake		Sentiment	
	Acc	F1-W	Acc	F1-W
Baseline (MONTEIRO et al., 2018)	0.89	0.89	-	-
Baseline (ARAUJO et al., 2016)	-	-	-	0.71
BertPT	**0.98**	**0.98**	0.77	0.76
AlbertPT	**0.98**	**0.98**	**0.79**	**0.78**
Multilingual	**0.98**	**0.98**	0.71	0.70

Note: Fake News has two classes; Sentiment Polarity, three classes. Both were evaluated using five-fold cross-validation.

3.2.5 Sentiment Polarity Classification on Tweets

In (ARAUJO et al., 2016) the performance of several models for sentiment polarity classification was evaluated. The authors reported the results from several methods using Macro F1 and Coverage. Thus, a classifier that outputs a high macro F1 with low coverage is not helpful because the most challenging examples were not classified. Most methods evaluated by them have low coverage indicating that they could not produce valid responses in the most challenging cases. The best reported macro F1 with more than 90% of messages classified was 0.71. *Coverage* was defined by the author as "the fraction of messages that a method can classify as either positive or negative in a given dataset". We use this score as the baseline for Portuguese, although the authors did not report how it was obtained.

There are three possible sentiments: negative, neutral, or positive. The writing

style is very informal, and many out of vocabulary were used. Expressions like “goooood daaaay”, “loool”, “kkk”, “=D” are frequent. So, if the model only relies on known words, this task may be difficult. In this situation, the fact that the model uses subword units to tokenize the text may help at least try to construct an interpretation using the sequence of units. Also, some examples expose the sentiment through emoticons. So, correctly interpreting these symbols can lead to the associated sentiment. The corpus has 774 instances: 297 positive, 213 negative, and 264 neutral. The mean length of the sentences is 23 tokens, with a standard deviation of 10. The original address[14] where this dataset (NARR; HULFENHAUS; ALBAYRAK, 2012) should be available seems to be down. We used the version assembled by (ARAUJO et al., 2016) and make it available[15].

To keep the results comparable, we use only the positive and negative samples. For the results shown in Table 3.4, we did a 5-fold cross-validation over all data. The facts that this corpus has several emoticons and out-of-vocabulary expressions make it hard for the models that were not trained using similar vocabulary.

3.2.6 News Category Classification

The Folha UOL News Dataset[16] contains 167,053 news from *Folha de São Paulo*[17], a Brazilian newspaper. It contains the headlines, complete articles, and categories (opinion, daily life, sports, culture, markets, world, and politics).

There is a public ranking of this classification task on Kaggle[18]. So far the best result used 15% of the data as a validation set and achieved 87.39% of validation accuracy. As the ranking leader in Kaggle, we removed instances that do not have a category nor text associated. A total of 134,055 news remained. The instances are well distributed across classes varying from 15,617 (in the least frequent) to 22,022 (in the majority class). The texts are well-written with a mean length of 441 tokens with a standard deviation of 289, indicating a wide variation. To keep results as comparable as possible, we followed the ranking leader and randomly split 15% to be used as our test set. As this could be an arbitrary split, we also ran the evaluation using stratified 5-fold cross-validation. The results are shown in Table 3.5. All models behaved almost identically, with a slight advan-

[14]<http://www.dai-lab.de/~narr/sentimentdataset>
[15]Omitted for anonymity
[16]<https://www.kaggle.com/marlesson/news-of-the-site-folhauol>
[17]<https://www.folha.uol.com.br/>
[18]<https://www.kaggle.com/marlesson/news-of-the-site-folhauol>

tage for the multilingual model. We noticed that the result from randomly splitting 15% and 5-fold cross-validation was almost identical. We believe that this happened because of the large number of instances and the reasonably balanced category distribution.

The other news category classification experiment used Público Dataset. Público[19] is a Portuguese Newspaper. The dataset containing its news is distributed as part of CHAVE corpus[20]. Each news text belongs to one out of nine categories: Science, Culture, Sports, General, Economy, Local, World, National, and Society. The texts date back to 1994 and 1995. However, since we wanted to compare against a published baseline, our evaluation used just the news from October 1995. The task is a multiclass classification with a single label. For this evaluation, we used 5-fold cross-validation. The best-reported baseline achieved 0.84 in F-measure when combining several classification models.

The news texts are well written, contain punctuation and diacritics. The major problem here is length diversity – while the mean length is 728 tokens, the standard deviation is 628. Both BERT and ALBERT models were not able to handle such long texts due to memory limitations. We imposed a limit of using only the first 512 tokens of the text.

Table 3.5 has the results for this task. Without any task-specific architecture, the performance in this dataset was almost identical to the current state-of-the-art in this task.

Table 3.5 – Folha Uol and Público News Results

Model	Folha UOL		Público	
	Acc	F1-W	Acc	F1-M
Kaggle Baseline	0.87	-	-	-
BertPT	0.94	**0.94**	-	-
AlbertPT	0.93	0.93	-	-
Multilingual	**0.94**	**0.94**	-	-
Baseline (GONÇALVES; QUARESMA, 2008)	-	**0.84**	**0.84**	
BertPT	**0.94**	**0.94**	**0.84**	**0.84**
AlbertPT	0.93	0.93	0.82	0.82
Multilingual	**0.94**	**0.94**	**0.84**	**0.84**

Note: Folha Uol has seven classes. We use 15% split for evaluation. Público News has nine classes. It was evaluated using five-fold cross-validation.

[19] <https://www.publico.pt/>
[20] <https://www.linguateca.pt/CHAVE/>

3.2.7 Emotion Classification

The BRNews Dataset (MARTINAZZO; DOSCIATTI; PARAISO, 2011) contains news extracted from major Brazilian newspapers. Each of the 1,002 instances was manually annotated within six possible emotions: joy, surprise, anger, disgust, fear, and sadness.

The texts from the news are short and well-written. They use punctuation and diacritics. The length of tokens varies from 21 to 62 tokens, which allows the model to capture most of the meaning of the sentences. Class distribution is imbalanced with 364, 266, 251, 58, 32, and 31, respectively. The most represented class has up to ten times the frequency of the two least represented. So, the advantage of using the classifier version that favors the least represented classes is expected.

We ran two kinds of experiments. In the first, we employed a standard multiclass classifier. Unfortunately, we do not have any baseline results, so we report the results for the BERT-based models. In the second experimental setting, we followed the same methodology as (BECKER; MOREIRA; SANTOS, 2017) and transformed the problem into six binary classifiers (one for each sentiment). For each one of these binary models, the accuracy and weighted F1 score were calculated. We take the mean result.

Table 3.6 shows the results for 10-fold cross-validation. The baseline for the binary version reaches up to 0.84 of F-measure. In this experiment, BertPT has a result similar to the baseline, but it was not able to overcome it.

Table 3.6 – Emotion Classification Results

Model	Six classes		Binary	
	Acc	F1-W	Acc	F1-W
Baseline (BECKER; MOREIRA; SANTOS, 2017)	-	-	-	**0.84**
BertPT	**0.51**	**0.47**	**0.84**	0.83
AlbertPT	0.41	0.28	**0.84**	0.81
Multilingual	0.49	0.46	**0.84**	0.80

Note: Evaluated using ten-fold cross-validation.

The length and writing style should be ideal for the models to classify most instances correctly. Inspecting some misclassifications, we see some unusual annotations. For example, “President Lula will be in Sergipe this Friday for inauguration of works: Lula will visit three cities. The ally Governor will be joining him.”. This instance was annotated as “fear”, but BertPT predicted “joy”. In another similar example, “President Lula flies over a flooded area in Piauí: Lula will speak to mayors from the hardest hit

regions. In the afternoon, he will be in Maranhão, another affected state.", the instance was annotated as "joy", which probably indicates why the model predicted joy for the previous example. So, we think that the fact that the models could not beat the established baseline is more due to dataset characteristics than the ability of the model to interpret its meaning.

3.3 Discussion

The different tasks and settings under which BertPT and AlbertPT were compared to Multilingual BERT and published baselines yield 38 possible pairwise comparisons. We analyzed how many times the performance of each model was better than, worse than, or equivalent to the performance of the others. If the proportional difference in scores was below 5%, then the two models were considered equivalent. These results are shown in Table 3.7. We can see that no single method is always better than the others. The baselines have fewer wins, which means that BERT-based models tend to yield better results. BertPT and AlbertPT achieved equivalent performances most of the time (28 out of 38). AlbertPT has an advantage (six wins versus four losses), and since its training is less expensive, then it would be the model of choice.

Overall, Multilingual BERT has more wins in comparison to other alternatives. Although the differences lie most of the time within the 5% range, it achieved better results in more tasks. We attribute this superiority to the use of more languages. It is possible that the presence of languages similar to Portuguese (such as Spanish, which has twice the volume of texts of Portuguese) could produce richer semantic representations.

Table 3.7 – Result Comparison

Model	BertPT			AlbertPT			Multilingual		
	>	=	<	>	=	<	>	=	<
Baseline	3	12	8	1	8	14	1	13	9
BertPT				4	28	6	2	24	12
AlbertPT							1	30	7

Note: Difference smaller than 5% were disregarded.

The limitation on the size of the input (which could have a higher impact on the news categorization task) ended up proving not to be an issue – classification accuracy was very high in both datasets. Classification errors happened in cases in which the distinction of the categories was quite fuzzy (*e.g.,* daily life and culture).

Multilingual BERT was not trained on informal texts, which were abundant in two of our tasks (offensive comment detection and sentiment polarity classification). As a result, the results of BertPT and AlbertPT were slightly superior in these tasks, showing that having more text styles on the training set is beneficial.

3.4 Summary

This experiment used BERT and ALBERT for seven different NLP tasks in Portuguese, and it obtained state-of-the-art results. These methods showed their suitability for *textual entailment* that might help the summarization model to avoid summaries that could not be entailed from the source. In the next chapter, we will focus on the approaches used when generating legal summaries.

4 RELATED WORK

The goal of this chapter is to present a survey of the existing work on legal text summarization. For a comprehensive review of general text summarization, the reader should refer to Nenkova and McKeown (2012), Kanapala, Pal and Pamula (2019).

Creating a summary requires representing the knowledge that is inside a textual document. As seen in Chapter 2, extractive techniques rely on the definition of a score function to determine the meaningful chunks of input text (usually complete sentences and even whole paragraphs). Abstractive techniques require the construction of a language model capable of generating the summarized text.

Legal text summarization is a topic that attracts the attention of the industry [1]. Many approaches have been tested over the years. A classification of these approaches helps give a big picture of which paths had already been explored and what results they obtained. Nevertheless, existing approaches often do not fit into a strict classification scheme as they may fit into different categories depending on which feature is used to classify them. The goal of our proposed classification is to understand and organize the main characteristic of each work.

4.1 Keyword-Based Approaches

Keyword-based approaches use predefined keywords and expressions to find relevant contexts. Often, these patterns are predefined, and the model just adjusts the weights for each expression.

Flexicon (GELBART; SMITH, 1991) stands for Fast Legal Expert Information Consultant. It is a retrieval system proposed to overcome standard Boolean searches applied in the legal domain. One of the features offered by this system is that it builds indexing information. Also, it generates headnotes, which is a simple helpful summary for quickly identifying whether this case is relevant for the user. The headnote is generated upon the construction of document profiles over the raw text. The authors used keywords to identify concepts, case citations, legislation, and facts. Paragraphs are weighted, reflecting their significance using the frequency of features and its inverse frequency in the catalog (similar to TFIDF). A filtering mechanism eliminates unimportant "quotations" or very short paragraphs. Also, their weighting mechanism uses dictionaries of key

[1] <https://www.casefleet.com/use-cases/summary-judgment-software>

phrases, position, fact terms, citation patterns, and length to determine the most relevant paragraphs. Finally, the relevant paragraphs are extracted and concatenated.

Salomon (MOENS; UYTTENDAELE, 1997) stands for Summary and Analysis of Legal texts for Managing On-line Needs. The system extracts relevant information from raw text and composes a summary. The goal is to generate summaries from Belgian criminal cases. This system tries to create a representation of the knowledge of the case. This knowledge is made up by defining seven categories concerning general and special cases. Salomon uses a parser with patterns to associated paragraphs to those categories. A tree representing the knowledge of the case is defined using rules also extracted using the parser. Irrelevant paragraphs are tagged as such. From each category, the relevant paragraphs are selected to compose the summary.

4.2 Rule-Based approaches

Legal knowledge may be represented using manually created ontologies or inference rules. However, it is not an easy task. If the model is too abstract, it may not capture all the rules necessary to build a helpful summary. On the other hand, it may be tough to manage if the model has too many facts.

Galgani, Compton and Hoffmann (2012) tried to overcome these difficulties using Ripple Down Rules (RDR) (COMPTON; JANSEN, 1990) to create incremental Knowledge Acquisition (KA). These rules require the definition of a set of attributes over which the rules are applied. To increment the set of features available for the rules, they also used Lawcite[2] service to look for a citation and its respective catchphrases. The features chosen range from the position where a citation was found, word statistics, and TFIDF. The main advantage of this technique is that it does not require previously labeled data. However, it requires the expert to manually identify exceptions to the rules for considering each case at hand. The new rules/exceptions defined by the expert are added to the system and checked for consistency with all cases previously correctly classified. Finally, the set of features is used to build and refine rules that determine the sentences to compose the summary. The dataset AustLII (Australasian Legal Information Institute) was used for evaluation.

[2]<http://www.lawcite.org>

4.3 Cluster-Based approaches

Cluster-based approaches try to group sentences by similar topics and summarize by looking for the most representative sentence from each group. A summary should present cover several topics that were addressed by the decision. Intuitively, the task can be seen as finding these topics and gathering the most representative sentences for each topic. Several difficulties arise from this approach, including deciding the number of topics, and encoding a representation of sentences and topics that allow a measure of similarity between them.

Pandya (2019) tried this extractive approach. Sentences were represented as TFIDF vectors, and similar sentences were clustered together using the k-means algorithm. The method was evaluated on the AustLII dataset, *i.e.,* the same used by Galgani, Compton and Hoffmann (2012). However, the evaluation was on only one (large) specific case. From the 200 pages in the original text, the summary was 20 pages long (around 150 sentences).

More recently, Zhong et al. (2019) presented an extractive summarizer using a CNN classifier based on Maximum Marginal Relevance (MMR) (CARBONELL; GOLDSTEIN, 1998) algorithm. The application was developed over 35,000 cases from the Corpus Board of Veterans' Appeals concerning a single issue of Post-traumatic Stress Disorder. The summaries are composed of a single sentence for each issue subject, origin court, veteran service location, and the board's decision. Moreover, two to six sentences that best summarize the reasons and evidence are added to the summary. Their work included a module to classify sentences according to their function in classes "Reasoning/Evidential Support" or "Others", following previous work that has explored rhetorical roles of sentences (GROVER; HACHEY; KORYCINSKI, 2003; GROVER et al., 2003; YOUSFI-MONOD; FARZINDAR; LAPALME, 2010) to create an extractive summary. The model was trained encoding sentences using Universal Sentence Encoder (CER et al., 2018). Encoded representations are fed to a CNN model to predict the outcome already extracted using regular expressions. An exciting aspect of their framework was the use of integrated gradients (SUNDARARAJAN; TALY; YAN, 2017; MUDRAKARTA et al., 2018) to determine which sentences are more predictive for the outcome. Proper sentences are partitioned using a type classifier and selecting a set of summary sentences using MMR. The results of the experiments showed that extracting sentences based on the outcome of the decision was not enough to produce good ROUGE results. Their iterative

train-attribute-mask pipeline could not capture all the relevant aspects judged by human evaluation.

4.4 Probabilistic-Based approaches

Probabilistic approaches follow the perception that features in the text can be used to decide which sentences should be selected to compose the summary. LetSum (FARZINDAR; LAPALME, 2004) is a probabilistic approach for grouping sentences by their themes. It splits the source decision into four themes: Introduction, Context, Juridical Analysis, and Conclusion. Later, it extracts relevant sentences for each theme. LetSum uses a corpus of 3,500 judgments from the Federal Court of Canada. The preprocessing phase consists of splitting text into paragraphs, sentences, tokens and applying POS tagging.

It uses regular expressions to determine the thematic segmentation into: Introduction, which describes the prior situation (who did what to whom); Context, chronological order of the facts or description; Juridical Analysis, comments from the judges, and application of the law; and Conclusion, that expresses the final disposition. The next phase consists of filtering less essential units, such as citations of law articles. It selects relevant textual units using heuristic functions related to the position of the paragraphs in the document, the position of the paragraphs in the thematic unit, and TFIDF. Finally, the summary is generated within the size limit of the abstract. The evaluation used a source decision of 3,500 words and generated a summary with 10% of its length.

Further work from the same group that created LetSum proposed Decision Express (YOUSFI-MONOD; FARZINDAR; LAPALME, 2010). This approach was an attempt to apply supervised learning using a Naive Bayes classifier. The authors defend this extractive approach arguing that a reformulation done by an abstractive approach would be less credible because it is not the direct citation of the decision. The process consists of extracting Introduction, Context, Reasoning, and Conclusion using a parser. From the summaries, the system tries to identify source sentences using Levenshtein edit distance. A Naive Bayes classifier is trained with the following classes: Not in Summary, Introduction, Context, Reasoning, and Conclusion. The features are crafted by hand using a set of surface, emphasis, and content features (for example, such as previously defined legal keywords and positions within paragraph or section).

CaseSummarizer (POLSLEY; JHUNJHUNWALA; HUANG, 2016) is a multi-

technique approach. It follows the standard pre-processing: stemming, lemmatization, case-normalization, and stopword removal. The sentences were scored using TFIDF normalized by the sentence length. They also used a POS tagger and tried to identify named entities. One of the objectives of this work is to avoid using a manually defined list of keywords or catchphrases but instead using weights for entities, dates, and proximity to section headings.

Anand and Wagh (2019) recently tried an extractive approach over Indian Supreme Court judgments from 1947 to 1993. They use the headnotes as summaries for the cases. The cases are split into sentences, and for each sentence, a binary classifier decides if it is important. The system builds several features using similarity based on the overlap of significant words, TFIDF, ROUGE-L, and InferSent (CONNEAU et al., 2017) Sentence Embeddings. The proposed architecture is relatively simple. It uses a CNN with pooling for each time step before feeding it to an LSTM. The desired summary length was fixed in 10% of the case length.

4.5 Comparison of Existing Approaches

Table 4.1 shows the main features from each one of the selected works. The reported results are not comparable to each other because they are based on entirely different data. Also, each work has adopted a different methodology for testing. Still, the results are presented here for reference.

Even though each country and dataset has distinguishing features, examining the related work, it is possible to notice that manually crafted features such as keywords, inference rules, and ontologies give space to machine learning techniques in which the features are learned directly from data. Also, it can be seen that all proposed techniques still rely on an extractive approach, which limits the range of possible summaries that can be generated. Despite many of these works point out that ROUGE evaluation has some drawbacks when evaluating the quality of a generated summary, this metric is still broadly used in the absence of a better one.

Table 4.1 – Summary of Existing Legal Summarization Approaches

	Author	Name	Main approach	Evaluation Dataset	ROUGE		
					R-1	R-2	R-L
Keyword	Gelbart and Smith (1991)	Flexicon	Weighted features like keywords, frequency, and position of words in the sentences and the collection	-	-	-	-
	Moens and Uyttendaele (1997)	Salomon	Uses catchphrases to build a knowledge representation for seven categories	Belgian criminal cases	-	-	-
Rule	Galgani, Compton and Hoffmann (2012)	-	Knowledge representation using RDR, weighted features like citations, positions of the sentence, TFIDF	AustLII	0.36	-	-
Cluster	Pandya (2019)	-	TFIDF Vectorizer with K-Means to group similar sentences	AustLII	0.28	0.06	0.34
	Zhong et al. (2019)	-	CNN classifier using MMR	Corpus of Board of Veterans' Appeals	0.27	0.10	-
Probabilistic	Farzindar and Lapalme (2004)	LetSum	Position within thematic group and TFIDF	Federal Court of Canada	0.57	0.31	0.45
	Yousfi-Monod, Farzindar and Lapalme (2010)	Decision Express	Naive Bayes classifier to decide Not In Summary, Introduction, Context, Reasoning or Conclusion	Federal Court of Canada	-	-	-
	Polsley, Jhunjhunwala and Huang (2016)	Case Summarizer	Weighted features like TFIDF, POS tagging, parsed Dates	Federal Court of Australia	0.19	0.11	0.06
	Anand and Wagh (2019)	-	InferSent for embeddings, CNN with pooling and LSTM	Indian Supreme Court	0.44	0.25	0.38

5 RULINGBR: A DATASET FOR LEGAL RULINGS

Each country has its set of rules and statutes that define the Law. The Brazilian Judicial System uses precedents but is more tied to written statutes. These rules define the method and the shape of the decisions produced by the judges. Roughly speaking, there are two families of judicial systems: Common Law and Civil Law. The difference between these two is in its primary source of Law to define the outcome of the present disputes: Common Law uses previously decided cases; Civil uses written statutes. As a general overview, Common Law is followed by Great Britain, The United States, Ireland, India, Australia. Continental Law is followed by France, Italy, Germany, and Brazil.

With these differences, the summaries from Court decisions also focus on different aspects. As the focus of this research is to summarize Brazilian decisions, it was necessary to build a dataset to enable running experiments with real data. The creation of this dataset and preliminary experiments were compiled in a paper published and presented at PROPOR 2018 (FEIJO; MOREIRA, 2018).

For text summarization in the legal domain, we searched for a source with many publicly available documents. Thus, we chose to use the *Supremo Tribunal Federal* (STF) as our source. The STF is the highest court in Brazil and has the final word interpreting the country's Federal Constitution. All of its decisions must be published online and are available in its internet portal[1].

5.1 Structure of the Documents

Each ruling is split into four sections: *summary*, *report*, *vote*, and *judgment*.

- The **summary** (*Ementa*) contains the main topics discussed in each case and how the judges decided. This is the *reference summary* that automatic methods should aim to produce. The summary has an average length of 191 tokens, ranging from 25 to a few hundred tokens.
- The **report** (*Relatório*) is a compilation of the main arguments and events that happened during the trial. Its length can vary broadly, from few dozens up to a few thousand tokens. In general, this section accounts for about 22% of the full content.
- The **vote** (*Voto*) may contain one vote, in case that the other judges agree with the

[1] <http://www.stf.jus.br/>

first judge, or individual votes for each judge, otherwise. The length of this section can also vary even more than the report. Because the votes need to address all the points raised by the petitioners, this tends to be the most extensive section covering around 69% of the full content.

- The **judgment** (*Acórdão*) Finally, the judgment section is, in general, short and compiles the outcome as granted or denied. This section represents around 2% of the full content.

The text of the rulings and their corresponding summaries have an average length of 2,661 and 191 tokens, respectively. The compression ratio (average number of tokens in the summary divided by the average number of tokens in the ruling) is 7.2%.

The conventional format for the summary section is to have a header that presents representative keywords, terms, and expressions designating the fundamental elements in the case; and a second part where the extract of the conclusion of the decision is given (GUIMARÃES, 2011). The topics covered by the summary are spread among the report, vote, and judgment. In general, none of these three sections by themselves are enough to summarize the ruling correctly. The summary section will be used as our *ground-truth summary*, and the combination of all other sections will compose the *source text* to be summarized.

5.2 Data Collection

To obtain the documents, the Scrapy (SCRAPINGHUB, 2018) library was used to browse the search pages and to download the documents. Only a few rulings from the years 2010 and 2011 could be successfully parsed. Thus most decisions are dated from 2012 to 2018.

The raw text we obtained contains some undesired pieces of texts such as headings, footers, page numbers, *etc.*. We used regular expressions to identify the starting and ending points of each section of interest and remove unwanted text. Finally, the text of the sections was dumped as in JavaScript Object Notation (JSON), one object per line.

In Figure 5.1, we show an extract from a short document using JSON. The ellipsis indicates the omission of content to save space.

It comprises around 10K court rulings written in Portuguese, amounting to about 176MB of data. The rulings cover abstract constitutional control, administrative, civil,

Figure 5.1 – Example of a Ruling in JavaScript Object Notation

{**“ementa"**: *“Embargos de declaração em recurso extraordinário com agravo. 2. Decisão monocrática. (...) 5. Agravo regimental a que se nega provimento.",*
“acordao": *“Vistos, relatados e discutidos estes autos, acordam os ministros do Supremo Tribunal Federal, em Segunda Turma, (...), por unanimidade, converter os embargos de declaração em agravo regimental e, a este, negar provimento, nos termos do voto do Relator.",*
“relatorio": *“(...) Trata-se de embargos de declaração opostos contra decisão que negou provimento a recurso, ao fundamento de que a natureza da matéria versada nos autos reveste-se de índole infraconstitucional. Aponta-se violação direta à Constituição Federal, em especial, aos artigos (...).",*
“voto": *“(...) Tendo em vista o princípio da economia processual, recebo os embargos de declaração como agravo regimental e, desde logo, passo a apreciá-lo. (...)"*}

Source: The Author

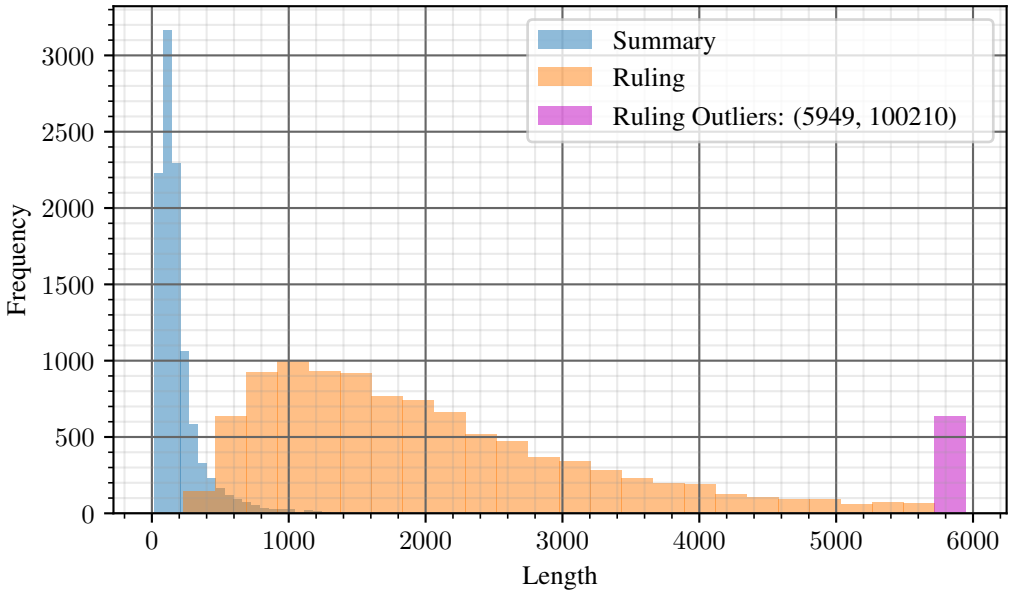
civil procedural, constitutional, consumer, criminal, criminal procedural, economic, electoral, environmental, financial, fundamental rights, labor, notarial, public international, social security, request of extradition, tax, and urban Law. There are cases from 18 judges. The final file can be downloaded from <https://github.com/diego-feijo/rulingbr/> There are around 26 million tokens in the entire dataset.

5.3 Length of Documents

We investigated whether there is a correlation between the summary length and all other sections combined (full document). This correlation would be vital for us to determine the desired summary size when using automatic summarizers. The calculated correlation coefficient was 0.39, which is considered weak and is reflected by a large dispersion.

The documents in the corpus significantly vary in length due to the several subjects covered by the decisions. To try to generalize a pattern, some outliers needed to be dropped. Using a token (word) as a measuring unit, we calculated the mean and the standard deviation for the summaries (99.53 ± 91.17) and the full contents (1397.44 ± 2101.73). To reduce the dispersion, we removed outliers. Input documents with fewer than 300 words or more than the mean plus three times the standard deviation were treated as outliers. Similarly, summaries with fewer than 19 words or more than the mean plus three times the standard deviation were removed. With this standardization, we suppressed 616 decisions for this analysis, representing 5.80% of the total. Full con-

Figure 5.2 – Ruling vs Summary Histogram



Note: There is no correlation between the summary and full contents length.
Source: The Author

tents mean became 1200.65, with a standard deviation of 893.86; Summary mean became 91.79, with a standard deviation of 62.92. The compression ratio after removing these outliers slightly increased from 7.2% to 7.6%. Figure 5.2 shows the histogram of the frequency distribution of the rulings and summaries length.

5.4 Summary

In this chapter, we presented the RulingBR Dataset. This dataset was built using rulings from the Brazilian Supreme Court (STF). We briefly explained the data assembly, the broad subjects, the size, and the length of documents. We also found out that there is no correlation between the size of the ruling and its summary. In the next chapter, we will present our method for improving the quality of generated legal summaries.

6 LEGALSUMM

As discussed in the Introduction, abstractive summarization might introduce extraneous subjects or facts. This problem may happen because the model often has one subject tied to some context or does not have enough training data. When training for summarization, the model learns to generate texts that it has often seen during training. Deep learning usually applies complex models that are limited in length. This represents a problem for applications in the legal area, where long texts are the standard. If the model cannot work with entire documents, the standard practice is to truncate the document at the maximum length. This approach may hide vital context data for the model correctly generating summaries. Our goal here is twofold. We aim to handle long texts and also to minimize the hallucinations generated by the model. We hypothesize that we may teach a model to distinguish if a summary belongs to a given ruling and that this can improve summarization quality.

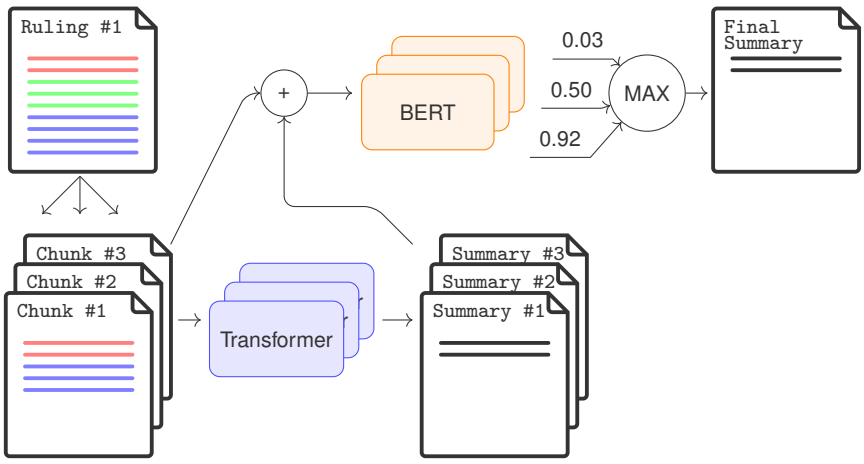
LegalSumm is suitable for long documents in which the focus of interest is dispersed throughout the text. It requires some structure of the source text to allow the extraction of coherent "views". LegalSumm does not require external taggers and may look at several parts of the rulings. This ability is handy when working with long sequences and summarizing topics spread through the source text. LegalSumm is not geared towards news and scientific articles because the focus of the summaries from these types of documents is usually either at the beginning (as in news articles) or the end of the document (as in scientific papers).

6.1 Overview

Figure 6.1 shows our LegalSumm proposal for abstractively generating text summaries from legal decisions. The input source document is split into smaller chunks. Each chunk is passed through a Transformer (in light blue), which generates a summary. Each chunk-summary pair is submitted to a set of BERT models (in light orange) that output scores. The summaries with the highest scores are kept.

Two practical difficulties arise from the fact that legal decisions are frequently long. First, complex models often are unable to handle long sequences due to high memory requirements. Second, even when the models can work with such long texts, the attention mechanism becomes too sparse and unable to focus on the relevant topics.

Figure 6.1 – LegalSumm Overview



Note: The input source document is split into smaller chunks. Each chunk is passed through a Transformer (in light blue), which generates a summary. Each chunk-summary pair is submitted do a set of BERT models (in light orange) that output scores. The summaries with the highest scores are kept.
Source: The Author

6.1.1 Building the Input Data

To overcome these practical problems, we propose splitting the text from a ruling into smaller samples, called *chunks*, that are generated according to predefined rules. The set of rules used for generating these views is called *strategy*. Each strategy defines how a chunk is generated from the text of the ruling. These chunks are depicted on the left bottom corner of Figure 6.1, and they could use the first or the last few paragraphs, or even an arrangement of sentences. Each one may be more biased to emphasize the initial, middle, or final parts of the ruling. This approach allows the model to handle long documents and also to focus on these restricted parts. LegalSumm requires at least two strategies because it needs to select the best summary within the number of candidates. Multiple strategies allow the generation of different "views", allowing our summarization module to generate an independent version of the summaries for each strategy.

Ideally, these chunks should contain coherent parts of the text. An incoherent text could jeopardize the subsequent evaluation step that assesses whether the summary could be inferred from the source text. Thus, the rules for selecting these portions should be preferably defined according to the inner structure of the text (paragraphs or sections). These strategies use the ruling structure (as described in Section 5.1) to split the text.

The strategy adopted here was to split the source text from the RulingBR dataset

into sections. The chunk was composed of an arrangement from the three sections: “report”, “vote”, and “judgment”. These sections may contain more than our operational limit, so the input data is always truncated at length 400. This limitation of 400 tokens required creating different strategies for assembling the chunk to be summarized, as discussed in Section 6.1.1. Each line in Table 6.1 represents a strategy for building the chunks. The first strategy defines that the “chunk” is composed using up to 300 tokens from the “report” section, concatenated with up to 100 tokens from the “judgment” section. For strategies 1 to 5, the generated chunks may be smaller than the maximum length allowed. This happens because the length of each section is enforced independently. For example, for strategy 4, if “report” has only 250 tokens, this will be the final input length under this strategy. Strategies 6, 7, and 8 allow more than 400 tokens, requiring posterior truncation at length 400.

Table 6.1 – Strategies for Generating the Chunks to be Summarized

#	1^{st} Section	Length	2^{nd} Section	Length	3^{rd} Section	Length
1	report	300	vote	0	judgment	100
2	report	0	vote	300	judgment	100
3	report	150	vote	150	judgment	100
4	report	400	vote	0	judgment	0
5	report	0	vote	400	judgment	0
6	report	400	vote	400	judgment	400
7	vote	400	judgment	400	report	400
8	judgment	400	report	400	vote	400

Each strategy (shown in the rows) uses the concatenation of the referred sections, limited to the corresponding length.

6.1.2 Generating Candidate Summaries

Transformer models (VASWANI et al., 2017) (shown in light blue in Figure 6.1) generate independent summaries for each chunk. The Transformer uses an Encoder-Decoder architecture with self-attention, and it can handle sequential data suitable for tasks such as translation and summarization. These alternative versions are vital for our framework because it can choose which version the model believes is the most truthful concerning the source.

6.1.3 Scoring Candidate Summaries

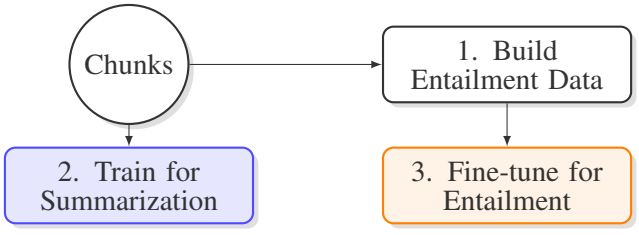
The candidate summaries are combined with the input chunks (round node with the + sign in Figure 6.1) and submitted to BERT (DEVLIN et al., 2018) models (in light orange) that predict scores for each pair, representing how confident the model is that this summary is related to the chunk. The input for BERT uses the special token *CLS* at the start, the chunk tokens, one special *SEP* token, the predicted summary tokens, another *SEP* token. The score is the output after the softmax of a binary classifier using the *CLS* token. The binary classifier gives the confidence for the positive and for the negative class. We use only the output for the positive class. It is a number between 0 and 1, representing how confident the model is that this summary matches the chunk. The score represents the verisimilitude that this summary was generated from a given ruling, based on its contents and writing style. There is one score for each input strategy.

After the scores generation, the summary with the highest score among the candidates is selected and used as the final output summary for each case.

6.2 Training Procedure

Figure 6.2 depicts the steps required for training LegalSumm. The training "Chunks" are generated following the same description presented in Section 6.1.1. These are the inputs for building the entailment data and for training the summarization models. We can see that the two most time-consuming training tasks (steps 2 and 3) can be done in parallel, reducing the total training time. In the following sections, we detail each one of these stages.

Figure 6.2 – The Training Procedure of the LegalSumm



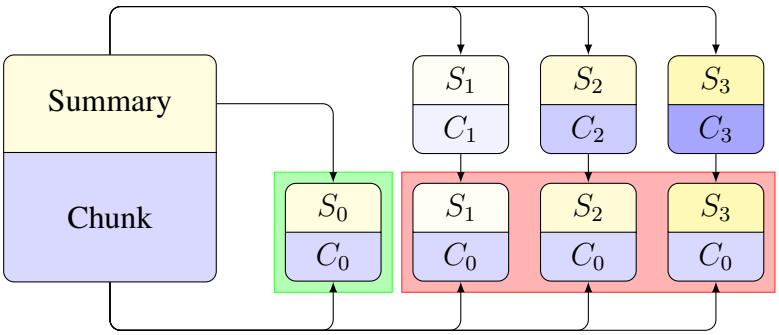
Source: The Author

6.2.1 Building Entailment Data

To reduce the number of hallucinations, we need to train a model to evaluate if a generated summary could be inferred from the source text. This task is closely related to Recognizing Textual Entailment (RTE) 2.7.2 task in Natural Language Processing (NLP). The goal of RTE is to define if one text entails, contradicts, or is neutral concerning another. We use BERT (DEVLIN et al., 2018) models for RTE because they showed state-of-the-art results in this task.

To teach the model to distinguish between an appropriate summary and one out-of-the-context, we need to feed it with real and fake examples. A real example is a ground-truth chunk-summary pair from the dataset. A fake example is formed using the same "chunk", but randomly selecting another summary from the dataset. Facts are the original chunk-summary pair, and fake are the ones that summary cannot be entailed from the chunk.

Figure 6.3 – Building Entailment Data
Bigram Search for Similar Cases: e.g., "habeas", "corpus"



Both generated fact and fake examples use the same source ruling text

Source: The Author

LegalSumm generates artificial entailment data following the procedure depicted in Figure 6.3. Each example (shown large on the left) comprises the chunk and its corresponding ground-truth summary. We use the original chunk and summary, tagging it as "fact" (*i.e.,* S_0, shown in green).

For the fake examples, the intuition is to simulate the effects of hallucinations, *i.e.,* the introduction of extraneous topics close to the broad subject of the ruling. To do that, we use the category label given in the dataset that represents the broad subject of the case (*e.g.,* Habeas Corpus, Habeas Data, *Writ* of Mandamus, Appealing in Civil Procedural Law, etc.). We randomly select ten rulings belonging to "related" categories

(to save space, Figure 6.3 depicts only three – S_1, S_2, and S_3). We consider that categories that share word bigrams in their labels are related. For example, a ruling categorized as "Appealing in Labor Procedural Law" is related to another ruling from the same category and to categories such as *"Appealing in* Criminal Law" or "Civil *Procedural Law*" as they share bigrams. Figure 6.3 uses different background shades to represent these differences from the source ruling. We keep these similar summaries and replace their ruling texts with the original chunk (shown using C_0). Finally, we tag these modified examples as "fake" (generated examples with red backgrounds on the right). With this technique, some fake examples will be more closely related to the source than others. We hypothesize that this method allows distinguishing distant subjects and is even more closely related in incremental degrees of difficulty.

6.2.2 Training Summarization Module

LegalSumm uses one Transformer (VASWANI et al., 2017) for each strategy because the model needs to learn to summarize chunks that are in the format defined by the strategy (see Table 6.1). The intuition is that each strategy covers distinct topics from the original ruling. This approach allows each summarizer to be more specialized; and a specialized model can learn more easily to distinguish the relevant parts of the chunks.

The standard Transformer model (VASWANI et al., 2017) was employed with 6 encoder and 6 decoder layers, embedding and hidden dimension of 512. One model is trained for each input strategy from Table 6.1. They were trained for 20K steps, using batch size 56, and learning rate 1×10^{-3}. Training took between two and six hours (training time varies because the input size of each strategy has a different length).

To train the summarization module, we used OpenNMT-py (KLEIN et al., 2017), more specifically, the standard Transformer model (VASWANI et al., 2017) with 6 encoder and 6 decoder layers, embedding and hidden dimension of 512. One model is trained for each input strategy from Table 6.1. They were trained for 20K steps, using batch size 56, and learning rate 1×10^{-3}. Training took between two and six hours (training time varies because the input size of each strategy has a different length).

We used these trained models to generate the candidate summaries. Some constraints were applied during decoding. We use beam search with size eight, avoid 3-ngram repetition (PAULUS; XIONG; SOCHER, 2017), and require a minimum of 25 tokens and a maximum of 256. We chose these limits because they represent the minimum and max-

imum lengths found in the dataset. The test set contains around 2K examples, and the prediction took 20 minutes for each input strategy.

6.2.3 Fine-tuning for Entailment

Training an entailment model from scratch requires the model to learn to "understand" the language and the complex relations from the topics. BERT (DEVLIN et al., 2018) is a method for pre-training language models that can later be fine-tuned for various NLP tasks. It has been applied to several such tasks, achieving results that outperform the state-of-the-art. Its authors propose a costly pre-training method using a large volume of data so that this model can understand the general topics of the language. This pre-training on general texts allows the model to generate rich representations of the source data. Later, this pre-trained model can be fine-tuned for specific tasks such as RTE.

The authors of BERT released a multilingual model trained on the Wikipedias from 104 languages (PIRES; SCHLINGER; GARRETTE, 2019). Portuguese is among the languages that compose the multilingual model so that it can be used with the dataset of Brazilian rulings. LegalSumm uses one entailment model for each one of the input strategies. There is no parameter sharing between these models.

To train our entailment module, we used the Hugging Face Transformers library (WOLF et al., 2019). We start from bert-base-multilingual-cased and fine-tune it for entailment. We append on top of the model a classification head for the model to choose if a given chunk-summary pair belongs to the "fact" of the "fake" class. Even though we converted all text to lowercase, we used the cased version of BERT multilingual because the uncased version also strips diacritics. Using lowercase for all data may jeopardize recognizing proper nouns, entities, and the start of the sentences. Considering that the dataset is not consistent with casing, we opted to use lowercase everywhere. Also, Portuguese uses diacritics and, while the text can be understood without them, removing them introduces noise as some discriminating features are lost – *e.g.,* the distinction between "baby" (*bebê*) and "s/he drinks" (*bebe*) is on the diacritical mark.

The input is the concatenation of a classification token, up to 400 tokens from the chunk, a special SEP token, and up to 110 tokens from the candidate summary. The output layer is just two classes representing how confident the model is that the summary entails the input data. Each fold contains 20% of testing data. From the remaining, we separated 10% for validation. For each example, we generate another ten fake examples using the

method shown in Section 6.3. We trained each model for 6K steps until validation loss stopped diminishing, batch size 32, learning rate 2×10^{-5}. Training took between two and six hours for each model, depending on the length of the inputs.

Furthermore, to avoid data leakage when selecting the fake examples, we ensure that these examples come from the training folds. This way, the test fold is protected and not mixed with the training folds.

Unfortunately, the restriction of 400 tokens from the chunk might not provide enough "coverage" of the ruling for the model to correctly entail that this chunk-summary is a valid combination. In this case, as the model does not have enough information, this sample would be noisy, probably jeopardizing the model performance. We address this issue using eight models with different chunking strategies. We expect the chunking strategy with enough information (*i.e.,* best "coverage") to provide the best summaries. In contrast, the situations in which the model has insufficient information and would not generate the best summaries receive a less confident score.

6.3 Summary

This chapter presented the LegalSumm proposed architecture to apply abstractive summarization in the juridical domain. We review the two main challenges identified: the restriction of length for Transformer-based models, and the entailment problem between the ruling and the generated summary. In the next chapter, we describe the experiments we have already done in-depth while researching methods for summarization.

7 EXPERIMENTS

This chapter describes experiments done while investigating tools and techniques used for generating text summaries. The experiment in Section 7.1 describes a comparative analysis using conventional extractive and abstractive models. Section 7.2 describes the comparative experiments using LegalSumm and strong baselines showing how this method can improve generated summaries.

7.1 Summarization Using RulingBR

This experiment investigated the suitability of extractive and abstractive approaches in summarizing legal rulings. The description of this experiment resulted in a paper published and presented at RANLP 2019 (FEIJO; MOREIRA, 2019). Given that the vast majority of works in the summarization area are focused solely on news datasets, we believe that testing summarization on a new domain is vital given the different nature of the input documents. We tested 13 methods; nine extractive, and four abstractive, over the RulingBR, described in Chapter 5.

7.1.1 Text Pre-Processing

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. Besides, we changed all text to lowercase and isolated the standard punctuation symbols from the alphabetic characters to avoid them being interpreted as a part of some word.

7.1.2 Vocabulary

Portuguese has a rich vocabulary, with words having lots of variant forms. Verbs, in particular, can have dozens of different suffixes corresponding to the various conjugations. In the legal domain, it is common to reference existing laws, specific dates, and names. Thus, it is improbable that the vocabulary generated during training will contain all possible words present in the test set. To deal with the problem of OOV words, we

used the SentencePiece package (KUDO; RICHARDSON, 2018), which implements the sub-word units with unigram representation. The combination of pieces generates words even when they are not present in the training set.

7.1.3 Source and Target Length

Extractive and Abstractive methods require the definition of the target length. To extractive strategies, a score is typically assigned to each sentence. The sentences are ranked by this scored and added to the output summary until the desired length is reached. Using the mean length of the reference summaries as the desired length will always yield an error because the generated predictions will be shorter or longer, penalizing precision or recall.

In the RulingBR dataset, the mean length for the test set is 190 tokens, with a minimum of 20 and a 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 with truncating the reference summaries at different points to observe the effect over the mean length of the test set.

Table 7.1 – Reference Summary Length

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 7.1 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 length and higher deviation (but with a significant error associated). The target length will influence ROUGE scores (as described in Section 2.8.4). Truncating the target length to 300 tokens leads to summaries of 158 tokens on average.

The definition of the source length matters because abstractive models require a fixed size input. Different lengths will require padding, which in turn will harm training. Through empirical observation, we found that the *relatório* (report) section from the source text usually contains most of the information present in the reference summary. So,

after removing some boilerplate text typically present at the beginning of the documents, we extract a sequence of words until it reaches the desired summary length.

Table 7.2 – Heuristic Method Evaluation

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

Note: Our simple baseline heuristic was evaluated for different source lengths. The results show that the F-measure is reasonably stable across different lengths.

This simple heuristic approach will be used as a baseline show in Table 7.2. The results are for different maximum lengths of *relatório* and *voto* and trying to find summaries with this same length. Truncating the source length has little effect over the F-measure of the ROUGE scores. To make a fair comparison between the algorithms, we aimed at generating summaries of similar lengths.

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*).

7.1.4 Extractive Models

The experiment used the conventional extractive models Luhn, LexRank, LSA, KLSum, SumBasic, and TextRank (described in Section 2.4). The Random baseline defines random scores for sentences, and the top-scoring sentences are selected to compose the summary. The Heuristic baseline described in the previous section is also used in the experiment.

7.1.5 Abstractive Models

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. These are models with attention components, as described in Section 2.5.3.

NMTSmall and NMTMedium are standard RNN models. They use an encoder-decoder architecture. The decoder employs Luong, Pham and Manning (2015) style attention model over the input. We trained the network to learn when to stop generating the summary. We appended an End-of-Document token to the instances during training to mark the end of the output. When the network generates this token, we truncate the output at this point. The model uses a beam search (as described in Section 2.7.3 of size four when decoding, and it is configured to ignore outputs 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 (described in Section 2.1.4), 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. NMTSmall 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. The Transformer model uses the configuration as originally proposed by Vaswani et al. (2017). The TransformerAAN uses cumulative average attention network in the decoder as described in (ZHANG; XIONG; SU, 2018). The objective is to reduce the required training and inference time.

7.1.6 Results and Discussion

The performance of the extractive algorithms shown in Table 7.3 was disappointing. TextRank algorithm appears twice in our results as we used both the Gensim and the Sumy implementations. Gensim's implementation replaced the standard keyword overlap with the BM25 algorithm (ROBERTSON et al., 1995). Except for 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 present in the source text. Most generated summaries selected just one very long sentence, while others used random, disconnected sentences.

Table 7.3 – Extractive Algorithm Evaluation

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
Heuristic	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

Our experiments varying the lengths of the input method (Table 7.2) have shown that even with more substantial source inputs, which could contain more tokens that should be present in the output, the performance was decreasing.

Our baseline results (FEIJO; 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 this experiment because we removed stopwords, and the reported results were for the entire dataset. Since in this experiment, we have a training phase for the abstractive models, ROUGE results were evaluated only for the test set.

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

Table 7.4 shows reasonably good results for both NMT and Transformer models. There was a small advantage for the standard Transformer model 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 considerable amounts of memory to run. So, the batches need to be smaller. As a consequence, it required 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 reported for other datasets are not directly comparable to our results. Still, they may serve as a reference. Zhang et al. (2019) reports that the current state-of-the-art for the CNN/Daily Mail dataset (NALLAPATI; XIANG; ZHOU,

[1]Using Sumy implementation
[2]Using Gensim implementation

Table 7.4 – Abstractive Model Evaluation

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

Note: The column "Len" represents the mean length of the generated summaries.

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 summarizer correctly captured the main topics. Nevertheless, as shown in Figure 7.1, 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 the minimum length restriction may have caused this.

Legal practitioners rely on summaries to do their jobs since it is impossible to read the entire 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 replace humans in this task. The current state is promising, but automatic systems are not always capable of generating valuable summaries. Hence, they could be used to prepare drafts, which still require human revision.

7.2 LegalSumm Experimental Evaluation

To evaluate the effectiveness of LegalSumm in improving the quality of the generated summaries, we have performed two experiments using the RulingBR dataset (described in Chapter 5).

The automatically generated summaries were scored using the official ROUGE script without stemming. The evaluation metrics are described in Section 2.8.4. Addi-

Figure 7.1 – Sample Summary Generated by the Transformer Model

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 ...

Note: The reference 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.
Source: The Author

tionally, to save space on the tables, we multiply the scores by ten in all experimental runs.

We organized the results of our experimental evaluation into four parts, each designed to answer one of the following questions. Each of these questions is addressed in the next subsections.

1. How useful is the entailment module in LegalSumm?
2. How does LegalSumm compare with baselines for text summarization?
3. How do legal experts rate the quality of the automatically generated legal summaries?
4. How does the number of fake summaries impact the results in LegalSumm?

7.2.1 How Useful is the Entailment Module in LegalSumm?

Table 7.5 shows the results of a comparison among LegalSumm and the eight strategies for assembling the input data introduced in Table 6.1. Recall that these strategies rely only on standard Transformer models. Thus, this comparison works as an internal baseline because they are generated by the summarization module without using the entailment module.

Looking at the scores, we find that strategies 6, 7, and 8 produced the best summaries among the baselines. The advantage of these strategies can be attributed to the fact that they use the maximum number of tokens allowed. With more “features”, the

summarization module produces better summaries.

Although it is inappropriate to say that ROUGE correctly evaluates entailment, it is safe to consider that a higher ROUGE score might indicate that the summary more closely represents the expected topics. In this case, we assume that if our entailment module can distinguish between "fact" and "fake" chunk-summary pairs, the ROUGE score should show some improvement.

Table 7.5 – LegalSumm Comparison to Transformer Baselines

Model	R1-F	R1-P	R1-R	R2-F	R2-P	R2-R	**RL-F**	RL-P	RL-R
LegalSumm	**43**(±3)	**63**(±1)	**37**(±3)	**27**(±3)	**39**(±3)	**23**(±3)	**35**(±3)	**51**(±1)	**30**(±3)
Transformer 1	37(±2)	55(±2)	31(±2)	20(±3)	30(±2)	17(±3)	29(±2)	43(±2)	25(±2)
Transformer 2	38(±3)	57(±2)	33(±3)	22(±3)	33(±3)	19(±3)	31(±3)	45(±2)	26(±3)
Transformer 3	39(±2)	58(±1)	33(±2)	23(±2)	33(±2)	20(±3)	31(±2)	46(±2)	27(±2)
Transformer 4	36(±2)	56(±1)	31(±2)	20(±2)	31(±1)	17(±2)	28(±2)	44(±1)	24(±2)
Transformer 5	39(±2)	59(±1)	33(±3)	23(±2)	34(±2)	20(±2)	31(±2)	47(±2)	27(±2)
Transformer 6	42(±3)	60(±2)	36(±3)	26(±3)	36(±3)	22(±3)	34(±3)	48(±2)	29(±3)
Transformer 7	42(±3)	60(±2)	36(±3)	26(±3)	36(±3)	**23**(±4)	34(±3)	48(±2)	29(±3)
Transformer 8	42(±2)	60(±2)	36(±3)	26(±3)	36(±2)	22(±3)	34(±2)	48(±2)	29(±3)

Note: Average ROUGE scores for summaries generated by each strategy using 5-fold cross-validation. The numbers between brackets are the standard deviations across folds. Best results in bold.

To illustrate the LegalSumm's ability to select a summary among the candidates, we show a shortened example in Table 7.6. The original summaries in Portuguese were translated into English so that a broader audience could understand them. The first line represents the ground-truth summary. The following lines represent the candidate summaries generated by each strategy. Imprecise or factual errors are highlighted in yellow. Missing information such as "credit cooperative" is not represented and would lower the score. The "Score" column shows the confidence that this is a summary from the input chunk. We can see that LegalSumm assigns higher scores to the summaries with fewer factual errors.

Summaries 2, 5, and 7 have more invalid information and received the lowest scores. Summaries generated with strategies 1, 4, and 6 contain fewer errors and received the highest scores. In this example, the summary from strategy 6 was chosen.

In an error analysis experiment, we scored each of the 10K summaries generated by each of the eight Transformer models and the summaries picked by LegalSumm using RL-F. If the entailment module made the optimal choice every time (*i.e.,* selected the summary that maximizes RL-F), then the average score for LegalSumm would be ten

Table 7.6 – Sample Scores Generated by the Entailment Module

Strategy	Score	Summary
Ground-Truth		labor. union contribution. credit cooperative. equivalent to a banking establishment. absence of the required pre-questioning. overviews 282 and 356 of the stf. interpretation of statutory legislation.
1	0.96	labor law. rural union contribution. proof. absence of pre-questioning. overviews 282 and 356 of the stf. review of facts and evidence. summary no. 279 of the stf. unfeasibility of the extraordinary appeal.
2	0.00	administrative. restructuring of the workers of the brazilian institute for the environment and renewable natural resources - ibama.
3	0.80	labor law. rural union contribution. prescriptive period. absence of pre-questioning. overviews 282 and 356 of the stf. analysis of statutory legislation.
4	0.96	tax law. rural union contribution. absence of pre-questioning. Precedent 282 of the stf. analysis of statutory legislation.
5	0.67	search and seizure action. fiduciary sale in guarantee. civil prison. decree-law n. 911/69. binding summary no. 25 of this court. No pre-questioning. overviews 282 and 356 of the stf. analysis of statutory legislation.
6	**0.99**	labor. wage differences. rural union contribution contract. absence of pre-questioning. overviews 282 and 356 of the stf.
7	0.76	civil procedure. execution. savings. inflationary purges. no pre-questioning. overviews 282 and 356 of the stf. analysis of statutory legislation.
8	0.84	labor law. rural union contribution. compensation. pre-questioning. overviews 282 and 356 of the stf. analysis of statutory legislation.

Note: Candidate summaries automatically generated by the summarization module and their scores. Factual errors are highlighted. All summaries were shortened to save space and translated into English to be understood by a broader audience.

points higher. The choices made by the entailment module are not optimal because of two main reasons (i) the goal of the training procedure is not to maximize ROUGE scores, which are not known at that stage; and (ii) if multiple good candidate summaries are available, LegalSumm will be penalized for not choosing the one that follows the same writing style as the ground-truth.

7.2.2 How does LegalSumm Compare with Baselines for Text Summarization?

In this section, we compare LegalSumm with existing text summarization systems, including BertSumExt and BertSumAbs by Liu and Lapata (2019), and BART by Lewis et al. (2020). Additionally, we also compare with the results published by Feijo and Moreira (2019) on the same dataset.

Next, we provide a brief description of these baselines. Notice that BertSumExt, BertSumAbs, and BART cannot readily work on the RulingBR dataset, which is in Portuguese. Thus, adaptations were needed. These models were trained using the concatenation of sections "Report", "Vote", and "Judgment", *i.e.,* the same contents seen by LegalSumm.

- **BertSumExt** (LIU; LAPATA, 2019) is an extractive approach that uses BERT to generate the features for the sentences and to decide which sentences should compose the final summary.
- **BertSumAbs** (LIU; LAPATA, 2019) is an abstractive model that uses a standard BERT as an encoder and trains a decoder with a causal mask from scratch. Because BertSumExt and BertSumAbs were trained using English-only texts, but we needed these models to work with Portuguese-only texts, we could not use any fine-tuned checkpoints. Thus, we used the original code shared by the authors and replaced "bert-base-cased" with "bert-base-multilingual-cased" (which includes Portuguese) and fine-tuned it for summarization using the RulingBR dataset. Fine-tuning BertSumExt required four hours, and BertSumAbs required 28 hours, both using one Nvidia Tesla V-100 GPU.
- **BART** (LEWIS et al., 2020) is also an encoder-decoder model that uses the same BERT architecture as encoder. The decoder uses a causal mask with a cross-attention with the encoder. Again, there is no BART model trained using Portuguese texts. Thus, we were required to run the entire pre-training. For this task, we use three million sentences available from (GOLDHAN; ECKART; QUASTHOFF, 2012). BART's "large" configuration was used, and pre-training used one Nvidia Tesla V-100 GPU for 210K steps. Pre-training took about 70 hours. After pre-training, each fold was fine-tuned for summarization. Fine-tuning took about two hours.

Table 7.7 shows the results for the comparison with external baselines. BertSumExt did not produce good summaries in our legal data. We believe this is due to differences between summarizing news articles and legal documents (as discussed in Section 1.1). On the other hand, BertSumAbs and BART produced competitive summaries, but required a very long training procedure. Even with these strong baselines, the results showed that LegalSumm outperformed or matched the ROUGE-F scores for all metrics evaluated. Despite the required training of several similar models, we improved ROUGE

Table 7.7 – LegalSumm Comparison to Summarization Baselines

Model	R1-F	R1-P	R1-R	R2-F	R2-P	R2-R	**RL-F**	RL-P	RL-R
LegalSumm	**43**(±3)	**63**(±1)	37(±3)	**27**(±3)	**39**(±3)	23(±3)	**35**(±3)	**51**(±1)	30(±3)
BertSumExt	23(±2)	38(±1)	20(±2)	6(±1)	10(±1	5(±1)	16(±1)	28(±1)	14(±1)
BertSumAbs	41(±3)	62(±1)	34(±3)	25(±3)	37(±2)	21(±3)	33(±3)	49(±1)	28(±3)
BART	**43**(±3)	56(±2)	**41**(±3)	25(±4)	32(±4)	**24**(±4)	32(±4)	41(±3)	**31**(±4)

Note: Average ROUGE scores for summaries generated by LegalSumm and external baselines using 5-fold cross-validation. The numbers between brackets are the standard deviations across folds. Best results in bold.

scores without using larger versions expensive to train. We ran paired t-tests between the RL-F scores (which is the most important of the metrics we reported) of LegalSumm and all baselines, including our eight Transformers. The results showed statistically significant differences between LegalSumm and all other baselines (p-values < 0.01 in all cases). Our method has the advantage of not requiring additional trained NER or POS tools, and it is suitable even for low-resource languages such as Portuguese.

We also compared LegalSumm to our previous work on the RulingBR Feijo and Moreira (2019). We explained this experiment in and in Section 7.1.6. In that work, we ran several experiments using both extractive and abstractive models. For the abstractive models, we used standard RNN and Transformer architectures. The minimum summary lengths were set to 100 and 120 tokens, respectively. The Transformer model with a minimum summary length of 100 tokens was the configuration that produced the best scores – and these are the results we reproduce here.

Table 7.8 shows that LegalSumm improved or matched all ROUGE F-scores in comparison to the basic encoder-decoder Transformer. The most significant advantage was on precision scores. LegalSumm can choose safer alternatives avoiding unusual terms or extended sentences that could append wrong information. This behavior positively impacts precision, but it could harm recall because more complete or extended candidates would not be selected. The baseline, however, achieves a higher recall. That happens because it does not have the restriction imposed by the entailment module and thus can generate more terms that are found in the ground-truth summaries. Since LegalSumm's advantage in the precision is greater than its loss in the recall, its F-scores were higher.

Table 7.8 – LegalSumm Comparison to our Previous Work

Approach	R1-F	R1-P	R1-R	R2-F	R2-P	R2-R	**RL-F**	RL-P	RL-R
LegalSumm	**44**	**64**	38	**29**	**42**	25	**36**	**52**	31
Feijo and Moreira (2019)	**44**	49	**46**	27	28	**28**	35	39	**37**

Note: ROUGE Scores on train/test splits. Best results are in bold.

7.2.3 How do Legal Experts Rate the Quality of the Automatically Generated Summaries?

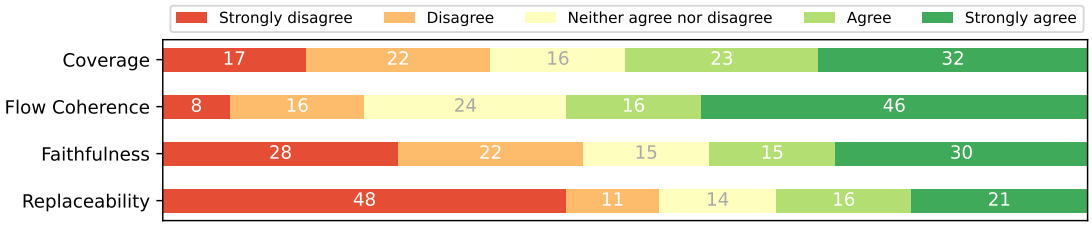
To answer this question, we designed an evaluation experiment in which legal experts evaluate randomly selected cases. Our experts were eleven volunteers with a law degree and over ten years of experience working with legal rulings. For each case, the legal experts were presented with (i) the full text of the ruling, (ii) the ground-truth summary, (iii) the summary generated by BertSumAbs, and (iii) the summary produced by LegalSumm. This assessment was a time-consuming task that required familiarity with the topic. Therefore, it was restricted to ten cases. We chose BertSumAbs as the comparison model because it obtained the highest RL-F scores among our external baselines. For each of the automatically generated summaries, the legal expert was asked to rate the summary concerning four aspects by assigning a score from one and five. A score of one represents strong disagreement, while five represents strong agreement with each of the following statements.

1. The summary covers important parts of the text.
2. The summary presents a coherent flow.
3. The summary is faithful to the facts and does not introduce extraneous facts.
4. The summary could replace the original (manually created) summary.

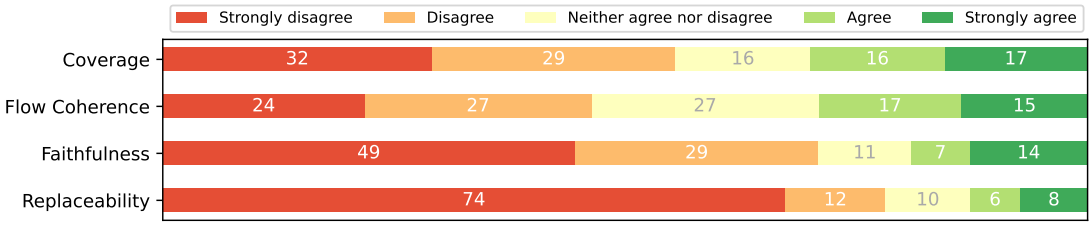
The results of the evaluation by the legal experts are shown in Figure 7.2. In comparison with BertSumAbs, LegalSumm was better in all four evaluated aspects. Regarding flow coherence and faithfulness, LegalSumm had about twice the number of positive assessments, and, for replaceability, the number was almost three times as high. In LegalSumm, the aspects with the most positive assessments were coherence (56% of the answers agreed that the flow was coherent) and coverage (50% of the answers). However, over half of the answers were in disagreement concerning being faithful to the facts and the possibility of using the generated summary as a replacement to the original one. These negative ratings mean that, despite the improvements, these generated summaries

Figure 7.2 – Legal Expert Evaluation

(a) LegalSumm



(b) BertSumAbs



Note: Scores given by experts to summaries generated by LegalSumm and BertSumAbs to each of the four statements regarding the quality of the summaries.
Source: The Author

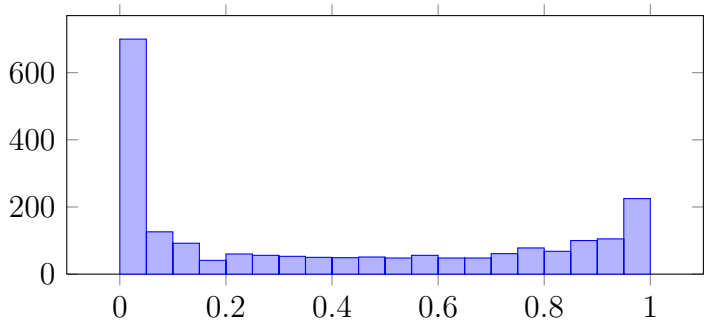
still lack the reliability to be used independently. Nevertheless, LegalSumm's summaries could be used as drafts that require manual checking, thus alleviating the burden on human summarizers.

7.2.4 Impact of the Variation of the Number of Fake Examples

Some points deserve a more profound analysis. First, according to Cao et al. (2018), state-of-the-art abstractive summarization models generate around 30% of summaries with one or more false facts. The definition of a false fact is when a triplet (subject + predicate + object) could not find support in the source text. Thus, when applying our entailment classifier over the test set, we expected that the summaries would contain a fair number of "hallucinations", with around 30% of low (< 0.5) scores, indicating that these summaries were considered as fake. However, the histogram shown in Figure 7.3 shows that the model classified 60% of cases as "fake". This bias is a consequence of the training data having ten times more fake samples than true samples.

To investigate the influence of generating ten fakes for each original example, we

Figure 7.3 – Scores of Generated by Entailment Module



Note: The Figure shows scores for a test fold in one of the chunking strategies.
Source: The Author

also experimented with 5 and 15. The number of "replacements" (*i.e.,* when the generated summary is replaced by another with higher confidence) increased with the higher number of fake examples. A ratio of 5 to 1 generated 1097 replacements, which increased to 1172 and 1501 for ratios of 10 to 1 and 15 to 1, respectively. Despite these changes, the results were almost identical with slight modifications in precision or recall, but not altering the F-score of any strategy. These results show that the entailment component is relatively tolerant to class imbalance and can be used in contexts where the source data has some structure, and the relevant data for the summaries are spread in the text.

7.2.5 Limitations

Using BERT for assessing the entailment restricts the maximum length supported by LegalSumm. BERT restricts the length up to 512 tokens. As BERT needs to evaluate both the chunk and the candidate summary, the chunk length had to be limited to a maximum of 400 tokens. Each summary length increment requires reducing the length of the chunks. Therefore, if the summaries needed to be twice as large, the chunks would be limited to almost this same length. In practice, LegalSumm would not be suitable for summaries longer than 200 tokens. To overcome this limitation, BERT should be replaced by another model that supports more than 512 tokens.

7.3 Summary

This section presented LegalSumm, a method for improving abstractive summarization of legal rulings. LegalSumm works with long documents by splitting the legal

document into predefined chunks that the model can handle and generating a candidate summary. Also, it can remove extraneous topics by selecting the most suitable summaries among these candidate summaries. Compared to other methods for avoiding "hallucination", LegalSumm has the advantage of not requiring NER or POS taggers. It improves the ROUGE scores without increasing the size of the summarizer.

We conducted different evaluation experiments. The first experiment showed that extractive approaches did not achieve good results. The abstractive strategies obtained much better results, but in some cases, they deviate from the subject of the source text.

The second experiment demonstrates the ability of LegalSumm to improve the quality of the summaries generated by standard Transformer models. We split this second experiment into four research questions: (i) how useful is the entailment module in LegalSumm; (ii) how does LegalSumm compare with baselines for text summarization; (iii) how do legal experts rate the quality of the automatically generated summaries; (iv) how does the number of fake summaries impact the results in LegalSumm.

The (i) question demonstrates that LegalSumm helps to improve the quality of generated summaries. The (ii) question compared LegalSumm to strong baselines including BertSumExt, BertSumAbs (LIU; LAPATA, 2019), and BART (LEWIS et al., 2020). The (iii) question compared LegalSumm to the results of (FEIJO; MOREIRA, 2019) on the same dataset. LegalSumm outperforms or matches all ROUGE F-scores from these baselines. Our third evaluation asked legal experts to assess the automatically generated summaries regarding coverage, coherence, faithfulness, and replaceability. In (iv) show the impact of varying the number of fake examples and found out that the entailment module is relatively tolerant to class imbalance. Finally, we discussed the limitations of the proposed method.

8 CONCLUSION

Text summarization is a research topic that is being updated frequently in the last few years. Recent advances in NLP brought a new boost and updated the challenges in this area.

This work presented LegalSumm, a method for improving the quality of the abstractive summarization models in the legal domain. We presented an overview of the most relevant background topics related to text summarization was presented. Also, we organized and discussed the main techniques applied to legal text summarization.

The research started with constructing a dataset of legal rulings, the data collection, cleaning, data representation, and the analysis of its size. The evaluation using standard extractive techniques using this dataset and its results were shown and analyzed. The results showed that extractive methods provided weak performance being unable to generate useful summaries.

The evaluation using abstractive models provided much better results, with summaries similar to those produced by humans. However, they also presented severe problems with repeating expressions and introducing subjects that were not present in the source documents.

The experiment using the BERT-based models showed the possibility of using pre-training and fine-tuning for Portuguese and validated the suitability of these methods for entailment. The LegalSumm architecture was presented, integrating state-of-the-art techniques, using pre-training, and models dedicated to summarization and entailment. We showed that this architecture improves the quality of the summaries in the legal domain.

Despite these improvements, our evaluation with legal experts demonstrated that these generated summaries still lack the reliability to be used independently. Nevertheless, LegalSumm's summaries could be used as drafts that require manual checking, thus alleviating the burden on human summarizers.

We can prospect some future works. In our summarization module, the Transformer architecture could be replaced with a more robust summarizer like BART (LEWIS et al., 2020). In the entailment module, another encoder that could deal with inputs longer than 512 would be beneficial.

Our chunking strategies took advantage of the rulings' internal structure. Despite the convenience of using these sections to define our views, our method has become too specific for rulings with these characteristics. The applicability of our method with other

kinds of legal documents requires the manual definition of valuable parts to generate the summaries. In future work, we could define a more general method applicable to other legal documents that do not have the same structure.

In our entailment module, we used a simple method for generating fake data. As future work, we could pursue better POS and NER tools trained in Portuguese that would be used to create more fine-grained entailment data. Then, with this improvement, we could tag the position of the factual transformations and train the model to detect and find these positions. With an entailment module capable of showing the error, we could use this model to replace or remove these parts from the summary.

In every experiment, we used all available samples as we intended to have a neutral writing style. In this situation, our training method was mixing content and writing style. Future work could experiment with splitting summaries per judge that follow the same writing style. This approach would favor the training to focus only on the content.

REFERENCES

ANAND, D.; WAGH, R. Effective deep learning approaches for summarization of legal texts. **Journal of King Saud University-Computer and Information Sciences**, Elsevier, 2019.

ARAUJO, M. et al. An evaluation of machine translation for multilingual sentence-level sentiment analysis. In: **ACM SAC**. [S.l.: s.n.], 2016.

BAHDANAU, D.; CHO, K.; BENGIO, Y. Neural machine translation by jointly learning to align and translate. **arXiv preprint arXiv:1409.0473**, 2014.

BANERJEE, S.; LAVIE, A. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In: **Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization**. [S.l.: s.n.], 2005. p. 65–72.

BARRIOS, F. et al. Variations of the similarity function of textrank for automated summarization. **arXiv preprint arXiv:1602.03606**, 2016.

BECKER, K.; MOREIRA, V. P.; SANTOS, A. G. dos. Multilingual emotion classification using supervised learning. **Inf. Process. Manage.**, v. 53, n. 3, p. 684–704, may 2017. ISSN 0306-4573.

BELICA, M. **sumy: Module for automatic summarization of text documents and HTML pages**. 2018. Available from Internet: <https://github.com/miso-belica/sumy>.

BELTAGY, I.; PETERS, M. E.; COHAN, A. Longformer: The long-document transformer. **arXiv preprint arXiv:2004.05150**, 2020.

BOJANOWSKI, P. et al. Enriching word vectors with subword information. **Transactions of the Association for Computational Linguistics**, MIT Press, v. 5, p. 135–146, 2017.

CAO, Z. et al. Faithful to the original: Fact aware neural abstractive summarization. In: **Thirty-Second AAAI Conference on Artificial Intelligence**. [S.l.: s.n.], 2018.

CARBONELL, J.; GOLDSTEIN, J. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In: **Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval**. [S.l.: s.n.], 1998. p. 335–336.

CELIKYILMAZ, A. et al. Deep communicating agents for abstractive summarization. **arXiv preprint arXiv:1803.10357**, 2018.

CER, D. et al. Universal sentence encoder for english. In: **Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations**. [S.l.: s.n.], 2018. p. 169–174.

CHEN, Y.; BANSAL, M. Fast abstractive summarization with reinforce-selected sentence rewriting. **CoRR**, abs/1805.11080, 2018. Available from Internet: <http://arxiv.org/abs/1805.11080>.

CHITNIS, R.; DENERO, J. Variable-length word encodings for neural translation models. In: **Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing**. [S.l.: s.n.], 2015. p. 2088–2093.

CHO, K. et al. On the properties of neural machine translation: Encoder-decoder approaches. **arXiv preprint arXiv:1409.1259**, 2014.

CHO, K. et al. Learning phrase representations using rnn encoder-decoder for statistical machine translation. **arXiv preprint arXiv:1406.1078**, 2014.

COHN, T. et al. Incorporating structural alignment biases into an attentional neural translation model. **arXiv preprint arXiv:1601.01085**, 2016.

COMPTON, P.; JANSEN, R. Knowledge in context: A strategy for expert system maintenance. In: **Proceedings of the Second Australian Joint Conference on Artificial Intelligence**. Berlin, Heidelberg: Springer-Verlag, 1990. (AI '88), p. 292–306. ISBN 0387520627.

CONNEAU, A. et al. Supervised learning of universal sentence representations from natural language inference data. **arXiv preprint arXiv:1705.02364**, 2017.

DAGAN, I. et al. **Recognizing textual entailment: Models and applications**. [S.l.]: Morgan & Claypool Publishers, 2013.

DEVLIN, J. et al. BERT: Pre-training of deep bidirectional transformers for language understanding. **arXiv preprint arXiv:1810.04805**, 2018.

EDMUNDSON, H. P. New methods in automatic extracting. **J. ACM**, ACM, New York, NY, USA, v. 16, n. 2, p. 264–285, abr. 1969. ISSN 0004-5411.

ERKAN, G.; RADEV, D. R. Lexrank: Graph-based lexical centrality as salience in text summarization. **Journal of Artificial Intelligence Research**, v. 22, p. 457–479, 2004.

FARZINDAR, A.; LAPALME, G. Letsum, an automatic legal text summarizing system. **Legal knowledge and information systems, JURIX**, p. 11–18, 2004.

FEIJO, D. d. V.; MOREIRA, V. P. Rulingbr: A summarization dataset for legal texts. In: **Computational Processing of the Portuguese Language (PROPOR 2018)**. [S.l.]: Springer International Publishing, 2018. p. 255–264.

FEIJO, D. d. V.; MOREIRA, V. P. Summarizing legal rulings: Comparative experiments. In: **Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)**. [S.l.: s.n.], 2019. p. 313–322.

FEIJO, D. d. V.; MOREIRA, V. P. Mono vs multilingual transformer-based models: a comparison across several language tasks. **arXiv preprint arXiv:2007.09757**, 2020. Available from Internet: <https://arxiv.org/abs/2007.09757>.

FIALHO, P. et al. Inesc-id@assin: Medição de similaridade semântica e reconhecimento de inferência textual. **Linguamática**, v. 8, n. 2, p. 33–42, Dez. 2016.

GALGANI, F.; COMPTON, P.; HOFFMANN, A. Combining different summarization techniques for legal text. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. **Proceedings of the workshop on innovative hybrid approaches to the processing of textual data**. [S.l.], 2012. p. 115–123.

GELBART, D.; SMITH, J. Beyond boolean search: Flexicon, a legal text-based intelligent system. In: **Proceedings of the 3rd international conference on Artificial intelligence and law**. [S.l.: s.n.], 1991. p. 225–234.

GOLDHAN, D.; ECKART, T.; QUASTHOFF, U. Building large monolingual dictionaries at the leipzig corpora collection: From 100 to 200 languages. In: **Proceedings of the 8th International Language Resources and Evaluation (LREC'12)**. [S.l.: s.n.], 2012.

GONÇALVES, T.; QUARESMA, P. Using linguistic information to classify portuguese text documents. In: **Mexican International Conference on Artificial Intelligence**. [S.l.: s.n.], 2008. p. 94–100.

GONG, Y.; LIU, X. Generic text summarization using relevance measure and latent semantic analysis. In: ACM. **Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval**. [S.l.], 2001. p. 19–25.

GOODRICH, B. et al. Assessing the factual accuracy of generated text. In: **Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining**. [S.l.: s.n.], 2019. p. 166–175.

GROVER, C. et al. Automatic summarisation of legal documents. In: **Proceedings of the 9th International Conference on Artificial Intelligence and Law**. [S.l.]: Association for Computing Machinery, 2003. (ICAIL '03), p. 243–251. ISBN 1581137478.

GROVER, C.; HACHEY, B.; KORYCINSKI, C. Summarising legal texts: Sentential tense and argumentative roles. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. **Proceedings of the HLT-NAACL 03 on Text summarization workshop-Volume 5**. [S.l.], 2003. p. 33–40.

GUIMARÃES, J. A. C. Elaboração de ementas jurisprudenciais: elementos teórico-metodológicos. **Série Monografias do CEJ**, v. 9, 2011.

HAGHIGHI, A.; VANDERWENDE, L. Exploring content models for multi-document summarization. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. **Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics**. [S.l.], 2009. p. 362–370.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. **Neural computation**, MIT Press, v. 9, n. 8, p. 1735–1780, 1997.

JONES, K. S. A statistical interpretation of term specificity and its application in retrieval. **Journal of Documentation**, v. 28, p. 11–21, 1972.

KANAPALA, A.; PAL, S.; PAMULA, R. Text summarization from legal documents: a survey. **Artificial Intelligence Review**, Springer, v. 51, n. 3, p. 371–402, 2019.

KITAEV, N.; KAISER, Ł.; LEVSKAYA, A. Reformer: The efficient transformer. **arXiv preprint arXiv:2001.04451**, 2020.

KLEIN, G. et al. OpenNMT: Open-source toolkit for neural machine translation. In: **Proceedings of ACL 2017, System Demonstrations**. Vancouver, Canada: Association for Computational Linguistics, 2017. p. 67–72.

KRYŚCIŃSKI, W. et al. Evaluating the factual consistency of abstractive text summarization. In: **Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)**. [S.l.: s.n.], 2020. p. 9332–9346.

KUDO, T. Subword regularization: Improving neural network translation models with multiple subword candidates. **CoRR**, abs/1804.10959, 2018. Available from Internet: <http://arxiv.org/abs/1804.10959>.

KUDO, T.; RICHARDSON, J. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In: **Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations**. Brussels, Belgium: Association for Computational Linguistics, 2018. p. 66–71. Available from Internet: <https://www.aclweb.org/anthology/D18-2012>.

LAN, Z. et al. Albert: A lite bert for self-supervised learning of language representations. **arXiv preprint arXiv:1909.11942**, 2019.

LEWIS, M. et al. 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**. Online: Association for Computational Linguistics, 2020. p. 7871–7880. Available from Internet: <https://aclanthology.org/2020.acl-main.703>.

LI, P.; BING, L.; LAM, W. Actor-critic based training framework for abstractive summarization. **arXiv preprint arXiv:1803.11070**, 2018.

LIN, C.-Y. Rouge: A package for automatic evaluation of summaries. In: **Text summarization branches out**. [S.l.: s.n.], 2004. p. 74–81.

LING, W. et al. Two/too simple adaptations of word2vec for syntax problems. In: **Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies**. [S.l.: s.n.], 2015. p. 1299–1304.

LIU, P. J. et al. Generating wikipedia by summarizing long sequences. **arXiv preprint arXiv:1801.10198**, 2018.

LIU, Y.; LAPATA, M. Text summarization with pretrained encoders. 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)**. [S.l.: s.n.], 2019. p. 3721–3731.

LUHN, H. P. The automatic creation of literature abstracts. **IBM Journal of research and development**, v. 2, n. 2, p. 159–165, 1958.

LUONG, M.; PHAM, H.; MANNING, C. D. Effective approaches to attention-based neural machine translation. **CoRR**, abs/1508.04025, 2015. Available from Internet: <http://arxiv.org/abs/1508.04025>.

MARNEFFE, M.-C. de et al. Universal Dependencies. **Computational Linguistics**, v. 47, n. 2, p. 255–308, 07 2021. ISSN 0891-2017. Available from Internet: <https://doi.org/10.1162/coli_a_00402>.

MARTINAZZO, B.; DOSCIATTI, M. M.; PARAISO, E. C. Identifying emotions in short texts for brazilian portuguese. In: **International Workshop on Web and Text Intelligence**. [S.l.: s.n.], 2011. p. 16.

MIHALCEA, R.; TARAU, P. TextRank: Bringing order into texts. In: **Proceedings of EMNLP-04and the 2004 Conference on Empirical Methods in Natural Language Processing**. [S.l.: s.n.], 2004.

MIKOLOV, T. et al. Efficient estimation of word representations in vector space. **arXiv preprint arXiv:1301.3781**, 2013.

MOENS, M.-F.; UYTTENDAELE, C. Automatic text structuring and categorization as a first step in summarizing legal cases. **Information Processing & Management**, Elsevier, v. 33, n. 6, p. 727–737, 1997.

MONTEIRO, R. A. et al. Contributions to the study of fake news in portuguese: New corpus and automatic detection results. In: **PROPOR**. [S.l.: s.n.], 2018. p. 324–334.

MUDRAKARTA, P. K. et al. Did the model understand the question? **CoRR**, abs/1805.05492, 2018. Available from Internet: <http://arxiv.org/abs/1805.05492>.

NALLAPATI, R.; XIANG, B.; ZHOU, B. Sequence-to-sequence rnns for text summarization. **CoRR**, abs/1602.06023, 2016. Available from Internet: <http://arxiv.org/abs/1602.06023>.

NARR, S.; HULFENHAUS, M.; ALBAYRAK, S. Language-independent twitter sentiment analysis. **Knowledge discovery and machine learning**, p. 12–14, 2012.

NENKOVA, A.; MCKEOWN, K. A survey of text summarization techniques. In: **Mining text data**. [S.l.]: Springer, 2012. p. 43–76.

NENKOVA, A.; PASSONNEAU, R. J. Evaluating content selection in summarization: The pyramid method. In: **Proceedings of the human language technology conference of the north american chapter of the association for computational linguistics: Hlt-naacl 2004**. [S.l.: s.n.], 2004. p. 145–152.

NENKOVA, A.; VANDERWENDE, L. The impact of frequency on summarization. **Microsoft Research, Redmond, Washington, Tech. Rep. MSR-TR-2005**, v. 101, 2005.

PANDYA, V. Automatic text summarization of legal cases: A hybrid approach. **5th International Conference on Advances in Computer Science and Information Technology (ACSTY-2019)**, Aug 2019.

PAPINENI, K. et al. Bleu: a method for automatic evaluation of machine translation. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. **Proceedings of the 40th annual meeting on association for computational linguistics**. [S.l.], 2002. p. 311–318.

PAULUS, R.; XIONG, C.; SOCHER, R. A deep reinforced model for abstractive summarization. **arXiv preprint arXiv:1705.04304**, 2017.

PELLE, R. P. de; MOREIRA, V. P. Offensive comments in the brazilian web: a dataset and baseline results. **BraSNAM**, 2017.

PENNINGTON, J.; SOCHER, R.; MANNING, C. D. Glove: Global vectors for word representation. In: **Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)**. [S.l.: s.n.], 2014. p. 1532–1543.

PIRES, T.; SCHLINGER, E.; GARRETTE, D. How multilingual is multilingual BERT? In: **Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics**. [S.l.: s.n.], 2019. p. 4996–5001.

POLSLEY, S.; JHUNJHUNWALA, P.; HUANG, R. Casesummarizer: a system for automated summarization of legal texts. In: **Proceedings of COLING 2016, the 26th international conference on Computational Linguistics: System Demonstrations**. [S.l.: s.n.], 2016. p. 258–262.

ŘEHŮŘEK, R.; SOJKA, P. Software Framework for Topic Modelling with Large Corpora. In: **Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks**. Valletta, Malta: ELRA, 2010. p. 45–50.

ROBERTSON, S. E. et al. Okapi at trec-3. **Nist Special Publication Sp**, NATIONAL INSTIUTE OF STANDARDS & TECHNOLOGY, v. 109, p. 109, 1995.

ROCHA, G.; CARDOSO, H. L. Recognizing textual entailment: challenges in the portuguese language. **Information**, v. 9, n. 4, p. 76, 2018.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning representations by back-propagating errors. **nature**, Nature Publishing Group, v. 323, n. 6088, p. 533–536, 1986.

SCHUSTER, M.; NAKAJIMA, K. Japanese and korean voice search. In: IEEE. **2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)**. [S.l.], 2012. p. 5149–5152.

SCRAPINGHUB. **Scrapy - A fast and powerful scraping and Web Crawling Framework**. 2018. Available from Internet: <https://scrapy.org>.

SEE, A.; LIU, P. J.; MANNING, C. D. Get to the point: Summarization with pointer-generator networks. In: **Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)**. [S.l.: s.n.], 2017. p. 1073–1083.

SENNRICH, R.; HADDOW, B.; BIRCH, A. Neural machine translation of rare words with subword units. **arXiv preprint arXiv:1508.07909**, 2015.

STEINBERGER, J.; JEZEK, K. Using latent semantic analysis in text summarization and summary evaluation. **Proc. ISIM**, v. 4, p. 93–100, 2004.

SUNDARARAJAN, M.; TALY, A.; YAN, Q. Axiomatic attribution for deep networks. In: PMLR. **International Conference on Machine Learning**. [S.l.], 2017. p. 3319–3328.

SUTSKEVER, I.; VINYALS, O.; LE, Q. V. Sequence to sequence learning with neural networks. In: **Advances in neural information processing systems**. [S.l.: s.n.], 2014. p. 3104–3112.

TURTLE, H. Text retrieval in the legal world. **Artificial Intelligence and Law**, Springer, v. 3, n. 1, p. 5–54, 1995.

VASWANI, A. et al. Attention is all you need. In: **Proceedings of the 31st International Conference on Neural Information Processing Systems**. [S.l.: s.n.], 2017. p. 6000–6010.

VINYALS, O.; FORTUNATO, M.; JAITLY, N. Pointer networks. In: **Advances in neural information processing systems**. [S.l.: s.n.], 2015. p. 2692–2700.

WOLF, T. et al. Huggingface's transformers: State-of-the-art natural language processing. **arXiv preprint arXiv:1910.03771**, 2019.

WU, Y. et al. **Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation**. 2016.

YOUSFI-MONOD, M.; FARZINDAR, A.; LAPALME, G. Supervised machine learning for summarizing legal documents. In: SPRINGER. **Canadian Conference on Artificial Intelligence**. [S.l.], 2010. p. 51–62.

ZHANG, B.; XIONG, D.; SU, J. Accelerating neural transformer via an average attention network. **CoRR**, abs/1805.00631, 2018. Available from Internet: <http://arxiv.org/abs/1805.00631>.

ZHANG, H. et al. Pretraining-based natural language generation for text summarization. **arXiv preprint arXiv:1902.09243**, 2019.

ZHONG, L. et al. Automatic summarization of legal decisions using iterative masking of predictive sentences. In: **Proceedings of the Seventeenth International Conference on Artificial Intelligence and Law**. [S.l.: s.n.], 2019. (ICAIL '19), p. 163–172. ISBN 9781450367547.

APPENDIX A — RESUMO EXPANDIDO

Com o incremento de disponibilidade de dados, a sumarização de textos tornou-se uma necessidade para encontrar informações relevantes rapidamente. Na área jurídica, documentos possuem peculiaridades relacionadas ao comprimento, jargão especializado e vocabulário. Estas características tornam difícil a utilização de técnicas convencionais para geração de resumos.

Avanços recentes nas abordagens usando redes neurais obtiveram elevados resultados em termos de qualidade. Contudo, estas abordagens vêm sendo usadas majoritariamente para criação de sumários curtos e no contexto jornalístico. Portanto, a sua aplicação no domínio jurídico segue como um problema em aberto.

Para avaliar a possibilidade de geração de resumos de decisões judiciais, construímos o dataset RulingBR a partir de decisões judiciais extraídas do site do Supremo Tribunal Federal. O dataset é composto por aproximadamente 10 mil decisões judiciais de 18 ministros. Experimentamos esses dados usando técnicas extrativas, pelas quais sentenças completas são usadas para compor o sumário gerado; e técnicas abstrativas, nas quais o sumário é gerado palavra por palavra após a leitura completa da decisão judicial.

Nossos experimentos demonstraram que as técnicas extrativas produziam resumos ruins, distantes do formato esperado para um resumo jurídico. Por outro lado, as técnicas abstrativas produzem resumos muito próximos dos gerados por humanos, mas frequentemente introduzem fatos falsos em um fenômeno conhecido como halucinação.

Considerando que a forma que produziu os melhores resultados foi a abstrativa, identificamos as duas maiores dificuldades para a sua aplicação em dados jurídicos. A primeira é lidar com o documentos longos, e a segunda é confiar no conteúdo gerado pelo sumarizador automático.

A seguir, considerando que a maioria dos modelos e ferramentas de NLP são treinados e estão disponíveis apenas em inglês, realizamos o treinamento completo de um modelo BERT (DEVLIN et al., 2018) em Português. Com este modelo treinado, realizamos experimentos que demonstraram a capacidade deste modelo para avaliar se um texto pode ser inferido a partir de outro (implicação textual).

Nesta tese, propomos LegalSumm, um método de sumarização para decisões judiciais baseado em Tranformers (VASWANI et al., 2017) e implicação textual (*textual entailment*). Para lidar com o problema dos documentos longos, nossa abordagem utiliza um técnica de divisão em pedaços de texto. Nós geramos oito pedaços de texto a partir da

decisão judicial e um sumário-candidato para cada.

Nós geramos dados fictícios falsos para treinar nosso modelo a distinguir a implicação textual. Para isso, a partir de cada exemplo contendo o texto da decisão e o seu resumo, procuramos por decisões com características similares e substituímos o resumo original pelo resumo de outra decisão e marcamos o exemplo como falso. Para cada par de decisão e resumo original, criamos dez exemplos falsos. Com esses dados, conseguimos treinar o modelo a reconhecer a implicação textual entre a decisão e o seu resumo.

A seguir, avaliamos a implicação textual entre a decisão e cada um dos oito sumários gerados automaticamente. Para cada, selecionamos o candidato que obtiver a maior pontuação.

Para avaliar a efetividade de nosso método, primeiramente avaliamos se esse método produz melhores resultados do que os resultados produzidos usando o modelo Transformer sozinho. Nosso experimento demonstrou que nosso método foi capaz de escolher resumos de forma a aumentar a avaliação geral em relação a qualquer uma das estratégias usadas para gerar os resumos.

A seguir, para avaliar nosso método em relação a métodos gerais de estado da arte para geração de resumos, treinamos em Português os modelos BertSumExt, BertSumAbs (LIU; LAPATA, 2019) e BART (LEWIS et al., 2020). Nossa abordagem superou todos esses modelos.

Para avaliar a qualidade dos resumos, realizamos uma pequena avaliação com especialistas na área jurídica. No experimento, selecionamos aleatoriamente dez decisões e apresentamos o conteúdo da decisão, o resumo original, um resumo gerado por nossa abordagem e outro resumo gerado pela abordagem BertSumAbs (a abordagem que obteve a melhor avaliação entre os métodos comparados). De acordo com as opiniões de especialistas jurídicos, nosso método produziu resumos melhores e mais confiáveis.

Os resultados demonstram que LegalSumm melhora a qualidade dos resumos gerados. Contudo, como o experimento com os especialistas também demonstrou, apesar da melhora, os resumos produzidos ainda não são aptos a substituir o papel do ser humano na confecção dos resumos, mas podem ajudar no trabalho oferecendo uma proposta de resumo, reduzindo a carga de trabalho do especialista para somente realizar a revisão do resumo gerado.