UNIVERSIDADE FEDERAL DO CEARÁ

CENTRO DE CIÊNCIAS

DEPARTAMENTO DE COMPUTAÇÃO

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

FELIPE MELO SOARES

COMPRESSÃO DE SENTENÇAS SOBRE DOMÍNIOS COM DISPONIBILIDADE LIMITADA DE DADOS ROTULADOS

FORTALEZA

2019

FELIPE MELO SOARES

COMPRESSÃO DE SENTENÇAS SOBRE DOMÍNIOS COM DISPONIBILIDADE LIMITADA DE DADOS ROTULADOS

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação do Centro de Ciências da Universidade Federal do Ceará, como requisito parcial à obtenção do título de mestre em Ciência da Computação. Área de Concentração: Banco de Dados.

Orientador: Prof. Dr. José Antonio Fernandes de Macêdo.

FORTALEZA

2019

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Sistema de Bibliotecas
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

S654c Soares, Felipe Melo.
Compressão de Sentenças sobre Domínios com Disponibilidade Limitada de Dados / Felipe Melo Soares. – 2019.
77 f. : il.

Dissertação (mestrado) – Universidade Federal do Ceará, Centro de Ciências, Programa de Pós-Graduação em Ciência da Computação, Fortaleza, 2019.
Orientação: Prof. Dr. José Antonio Fernandes de Macêdo.

1. Compressão de Sentenças. 2. Processamento de Linguagem Natural. 3. Aprendizagem Profunda. I. Título.

CDD 005

FELIPE MELO SOARES

COMPRESSÃO DE SENTENÇAS SOBRE DOMÍNIOS COM DISPONIBILIDADE LIMITADA DE DADOS ROTULADOS

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação do Centro de Ciências da Universidade Federal do Ceará, como requisito parcial à obtenção do título de mestre em Ciência da Computação. Área de Concentração: Banco de Dados.

Aprovada em: 27 de Novembro de 2019

BANCA EXAMINADORA

Prof. Dr. José Antonio Fernandes de Macêdo (Orientador)
Universidade Federal do Ceará (UFC)

Prof. Dr. Ticiana Linhares Coelho da Silva
Universidade Federal do Ceará (UFC)

Prof. Dr. Vládia Célia Monteiro Pinheiro
Universidade de Fortaleza (UNIFOR)

Prof. Dr. João Paulo Pordeus Gomes
Universidade Federal do Ceará (UFC)

Aos meus pais, Lucimar e Soares, por acreditarem em mim desde sempre e não medirem esforços, mesmo nas dificuldades, para me ajudarem a chegar até aqui. Ao meu irmão, Gabriel, por ter sido sempre o meu melhor amigo. E a minha esposa, Janiele, pelo amor, compreensão e dedicação e por estar sempre ao meu lado.

AGRADECIMENTOS

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001

Agradeço à Deus antes de tudo, pelo dom da vida e por estar presente mesmo nos dias mais difíceis.

Agradeço a minha família, por todo carinho, amor e cuidado, em especial aos meus pais por terem sempre lutado a vida toda para me darem a oportunidade de crescer e alcançar os meus objetivos.

Agradeço a minha esposa, Janiele, pela paciência infinita, pela compreensão, pelo apoio dia após dia, e por não me deixar desistir nunca.

Agradeço ao meu orientador, José Antônio Macedo, por ter acreditado em mim, pelas oportunidades que me ofereceu e por ser a mente por trás de tantas pesquisas e aplicações tão importantes para a sociedade.

Agradeço a minha coorientadora, Ticiana Linhares, por toda a ajuda que me deu, pelo apoio e comprometimento e por ter acreditado que eu seria capaz.

Agradeço aos meus companheiros de trabalho, pelo valores de excelência compartilhados, por tudo que eles permitiram que eu aprendesse e pelo dia a dia leve e descontraído que tornou essa jornada mais agradável.

Agradeço ao grupo Insight Data Science Lab e a todos os seus membros, por terem me acolhido no início dessa caminhada, por todas as experiências compartilhadas e pela dedicação. Um agradecimento especial ao gerente de projetos do grupo, David Araújo, por entender a situação de cada pessoa que gerencia, por se preocupar com o bem estar do grupo e pela relação de confiança.

Agradeço à Universidade Federal do Ceará, em particular ao programa de Mestrado e Doutorado em Ciências da Computação, pela excelência de seus professores e servidores e por possibilitar uma oportunidade tão singular.

Por fim, ao Doutorando em Engenharia Elétrica, Ednardo Moreira Rodrigues, e seu assistente, Alan Batista de Oliveira, aluno de graduação em Engenharia Elétrica, pela adequação do *template* utilizado neste trabalho para que o mesmo ficasse de acordo com as normas da biblioteca da Universidade Federal do Ceará (UFC).

“Existem coisas melhores adiante do que qualquer outra que deixamos para trás.”
(C. S. Lewis)

RESUMO

Dia após dia, volumes gigantescos de dados são produzidos na Web. São grandes quantidades de vídeos, imagens e textos que armazenam informação de maneira não estruturada. Sistemas de sumarização de textos foram criados com o intuito facilitar a apresentação de grandes quantidades de dados textuais para usuários assim como para facilitar a recuperação de informações nesses tipos de dados. A tarefa de compressão de sentenças surgiu para melhorar a qualidade dos sumários gerados por esses sistemas. No entanto, em domínios com pouca disponibilidade de dados rotulados para a compressão de sentenças, modelos baseados em redes neurais tem grandes dificuldades de extrair todas as informações que eles precisam. Para melhorar a performance destes modelos neste cenário, é possível que parte da informação das palavras seja extraída e adaptada antes de ser utilizada para o treinamento. Portanto, este trabalho propõe um método de compressão de sentenças capaz de atingir resultados competitivos mesmo utilizando quantidades de dados inferiores ao normalmente utilizado na literatura através da utilização de um conjunto de atributos linguísticos extraídos das palavras, aliada a uma estratégia de redução de palavras raras aplicada sobre as sentenças.

Palavras-chave: compressão de sentenças; sumarização automática de textos; processamento de linguagem natural; redes neurais recorrentes; rotulação de sequências.

ABSTRACT

Huge volumes of data are produced every day on the Web. These are a big amount of videos, images, and texts that store unstructured information. Text summarization systems were created to facilitate the presentations of large amounts of textual data as well as to aid information retrieval over this type of data. The sentence compression has been developed due to the need for better summaries generated by these systems. However, when trained over domains with restricted amounts of labeled data for sentence compression, neural netword-based models tend to not be able to extract important features. Thus, to improve the performance of these models in this scenario, some pieces of information must be extracted and adapted before being used for training. Thus, we propose a sentence compression model capable of achieving competitive results, even when trained with smaller amounts of data, compared with other neural network-based models, by using a set of linguistic features extracted from words alongside a rare words reduction strategy over the sentences.

Keywords: sentence compression; automatic text summarization; natural language processing; recurrent neural networks; sequence labeling.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

LSTM	*Long Short-term Memory*
RNN	*Recurrent Neural Networks*
ReLU	*Rectified Linear Unit*
NMT	Neural Machine Translation
PLI	Programação Linear Inteira
PLN	Processamento de Linguagem Natural

SUMÁRIO

1 INTRODUÇÃO

Com a popularização da internet, houve um massivo crescimento na quantidade de dados não estruturados disponíveis para os usuários dessa rede. Dentre as várias formas como esses dados podem ser encontrados, uma das mais difíceis de se extrair informação relevante são aqueles que se apresentam como textos; documentos como artigos, notícias, postagens de *blogs*, *reviews* de produtos, tópicos de discussão em fóruns, entre outros. Dada toda essa quantidade de conteúdo textual continuamente produzido na *Web*, tem sido um desafio para seus usuários consumir este conteúdo em toda a sua integridade. Assim, sistemas de sumarização de texto têm um papel muito importante para esse domínio.

Sistemas de sumarização automática de texto normalmente produzem sumários extraindo as sentenças mais relevantes do documento original (GAMBHIR; GUPTA, 2017). Grande parte deles é focada em duas etapas para construir um sumário.

1. Eles atribuem uma nota para cada uma das sentenças do documento original de acordo com sua importância. Essa nota de importância, ou saliência, mede a quantidade de informação relevante que cada sentença agrega ao documento e pode ser penalizada pelo quanto ela aumenta o tamanho do sumário ou quanta informação irrelevante ela também carrega.
2. Dentre as sentenças com maiores saliências, os sistemas selecionam aquelas que realmente irão compor o sumário. Essas sentenças são selecionadas com o objetivo de minimizar a redundância do sumário ao mesmo tempo que tenta atingir uma maior abrangência de informações diferentes.

Ainda que muito comum, essa abordagem não leva em conta que, quando os documentos originais foram redigidos, não havia qualquer restrição de concisão para o tamanho das sentenças, gerando frequentemente sentenças longas que armazenam não apenas informações relevantes, mas também muita informação acessória, isto é, que apesar de importante para a própria sentença, pode não ser importante para o documento todo. Por exemplo uma sentença longa como *"Myanmar opposition leader Aung San Suu Kyi will visit Australia next week, with meetings scheduled with the prime minister and foreign minister, her party said on Wednesday."* cuja informação mais importante poderia se resumir a "*Aung San Suu Kyi will visit Australia.".*

Um sumário é um documento textual de tamanho restrito, assim essas sentenças não são apropriadas para fazerem parte de um, pois apesar de adicionarem-lhe um conteúdo valioso, elas também ocupam muito espaço, impedindo que outras sentenças que contenham

outras informações relevantes diferentes também enriqueçam seu conteúdo. Assim, é comum que sistemas de sumarização automática apenas descartem sentenças assim, ainda que elas carreguem informações essenciais para o documento original. Uma maneira muito conveniente de lidar com esse problema é utilizando um terceiro passo para reduzir o tamanho das sentenças do documento original com o objetivo de extrair delas apenas suas partes mais relevantes (ANTUNES, 2018). Esse passo configura uma tarefa de redução ou compressão de sentenças.

A compressão de sentenças é uma tarefa de Processamento de Linguagem Natural (PLN) que tem como objetivo reduzir o tamanho de uma sentença removendo delas informações pouco relevantes sem que elas deixem de ser gramaticalmente corretas e mantendo seu sentido original. Por exemplo a sentença *"A man was seriously injured while trying to catch the trolley in the College East area."* pode ser reduzida à sentença *"A man was seriously injured while trying to catch the trolley."*. Muitos métodos de executar essa tarefa de maneira automática foram propostos aos longos dos anos. Os primeiros deles desenvolveram modelos estatísticos capazes de calcular a probabilidade de uma certa configuração de palavras formarem a melhor compressão para uma dada sentença (KNIGHT; MARCU, 2000; MCDONALD, 2006). Outros trabalhos propuseram realizar a compressão através de operações de reescrita utilizando regras bem definidas (COHN; LAPATA, 2007; COHN; LAPATA, 2008). Alguns dos métodos mais recentes modelaram o problema como um problema de otimização linear que deveria ser capaz de encontrar uma versão comprimida da sentença original dada uma série de restrições (CLARKE; LAPATA, 2008; FILIPPOVA; STRUBE, 2008).

Com o recente crescimento da disponibilidade de poder computacional e com a popularização das redes neurais profundas, métodos que usam diferentes variações de redes neurais recorrentes tem contribuído com avanços notáveis para o estado da arte de compressão de sentenças. No entanto, esses modelos de redes neurais ainda sofrem com algumas limitações.

Primeiramente, uma grande quantidade de dados rotulados, isto é, conjuntos de sentenças originais com suas respectivas versões comprimidas, é necessária para treinar esses modelos. O trabalho de (FILIPPOVA *et al.*, 2015) utiliza uma base de dados composta por 2 milhões de pares de sentenças gerados sobre um conjunto de notícias utilizando o método proposto por (FILIPPOVA; ALTUN, 2013) e não disponibilizada publicamente. Os trabalhos de (RUSH *et al.*, 2015; CHOPRA *et al.*, 2016; ZHOU *et al.*, 2017; FU *et al.*, 2018), por sua vez, utilizam uma base de dados de aproximadamente 4 milhões de sentenças extraídas da base de dados do *english gigaword*, uma base fechada e de difícil obtenção. Conseguir essa quantidade

de dados pode ser uma tarefa árdua ou até mesmo impossível para alguns domínios.

Outro ponto importante é que, uma vez que esses modelos são treinados em um domínio específico, especialmente quando utilizando quantidades inferiores de dados, eles tendem a absorver certos aspectos desse domínio, como construções de sentenças ou níveis de compressão específicos, que pode acabar por fazer com que esses modelos não atinjam a mesma performance quando utilizados com sentenças de um domínio no qual ele não foi treinado, como concluiu (WANG *et al.*, 2017).

Modelos baseados em redes neurais para PLN precisam dessa grande quantidade de dados porque os trechos de informações textuais possuem alta interdependências com relações de difícil mapeamento. Quando o único atributo utilizado nesses modelos é a própria palavra, essa necessidade se torna ainda mais aparente, pois uma vez que eles aprendem uma tarefa através da co-ocorrência de unidades textuais dos dados de entrada, a falta de dado significa que não haverá exemplos o suficiente para que eles sejam capazes de generalizar de fato as tarefas para as quais eles deveriam ser treinados.

Um fator agravante para essa situação está, ainda, no tamanho do vocabulário e na ocorrência de palavras raras. Segundo a Lei de Zipf, em um corpus de linguagem natural, a palavra mais frequente ocorre aproximadamente duas vezes mais que a segunda palavra mais frequente, três vezes mais que a terceira e assim por diante (POWERS, 1998). Deste modo, é comum que, quão maiores forem os vocabulários de um corpus mais estes modelos terão que lidar com uma grande quantidade de palavras que ocorrem pouquíssimas vezes, e que portanto são pouco conclusivas sobre suas importâncias na composição de uma sentença.

Tendo tudo isso em vista, essa dissertação aborda o problema de compressão de sentenças utilizando métodos baseados em redes neurais, levando em conta, porém, uma restrição na quantidade de dados disponível para o treinamento de modelos.

Tendo isso em vista, essa dissertação aborda o problema de tentar melhorar a compressão de sentenças utilizando métodos baseados em redes neurais para domínios que possuem quantidades restritas de dados disponíveis para treinamento.

1.1 Objetivos

Dado este cenário, onde deseja-se comprimir sentenças, que potencialmente possuam informações descartáveis, de um domínio com pouca disponibilidade de dados para treinamento de modelos, esse trabalho tem como objetivo propor uma metodologia de compressão de

sentenças utilizando redes neurais que seja capaz de obter resultados competitivos com o estado da arte mesmo quando utilizando quantidades de dados muito inferiores às normalmente utilizadas por modelos baseados em redes neurais para esse propósito:

- Elaborar uma etapa de pré-processamento dos dados com o objetivo de reduzir o vocabulário de um conjunto de textos minimizando assim a ocorrência de palavras raras e facilitando o treinamento de modelos baseados em redes neurais;
- Construir um modelo de compressão de sentenças baseado em redes neurais que seja competitivo com o estado da arte quando treinado com quantidades inferiores de dados.

1.2 Contribuições

Como resultado dessa dissertação um modelo de compressão de sentenças baseado em redes neurais proposto em (WANG *et al.*, 2017) foi estendido ao adicionar-se a ele um módulo capaz de codificar toda uma sentença em uma representação vetorial para ser utilizada como mais um dos atributo de entrada de modelo principal, um rotulador de sequências responsável por classificar se cada palavra de uma sentença deve ser mantida ou removida para formar sua versão comprimida. Além disso, foi elaborada uma etapa de pré-processamento dos dados que tem como objetivo reduzir a ocorrência de palavras raras nas sentenças utilizadas através da substituição dessas palavras por outras mais comuns e de semântica similar, a fim de facilitar o treinamento do modelo baseado em redes neurais.

1.3 Organização do Trabalho

Esta dissertação está organizada da seguinte forma: O Capítulo 2 apresenta conceitos e definições importantes sobre a tarefa de compressão de sentenças. Além disso, nele também são apresentadas algumas técnicas de processamento de linguagem natural envolvidas na solução proposta assim como o problema de rotulação de sequências utilizando redes neurais. O Capítulo 3 apresenta um conjunto de trabalhos relacionados que, assim como esta dissertação, propõem soluções para a tarefa de compressão de sentenças extrativa. O Capítulo 4 apresenta a solução proposta para a compressão de sentenças em domínios com pouca disponibilidade de dados rotulados. O Capítulo 5 apresenta os experimentos, e seus respectivos resultados, realizados para comparar a qualidade das compressões realizadas pela solução proposta com outro modelo que

assim como este lida com compressão de sentenças com pouca quantidade de dados, na tentativa de verificar o quão próximos das compressões de referência cada modelo foi capaz de chegar. Por fim, o Capítulo 6 conclui este trabalho resumindo os resultados obtidos e apresentando futuros passos para o avanço desta linha de pesquisa.

2 FUNDAMENTAÇÃO TEÓRICA

Sistemas de sumarização automática de textos tem como objetivo capturar de maneira resumida a maior quantidade possível de informações de um documento ou de múltiplos documentos (TAS; KIYANI, 2007). Esses sistemas podem ser classificados em duas categorias, de acordo como tipo de sumário que eles geram: abstrativos e extrativos. Em sistemas abstrativos, o sumário gerado é obtido através da geração de linguagem natural a partir do texto original através de técnicas de reescrita de texto ou abordagens generativas. Sistemas extrativos, por sua vez, geram o sumário a partir do agrupamento de fragmentos do texto original (expressões, sentenças ou parágrafos). Na maioria dos sistemas extrativos, adota-se a sentença como unidade básica de construção do sumário. Desta forma, sistemas extrativos são mais simples que os abstrativos, contudo podem sofrer com sentenças longas, que podem carregar grandes quantidades de dados irrelevantes para o sumário final (TAS; KIYANI, 2007).

Sistemas de sumarização extrativos são em geral compostos por duas etapas principais: 1) ranqueamento de sentenças e 2) seleção de sentenças. Durante o ranqueamento, as sentenças são analisadas quanto a quantidade de informação importante que ela carrega e lhes são atribuídos escores de saliência, que indicam o quão importante cada sentença pode ser para o sumário. A seleção de sentenças então, tem como objetivo escolher as sentenças mais adequadas para compor o sumário, levando em conta não apenas a quantidade de informação relevantes, mas também a quantidade de informação possivelmente redundante que cada sentença agregará ao sumário final (TAS; KIYANI, 2007).

Apesar de imitar o comportamento humano ao tentar selecionar sentenças importantes para compor o sumário, esses sistemas extrativos deixam de lado uma etapa importante muito comum no processo de sumarização como ele é realizado por pessoas. Ao invés de simplesmente extrair sentenças de um texto e combinar umas as outras, seres humanos geralmente as modificam antes de colocá-las no sumário, a fim de garantir que o sumário se mantenha conciso e coerente (JING, 2000).

Uma série de operações de edição podem ser aplicadas para fazer essa mudança nas sentenças, no entanto o objetivo é sempre o mesmo: reduzir o tamanho da sentença para que ela se torne mais adequada para um sumário. A essa tarefa de reduzir o tamanho de uma sentença dá-se o nome de compressão, redução ou sumarização de sentença.

2.1 Compressão de Sentenças

Os termos compressão, redução e sumarização de sentenças vem sendo usado desde o início dos anos 2000 para descrever a tarefa de reduzir o tamanho de uma sentença, seja removendo partes dessa sentença seja reescrevendo-a do zero, com o objetivo de obter-se outra sentença, menor, que conservasse a maior quantidade possível de informação da original, em especial sem que sua informação central fosse perdida, e que ela se mantivesse gramaticalmente correta. Por exemplo, a sentença *"Sportswear company Adidas has suspended its contract with Tyson Gay after the former double world sprint champion failed an out-of-competition dope test."* tem como informação principal a suspensão do contrato entre a companhia *Adidas* e *Tyson Gay* e poderia ser reescrita como *"Adidas suspended its contract with Tyson Gay."*. A segunda seria portanto uma compressão das informações da primeira.

A tarefa de compressão de sentenças pode ser vista também como uma especialização de sumarização automática de textos que trabalha a nível de sentença. Assim como a sumarização de textos, a compressão de sentenças pode ser dividida em duas categorias quanto ao caráter da compressão: *compressão de sentenças extrativa* e *compressão de sentenças abstrativa*.

2.1.1 Compressão de Sentenças Extrativa

Também conhecida por compressão de sentenças por remoção de palavras, esse é o tipo de compressão predominante na literatura (KNIGHT; MARCU, 2000; MCDONALD, 2006; COHN; LAPATA, 2007; COHN; LAPATA, 2008; CLARKE; LAPATA, 2008; FILIPPOVA; STRUBE, 2008; GALANIS; ANDROUTSOPOULOS, 2010; BERG-KIRKPATRICK *et al.*, 2011; FILIPPOVA; ALTUN, 2013; FILIPPOVA *et al.*, 2015; WANG *et al.*, 2017), por ser uma problema mais simples de resolver que a compressão abstrativa e que garante bons resultados. Por muitas vezes esses trabalhos acabam por definir a compressão de sentenças por remoção de palavras como sinônimo de compressão de sentenças em si, quando na verdade se trata de uma subcategoria dela.

Na compressão de sentenças extrativa, o objetivo é reduzir o tamanho de uma sentença utilizando apenas operações de remoção de palavras, isto é, a compressão é feita apenas removendo palavras da sentença original, sem alterar sua ordem ou adicionar novas palavras. Deste modo, as sentenças comprimidas através de um método extrativo não possuem palavras que não ocorram nas sentenças originais, assim as palavras das sentenças comprimidas são

sempre subconjuntos das palavras da original. Além disso, a ordem das palavras das sentenças se mantém inalterada. Por exemplo, a sentença *"The American Samoa Governor Lolo Matalasi Moliga has approved two tax exemption certificates since taking office in January."* poderia ser comprimida ao se removerem as palavras *since*, *taking*, *office*, *in* e *January* do final da frase, sobrando então a sentença: *"The American Samoa Governor Lolo Matalasi Moliga has approved two tax exemption certificates."*

Ainda que a compressão de sentenças extrativa se limite a apenas remover palavras da sentença original, o que restringe bastante as possibilidades de compressão, em um estudo feito em (COHN; LAPATA, 2008) concluiu-se que aproximadamente 70% das operações de edição feitas por uma pessoa ao comprimir uma sentença são operações de remoção. Portanto, os resultados desse tipo de compressão não são assim tão distantes da realidade.

2.1.2 Compressão de Sentenças Abstrativa

A compressão de sentenças abstrativa é comumente conhecida na literatura apenas como sumarização de sentenças (RUSH *et al.*, 2015; CHOPRA *et al.*, 2016; ZHOU *et al.*, 2017; FU *et al.*, 2018). Não existe um trabalho específico que defina o termo sumarização de sentenças como um outro nome para compressão de sentenças abstrativa, contudo a definição comum de sumarização de sentenças é exatamente a mesma definição de compressão de sentença sem a limitação de utilização de operações de remoção de palavras. Assim, a compressão de sentenças abstrativa pode ser definida como a categoria de compressão de sentenças que não tem como restrição a utilização de operações de remoção de palavras apenas.

Além de remover palavras, um ser humano, ao comprimir ou reduzir uma sentença, também faz uso de outras operações de edição. As mais comuns ainda é operação de remoção, seguida pelas substituições, inserções de palavras e reordenações de frases (COHN; LAPATA, 2008). No entanto, empregar essas outras operações é uma tarefa potencialmente mais complexa que a remoção, uma vez que as relações entre as palavras de um texto são extremamente intrincadas e difíceis de mapear, além de serem dinâmicas e dependentes de contexto, e essas operações alterarem mais profundamente a estrutura da sentença que a simples remoção de palavras.

Devido a essas dificuldades, houve poucos trabalhos que abordassem essa categoria nos primeiros anos da área e só recentemente com o advento das redes neurais profundas eles começaram a ter um avanço mais significativo (RUSH *et al.*, 2015; CHOPRA *et al.*, 2016), uma

vez que essas redes conseguem mapear por si próprias essas relações entre as palavras, quando submetidas a um conjunto de dados de treino suficientemente abrangente.

2.2 Rotulação de Sequências com Redes Neurais Recorrentes

A rotulação de sequência é um tipo de tarefa de aprendizagem de máquina que tem o objetivo de, dada uma sequência de elementos observáveis, atribuir um rótulo que descreve uma categoria, de um conjunto finito de categorias, a cada elemento dessa sequência. Apesar de ser possível tratar a rotulação de sequência como uma série de classificações individuais de cada elemento de uma sequência, pode haver dependências entre as classificações desses elementos que seriam perdidas, e que portanto prejudicariam a qualidade da rotulação da sequência como um todo. Por causa dessa dependência na classificação dos elementos de uma sequência, muitos algoritmos assumem que a sequência de rótulos de uma sequência de elementos obedecem a propriedade de Markov, isto é, eles assumem que a decisão do rótulo de um elemento depende apenas da decisão do rótulo do elemento anterior, assim essa sequência de rótulos formariam um processo de Markov. Os algoritmos mais utilizados para resolver esse problema são os modelos ocultos de Markov, os campos aleatórios condicionais, as máquinas de vetor de suporte estruturadas e, mais recentemente, as redes neurais recorrentes.

Uma vez que, textos escritos por humanos (linguagem natural) são essencialmente sequências de unidades linguísticas, que podem ser letras, palavras, sentenças, etc., e que possuem um intrincado conjunto de dependências entre si, é bastante comum encontrar tarefas de processamento de linguagem natural relacionadas à rotulação de sequências. Alguns exemplos são a marcação de classe gramatical, que tem por objetivo atribuir um rótulo de classe gramatical para cada palavra de uma sentença, ou o reconhecimento de entidades nomeadas, que busca encontrar palavras que identificam entidades nomeadas e classificá-las em categorias diferentes atribuindo-lhes rótulos específicos para cada uma dessas categorias. A compressão de sentenças extrativa ou por remoção de palavra é uma terceira possibilidade de tarefa que pode ser interpretada como uma rotulação de sequências, na qual o objetivo é classificar cada palavra da sentença com um de dois rótulos: mantida ou removida. Assim, todas as palavras rotuladas como removidas são eliminadas da sentença e a sequência de palavras restantes formam a sentença comprimida.

Com o avanço da pesquisa em redes neurais profundas e o aumento da disponibilidade de poder computacional, as redes neurais recorrentes têm sido amplamente utilizadas para

a rotulação de sequências e tem avançado significativamente o estado da arte. Por exemplo, o recente trabalho (PANCHENDRARAJAN; AMARESAN, 2018) utiliza redes neurais recorrentes combinado com campos aleatórios condicionais para fazer reconhecimento de entidades nomeadas. O trabalho de (FILIPPOVA *et al.*, 2015) é o primeiro que se saiba a modelar o problema de compressão de sentenças como um problema de rotulação de sequências e tentar resolvê-lo utilizando redes neurais recorrentes.

2.2.1 Redes Neurais Recorrentes

Seres humanos ao consumirem praticamente qualquer informação não esquecem sempre a última coisa que processaram antes de absorverem a próxima. A próxima informação obtida é sempre dependente das informações obtidas anteriormente e armazenadas na memória. Redes neurais convencionais não são capazes de imitar esse comportamento. Elas simplesmente recebem um conjunto de informações e retornam uma saída para ela, sem levar em conta qualquer processamento anterior. Redes neurais recorrentes ou *Recurrent Neural Networks* (*RNN*), por sua vez, são um tipo específico de rede neural artificial desenvolvidas para lidar com dados sequenciais. Diferentemente, das redes neurais convencionais, as *RNN* possuem uma espécie de memória que as permite processar entradas ao longo de uma sequência temporal, isto é, essas redes conseguem processar informações que dependem de informações anteriormente processadas.

Elas reproduzem esse comportamento ao se retroalimentarem com seus estados ocultos a cada passo de uma sequência. A Figura 1 mostra uma rede neural recorrente que recebe como entrada um vetor x_t e retorna um vetor de saída h_t, muito semelhante a uma rede neural convencional, contudo ela possui ainda um ciclo que sai dela e volta para ela representando o estado oculto, ou sua memória, que a retroalimenta. A Figura 2 mostra uma parte dessa mesma rede recorrente desdobrada em uma sequência. Nessa figura é possível ver como o vetor h_t é emitido pela rede no momento t e volta a alimentá-la no momento $t+1$.

Formalmente, uma rede neural recorrente pode ser descrita na seguinte forma:

$$h_t = \phi(W \cdot x_t + U \cdot h_{t-1}) \tag{2.1}$$

onde, para um dado passo t, x_t representa a entrada da rede, h_t representa o estado oculto da rede no passo t, W e U são matrizes de pesos utilizadas respectivamente para calcular

Figura 1 – Diagrama representando uma rede neural recorrente.

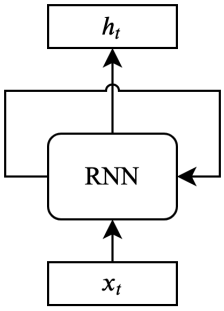
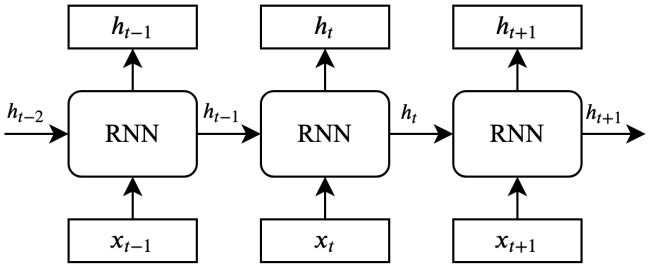


Figura 2 – Diagrama representando uma rede neural recorrente desdobrada ao longo de uma sequência.



a influência da entrada x_t e do estado oculto do passo anterior h_{t-1} para a saída da rede e ϕ é a função de ativação.

As redes neurais recorrentes podem trabalhar de duas maneiras principais. Retornando um resultado para cada um dos elementos de uma sequência, como é possível ver na Figura 3, ou retornando um único resultado no final do processamento da sequência completa, como é possível ver na Figura 4. Para a tarefa de rotulação de sequências, a primeira abordagem é bastante conveniente, uma vez que esse tipo de rede obedece a propriedade de Markov, ao depender apenas do estado do passo anterior e de uma observação para determinar o próximo estado (ou predição).

Figura 3 – Rede neural recorrente classificando cada elemento de uma sequência.

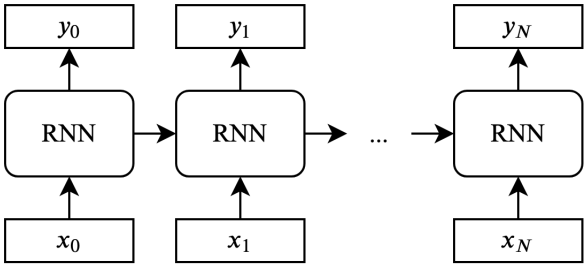
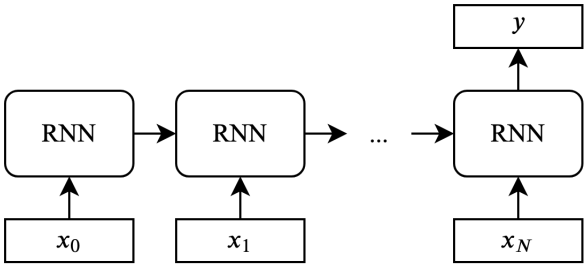


Figura 4 – Rede neural recorrente classificando uma sequência completa.



2.2.2 Retro-propagação ao Longo do Tempo

Como as redes neurais recorrentes possuem uma dependência temporal que não existe nas redes neurais convencionais, o algoritmo de retropropagação (ou backpropagation), utilizado para atualizar os pesos das matrizes internar das redes durante a fase de treinamento, tem que ser ligeiramente modificado, para levar em conta a rede desdobrada ao longo do tempo. Essa versão modificada da retropropagação é chamada de retropropagação ao longo do tempo ou, no inglês, *backpropagation through time*.

O primeiro passo do algoritmo da retropropagação ao longo do tempo é desdobrar a rede recorrente ao longo da sequência, a fim de montar uma estrutura semelhante a uma rede neural convencional multicamadas, com a particularidade de que os pesos das camadas são compartilhados entre si, por se tratarem na verdade, da mesma camada reproduzida ao longo dos passos da sequência. Figura 5 ilustra essa etapa do algoritmo. Os vetores E_t representam o erro ou o custo das predições da rede, calculados para cada passo t.

Então para cada passo t, o algoritmo da retropropagação é utilizado para calcular os gradientes de erro em relação a todos os parâmetros na rede. A Figura 6 ilustra como os gradientes são propagados pela rede desdobrada para um dado passo t.

Figura 5 – Rede neural recorrente desdobrada ao longo do tempo para a propagação dos gradientes.

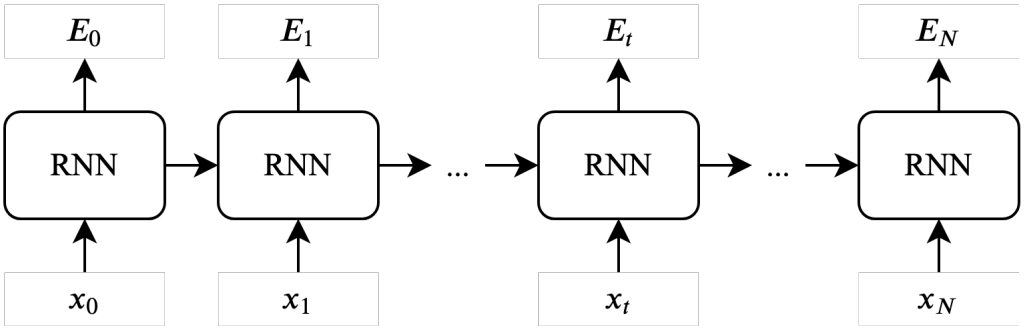
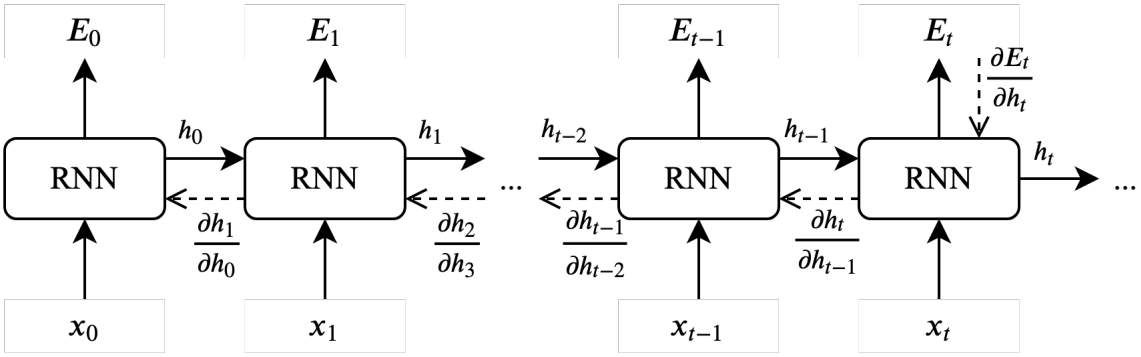


Figura 6 – Gradientes de erro sendo retro-propagados por uma rede neural recorrente desdobrada ao longo do tempo em um passo t.



2.2.3 O Problema da Explosão e Dissipação de Gradientes

Ao calcular-se o gradiente do erro em relação a matriz de pesos W para uma sequência longa é feita uma grande quantidade de multiplicações para obter-se a contribuição dos passos mais distantes do começo da sequência. Essa série de multiplicações em cadeia pode acabar por gerarem um problema para o treinamento de redes neurais recorrentes em sequências mais longas.

Quando os valores desses gradientes calculados para cada passo são maiores que um, as sucessivas multiplicações podem fazer com que o valor final do gradiente cresça exponencialmente, criando uma contribuição excessiva na atualização dos pesos da rede, ao passo que, quando esses valores são menores que um, eles podem fazer com que as multiplicações gerem gradientes cada vez mais próximos de zero, até chegar um ponto em que a contribuição de um passo t para o treinamento seja insignificante, impedindo assim que a rede aprenda dependências entre elementos muito distantes uns dos outros na sequência. Esses são, respectivamente, os problemas de explosão e dissipação de gradientes.

Quanto ao problema da explosão de gradientes, não é tão difícil de encontrar uma solução eficiente. O trabalho (PASCANU *et al.*, 2012) propõe uma truncamento no gradiente quando ele atingir um certo limiar preestabelecido. Essa é uma solução simples e efetiva para resolver a explosão de gradiente, mas o mesmo não se aplica para o problema da dissipação de gradiente.

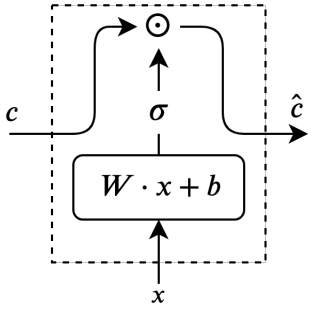
Uma forma de mitigar esse problema é a utilização da função *Rectified Linear Unit* (*ReLU*) como função de ativação ao invés da tangente hiperbólica ou da função sigmoide. No entanto, uma solução mais popular é a utilização de arquiteturas de redes de *LSTM* (HOCHREITER; SCHMIDHUBER, 1997).

2.2.4 Long Short-Term Memory (LSTM)

Redes de Memória de Longo Prazo, ou *Long Short-Term Memory* (*LSTM*) são variações de redes neurais recorrentes desenvolvidas para lidar com o problema que as versões mais simples dessas redes tinham para entender dependências longas entre elementos de uma sequência (HOCHREITER; SCHMIDHUBER, 1997). A ideia por trás da *LSTM* está na introdução de um vetor C_t que armazena o estado de uma célula de *LSTM* para um dado passo t e pelo qual é possível propagar o erro de forma constante entre dois passos da rede, evitando assim os problemas de explosão e dissipação de gradiente. Além disso, ela introduz outras três estruturas utilizadas para controlar como a informação flui através do estado da célula: os portões de esquecimento, de entrada e de saída.

Esses portões são estruturas simples que controlam o estado da célula decidindo o que deve ser removido ou adicionado do mesmo através de pequenas interações lineares. São compostos basicamente de uma rede neural convencional com uma função sigmoide como função de ativação e uma multiplicação elemento a elemento. Como a função sigmoide retorna valores entre zero e um, essa estrutura tem o poder de controlar quanta informação deve passar através do estado da célula. A Figura 7 ilustra a estrutura de um portão.

Figura 7 – Detalhes de uma estrutura de portão utilizada em células de *LSTM*.



2.2.4.1 Portão de Esquecimento

O portão de esquecimento é o primeiro portão a ser ativado durante a execução de um passo de uma *LSTM*. Ele tem a função de decidir quais informações do estado da célula devem ser mantidas durante esse passo. Ele recebe como entrada o estado oculto da célula no passo anterior h_{t-1} e a entrada do passo atual x_t e retorna um vetor f_t com valores entre zero e um com a mesma dimensão do estado da célula C_t. Quanto mais próximo de um for o valor de

uma dada posição desse vetor, mais informação será mantida nessa mesma posição do estado da célula e quanto mais próximo de zero, mais essa informação será esquecida.

2.2.4.2 Portão de Entrada

Após o portão de esquecimento, o próximo passo é decidir que novas informações são relevantes para serem adicionadas ao estado da célula. Para isso, a entrada do passo atual x_t e o estado oculto do passo anterior h_{t-1} são passados por um outro portão chamado portão de entrada gerando um vetor i_t. Essa estrutura tem o papel de ponderar quanta informação da entrada do passo atual deve ser persistida no estado da célula. Para codificar essas informações de entrada em um vetor candidato $\tilde{C}_t$, x_t e h_{t-1} são passados por uma rede neural ativada com uma função de tangente hiperbólica. Então os vetores $\tilde{C}_t$ e i_t são multiplicados, elemento a elemento, e o resultado é somado ao estado atual da célula.

2.2.4.3 Portão de Saída

Por fim, o portão de saída é o responsável por determinar que informações do estado da célula serão utilizadas na saída da célula de *LSTM*. Para isso, o estado da célula é projetado por uma função de tangente hiperbólica, e o resultado é multiplicado, elemento a elemento, pelo vetor de saída o_t do portão de saída, uma rede neural ativada pela função sigmoide, tendo x_t e h_{t-1} como entrada. O estado da célula nesse ponto contém toda a informação que não foi esquecida no portão de esquecimento somada a toda a informação adicionada pelas informações do passo atual pelo portão de entrada. O vetor o_t então tem a função de controlar quanto dessas informações devem sair da célula e passar para o passo $t+1$.

A Figura 8 ilustra a arquitetura e as Equações 2.2, 2.3, 2.4, 2.5, 2.6 e 2.7 descrevem a execução de uma célula de *LSTM*.

$$h_t = o_t \odot tanh(C_t) \tag{2.2}$$

$$o_t = \sigma(W_o \cdot (h_{t-1} \oplus x_t) + b_o) \tag{2.3}$$

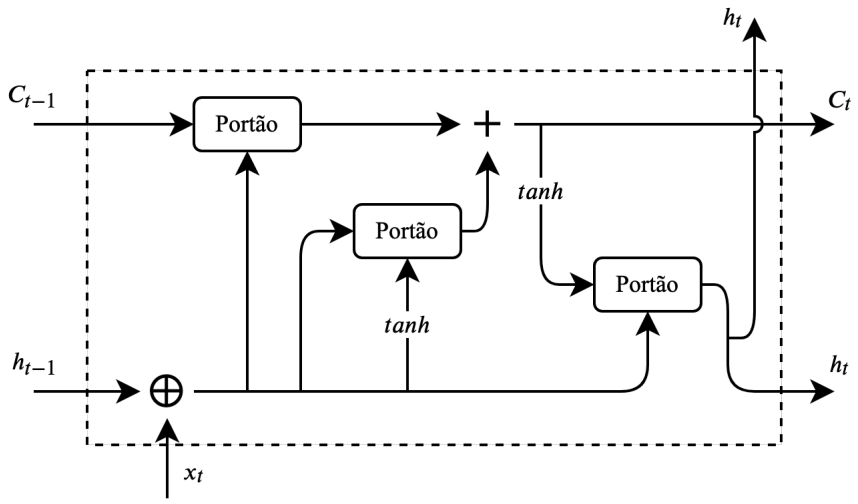
$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \tag{2.4}$$

$$\tilde{C}_t = tanh(W_C \cdot (h_{t-1} \oplus x_t) + b_C) \tag{2.5}$$

$$i_t = \sigma(W_i \cdot (h_{t-1} \oplus x_t) + b_i) \tag{2.6}$$

$$f_t = \sigma(W_f \cdot (h_{t-1} \oplus x_t) + b_f) \tag{2.7}$$

Figura 8 – Célula de *LSTM* detalhada.



onde $\odot$ representa um multiplicação elemento a elemento entre dois vetores e $\oplus$ representa uma operação de concatenação entre dois vetores.

2.2.5 LSTM Bidirecional (BiLSTM)

Redes *LSTM* lidam muito bem com o problema de dependências distantes entre elementos de uma sequência, mas ela ainda não consegue lidar com dependências passadas e futuras ao mesmo tempo, isto é, quando um ou mais elementos pode depender tanto de elementos que vieram antes na sequência como de elementos que virão depois. Em linguagem natural, esse cenário é bastante comum. Por exemplo, na sentença *"It was pretty easy to do that."*, apenas com a informação das palavras anteriores a palavra *pretty* se torna incompleta e só pode ser totalmente compreendida ao observar-se a palavra *easy*.

Para lidar com essa situação (GRAVES *et al.*, 2013) propôs uma composição de LSTMs de modo que uma sequência fosse processada em ordem por uma *LSTM* (*forward*) e em ordem inversa por outra (*backward*), concatenando assim os resultados de ambas as redes no fim para cada elemento, como é possível ver pela Equações 2.8, 2.9 e 2.10. Essa arquitetura que combina duas LSTMs para processar dependências entre elementos nos dois sentidos de uma sequência é chamada de LSTM bidirecional.

$$h_i^F = LSTM_{forward}(x_i) \tag{2.8}$$

$$h_i^B = LSTM_{backward}(x_i) \tag{2.9}$$

$$h_i = h_i^F \oplus h_i^B \tag{2.10}$$

onde $\oplus$ representa uma operação de concatenação entre dois vetores.

2.3 Tarefas de Processamento de Linguagem Natural Envolvidas

Para construir modelos de compressão de sentenças baseados em redes neurais que dependessem menos da quantidade de dados disponível para treinamento é necessário que parte do trabalho do modelo seja feita antes do treinamento e passado para ele como entrada em forma de atributos. O processo de extrair das sentenças esses atributos, envolve a utilização de outras tarefas de processamento de linguagem natural, como a marcação de classes gramaticais, a análise de dependências gramaticais, entre outras. Esta seção descreve as outras tarefas relacionadas à construção do modelo de compressão de sentenças que foram utilizadas na elaboração deste trabalho.

2.3.1 Marcação de Classes Gramaticais

A tarefa de marcação de classes gramaticais pode ser definida como uma tarefa de rotulação de sequências cujo objetivo é atribuir um rótulo que representa uma classificação gramatical a cada palavra de um *corpus* baseado na relação dessa palavra com outras palavras próximas ou correlacionadas. Por exemplo, a seguinte sentença foi escrita com cada uma de suas palavras seguidas por suas respectivas classes gramaticais, entre colchetes:

Mary [SUBSTANTIVO] ***plays*** [VERBO] ***the*** [ARTIGO] ***piano*** [SUBSTANTIVO].

Na escola geralmente, é ensinado um conjunto de classes gramaticais simplificado composto por 10 classes: substantivo, verbo, adjetivo, pronome, artigo, numeral, preposição, conjunção, interjeição e advérbio. Esse conjunto mais simples de classes pode variar bastante de um idioma para o outro ou ter diferentes níveis de detalhamento, dependendo da aplicação para a qual ele é utilizado. Sistemas computacionais geralmente utilizam conjuntos maiores e mais detalhados com classes que descrevem com mais granularidade a classificação gramatical de uma palavra. Por exemplo, eles podem discriminar substantivos no singular de substantivos no plural, verbos em diferentes conjugações e tempos verbais, diferentes tipos de adjetivos, como comparativos ou superlativos, etc.

Um conjunto de classes muito popular para rotular sentença em língua inglesa é o conjunto do *Penn Treebank* (MARCUS *et al.*, 1993). Ele é composto de 36 rótulos de classes gramaticais e 12 outros rótulos que descrevem pontuações e símbolos especiais. O conjunto

de classes utilizado nesse trabalho é a versão do *OntoNotes 5* (WEISCHEDEL *et al.*, 2013) do conjunto do *Penn Treebank* e pode ser visto na Tabela 1.

Tabela 1 – Rótulos de classes gramaticais do *OntoNotes 5*.

RÓTULO	**DESCRIÇÃO**
$	símbolo de moeda
“	aspas abrindo
”	aspas fechando
,	vírgula
-LRB-	parênteses esquerdo
-RRB-	parênteses direito
.	ponto final
:	dois-pontos
ADD	email
AFX	afixo
CC	conjunção coordenativa
CD	número cardinal
DT	determinador
EX	there existencial
FW	palavra estrangeira
GW	palavra adicional em expressão com múltiplas palavras
HYPH	hífen
IN	conjunção subordinativa ou prepositiva
JJ	adjetivo
JJR	adjetivo comparativo
JJS	adjetivo superlativo
LS	marcador de listagem
MD	verbo auxiliar modal
NFP	pontuação superflua
NIL	ausência de rótulo
NN	substantivo singular ou coletivo
NNP	substantivo próprio singular
NNPS	substantivo próprio plural
NNS	substantivo plural
PDT	pré-determinador
POS	terminação possessiva (’s)
PRP	pronome pessoal
PRP$	pronome possessivo
RB	advérbio

RBR	advérbio comparativo
RBS	advérbio superlativo
RP	partícula adverbial
SP	espaço
SYM	símbolo
TO	"to"infinitivo
UH	interjeição
VB	verbo na forma básica
VBD	verbo no passado
VBG	verbo no gerúndio ou particípio presente
VBN	verbo no particípio passado
VBP	verbo no presente não na 3ª pessoa do singular
VBZ	verbo no presente na 3ª pessoa do singular
WDT	determinador wh-
WP	pronome pessoal wh-
WP$	pronome possessivo wh-
WRB	advérbio wh-
XX	desconhecido
_SP	

2.3.2 Análise de Dependências

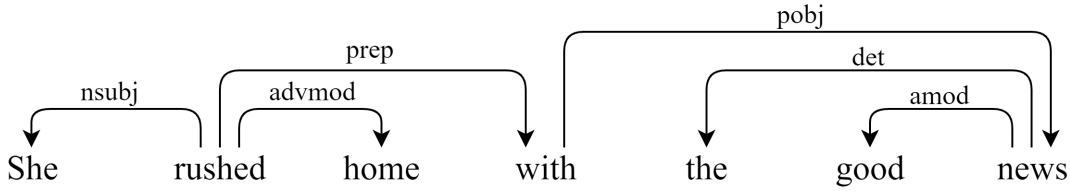
A análise de dependência é uma tarefa de processamento de linguagem natural que tem por objetivo, descobrir as relações de dependência gramatical entre as palavras de uma sentença. Dependência gramatical é um conceito que descreve como uma palavra se conecta com outras palavras em uma sentença. O verbo principal de uma sentença é também o seu pivô e qualquer outra palavra está ligada a ele de maneira direta ou indireta. Cada palavra tem uma ligação unidirecional de dependência com uma única palavra apenas, isto é, ela depende somente de uma única palavra na sentença toda. Desta forma, uma palavra pode ter várias palavras que dependem dela, como no caso do verbo principal de uma frase, mas tem apenas uma palavra da qual ela depende, ou nenhuma no caso do verbo principal.

Essas relações de dependência, no entanto, não são únicas entre quaisquer duas palavras. Existem várias formas possíveis de classificar como uma palavra modifica ou é modificada por outra. Por exemplo, na sentença *"She rushed home with the good news."*, a palavra *good* é um adjetivo que existe para modificar o substantivo *news*, enquanto *news* é o

substantivo que completa a preposição *with*. Não existe um conjunto fechado de tipos possíveis de relações de dependência, porém existe um conjunto de tipos tido como universal, isto é, com tipos de relação presentes em qualquer língua, chamado *Universal Dependency Relations* ou *Universal Stanford Dependencies* (MARNEFFE *et al.*, 2014).

Um analisador de dependências, portanto, tem o papel de analisar uma sentença descobrindo as relações de dependência entre as palavras e seus tipos. A estrutura naturalmente gerada pelas conexões de dependência entre as palavras é chamada de árvore de dependência gramatical. A Figura 9 mostra a árvore de dependência de uma pequena sentença com os tipos das relações descritos em cada aresta entre duas palavras.

Figura 9 – Exemplo de árvore de dependência de uma sentença.



Este trabalho utiliza um conjunto expandido de tipos de relação de dependência descrito em detalhes na Tabela 2

Tabela 2 – Tipos de relação de dependência utilizados no trabalho.

RELAÇÃO	**DESCRIÇÃO**
acl	modificador oracional de substantivo (oração adjectiva)
acomp	complemento adjetivo
advcl	modificador de oração adverbial
advmod	modificador adverbial
agent	agente
amod	modificador adjetivo
appos	aposto
attr	atributo
aux	verbo auxiliar
auxpass	verbo auxiliar (passivo)
case	marcador de caso gramatical
cc	conjunção coordenativa
ccomp	oração complementar
compound	composto
conj	oração coordenativa dependente
csubj	sujeito oracional
csubjpass	sujeito oracional (passivo)

dative	dativo
dep	dependente não classificado
det	determinador
dobj	objeto direto
expl	expletivo (there existencial)
intj	interjeição
mark	marcador
meta	metamodificador
neg	modificador de negação
nn	modificador de substantivo composto
nounmod	modificador de nominal
npmod	locução substantiva como modificador adverbial
nsubj	sujeito nominal
nsubjpass	sujeito nominal (passivo)
nummod	modificador numérico
oprd	predicativo do objeto
obj	objeto
obl	nominal oblíquo
parataxis	modificador parentético
pcomp	complemento de preposição
pobj	objeto de preposição
poss	modificador possessivo
preconj	conjunção pré-correlativa
prep	modificador preposicional
prt	partícula
punct	pontuação
quantmod	modificador de quantidade
relcl	modificador de oração relativa
root	raiz
xcomp	complemento de oração aberto

2.3.3 Lematização

É bastante comum em vários idiomas a existência de palavras que são flexionadas em outras palavras. Um exemplo muito comum desse comportamento são os verbos, que podem aparecer sob diferentes formas, dependendo da conjugação utilizada. Em inglês, o verbo *to think* pode aparecer como *think*, *thinks*, *thinking* ou *thought*. Todas essas formas, no entanto, derivam

de uma forma original, no caso do exemplo essa palavra é *think*. A essa forma original, também conhecida como forma de dicionário, de uma palavra dá-se o nome de lema.

A lematização é, portanto, uma tarefa de processamento de linguagem natural cujo objetivo é obter o lema das palavras de um corpus. Apesar de parecer uma tarefa simples, ela depende fortemente de informações sintáticas da palavra para obter o lema correto de uma palavra. Por exemplo, a palavra *thought* pode ser tanto um verbo como um substantivo em uma frase. Se ela aparecer como um verbo o seu lema será *think*, pois é forma de dicionário do verbo *to think*, mas se ela aparecer como um substantivo seu lema será *thought*.

2.3.4 Reconhecimento de Entidades Nomeadas

Entidades nomeadas são palavras que descrevem coisas do mundo real como pessoas, lugares, instituições, eventos, etc. John, Nova York, Paris, Cristo Redentor, são exemplos comuns de entidades nomeadas. O reconhecimento de entidades nomeadas, em geral, tem dois objetivos: 1) identificar entidades nomeadas em documentos textuais e 2) atribuir uma categoria para as entidades nomeadas identificadas. Por exemplo, na sentença *"Will Smith no longer lives in Hollywood."*, as palavras *Will Smith* e *Hollywood* são entidades nomeadas, mas de categorias diferentes. *Will Smith* representa um nome de pessoa, enquanto *Hollywood* representa o nome de um lugar.

Não existe um conjunto padrão de categorias de entidades nomeadas. Alguns modelos de reconhecimento de entidades nomeadas utilizam conjuntos mais simplificados de categorias, enquanto outros utilizam conjuntos com uma maior variedade de categorias. Modelos recentes treinados sobre o corpus da *Wikipedia* utilizam um esquema com apenas quatro categorias: PER (nomes de pessoas), LOC (nomes de lugares no geral), ORG (nomes de organizações) e MISC (qualquer outro tipo de entidade nomeada) (NOTHMAN *et al.*, 2013). Neste trabalho foi utilizado, contudo, um conjunto mais detalhado de categorias como é possível ver na Tabela 3.

Tabela 3 – Categorias de entidades nomeadas utilizadas no trabalho.

CATEGORIA	**DESCRIÇÃO**
PERSON	Pessoas, incluindo ficcionais
NORP	Nacionalidades or grupos religiosos ou políticos
FAC	Construções, aeroportos, estradas, pontes, etc.
ORG	Companhias, agências, instituições, etc.
GPE	Países, cidades e estados

LOC	Locais que não sejam GPE
PRODUCT	Objetos, veículos, comidas, etc.
EVENT	Nomes de furacões, batalhas, guerras, eventos esportívos, etc.
WORK_OF_ART	Títulos de livros, músicas, etc.
LAW	Documentos nomeados feitos nas leis
LANGUAGE	Qualquer nome de idioma
DATE	Datas ou períodos relativos ou absolutos
TIME	Momentos menores que um dia
PERCENT	Porcentagem, incluindo o "%"
MONEY	Valores monetários, incluindo a unidade
QUANTITY	Quantificadores, como peso ou distância
ORDINAL	Números ordinais
CARDINAL	Numerais que não se enquadrem em nenhuma outra categoria

3 TRABALHOS RELACIONADOS

Diversas soluções foram propostas para a tarefa de compressão de sentenças. Inicialmente, praticamente todos os trabalhos focaram em executar a tarefa de maneira extrativa, uma vez que essa era uma abordagem inicial mais simples de resolver, contudo alguns trabalhos focados em compressão abstrativa de sentenças também foram propostos, em sua grande maioria, nos últimos anos. Essas soluções se dividiam principalmente em abordagens baseadas em fontes de conhecimento (JING, 2000), em modelos de *Noisy-Channel* (KNIGHT; MARCU, 2000; GALLEY; MCKEOWN, 2007), em aprendizagem discriminativa de margens largas (MCDONALD, 2006), em modelos baseados em regras de reescrita (COHN; LAPATA, 2007; COHN; LAPATA, 2008), em otimização (CLARKE; LAPATA, 2008; FILIPPOVA; STRUBE, 2008) e, mais recentemente, em redes neurais (FILIPPOVA *et al.*, 2015; RUSH *et al.*, 2015; CHOPRA *et al.*, 2016; WANG *et al.*, 2017). Neste último com um destaque para as abordagens extrativas, uma vez que essas são naturalmente demandam quantidades inferiores de dados que as abordagens abstrativas.

3.1 Abordagens Baseadas em Fontes de Conhecimento

O trabalho de (JING, 2000) é o primeiro trabalho a tentar resolver de forma automática a compressão de sentenças. Apoiado pelo argumento de utilizar compressão de sentenças para melhorar a qualidade de sumários extrativos, ele chama a então desconhecida tarefa de redução de sentenças e propõe uma abordagem que se baseia em extrair informações de múltiplas fontes de conhecimento para apoiar as decisões de compressão de uma sentença. Com essas fontes de dados ele utiliza um conjunto de passos que tem como objetivo descobrir quais partes das sentenças não podem ser removidas, quão importante para o contexto é cada palavra e qual o impacto de removê-la, para assim no final ele ter informação suficiente para decidir, para cada palavra, se ela deve ser removida ou não.

3.2 Abordagens Baseadas em *Noisy-Channel*

(KNIGHT; MARCU, 2000) propôs em seu trabalho dois modelos diferentes para compressão de sentenças: um modelo de noisy-channel probabilístico, então utilizado em outras tarefas de PLN, e um modelo baseado em árvore de decisão. Para o modelo de noisy-channel a tarefa foi modelada com objetivo de, dada uma sentença t com informações pouco relevantes ou

ruídos, encontrar uma sentença *s* sobre a qual fora adicionada essa informação extra. Para isso, foi utilizado um modelo baseado em gramática probabilística livre de contexto com o objetivo de calcular as probabilidades de expansão da sentença *s* na sentença *t*. O modelo baseado em árvore de decisão, por sua vez, utiliza um conjunto de atributos baseados em operações de reescrita da árvore sintática das sentenças. Assim, foi mapeado o conjunto de operações possíveis sobre a árvores sintática das sentenças e esse conjunto foi mapeado em um conjunto de atributos utilizados em um modelo de decisão. O trabalho de (GALLEY; MCKEOWN, 2007), por usa vez, seguiu a abordagem de noisy-channel proposta por (KNIGHT; MARCU, 2000) e apresentou um sistema de compressão de sentenças que utilizava informações de dependência das palavras para formular regras de remoção de palavras baseadas em uma gramática livre de contexto síncrona, o que ele mostrou que melhorou a performance de seu predecessor.

3.3 Abordagens Baseadas em Aprendizagem Discriminativa de Margens Largas

Em (MCDONALD, 2006), a compressão de sentenças é modelado como um problema de encontrar o melhor conjunto de pares de palavras que formam a compressão de uma sentença original. Essa modelagem supõe que cada par de palavras adjacentes em uma sentença possui uma pontuação que descreve o quão importante é que duas palavras estejam adjacentes na sentença comprimida. As palavras, para essa modelagem, foram representadas por um conjunto de atributos sintáticos extraídos delas por outros métodos específicos. Esses atributos foram chamados de atributos atenuados, pois como eram o resultado de outros modelos, eles eram na verdade uma estimativa dos atributos sintáticos das palavras. O objetivo então era encontrar os pares de palavras que maximizavam a soma dessas pontuações para formar a versão comprimida da sentença original. Para aprender como atribuir essas pontuações aos pares de palavras foi utilizado um método de aprendizagem discriminativa de margens largas chamado *Margin Infused Relaxed Algorithm* (MIRA).

3.4 Abordagens Baseadas em Regras de Reescrita

O trabalho de (COHN; LAPATA, 2007) modelou a compressão de texto como um problema de reescrita de árvores. Ele propõe um método de tradução de árvores para problemas de reescrita de texto que utilizava uma gramática de substituição síncrona de árvore. A cada regra da gramática de substituição síncrona de árvore foi atribuído um peso aprendido através de

um conjunto de dados de treino. Assim, o objetivo do modelo era buscar por todo o espaço de possíveis compressões de uma sentença pela sentença com melhores pontuações utilizando as regras da gramática. (COHN; LAPATA, 2008) estendeu os conceito do trabalho anterior para lidar com a compressão de sentenças abstrativa ao inserir regras de reescrita que contemplavam substituições, inserções e reordenações das sentenças. Este foi o primeiro trabalho, e o único por um tempo, a tentar tratar compressão de sentenças abstrativa.

3.5 Abordagens Baseadas em Otimização

O trabalho de (CLARKE; LAPATA, 2008) modela a compressão de sentença como um problema de Programação Linear Inteira (PLI) cujo objetivo é encontrar a uma compressão ótima para uma sentença dado um conjunto de restrições sobre características da linguagem e específicas da tarefa de compressão de sentenças. Nesse trabalho, formulações anteriores de compressão de sentenças são adaptadas para a abordagem de PLI com o objetivo de investigar se esses modelos tem suas performances melhoradas pela adição de restrições no processo de compressão das sentenças. (FILIPPOVA; STRUBE, 2008) em seu trabalho propõe uma abordagem de compressão de sentenças não supervisionada que se baseia em fazer a redução das sentenças removendo subárvores de suas árvores de dependências. Ele modela essa tarefa, assim como (CLARKE; LAPATA, 2008), como um problema de PLI com o objetivo de determinar que relações de dependência, isto é, as arestas da árvore de dependências, deveriam ser removidas e como a árvore deveria se adaptar a essa remoção.

3.6 Abordagens Baseadas em Redes Neurais

O trabalho de (FILIPPOVA *et al.*, 2015) é o primeiro a modelar a tarefa de compressão de sentenças como uma rotulação de sequências e o primeiro a utilizar redes neurais para classificar as palavras de uma sentença como *mantidas* ou *removidas*. Ele propôs um modelo inspirado nos, então recentes, avanços obtidos em Neural Machine Translation (NMT).

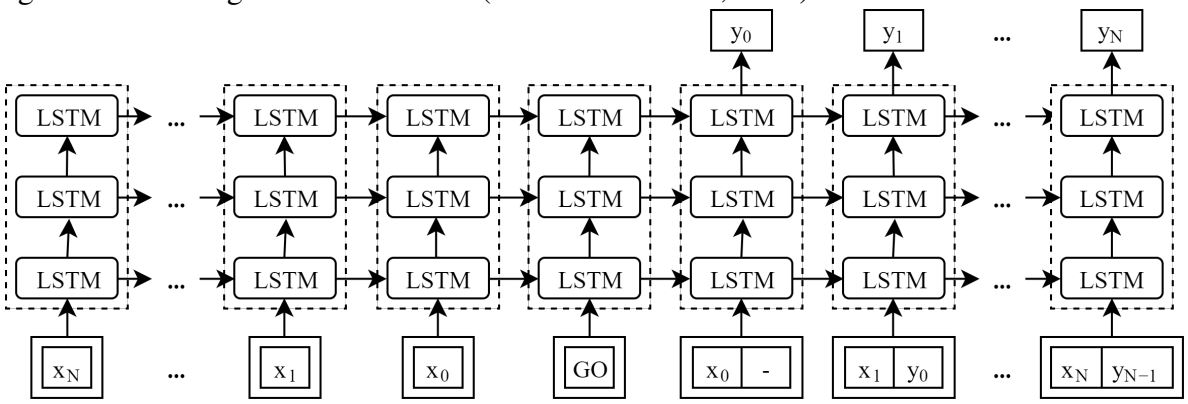
O modelo de (FILIPPOVA *et al.*, 2015) era uma rede neural recorrente composta de três camadas de *LSTM* empilhadas com o objetivo de permitir que as camadas superiores aprendessem representações mais complexas das sentenças de entrada. Entre as camadas de *LSTM* foram posicionadas camadas de *dropout* para evitar o *overfitting*. Estas camadas ignoram a fase de atualização dos pesos para uma percentagem aleatória de neurônios da camada anterior a

ela, assim evitando que o modelo acabe aprendendo demais as dependências dos dados de entrada e não generalize bem a tarefa. De maneira similar à arquitetura de codificador-decodificador, as sentenças alimentavam o modelo, palavra a palavra em ordem inversa, até um *token* especial de marcação de fim da sentença ser passado. Então cada palavra da sentença era mais uma vez passada para o modelo e ele classificava-as como 1, se ela fosse mantida nas sentenças comprimidas, ou 0, se ela fosse removida. A Figura 10 ilustra esse processo.

Para representar as palavras de entrada, foi utilizado apenas os vetores de *embedding* das palavras, treinados com o modelo de *Skipgram* (MIKOLOV *et al.*, 2013) e a classificação da palavra anterior. Assim, na fase de treinamento o modelo era alimentado com os rótulos corretos das palavras anteriores e durante a fase de decodificação, ele era alimentado com o rótulo gerado no passo anterior da rede neural recorrente.

Este trabalho utilizou o método proposto em (FILIPPOVA; ALTUN, 2013) para gerar um conjunto de dados de notícias do *Google News* composto por 2 milhões de sentenças e as suas respectivas formas comprimidas de referências. Esse conjunto de dados foi então utilizado para treinar o modelo por ele proposto.

Figura 10 – Visão geral do modelo de (FILIPPOVA *et al.*, 2015)



Preocupado com a performance de modelos de compressão de sentenças baseados em redes neurais em dados de domínios sobre os quais eles não foram treinados, o trabalho de (WANG *et al.*, 2017) estendeu o trabalho de (FILIPPOVA *et al.*, 2015) atualizando a arquitetura de *LSTM* utilizada para uma arquitetura baseada em pilhas bidirecionais de *LSTM* (GRAVES *et al.*, 2013), adicionando atributos sintáticos e adicionando um pós-processamento na saída do modelo baseado em PLI.

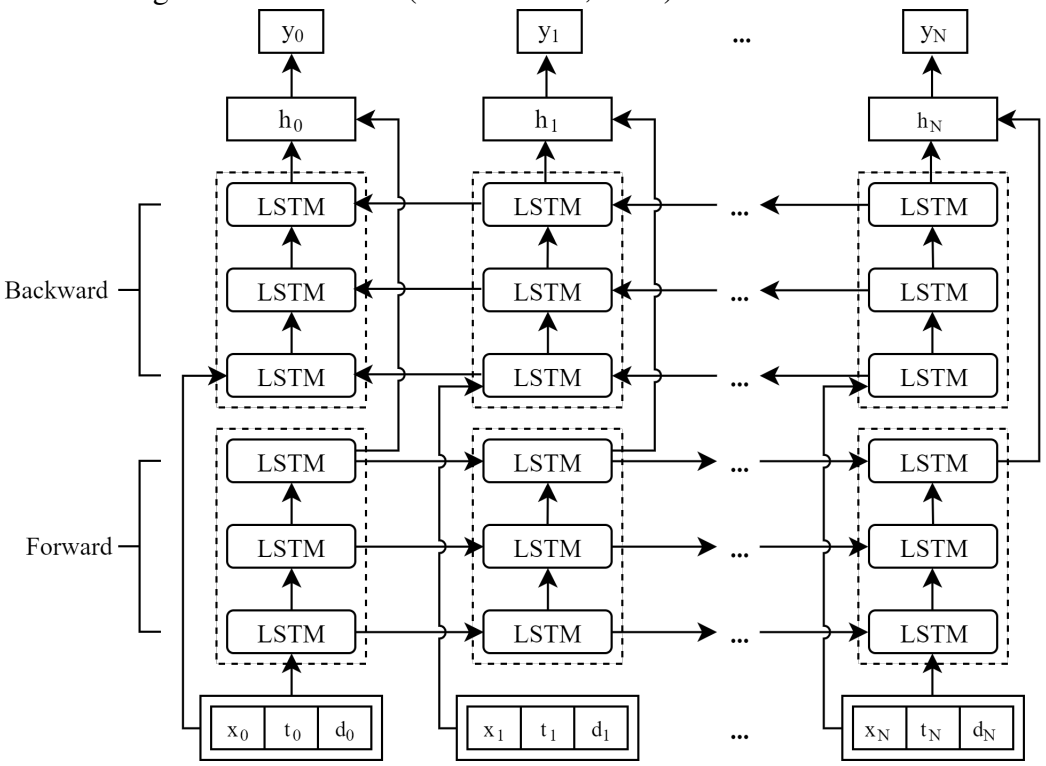
O modelo de (WANG *et al.*, 2017) utilizava duas pilhas de três camadas de *LSTM*, uma que recebia as sentenças em ordem, e outra que recebia a sentença em ordem inversa. As

saídas de ambas as pilhas eram então concatenada em vetores que representavam as palavras em ambas as ordens, melhorando assim a capacidade do modelo de lidar com informações contextuais. É possível ver uma representação da arquitetura descrita na Figura 11.

Além disso, ele adicionou a classe gramatical das palavras e o tipo de dependência que a palavra tinha com sua pai como atributos sintáticos à entrada do modelo, sendo ambas representadas por vetores de *embedding* a serem treinados juntamente com o modelo. Esses vetores foram então concatenados todos ao vetor de *embedding* da própria palavra, aqui utilizando os vetores pré-treinados do *GloVe* (PENNINGTON *et al.*, 2014). Na Figura 11 para uma sentença $\mathscr{W} = w_0, w_1, ..., w_N$, x_i representa o vetor de *embedding* da palavra w_i, t_i representa o vetor de *embedding* da classe gramatical de w_i e d_i representa o vetor de *embedding* do tipo de dependência da palavra w_i, para $w_i \in W$.

Por fim, este trabalho também propôs a utilização de PLI para encontrar uma combinação ótima de rótulos de compressão do modelo que gerassem uma sentenças que obedecessem um conjunto de restrições.

Figura 11 – Visão geral do modelo de (WANG *et al.*, 2017)



3.7 Discussão

Dada a dependência que modelos de compressão de sentenças abstrativos tem de grandes quantidades de dados, não seria viável desenvolver uma abordagem abstrativa para lidar com pequenas quantidades de dados. Por conta disso, esses trabalhos não são de interesse deste trabalho e, consequentemente, não foram incluídos nesta relação de trabalhos relacionados. Para o cenário em que se desejava desenvolver um método de compressão de sentenças que fosse competitivo com o estado da arte, mas que conseguisse lidar de maneira mais eficiente com pequenas quantidades de dados, este trabalho estende o trabalho de (WANG *et al.*, 2017), que por sua vez já era uma extensão do trabalho de (FILIPPOVA *et al.*, 2015).

O trabalho de (FILIPPOVA *et al.*, 2015) apoiava-se no argumento de que, com quantidades suficientemente grandes de dados, não era necessária a utilização de atributos sintáticos para o treinamento de um modelo de compressão de sentenças. De fato, ele prova isso empiricamente através de sua experimentação, mas não avalia seu modelo em sentenças de conjuntos de dados de outros domínios além daquele sobre o qual elas foram treinadas. Em seu trabalho (WANG *et al.*, 2017) argumenta que esses modelos, não necessariamente têm uma boa performance em sentenças de fora dos domínios das sentenças que foram utilizadas para treiná-los. Assim, (WANG *et al.*, 2017) estende o trabalho de (FILIPPOVA *et al.*, 2015) adicionando a classe gramatical das palavras e o tipo de dependência delas com suas pais como atributos sintáticos, que ele argumenta ajudar na capacidade de generalização do modelo. Aliado a isso, ele aplica um *framework* de PLI, com o objetivo de aplicar certas restrições às sentenças comprimidas, tornando seu modelo um híbrido entre modelos baseados em redes neurais e modelos baseados em otimização. A adição dessa última etapa melhora a performance do modelo original para sentenças de fora do domínio de treinamento, mas acaba por prejudicar a performance desse modelo para sentenças do próprio domínio, o que ainda cumpre o objetivo de seu trabalho.

Neste trabalho, ao invés de tentar desenvolver um modelo capaz de generalizar a tarefa de compressão de sentenças para qualquer domínio, a fim de que domínios com pouca disponibilidade de dados rotulados não sejam prejudicados, como fez (WANG *et al.*, 2017) em seu trabalho, o objetivo era desenvolver uma técnica que permitisse a generalização da tarefa apenas para um mesmo domínio, mas que demandasse quantidades inferiores de dados. Para estender o trabalho de (WANG *et al.*, 2017), o trabalho proposto apoiou-se em dois pontos principais: 1) ampliar a representação das palavras adicionando-lhes mais informações

pertinentes e 2) elaborar uma estratégia mais inteligente de substituição de palavras com poucas ocorrências que a simples substituição por um *token* especial de palavra desconhecida.

Deste modo, o modelo proposto, assim como o de (WANG *et al.*, 2017), utiliza uma pilha bidirecional de *LSTM* para classificar as palavras entre mantidas ou removidas. Como atributos ele utiliza uma representação vetorial da sentença original para armazenar o contexto da sentença, informações como posição das palavras e de suas pais para construir essa relação entre as palavras e a quantidade de palavras dependentes de uma palavra para tornar explícita a quantidade de informação que ela carrega, além da própria palavra, classe gramatical e tipo de dependência da palavra com sua pai. Diferente do modelo de (WANG *et al.*, 2017), os atributos sintáticos utilizados não estão necessariamente rotulados nos dados. Eles são extraídos através de técnicas específicas. No caso da classe gramatical é feita uma marcação de classes gramaticais e no caso das relações de dependência, posição da palavra pai e quantidade de palavras dependentes, é feita uma análise de dependência e essas informações são extraídas da árvore de dependência gramatical gerada. Esses atributos sintáticos, assim como em (MCDONALD, 2006), são chamados de atenuados, pois representam apenas estimativas das informações reais portanto, podem agregar um fator de erro ao modelo de compressão. No entanto, não precisar dessas informações rotuladas nos dados diminui o custo de se construir um conjunto de dados para compressão de sentenças em domínios específicos. Como a etapa de PLI prejudicou o modelo original de (WANG *et al.*, 2017) para sentenças do domínio utilizado para treinamento, esta parte não foi utilizada no trabalho proposto por essa dissertação, uma vez que o objetivo era obter a melhor performance possível nas sentenças do próprio domínio, mesmo com poucos dados.

Finalmente, ao invés de apenas substituir palavras que ocorrem poucas vezes com um *token* de *desconhecida*, que seria mapeado para um vetor de *embedding* nulo ou aleatório, é adotada uma estratégia mais inteligente, na qual as palavras são substituídas por palavras com semânticas relacionadas, assim essas palavras são melhor aproveitadas pelo modelo. A Tabela 4 mostra uma comparação entre os modelos mais relacionados ao trabalho proposto.

Tabela 4 – Comparação entre trabalhos de compressão de sentenças extrativos

	Rede Neural Recorrente	**Atributos**	**Palavras Raras**
(MCDONALD, 2006)	Não utiliza	Conjunto extenso de atributos sintáticos atenuados	Não faz substituição
(FILIPPOVA *et al.*, 2015)	Pilha de LSTM	Palavra + Rótulo da palavra anterior	Substituição por *token* de *desconhecida*
(WANG *et al.*, 2017)	Pilha de BiLSTM	Palavra + Atributos sintáticos	Substituição por *token* de *desconhecida*
Modelo Proposto	Pilha de BiLSTM	Atributos atenuados semânticos, estruturais e de dependência	Substituição por palavras similares

4 MODELO NEURAL PARA COMPRESSÃO DE SENTENÇAS

Este capítulo apresenta a proposta de solução desta dissertação para o problema de compressão automática de sentenças utilizando um modelo baseado em redes neurais com quantidades reduzidas de dados de treinamento. Assim como em (FILIPPOVA *et al.*, 2015) e (WANG *et al.*, 2017), nesta dissertação o problema de compressão de sentenças foi modelado como um problema de rotulação binária de sequências, no qual cada *token* de uma sequência deve ser rotulado como *mantido* ou *removido*.

Para o desenvolvimento deste trabalho, o trabalho de (WANG *et al.*, 2017) foi utilizado como referência, uma vez que ele também tem uma limitação na quantidade de dados de treinamento disponível. Seu modelo original foi estendido mudando-se o conjunto de atributos de entrada utilizado para um conjunto mais amplo de atributos e utilizando-se uma estratégia de pré-processamento das sentenças visando a redução do vocabulário do *corpus* e, consequentemente, minimização da quantidade de palavras raras, que pouco contribuem para o treinamento do modelo. A Tabela 5 mostra com mais detalhes a diferença entre os atributos utilizados nos dois trabalhos.

A tarefa de compressão de sentenças ao ser modelada como um problema de rotulação de sequências pode ser formulada da seguinte maneira:

Definição 4.0.1 *Seja $\mathscr{V}$ um vocabulário de palavras e $\mathscr{W} = (w_0, w_1, \ldots, w_N)$ uma sentença composta por uma sequência de palavras $w_i \in \mathscr{V}$, $\forall i \in \{0, \ldots, N\}$. Deseja-se comprimir $\mathscr{W}$ com o objetivo de gerar uma nova sentença $\mathscr{U} = (u_0, ..., u_M)$ tal que $N \geq M$ e $\forall j \in \{0, ..., M\}, \exists i \in \{0, ..., N\}$, tal que $u_j = w_i$ e $j \leq i$. Além disso, $\mathscr{U}$ deve manter o máximo possível de informação de $\mathscr{W}$ e deve se manter gramaticalmente correta.*

Ao modelar essa tarefa como um problema de rotulação de sequências o objetivo é obter um modelo M tal que $\forall w_i \in \mathscr{W}$, $M(w_i) = 1$, se $w_i \in \mathscr{U}$, e $M(w_i) = 0$, caso contrário.

Tabela 5 – Diferenças dos atributos entre o modelo de referência e o modelo proposto

Modelo	Palavra	Sentença Codificada	Classe Gramatical	Posição da Palavra	Nós na Subárvore de Dependência	Relação de Dependência	Posição da Palavra Pai
Modelo de Referência (WANG *et al.*, 2017)	X		X			X	
Modelo Proposto	X	X	X	X	X	X	X

4.1 Atributos das Palavras

Para compressão de sentenças, o trabalho de (WANG *et al.*, 2017) já mostrou que utilizar as classes gramaticais das palavras, assim como as relações de dependência dessas palavras com suas palavras pais como atributos beneficiam o treinamento de modelos de compressão de sentença com quantidades inferiores de dados. No trabalho de (WANG *et al.*, 2017), por exemplo, o modelo proposto é treinado utilizando uma quantidade de dados mais de 200 vezes menor que em trabalhos anteriores (FILIPPOVA *et al.*, 2015; RUSH *et al.*, 2015; CHOPRA *et al.*, 2016). No entanto, existem outras informações que podem ser extraídas das palavras de uma sentença que podem também ser importantes para o desenvolvimento de um bom modelo de compressão de sentenças.

Na tentativa de encontrar um conjunto de atributos que pudesse melhorar o desempenho do modelo de (WANG *et al.*, 2017), foi elaborado um conjunto de atributos que estendia o conjunto de atributos do trabalho de (WANG *et al.*, 2017). Inspirado pela ideia de buscar informações em múltiplas fontes de conhecimento utilizado por (JING, 2000), o processo de selecionar os atributos de uma palavra contou com três objetivos principais: 1) obter informações semânticas e contextuais de uma palavra, 2) obter informações estruturais de uma palavra que mostrassem o papel e importância dela na sentença e 3) tornar explícitas as relações de dependência entre as palavras de uma sentença, pois uma vez que uma palavra é removida ou mantida, essa decisão pode repercutir de maneiras diferentes para cada uma das palavras descendentes dela. A seguir são apresentados cada um dos atributos utilizados, assim como as razões que contribuíram para sua escolha e a forma como ele foi representado para o modelo.

4.1.1 Atributos Semânticos

Foram escolhidos dois atributos que têm como objetivo guardar a informação semântica das palavras: a palavra em si e a sentença completa.

A palavra em si é o principal atributo do modelo. Ela contém toda e qualquer informação relevante que não tenha sido manualmente extraída em outros atributos e que pode ainda ser importante para a classificação das palavras de uma sentença. Uma vez que é muito difícil descobrir manualmente todos os atributos relevantes para o modelo, manter a palavra como um atributo garante que qualquer outra informação valiosa não seja simplesmente descartada. Dada uma sentença $\mathscr{W} = (w_0, w_1, \ldots, w_N)$, cada palavra $w_i \in \mathscr{W}$ é representada como um vetor

de *embedding* e_i pré-calculado que não é modificado durante o treinamento do modelo.

O outro atributo semântico é a sentença completa, da qual cada palavra faz parte. A sentença representa, resumidamente, o contexto no qual as palavras estão inseridas, portanto, aparenta ser uma informação importante para a tomada de decisão do modelo. Dada uma sentença $\mathscr{W}$, ela é representada por um vetor de *embedding* h_e calculado utilizando um codificador de sentenças construído com uma pilha bidirecional de *LSTM*, estrutura similar a utilizada por (WANG *et al.*, 2017) para classificar as palavras da sentença. Esse codificador de sentenças é posteriormente apresentado em detalhes.

4.1.2 Atributos Estruturais

As classes gramaticais das palavras, suas posições e quantidade de descendentes na árvore de dependência gramatical são utilizadas como atributos estruturais para o modelo.

O rótulo de classe gramatical de uma palavra é uma categorização linguística e pode ser utilizado para defini-la de uma maneira individual. Esses rótulos podem classificar uma palavra mais amplamente, especificando apenas a que classe gramatical ela pertence, ou de uma forma mais refinada, indicando também em que tipo de subcategoria de classe gramatical aquela palavra se encaixa. Para esse modelo foi utilizado um conjunto de rótulos de classes gramaticais mais refinados, de modo que as palavras pudessem ser precisamente diferenciadas, como é possível ver na Tabela 1. Assim, dada uma sentença $\mathscr{W} = (w_0, w_1, \ldots, w_N)$, cada rótulo relativo à palavra $w_i \in \mathscr{W}$ é representado como um vetor de *embedding* t_i, a ser calculado durante o treinamento do modelo. Para obter-se as classes gramaticais das palavras, as sentenças passaram por um processo de marcação de classes gramaticais utilizando um modelo específico para esse fim.

A posição da palavra na sentença também foi adicionada como um atributo estrutural na intenção de se criar uma referência numérica para cada palavra dentro da sentença e foi representado como um vetor de *one-hot*. Por fim, com o intuito de obter-se um valor numérico que representasse a quantidade de informação da sentença agregada a cada palavra as sentenças passaram por um processo de análise de dependência, no qual sua árvore de dependência gramatical foi construída e da qual foi feita a contagem da quantidade de palavras descendentes de cada palavra nessa árvore. Este último foi representado pelo seu próprio valor, sendo ele um atributo numérico.

A Figura 12 mostra a árvore de dependência gramatical da sentença *"The lake is*

a long way from here." e a Tabela 6 mostra os atributos estruturais extraídos para cada uma de suas palavras. Os significados dos rótulos das classes gramaticais e dos tipos de relação de dependência pode ser examinados na Tabela 1 e na Tabela 2, respectivamente:

Figura 12 – Exemplo de árvore de dependência gramatical para uma sentença.

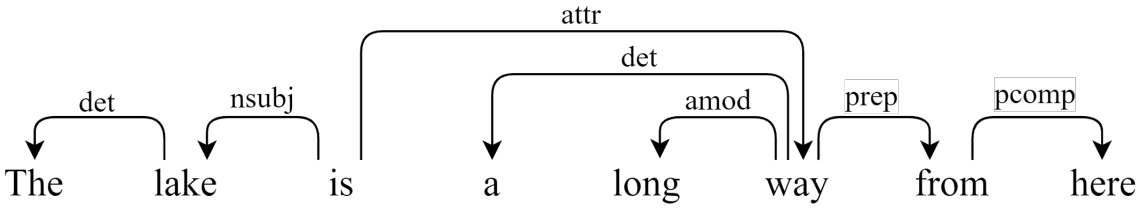


Tabela 6 – Exemplo de atributos estruturais para uma sentença.

Palavras	Rótulo de Classe Gramatical	Posição	Número de Palavras Dependentes
the	DT	0	0
lake	NN	1	1
is	VBZ	2	8
a	DT	3	0
long	JJ	4	0
way	NN	5	4
from	IN	6	1
here	RB	7	0
.	.	8	0

4.1.3 Atributos de Dependência

A árvore de dependência gramatical é uma estrutura amplamente utilizada para representar a estrutura sintática de uma sentença. Nela as palavras são nós de uma árvore e cada aresta mapeia um tipo de relação de dependência sintática entre duas palavras. A informação de que uma palavra foi removida da sentença é extremamente relevante para a decisão do que deve ser feito com outras palavras que descendem desta. Por exemplo, se um substantivo for removido de uma sentença, é possível que não faça sentido manter um adjetivo que o modificava.

Com a intenção de tornarem explícitas as dependências entre as palavras, foi feita a análise de dependências das sentenças e foram utilizados como atributos tanto a posição da palavra pai de cada palavra na árvore de dependência gramatical quanto o rótulo da relação de dependência entre essas duas. A posição da palavra pai, tal qual a posição da própria palavra, foi representado através de um vetor de *one-hot* e a relação de dependência entre uma palavra e sua pai foi representado como um vetor de *embedding* d_i a ser calculado durante o treinamento de

Tabela 7 – Exemplo de atributos de dependência para uma sentença.

Palavras	Relação de Dependência	Posição da Palavra Pai
the	det	1
lake	nsubj	2
is	root	2
a	det	5
long	amod	5
way	attr	2
from	prep	5
here	pcomp	6
.	punct	2

todo o modelo.

A Tabela 7 mostra os atributos de dependência para cada palavra da sentença "*The lake is a long way from here.*"segundo sua árvore de dependência gramatical presente na Figura 12. Os significados dos rótulos de tipo de relação de dependência podem ser examinados na Tabela 2:

4.1.4 Representação de Entrada das Palavras

Dada uma sentença $\mathscr{W} = (w_0, w_1, ..., w_N)$, seja h_e o vetor que representa a sentença $\mathscr{W}$ codificada, e_i o vetor de *embedding* que representa a palavra w_i, t_i o vetor de *embedding* que representa o rótulo de classe gramatical de w_i, d_i o vetor de *embedding* que representa a relação entre w_i e sua pai, o_i o vetor *one-hot* de i, p_i o vetor *one-hot* da posição da palavra pai de w_i e, por fim, s_i o total de palavras descendentes de w_i, para toda palavra $w_i \in \mathscr{W}$, o modelo proposto combina todos esses atributos concatenando-os em um vetor único a ser utilizado como entrada para o classificador de palavras:

$$x_i = e_i \oplus t_i \oplus d_i \oplus o_i \oplus p_i \oplus s_i \tag{4.1}$$

$$\hat{x}_i = h_e \oplus x_i \tag{4.2}$$

onde $\oplus$ representa uma operação de concatenação entre dois vetores.

4.2 Estratégia de Redução de Palavras Raras

Em uma tentativa de minimizar o impacto de uma pequena quantidade de dados rotulados que um domínio possa possuir, foi desenvolvida uma estratégia de redução de voca-

bulário focada em minimizar a ocorrência de palavras raras. A ideia central dessa estratégia é encontrar palavras raras que possuem papéis semelhantes entre as sentenças e substituí-las por palavras comuns entre si. Por exemplo, as palavras *John*, *Mary* e *Peter* são todas palavras que descrevem nomes de pessoas e pode acontecer delas ocorrerem poucas vezes em um corpus de sentenças, contudo elas poderiam ser todas substituídas apenas pela palavra *name*. A palavra *name* não só descreve as três palavras anteriores, como, por ser uma palavra comum, é facilmente representável através de um vetor de *embedding* pré-treinado utilizando modelos de *embedding* de palavras como o do *GloVe* (PENNINGTON *et al.*, 2014) ou de *Skipgram* (MIKOLOV *et al.*, 2013). Desta forma, ao invés de três palavras que ocorrem algumas poucas vezes no conjunto de treinamento cada uma, as ocorrências das três são somadas em uma única palavra. Este processo transforma os dados de treinamento ao fazer com que existam mais sentenças com estruturas similares para serem utilizadas na fase de treinamento do modelo.

Uma categoria de palavra que aparece com bastante frequência em qualquer texto e que tende a ter uma baixa frequência de ocorrência são as entidades nomeadas. Entidades nomeadas são palavras que dão nome a coisas, pessoas ou lugares e que, portanto, aparecem frequentemente em sentenças, mas raramente se repetem. Em um conjunto de dez sentenças podem ocorrer o nome de dez pessoas diferentes que serão contadas como dez palavras diferentes, mesmo que a informação mais relevante sobre elas seja que elas representam nomes de pessoas. Para evitar esse problema, as palavras que representavam entidades nomeadas das sentenças foram identificadas através de um processo de reconhecimento de entidades nomeadas e, para cada categoria de entidade nomeada, elas foram posteriormente substituídas por palavras em comum. Analogamente ao exemplo acima, nomes de construções, estradas, países e cidades são essencialmente nomes de lugares, assim palavras identificadas como entidades nomeadas de alguma dessas categorias foram substituídas por *location*. O mesmo acontece para nomes de pessoas, objetos, organizações, grupos, números, etc. Para um exame mais detalhado das substituições, a Tabela 3 mapeia cada categoria de entidade nomeada utilizada com sua palavra substituta.

Contudo, nem todas as palavras com baixa frequência de ocorrência são entidades nomeadas. Outras palavras podem sofrer com esse problema, como verbos em conjugações específicas, adjetivos ou substantivos modificados com sufixos, etc. Para resolver esse problema, uma etapa de lematização foi aplicada nas sentenças paralelamente ao reconhecimento de entidades nomeadas. A lematização é uma técnica de processamento natural que busca reduzir

cada palavra de um texto ao seu lema. O lema de uma palavra é a própria palavra em sua forma mais primitiva, sem inflexões ou terminações. Por exemplo, as palavras *good*, *better* e *best* tem como lema a palavra *good*, assim como as palavras *play*, *player*, *played* ou *playing* tem como lema a palavra *play*.

Durante a etapa de substituição propriamente dita, cada palavra é substituída por uma palavra comum seguindo a seguinte estratégia: se a palavra é uma entidade nomeada, ela é substituída pela palavra substituta de sua categoria de entidade nomeada, segundo a Tabela 8, caso contrário, ela é substituída por seu lema. No fim desta etapa, uma sentença como *"John and Mary were playing volleyball last week."*, seria escrita da seguinte maneira:

name and name be play volleyball date date

É importante notar que, apesar dessa sequência de palavras resultante não formar uma sentença coerente ou gramaticalmente correta, ela fornece ao modelo uma forma mais simples de entender os relacionamentos entre as palavras que a compõe.

Outro ponto importante é que fazer esse tipo de substituição de palavras específicas por palavras mais comuns e gerais ocasiona uma perda de informação. No entanto, ao ser utilizado neste trabalho, parte dessas informações possivelmente descartadas são previamente extraídas das palavras e utilizadas em atributos como as classes gramaticais e as relações de dependência entre elas.

4.3 Arquitetura EncBiLSTM

Como em (FILIPPOVA *et al.*, 2015), o modelo treinado pode, dada uma sentença $\mathscr{W} = (w_0, w_1, \ldots, w_N)$ onde cada w_i representa uma palavra da sentença, classificar uma palavra w_i em *mantida* ou *removida*. Para isso, o modelo proposto inspira-se na arquitetura de codificador-decodificador proposta em (CHO *et al.*, 2014) e (SUTSKEVER *et al.*, 2014). Essa arquitetura é composta de dois módulos principais: um codificador responsável por aprender uma representação vetorial para uma sequência de *tokens* e um decodificador responsável por gerar uma sequência de *tokens* a partir de um vetor que representa uma sequência codificada.

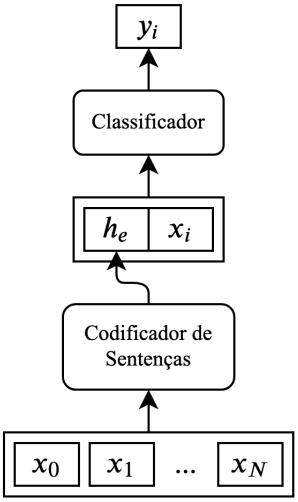
Como uma arquitetura capaz de transformar uma sequência de *tokens* em outra de tamanhos variados, ela é amplamente utilizada em diversos problemas de PLN como tradução de textos (SUTSKEVER *et al.*, 2014; BAHDANAU *et al.*, 2014; LUONG *et al.*, 2015), reconhecimento de discurso (PRABHAVALKAR *et al.*, 2017) ou criação de legendas para

Tabela 8 – Relação das palavras substitutas de cada categoria de entidade nomeada.

CATEGORIA	PALAVRA SUBSTITUTA
PERSON	*name*
NORP	*name*
FAC	*name*
ORG	*name*
GPE	*name*
LOC	*name*
PRODUCT	*name*
EVENT	*name*
WORK_OF_ART	*name*
LAW	*law*
LANGUAGE	*language*
DATE	*date*
TIME	*moment*
PERCENT	*number*
MONEY	*number*
QUANTITY	*measurement*
ORDINAL	*number*
CARDINAL	*number*

vídeoss (VENUGOPALAN *et al.*, 2015). Apesar de se inspirar na arquitetura de codificador-decodificador, o modelo aqui proposto, no entanto, não implementa os mesmos módulos que a arquitetura original. Assim como no codificador-decodificador original, o codificador da arquitetura é responsável por gerar uma representação vetorial de uma sentença. Essa representação vetorial é então utilizada como atributo de entrada, ao lado dos outros atributos das palavras da sentença. Como um rotulador de sequências, o modelo compressor de sentenças não precisa mapear duas sequências de tamanhos distintos, portanto não seria necessário utilizar um decodificador como na arquitetura original. No lugar do decodificador, a arquitetura do modelo utiliza um classificador que recebe como entrada uma sequência de palavras representadas por seus atributos como é possível ver na Figura 13. É importante notar que o vetor de *embedding* da palavra utilizado aqui é da palavra substituta, segundo a estratégia de redução de vocabulário descrita, e não da palavra original. Ambos, codificador de sentenças e classificador são apresentados em detalhes a seguir.

Figura 13 – Visão geral do modelo proposto.



4.3.1 Codificador de Sentenças

O codificador foi implementado utilizando uma arquitetura de pilha bidirecional de *LSTM* (GRAVES *et al.*, 2013; WANG *et al.*, 2017). Comparado com uma arquitetura de *LSTM* comum, a bidirecional tem uma maior capacidade de capturar informações contextuais das sentenças, como dependências entre palavras sucessivas ou não sucessivas. Como é possível ver na Figura 14, a arquitetura é composta de duas pilhas de redes de *LSTM* intercaladas com camadas de *dropout* (SRIVASTAVA *et al.*, 2014). A camada de *dropout* funciona como uma regularização para um modelo de redes neurais de múltiplas camadas, portanto é responsável por evitar que o modelo acabe se adaptando demais aos dados de treino, tornando-o ineficiente para novas sentenças. Ela faz isso associando-se a uma camada de rede neural e criando uma espécie de máscara que impede que uma parte dos neurônios dessa camada sejam atualizados durante a fase de atualização dos pesos, no treinamento.

Para uma sequência de palavras representadas por seus atributos $X = (x_0, x_1, ..., x_N)$, as pilha $Forward_E$ e $Backward_E$ são alimentadas com cada palavra x_i tal que $x_i \in X$. A pilha $Forward_E$ recebe as palavras em sua ordem original, enquanto a pilha $Backward_E$ recebe as mesmas palavras, porém em ordem inversa. Como, a cada nova palavra as pilhas armazenam informações de todas as palavras anteriores, as saídas h_e^F e h_e^B das pilhas $Forward_E$ e $Backward_E$, respectivamente, para a última palavra da sequência representam a sequência completa codificada na ordem original e em ordem inversa. Os vetores h_e^F e h_e^B são então concatenados para formar um novo vetor h_e que representa a sentença codificada ao mesmo tempo em ambas as direções.

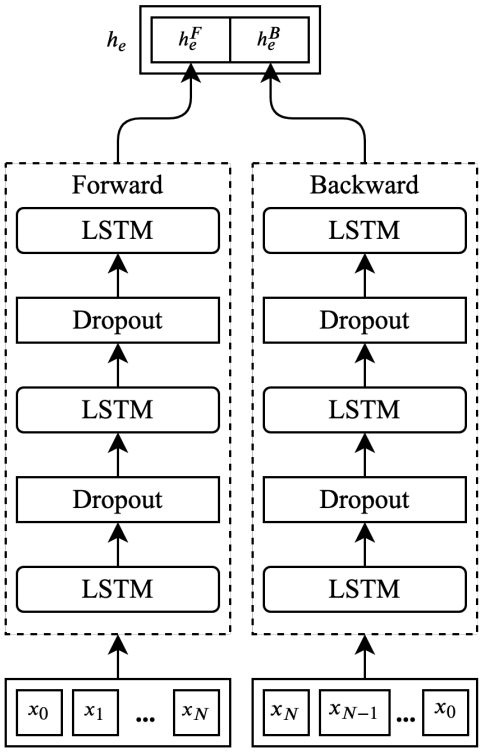
$$h_e^F = Forward_E(X) \tag{4.3}$$

$$h_e^B = Backward_E(X) \tag{4.4}$$

$$h_e = h_e^F \oplus h_e^B \tag{4.5}$$

onde $\oplus$ representa uma operação de concatenação entre dois vetores.

Figura 14 – Visão geral do codificador de sentenças.



4.3.2 Classificador

O classificador da arquitetura é similar ao proposto por (WANG *et al.*, 2017). A principal diferença está na forma como os dados de entrada são passados para o modelo, ou seja, seus atributos. Ao invés de receber apenas os atributos relativos às palavras de uma sentença, o modelo proposto também recebe um vetor representando a própria sentença codificada concatenada aos atributos de cada palavra. Assim, é esperado que no momento de classificação de cada palavra, o modelo tenha mais informação de entrada que o modelo original de (WANG *et al.*, 2017), melhorando assim sua capacidade de decidir se a palavra deve ser mantida ou removida da sentença comprimida. A Figura 15 descreve o funcionamento do codificador ao

longo da classificação das palavras de uma sentença e as Equações 4.6, 4.7, 4.8, 4.9 descrevem o processo de classificação para uma palavra.

$$h_i^F = Forward_C(\hat{x}_i) \tag{4.6}$$

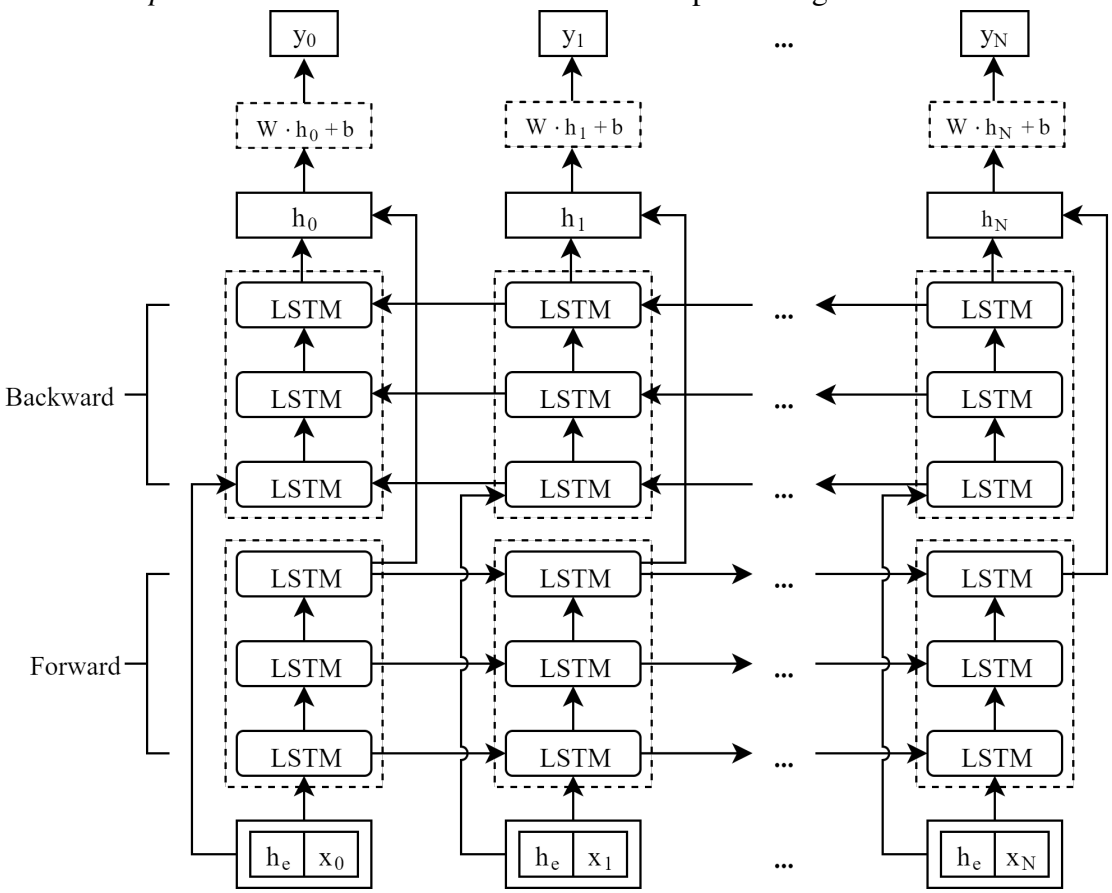
$$h_i^B = Backward_C(\hat{x}_i) \tag{4.7}$$

$$h_i = h_i^F \oplus h_i^B \tag{4.8}$$

$$y_i = W \cdot h_i + b \tag{4.9}$$

onde para uma sentença $\mathscr{W} = (w_0, w_1, ..., w_N)$, $\hat{x}_i$ representa os atributos, incluindo a sentença codificada, relativos à palavra w_i, h_i^F e h_i^B representam os estados ocultos de w_i codificados na ordem original e em ordem inversa, respectivamente, h_i representa o estado oculto de w_i em ambas as direções ao mesmo tempo e y_i representa o rótulo, *mantido* ou *removido*, de w_i, tal que $w_i \in \mathscr{W}$. O símbolo $\oplus$ representa uma operação de concatenação entre dois vetores.

Figura 15 – Classificador detalhado desdobrado ao longo de uma sentença. As camadas de *dropout* foram intencionalmente omitidas em prol da legibilidade.



4.4 Funcionamento

Original: Luis Suarez was spotted in London this afternoon and this has led the Daily Star to link the Liverpool striker to a potential move to Chelsea or Arsenal. Comprimida: Luis Suarez was spotted in London.

- Uma frase é pré-processada e os atributos de suas palavras são extraídos. - Essa frase é codificada em um vetor. - O vetor da frase é concatenado em cada palavra da frase - Cada palavra (conjunto de atributos) da frase é classificada como "mantida"ou "removida- As palavras classificadas como removidas são removidas da frase original

5 EXPERIMENTAÇÃO

Essa seção apresentará os experimentos realizados com o objetivo de avaliar a qualidade das compressões geradas pelo modelo proposto, treinado com diferentes quantidades de dados e seus respectivos resultados.

As sentenças geradas pelo modelo proposto foram comparadas com as geradas por uma reimplementação do modelo de (WANG *et al.*, 2017), visto que o modelo original não está disponível ao público. A escolha desse trabalho como objeto de comparação se deu por três motivos: 1) este é um modelo que também lida com poucas quantidades de dados para treinamento, ainda que este não seja seu foco, 2) ele treina seu modelo de compressão de sentenças utilizando os mesmos dados utilizados neste trabalho e 3) em seu trabalho original ele fez comparações com outros trabalhos presentes na literatura (FILIPPOVA *et al.*, 2015; CLARKE; LAPATA, 2008), então comparar com ele é como comparar indiretamente com todos esses trabalhos.

Em seu trabalho, (WANG *et al.*, 2017) propões três modelos que estendem do modelo de (FILIPPOVA *et al.*, 2015), um que utiliza apenas a pilha bidirecional de *LSTM*, sem a incorporação de atributos sintáticos, um que utiliza a pilha bidirecional de *LSTM* com atributos sintáticos, e um terceiro que utiliza o modelo com atributos sintáticos combinado com uma etapa de programação linear inteira para resolver a sequência de rótulos das sentenças, com um conjunto de restrições de linguagem. Com o objetivo de ter um comparação mais justa o modelo proposto por este trabalho foi comparado com o modelo que utilizava a pilha bidirecional de *LSTM* com atributos sintáticos, uma vez que este foi o modelo que obteve melhores resultados para sentenças do domínio sobre o qual ele foi treinado. Portanto, foi implementado o modelo de (WANG *et al.*, 2017) que utiliza uma pilha bidirecional de *LSTM* e os vetores de *embedding*, classes gramaticais e dependências das palavras como atributos.

5.1 Conjunto de Dados

O conjunto de dados utilizado nos experimentos são os 10 mil pares de sentenças, originais e comprimidas do *Google News* disponibilizados publicamente em (FILIPPOVA *et al.*, 2015). As sentenças possuíam anotações de classes gramaticais e relações de dependência das palavras, no entanto, essas informações foram descartadas com o intuito de simular um cenário em que essas informações não estão disponíveis. Esses pares de sentenças foram extraídos de

maneira automática do *Google News* através de um método desenvolvido em (FILIPPOVA; ALTUN, 2013). Foi determinado um tamanho máximo de 120 palavras para as sentenças assim como em (FILIPPOVA *et al.*, 2015); Aquelas com mais de 120 palavras foram descartadas deixando o conjunto com um total de 9994 sentenças. Dessas, 8 mil foram separadas para treino, mil para validação em desenvolvimento e o restante para teste. Para avaliar como o desempenho do modelo variava com a quantidade de dados usada para treinar, foram utilizados três subconjuntos dos dados de treino: o primeiro utilizava todas as 8 mil sentenças do conjunto, o segundo utilizava 5 mil sentenças e o menor deles utilizava 2 mil sentenças.

5.2 Configuração dos Experimentos

Durante os experimentos, todos os modelos foram treinados para minimizar a função de entropia cruzada multi-classe utilizando o Adam (KINGMA; BA, 2014) como otimizador a uma taxa de aprendizado inicial de 0,001. A dimensão das camadas ocultas de todas as *LSTM* foram definidas para 100, assim como a dimensão dos vetores de *embedding* das palavras. Estes foram inicializados com vetores pré-treinados do modelo do *GloVe* de mesmas dimensões e não foram atualizados durante o treinamento, exceto no caso do modelo de (WANG *et al.*, 2017) onde, tal como descrito em seu trabalho original, os vetores de *embedding* das palavras continuaram a ser atualizados durante o treinamento. As dimensões dos vetores de *embedding* das classes gramaticais e das relações de dependência foram ambas definidas como 40 e, diferentemente dos vetores de *embedding* das palavras, estes foram treinados juntamente com o resto dos modelos. As camadas de *dropout* que intercalavam as camadas de *LSTM* foram configuradas para ignorar aleatoriamente 50% dos neurônios da camada anterior durante a fase de atualização dos parâmetros da rede. Finalmente, todos os modelos foram treinados com um tamanho de *batch* de 30 por 20 épocas cada. A partir de 20 épocas, os modelos não apresentavam mais melhoras para os dados de validação.

5.3 Variações dos Modelos

Foram avaliadas quatro variações de modelos de compressão de sentenças baseados em redes neurais. A Tabela 9 resume as relações entre os modelos e suas características descritas abaixo:

***BiLSTM SynFeat (baseline)*:** Como previamente apresentado, este é o modelo

proposto em (WANG *et al.*, 2017) que utiliza atributos sintáticos, mas não utiliza programação linear inteira. Ele usa uma pilha bidirecional de *LSTM* como classificador combinado com o vetor de *embedding* de cada palavra, sua classe gramatical e relação de dependência com a palavra pai como atributos de entrada. São apresentados tanto os resultados reportados no trabalho original onde esse modelo é proposto, quanto os resultados da reimplementação.

***EncBiLSTM SynFeat*:** Este modelo possui um classificador igual ao do *BiLSTM SynFeat*, porém ele também utiliza um módulo codificador para encontrar uma representação vetorial para a sentença completa e combina essa representação com os atributos de entrada utilizados no *BiLSTM SynFeat*.

***EncBiLSTM SynFeat+*:** Este modelo possui a mesma arquitetura que o modelo *EncBiLSTM SynFeat*, utilizando o mesmo codificador de sentenças e o mesmo classificador, porém ele utiliza a posição da palavra, a posição da palavra pai e a quantidade de nós na subárvore de dependência como atributos de entrada, além do vetor de *embedding* da palavra, sua classe gramatical e relação de dependência, como os modelos anteriores.

***ERPR EncBiLSTM SynFeat+*:** Este modelo possui exatamente a mesma arquitetura e atributos de entrada do modelo *EncBiLSTM SynFeat+*, porém ele foi treinado com a aplicação da **E**stratégia de **R**edução de **P**alavras **R**aras.

Tabela 9 – Relação entre modelos avaliados e suas características.

Modelo	Palavra	Sentença Codificada	Classe Gramatical	Posição da Palavra	Nós na Subárvore de Dependência	Relação de Dependência	Posição da Palavra Pai	Redução de Palavras Raras
BiLSTM SynFeat	X		X			X		
EncBiLSTM SynFeat	X	X	X			X		
EncBiLSTM SynFeat+	X	X	X	X	X	X	X	
ERPR EncBiLSTM SynFeat+	X	X	X	X	X	X	X	X

5.4 Métricas de Avaliação

O conjunto de dados utilizado para avaliação era composto de 994 sentenças previamente separadas do resto das sentenças. Para avaliar a capacidade dos modelos de reproduzirem as compressões de referência, foram calculadas as médias da acurácia a nível de palavra, do *F1 Score* e a da diferença da taxa de compressão padrão das sentenças entre as sentenças geradas pelos modelos e as compressões de referência do conjunto de validação. Para avaliar a qualidade das compressões de maneira automática foram utilizados as métricas *BLEU* (PAPINENI *et al.*, 2002) e algumas variações do *ROUGE* (LIN, 2004).

Acurácia: A acurácia a nível de palavras mede a taxa de palavras que foram

corretamente classificadas para uma sentença, isto é, é a razão entre a soma de todas as palavras que foram corretamente classificadas como *mantidas* (*Verdadeiros Positivos*) com todas as que foram corretamente classificadas como *removidas* (*Verdadeiros Negativos*) sobre a quantidade total de palavras da sentença.

$$Acuracia = \frac{VerdadeirosPositivos + VerdadeirosNegativos}{Total} \tag{5.1}$$

F1 Score: O *F1 Score* é a média harmônica entre outras duas métricas: a precisão (P) e a revocação (R). A precisão mede quantas palavras classificadas como *mantidas* foram corretamente classificadas, isto é, é a razão entre as palavras corretamente classificadas como *mantidas* (*Verdadeiros Positivos*) sobre a soma de todas as palavras que foram assim classificadas (*Verdadeiros Positivos* + *Falsos Positivos*). A revocação mede quantas das palavras que deveriam ter sido mantidas, foram realmente mantidas, ou seja, é a razão entre as palavras corretamente classificadas como mantidas e a quantidade de palavras que deveriam ter sido classificadas como *mantidas* (*Verdadeiros Positivos* + *Falsos Negativos*).

$$Precisao = \frac{VerdadeirosPositivos}{VerdadeirosPositivos + FalsosPositivos} \tag{5.2}$$

$$Revocacao = \frac{VerdadeirosPositivos}{VerdadeirosPositivos + FalsosNegativos} \tag{5.3}$$

$$F1 = \frac{2 \times Precisao \times Revocacao}{Precisao + Revocacao} \tag{5.4}$$

Diferença da Taxa de Compressão Padrão (DTC): A taxa de compressão é a razão entre o comprimento da sentença comprimida e o comprimento da sentença original, ambos em palavras. A diferença da taxa de compressão padrão mede a diferença, em valores absolutos, entre a taxa de compressão de uma sentença comprimida dos dados de validação e a versão comprimida da mesma sentença gerada por um modelo. Ela serve para medir o quão próximo do mesmo nível de compressão das sentenças de referência os modelos chegaram.

BLEU: Esta é uma métrica originalmente proposta para a tarefa de tradução entre idiomas que alega ter uma alta correlação com a avaliação feita por humanos. Ela tenta medir a

qualidade de uma sentença ao verificar o quão semelhante uma sentença gerada é de um conjunto de sentenças de referência. Ela retorna valores entre 0 e 1, sendo que quanto mais próximo de 1, mais parecida a sentença é das suas referências. Cada sentença gerada teve como referência apenas a sentença comprimida de referência dos dados durante o cálculo da BLEU.

ROUGE: Este é um conjunto de métricas amplamente utilizada na literatura para avaliar a qualidade de sumarizações de textos. Ela compara textos gerados com textos de referência para determinar um valor a partir da sobreposição de partes dos textos, normalmente n-gramas. As métricas mais comuns da ROUGE são a *ROUGE-1* (R-1), *ROUGE-2* (R-2) e *ROUGE-L* (R-L) que funcionam calculando a sobreposição de unigramas, bigramas e subsequência mais longa, respectivamente.

5.5 Resultados

Os resultados foram analisados com o objetivo de verificar o quão bem cada modelo foi capaz de reproduzir as compressões do conjunto de sentenças de validação. Essa verificação se deu das seguintes formas: 1) utilizando métricas simples de aprendizagem de máquina como *F1 score* e acurácia, 2) utilizando as métricas *BLEU* e *ROUGE* para verificação de qualidade de textos gerados, e 3) fazendo uma análise textual de um pequeno sub-conjunto aleatório de sentenças comprimidas.

5.5.1 Métricas de Aprendizagem de Máquina

A Tabela 10 mostra os resultados dos experimentos para as métricas de aprendizagem de máquina para cada um dos modelos examinados treinados com 8000 sentenças. Um primeiro ponto a destacar é a diferença entre os resultados do *baseline* reportados no trabalho de (WANG *et al.*, 2017) e os resultados obtidos pelo modelo reimplementado *BiLSTM SynFeat*. Isso pode acontecer por conta de vários fatores como etapas de pré-processamento ou até mesmo a inicialização dos parâmetros das redes neurais dos modelos. Nesse caso específico, é possível que isso ocorra por outro motivo. No trabalho de (WANG *et al.*, 2017), não fica claro se as classes gramaticais e as relações de dependência utilizadas como atributos de entrada foram descobertas durante o pré-processamento das sentenças através de passos de marcação de classes gramaticais e análise de dependência ou se foram utilizadas as informações que já estavam contidas nos dados. No primeiro caso, pode acontecer de o modelo ter feito classificações erradas

fazendo com que esse erro se acumule para os modelos de compressão examinados, o que não deve acontecer no segundo caso, pois supostamente as informações contidas nos dados estão sempre corretas.

Na tentativa de avaliar um cenário em que os dados utilizados para treinamento não possuem todas as informações de atributos necessárias, todos os modelos implementados neste trabalho utilizaram atributos extraídos através de passos de pré-processamento (marcação de classe gramatical e análise de dependência). De todo modo, não há garantias que os valores dos atributos utilizados no trabalho original para cada palavra sejam os mesmos dos utilizados neste trabalho, apenas que eles têm o mesmo sentido e objetivo. Isso provavelmente é o que gera essa divergência de resultados.

Com relação às arquiteturas dos modelos avaliados, ao adicionar a sentença codificada (juntamente com o codificador de sentenças) no modelo *EncBiLSTM SynFeat* nota-se que o desempenho não apresentou uma melhoria significativa. Isso possivelmente se deve ao fato de que o acréscimo do codificador, apesar de melhorar a capacidade do modelo de decidir sobre a classificação de uma palavra dada toda a sentença, também dobrou o tamanho do modelo em quantidade de parâmetros, dificultando mais seu treinamento. Para contrabalancear esse aumento do tamanho do modelo foram adicionadas a posição, a posição da palavra pai e a quantidade de palavras dependentes de cada palavra como atributos em *EncBiLSTM SynFeat+*. Os resultados deste modelo já se mostra ligeiramente superior ao *BiLSTM SynFeat*, mais ainda próximos. Ao aplicar a estratégia de redução de palavras raras no modelo *ERPR EncBiLSTM SynFeat+*, a performance tem um ganho considerável em relação ao *BiLSTM SynFeat*. Como esperado, a substituição de palavras por outras que carregam parte da semântica das originais teve um impacto positivo no desempenho do modelo.

A Tabela 10 mostra ainda que a diferença entre as taxas de compressão que apresentou melhores resultados foi a do modelo *EncBiLSTM SynFeat*. Esse valor de fato não é um significativo medidor de performance, mas sim uma espécie de parâmetro de controle. Valores próximos do zero são desejados, uma vez que isso significaria que o modelo testado conseguiu imitar perfeitamente o nível de compressão das sentenças com as quais ele foi treinado, contudo, valores próximos de zero também são admitidos como bons resultados. A situação preocupante se dá quando esse valor se distancia demais do zero, o que significaria que o modelo em questão está na realidade ou comprimindo demais as sentenças, descartando assim mais informação do que deveria, ou sendo muito conservador, mantendo informação irrelevante na sentença

comprimida. Portanto, apesar de o modelo *EncBiLSTM SynFeat* ter tido o melhor resultado, o resultado do *ERPR EncBiLSTM SynFeat+* também pode se caracterizar como positivo, ao passo que o resultado do *EncBiLSTM SynFeat+* pode sinalizar um comportamento indesejado.

As Figuras 16 e 17 mostram a progressão do desempenho dos modelos para *F1 score* e acurácia, respectivamente, ao longo de uma variação na quantidade de sentenças disponíveis para treinamento. Para o *F1 score* com 8000 sentenças de treinamento, como já foi discutido, todos os modelos, exceto o *ERPR EncBiLSTM SynFeat+* e o *BiLSTM SynFeat* tiveram performances muito semelhantes. À medida que a quantidade de sentenças disponíveis é reduzida, contudo, a performance do modelo *EncBiLSTM SynFeat+*, que possui essencialmente mais atributos que o *BiLSTM SynFeat*, já começa a se afastar dos demais, enquanto a performance do *ERPR EncBiLSTM SynFeat+* se mantém a maior. Para 2000 sentenças, a performance de todos os modelos caem bastante, mas os modelos *EncBiLSTM SynFeat+* e *ERPR EncBiLSTM SynFeat+* se mantêm superiores ao *BiLSTM SynFeat* e ao modelo proposto mais simples. O modelo *EncBiLSTM SynFeat*, por outro lado passa a ter uma performance inferior ao *BiLSTM SynFeat*, pois com quantidades tão inferiores de sentenças, o modelo *EncBiLSTM SynFeat* tem parâmetros demais para treinar e não possui dados suficientes para isso, enquanto o *BiLSTM SynFeat* por ter menos parâmetros consegue se sair melhor nesse cenário. Para a acurácia, os modelos *EncBiLSTM SynFeat+* e *ERPR EncBiLSTM SynFeat+* passam por todas as quantidades de sentenças superiores aos outros modelos, com o *ERPR EncBiLSTM SynFeat+* acima de todos.

Curiosamente, os valores de *F1 score* e acurácia para os modelos *EncBiLSTM SynFeat+* e *ERPR EncBiLSTM SynFeat+* praticamente se tocam quando treinados com 2000 sentenças. Isso ocorre, porque a quantidade de palavras raras é de certo modo proporcional ao tamanho do vocabulário, que por sua vez, é proporcional a quantidade de sentenças no conjunto de dados. Assim, quando o conjunto de sentenças é muito reduzido, o impacto de reduzir palavras raras também é menor, assim o modelo *ERPR EncBiLSTM SynFeat+* praticamente se reduz ao modelo *EncBiLSTM SynFeat+*. No entanto, para quantidades maiores de sentenças, a contribuição da substituição de palavras raras é clara. É importante notar que, a maior quantidade de sentenças utilizada neste trabalho ainda é mais de 200 vezes menor que as quantidades utilizadas em trabalhos anteriores, tais como (FILIPPOVA *et al.*, 2015; RUSH *et al.*, 2015; CHOPRA *et al.*, 2016).

Tabela 10 – Resultados dos modelos treinados com 8000 sentenças para métricas simples

Modelo	**F1 Score**	**Acurácia**	**DTC**
(WANG *et al.*, 2017)	0,80	0,82	2%
BiLSTM SynFeat (baseline)	0,79	0,81	2%
EncBiLSTM SynFeat	0,80	0,82	**1%**
EncBiLSTM SynFeat+	0,80	0,83	4%
ERPR EncBiLSTM SynFeat+	**0,82**	**0,84**	2%

Figura 16 – Relação entre o *F1 score* dos modelos e a quantidade de sentenças utilizadas para treinamento.

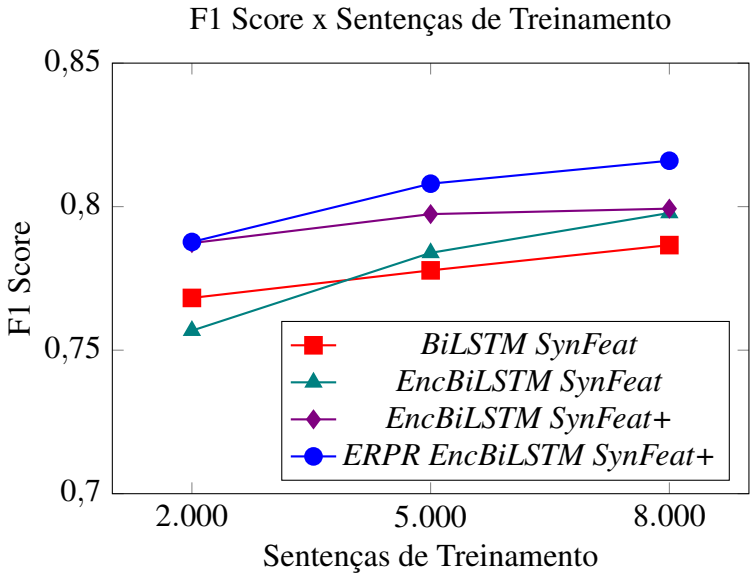
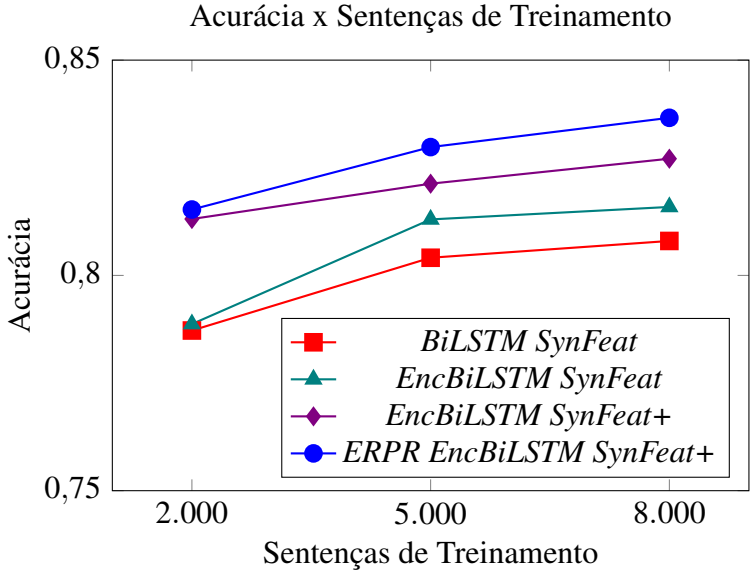


Figura 17 – Relação entre a acurácia dos modelos e a quantidade de sentenças utilizadas para treinamento.



5.5.2 Métricas de Qualidade das Compressões

A Tabela 11 mostra os resultados dos experimentos para as métricas de qualidade de compressão para cada um dos modelos examinados treinados com 8000 sentenças. A linha relativa ao trabalho original de (WANG *et al.*, 2017) nessa tabela está vazia porque o mesmo não reportou resultados sobre as métricas utilizadas.

A *BLEU* e a *ROUGE* são utilizadas neste trabalho com o objetivo de medir não só a capacidade dos modelos de escolherem as palavras corretas para formar as compressões das sentenças, mas também verificar como as sentenças comprimidas geradas se relacionam com as sentenças comprimidas de referência. A *BLEU*, como previamente apresentado, é uma métrica que foi proposta para o problema de tradução entre duas linguagens, como forma de verificar a qualidade de modelos de tradução, mas tem sido utilizada em outras tarefas que envolvam a geração de linguagem natural. É possível ver que existe uma clara progressão nos resultados dessa métrica para os modelos examinados. O modelo do *BiLSTM SynFeat* possui o menor resultado para essa métrica, o que pode acontecer devido a sua menor capacidade de entender as dependências entre as palavras. Essa falha é progressivamente melhorada com a adição da sentença codificada no modelo *EncBiLSTM SynFeat* e depois na adição de mais informações de dependência no modelo *EncBiLSTM SynFeat+*. Finalmente, o modelo *ERPR EncBiLSTM SynFeat+* é o que tem de longe o melhor resultado dessa métrica. Uma vez que muitas nomeadas são compostas, isto é, são constituídas de duas ou mais palavras, a estratégia de redução de palavras raras pode ter contribuído para facilitar a correlação entre as palavras de uma mesma entidade assim como deve ter facilitado a correlação entre verbos auxiliares e principais em locuções verbais.

A variação da *ROUGE* para unigramas, *ROUGE-1*, se comportou de maneira muito similar ao *F1 score* para todos os modelos. Isso se deve ao fato de que o *ROUGE-1* trabalha sobre unigramas, e possui um cálculo muito semelhante ao cálculo do *F1 score*. Algo muito semelhante acontece no caso do *ROUGE-L*, pois apesar de ela não trabalhar a nível de unigrama, ao calcular a subsequência mais longa ela chegará a um resultado parecido com o resultado do *ROUGE-1*. O motivo desse comportamento é que as sentenças comprimidas são sempre subsequências das sentenças originais, praticamente reduzindo uma métrica à outra. A *ROUGE-2*, por outro lado, como trabalha a nível de bigrama, obteve um resultado diferente das demais variações da *ROUGE*, resultado esse que corrobora com os resultados da *BLEU*.

A Figura 18 mostra a progressão do desempenho dos modelos para a métrica *BLEU*

ao longo de uma variação na quantidade de sentenças disponíveis para treinamento. Para 8000 sentenças, os resultados de todos os modelos são bem próximos, à exceção do *ERPR EncBiLSTM SynFeat+*. Esse comportamento é estranho, uma vez que o modelo *EncBiLSTM SynFeat+* na realidade tem sua performance melhorada ao treinar com 5000 sentenças, quando o esperado fosse que ele piorasse. Isso pode ter acontecido devido alguma anomalia de treinamento, no qual o modelo treinado para 8000 sentenças teve uma inicialização de parâmetros ruim e ficou preso em um mínimo local durante a otimização desses parâmetros. Assim como aconteceu com o *F1 score* e com a acurácia, com a redução da quantidade de dados de treinamento, a tendência dos modelos foi perder desempenho, inclusive copiando o mesmo comportamento visto na Figura 16 onde o resultado do *BiLSTM SynFeat* ultrapassa o do *EncBiLSTM SynFeat*, devido o excesso de parâmetros para treinar do segundo em relação ao primeiro. Um comportamento muito semelhante ocorre nos gráficos da Figura 19, onde é mostrada a progressão do desempenho dos modelos para as três variações da métrica *ROUGE* utilizadas ao longo de uma variação na quantidade de sentenças disponíveis para treinamento. Em todos os casos o *BiLSTM SynFeat* é o pior para 8000 sentenças de treinamento, mas com a redução na quantidade de dados ele e o *EncBiLSTM SynFeat* vão se aproximando de modo que com a menor quantidade de sentenças testada, o *BiLSTM SynFeat* se sai melhor que o outro. No fim, o modelo *ERPR EncBiLSTM SynFeat+* que utiliza os atributos extras para melhor mapear dependências entre palavras e a estratégia de redução de palavras raras é o que obtém melhores resultados, comparado com os outros, em qualquer uma das quantidades de sentenças avaliadas.

Tabela 11 – Resultados dos modelos treinados com 8000 sentenças.

Modelo	**BLEU**	**R-1**	**R-2**	**R-L**
(WANG *et al.*, 2017)	-	-	-	-
BiLSTM SynFeat	0,54	0,77	0,64	0,73
EncBiLSTM SynFeat	0,55	0,78	0,65	0,74
EncBiLSTM SynFeat+	0,56	0,78	0,67	0,74
ERPR EncBiLSTM SynFeat+	**0,61**	**0,81**	**0,70**	**0,77**

5.5.3 Análise Textual das Sentenças Comprimidas

A Tabela 12 mostra uma comparação entre o *BiLSTM SynFeat* e o *ERPR EncBiLSTM SynFeat+*. No primeiro exemplo, ambos os modelos geraram compressões de qualidade, isto é, nenhuma das duas possuem erros gramaticais ou falhas na legibilidade. Além disso, eles descartaram um pedaço de informação que, de fato, não é tão relevante para a sentença, mas

Figura 18 – Relação entre o valor da métrica BLEU dos modelos e a quantidade de sentenças utilizadas para treinamento.

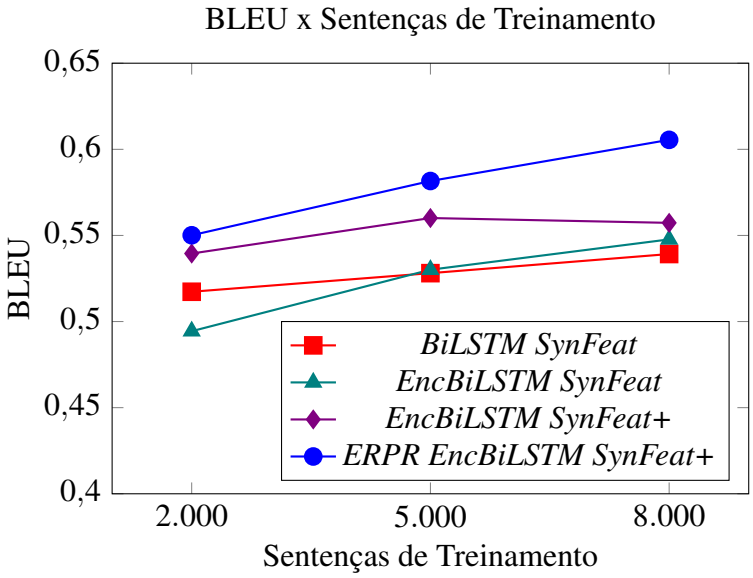
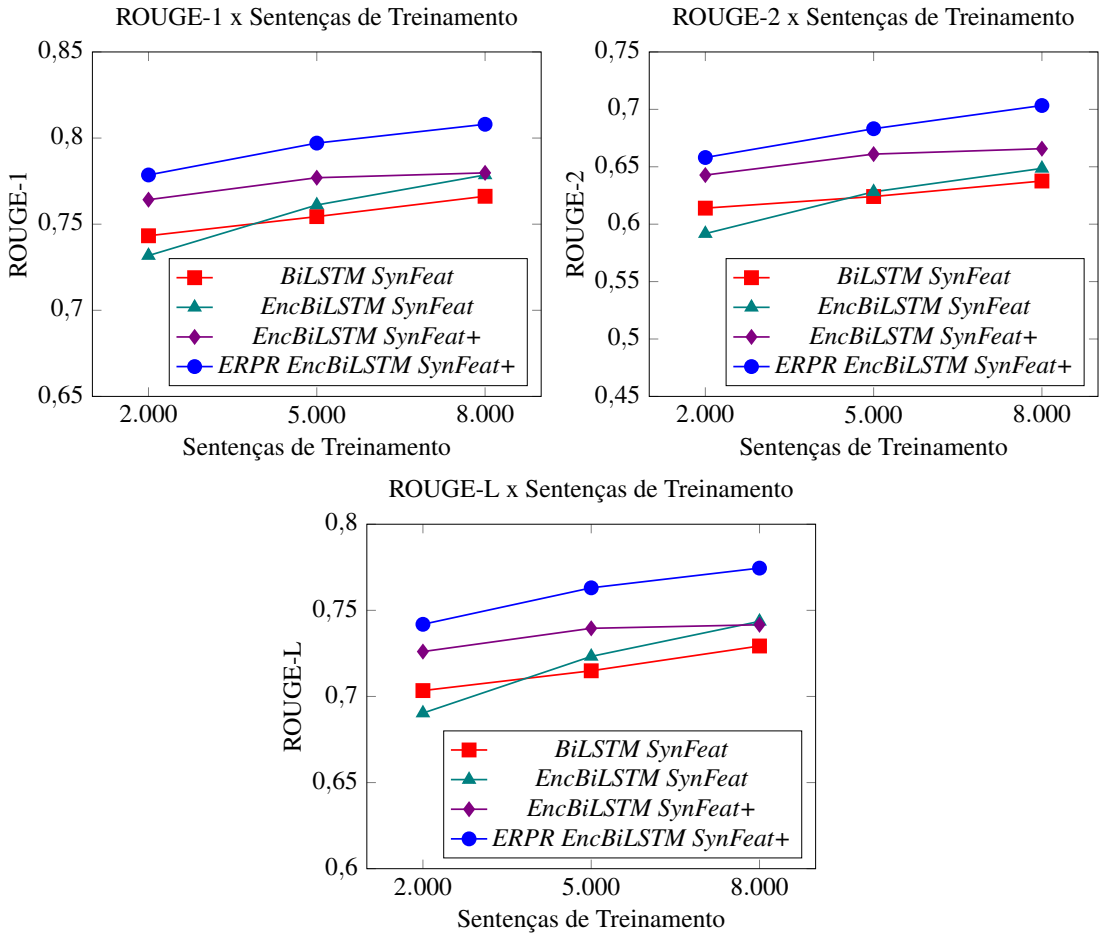


Figura 19 – Relação entre os valores da métrica *ROUGE* dos modelos e a quantidade de sentenças utilizadas para treinamento.



que a referência optou por manter. Isso mostra que, ainda que os modelos aprendam com compressões de referência, elas não são verdades absolutas, o que torna ainda mais complexo

fazer uma avaliação justa. No segundo exemplo o *BiLSTM SynFeat* comprimiu demais a sentença original o que acabou gerando uma sentença aparentemente incompleta, enquanto o modelo *ERPR EncBiLSTM SynFeat+* gerou uma compressão exatamente igual a referência. No terceiro exemplo acontece algo semelhante ao primeiro. A sentença do *BiLSTM SynFeat* está gramaticalmente correta e coerente e conseguiu descartar ainda mais informação irrelevante que a referência. O modelo *ERPR EncBiLSTM SynFeat+* novamente conseguiu reproduzir exatamente a mesma sentença da referência. Com esses três primeiros exemplos é possível notar que o *BiLSTM SynFeat* parece tender a comprimir mais as sentenças quando comparado com o *ERPR EncBiLSTM SynFeat+*, o que algumas vezes faz com que o segundo mantenha mais informação irrelevante como é possível ver no quarto exemplo.

O quinto exemplo mostra como ambos os modelos conseguem, em certas ocasiões, decidir melhor que a referência quanto a escolha de manter ou não certas palavras. No exemplo, a referência opta por retirar a palavra *refugee* da sentença, o que acaba por causar prejuízo ao sentido da sentença. Ambos os modelos, no entanto, optaram por mantê-la completando assim o sentido da palavra seguinte: *camp*. O último exemplo mostra uma sentença na qual ambos os modelos não obtiveram sucesso ao tentar comprimir. Interessante notar que os modelos decidiram por manter partes diferentes da sentença original. Nesse exemplo, nota-se que o modelo *ERPR EncBiLSTM SynFeat+* descartou o sujeito da sentença. Apesar de não ser um comportamento comum, isso também aconteceu em outras sentenças o que pode indicar que os modelos têm dificuldade de entender quando a informação mais importante da sentença não está diretamente ligada ao verbo principal dela, como é o caso desse exemplo.

Para um estudo mais horizontal, a Tabela 13 mostra a mesma comparação para um conjunto de 10 sentenças escolhidas aleatoriamente do conjunto de validação.

Tabela 13 – Comparação entre as sentenças aleatoriamente escolhidas do conjunto de validação.

Original: The Hill is reporting that Patriot Act author Rep. James Sensenbrenner Jr. wants Director of National Intelligence James Clapper fired and prosecuted for lying to Congress. **Referência:** James Sensenbrenner Jr. wants Director of James Clapper fired and prosecuted for lying. ***BiLSTM SynFeat*:** The Hill is reporting Patriot Act author Rep. James Sensenbrenner Jr. wants Director of National Intelligence James Clapper fired. ***ERPR EncBiLSTM SynFeat+*:** Patriot Act author Rep. James Sensenbrenner Jr. wants Director of National James Clapper fired and prosecuted for lying to Congress.
Original: Attempts to smuggle $1.3 million worth of heroin and 369 pounds of marijuana into the United States via San Diego ports of entry were thwarted on Monday, border officials announced this morning.

Referência: Attempts to smuggle $1.3 million worth of heroin and 369 pounds of marijuana were thwarted.
***BiLSTM SynFeat*:** Attempts to worth of heroin marijuana into were thwarted.
***ERPR EncBiLSTM SynFeat+*:** Attempts to smuggle $1.3 million worth of heroin marijuana were thwarted on.

Original: Helen Flanagan has confessed that she wants to get back with her ex-boyfriend Scott Sinclair.
Referência: she wants to get back with her ex-boyfriend Scott Sinclair.
***BiLSTM SynFeat*:** Helen Flanagan has she wants to get back with her ex-boyfriend Scott Sinclair.
***ERPR EncBiLSTM SynFeat+*:** she wants to get back with her ex-boyfriend Scott Sinclair.

Original: A man was robbed at knifepoint near Tulane University Monday night, according to a Tulane Police crime alert.
Referência: A man was robbed at knifepoint near Tulane University Monday night.
***BiLSTM SynFeat*:** A man was robbed at knifepoint near Tulane University.
***ERPR EncBiLSTM SynFeat+*:** A man was robbed at knifepoint near Tulane University.

Original: Sportswear company Adidas has suspended its contract with Tyson Gay after the former double world sprint champion failed an out-of-competition dope test.
Referência: Adidas has suspended its contract with Tyson Gay.
***BiLSTM SynFeat*:** Adidas has suspended its contract with Tyson Gay.
***ERPR EncBiLSTM SynFeat+*:** Adidas has suspended its contract with Tyson Gay.

Original: A Richmond man who was hit by a vehicle Saturday night is alive but still in critical condition at the University of Kentucky Medical Center.
Referência: A man who was hit by a vehicle is in critical condition.
***BiLSTM SynFeat*:** A Richmond man who was hit is alive but in critical condition Medical.
***ERPR EncBiLSTM SynFeat+*:** A Richmond man who was hit by a vehicle is alive in.

Original: POLICE have said they will take no action against Portsmouth councillor Mike Hancock after reading a dossier about an alleged sexual assault.
Referência: they will take no action against Portsmouth councillor Mike Hancock.
***BiLSTM SynFeat*:** POLICE have they will take no action against Portsmouth councillor Mike Hancock.
***ERPR EncBiLSTM SynFeat+*:** they will take no action against Portsmouth councillor Mike Hancock.

Original: The TMO & Steve Walsh, match referee, agree that a French forward pass has denied France a last minute try in the France v Ireland rugby match in Round 5 of the RBS 6 Nations at Stade de France, Paris on Super Saturday 15th March 2014.
Referência: a forward pass has denied France a last minute try in the France v Ireland match on 15th March 2014.
***BiLSTM SynFeat*:** The TMO & Steve Walsh agree a French forward pass has denied France a 15th 2014.
***ERPR EncBiLSTM SynFeat+*:** a French pass has denied France.

Original: Governor Mike Beebe is proposing a $5 billion budget for the coming year that would boost funding for education, human services and the state's prison system.
Referência: Governor Mike Beebe is proposing a $5 billion budget for the coming year.
***BiLSTM SynFeat*:** Mike Beebe is proposing a $5 billion budget for.

ERPR EncBiLSTM SynFeat+: Mike Beebe is proposing a $5 billion budget for.

Original: The first PlayStation 4s have gone on sale in North America, two weeks before they're released in Europe.
Referência: The first PlayStation 4s have gone on sale.
***BiLSTM SynFeat*:** The PlayStation 4s have gone on sale in America.
ERPR EncBiLSTM SynFeat+: The first PlayStation 4s have gone on sale in North America.

Tabela 12 – Comparação entre as sentenças comprimidas geradas pelos modelos examinados e as sentenças de referência.

Original: A 37-year-old woman was arrested Tuesday night after allegedly crashing her vehicle into CVS Pharmacy in Burbank while drunk, police said. **Referência:** A woman was arrested after allegedly crashing her vehicle into CVS Pharmacy while drunk police said. ***BiLSTM SynFeat*:** A woman was arrested after crashing her vehicle into CVS Pharmacy. ***ERPR EncBiLSTM SynFeat+*:** A woman was arrested after allegedly crashing her vehicle into CVS Pharmacy.
Original: Matthew McConaughey has won the Academy Award for best actor for his performance as a desperate and determined AIDS patient in "Dallas Buyers Club." **Referência:** Matthew McConaughey has won the Academy Award for best actor ***BiLSTM SynFeat*:** McConaughey has won the Academy Award for. ***ERPR EncBiLSTM SynFeat+*:** Matthew McConaughey has won the Academy Award for best actor.
Original: A teen beat a chihuahua with a wooden board after it approached his pit bull on Wednesday. **Referência:** A teen beat a chihuahua with a wooden board. ***BiLSTM SynFeat*:** A teen beat a chihuahua with a board. ***ERPR EncBiLSTM SynFeat+*:** A teen beat a chihuahua with a wooden board.
Original: SCMP reports that Gu Kailai's testimony against her husband, Bo Xilai, during his trial in Jinan this month will almost certainly lead to more charges against her for economic crimes. **Referência:** Gu Kailai's testimony against her husband Bo Xilai will lead to more charges against her. ***BiLSTM SynFeat*:** SCMP Gu Kailai's Bo Xilai will certainly lead to more charges against her crimes. ***ERPR EncBiLSTM SynFeat+*:** SCMP Gu Kailai's testimony will almost lead to more charges against her for economic crimes.
Original: President Mahmoud Abbas ordered Saturday dispatch of immediate food supplies to Yarmouk refugee camp in Syria due to the difficult situation the camp is going through. **Referência:** Mahmoud Abbas ordered dispatch of immediate food supplies to Yarmouk camp in Syria. ***BiLSTM SynFeat*:** Mahmoud Abbas ordered dispatch of immediate food supplies to Yarmouk refugee camp. ***ERPR EncBiLSTM SynFeat+*:** Mahmoud Abbas ordered dispatch of immediate food supplies to Yarmouk refugee camp.
Original: Johannesburg - Nelson Mandela's former aide Zelda la Grange deserves kudos for everything she did for the former president, Defence Minister Nosiviwe Mapisa-Nqakula said on Sunday. **Referência:** Zelda la Grange deserves kudos. ***BiLSTM SynFeat*:** Nelson Mandela's former aide. ***ERPR EncBiLSTM SynFeat+*:** deserves kudos for everything she.

6 CONSIDERAÇÕES FINAIS

6.1 Conclusão

Neste trabalho foi proposto um modelo de compressão de sentenças por remoção de palavras baseado em redes neurais capaz de atingir uma performance competitiva quando treinado com quantidades de dados inferiores ao que normalmente é utilizado na literatura para treinar esse tipo de modelo. O objetivo era encontrar uma forma de contra balancear o fato de o modelo não ter uma grande quantidade de exemplos diferentes nos quais se basear para aprender a tarefa de reduzir o tamanho de uma sentença apenas removendo palavras dela. Uma forma de alcançar esse objetivo era fazendo pelo modelo a extração de atributos, uma tarefa que normalmente é delegada à rede neural, mas que demanda quantidades massivas de dados.

Tendo isso em vista, um conjunto de atributos sobre as palavras de uma sentença foi elaborado e extraído durante o pré-processamento das sentenças para ser utilizado e estender um modelo do estado da arte construído utilizando uma pilha bidirecional de *LSTM*. Esse conjunto era composto de três tipos de atributos: semânticos, estruturais e de dependência. Os semânticos eram compostos pelos vetores de *embedding* das palavras e pela sentença codificada através um codificador de sentenças acoplado ao modelo principal e tinham o objetivo de carregar informações de significado e contexto para as palavras. Os estruturais, compostos pela classe gramatical, a posição e a quantidade de palavras dependentes de cada palavra, tinham como objetivo informar o modelo sobre a organização das sentenças, qual o papel individual de cada palavra e sua importância e posicionamento para a sentença. Por fim, os atributos de dependência tinham o objetivo de tornar explícitas as relações de dependência entre as palavras através da posição da palavra pai de cada palavra e do tipo de relação de dependência que elas formavam. Como acima mencionado, para adicionar a informação da sentença codificada, foi necessário acoplar ao modelo completo um módulo responsável por codificar a sentença em uma representação vetorial.

Contudo, extrair esses atributos resolvia apenas parte do problema. A presença de palavras com baixa frequência de ocorrência era outro fator dificultador do treinamento do modelo, principalmente para quantidades pequenas de dados que formavam grandes vocabulários. Nesses casos, a presença de um grande número de palavras que aparecem poucas vezes era um problema para o treinamento do modelo, uma vez que com tão pouca informação era muito difícil para o modelo tirar conclusões coerentes. Estratégias anteriores resolviam esse problema

simplesmente substituindo palavras com um baixa frequência de ocorrência por um *token* de *desconhecido*, mas isso acabava mapeando essas palavras para um vetor de *embedding* nulo, isto é, que não continha qualquer informação relevante sobre essas palavras. Para minimizar esse problema foi adotada uma estratégia de substituição de palavras mais inteligente. Ao invés de substituir todas as palavras com poucas ocorrências por um *token* de *desconhecido*, algumas delas foram substituídas por palavras que carregavam parte de sua semântica original e que possuíam uma probabilidade maior de ter um vetor de *embedding* não nulo. Todas as palavras que eram entidades nomeadas foram identificadas e substituídas por palavras em comum, por categoria, enquanto quaisquer outras palavras foram substituídas pelos seus lemas.

Com o objetivo de avaliar o desempenho desse modelo, ele foi utilizado para comprimir um conjunto de aproximadamente mil sentenças que possuíam versões suas comprimidas de referência. Sobre essas compressões geradas foi calculado um conjunto de métricas amplamente utilizadas tanto para avaliar o desempenho de modelos de compressão de sentenças por remoção de palavras quanto para avaliar tarefas de geração de textos em geral. Três modelos foram testados contra um modelo do estado da arte de compressão de sentenças baseado em redes neurais que serviu como *baseline*. O primeiro modelo era a versão mais simples do proposto. Era basicamente o *baseline* adicionando-lhe a sentença completa codificada como atributo extra. Apesar de ter uma performance superior a do *baseline* para quantidades razoáveis de dados (8.000 sentenças), este modelo teve um desempenho muito ruim com quantidades inferiores de dados, chegando a perder para o *baseline* com 2.000 sentenças. O segundo já utilizava todo o conjunto de atributos selecionados e, como esperado, já teve uma performance melhor que o *baseline* para toda a faixa de variação de quantidade de dados utilizada. O último modelo era a versão que utilizava os atributos selecionados e a estratégia de redução de palavras raras, e portanto a versão final do modelo proposto. Este obteve uma performance superior ao *baseline* e a todos os outros em todas as métricas avaliadas para 8.000 e 5.000 sentenças, mas teve uma performance muito similar ao segundo modelo para 2.000 sentenças, o que é justificado pelo vocabulário pequeno para esse caso e que, portanto, não teve benefícios significativos da estratégia de redução de palavras raras.

Uma avaliação textual elencou alguns pontos positivos e negativos da abordagem, comparada tanto com o *baseline* quanto com as sentenças de referência. O modelo proposto foi mais conservador e preciso que o *baseline* em muitas compressões, o que evitou em muitos casos que ele formasse sentenças gramaticalmente incorretas ou incoerentes. Em muitos casos ele foi

capaz de reproduzir perfeitamente as sentenças de referências ou, pelo menos, errar de modo que não afetasse a coerências da sentença gerada. Contudo, em sentenças nas quais a informação mais importante não estava diretamente ligada ao seu verbo principal ou em algumas sentenças que possuíam um complemento final, o modelo, assim como o *baseline*, teve dificuldades em gerar sentenças coerentes.

6.2 Trabalhos Futuros

Como trabalhos futuros, seria interessante experimentar outras formas, menos custosas de codificar uma sentença em uma representação vetorial. Treinar um codificador juntamento com o resto do modelo traz um custo que poderia ser evitado quando se está lidando com poucas quantidades de dados. A estratégia de redução de palavras raras trouxe uma melhoria significativa para o modelo e ela pode tanto ser evoluída quanto avaliada em outras tarefas de rotulação de sequências aplicadas a PLN. Além disso, existe a necessidade de avaliar as sentenças comprimidas pelos modelos avaliados quanto a outros critérios como a legibilidade das sentenças e sua informatividade, isto é, a quantidade de informação mantida na sentença comprimida. A manutenção da informação é uma característica importante da compressão de sentenças e, portanto é um fator determinante do exame de qualidade de qualquer modelo proposto para executar essa tarefa, contudo medir esse tipo de característica é uma tarefa árdua, uma vez que normalmente essa tarefa é feita através do uso de avaliadores humanos.

REFERÊNCIAS

ANTUNES, J. B. **Uma Abordagem para Sumarização Automática Semi-Extrativa**. Tese (Doutorado) — Universidade Federal de Pernambuco, 2018.

BAHDANAU, D.; CHO, K.; BENGIO, Y. Neural machine translation by jointly learning to align and translate. **CoRR**, abs/1409.0473, 2014.

BERG-KIRKPATRICK, T.; GILLICK, D.; KLEIN, D. Jointly learning to extract and compress. In: **Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies**. Portland, Oregon, USA: Association for Computational Linguistics, 2011. p. 481–490.

CHO, K.; MERRIENBOER, B. van; GULCEHRE, C.; BAHDANAU, D.; BOUGARES, F.; SCHWENK, H.; BENGIO, Y. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. 2014. ISSN 09205691.

CHOPRA, S.; AULI, M.; RUSH, A. M. Abstractive sentence summarization with attentive recurrent neural networks. In: **Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies**. San Diego, California: Association for Computational Linguistics, 2016. p. 93–98.

CLARKE, J.; LAPATA, M. Global inference for sentence compression an integer linear programming approach. **J. Artif. Int. Res.**, AI Access Foundation, USA, v. 31, n. 1, p. 399–429, mar. 2008. ISSN 1076-9757.

COHN, T.; LAPATA, M. Large margin synchronous generation and its application to sentence compression. In: **Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)**. Prague, Czech Republic: Association for Computational Linguistics, 2007. p. 73–82.

COHN, T.; LAPATA, M. Sentence compression beyond word deletion. In: **Proceedings of the 22Nd International Conference on Computational Linguistics - Volume 1**. Stroudsburg, PA, USA: Association for Computational Linguistics, 2008. (COLING '08), p. 137–144. ISBN 978-1-905593-44-6.

FILIPPOVA, K.; ALFONSECA, E.; COLMENARES, C. A.; KAISER, L.; VINYALS, O. Sentence Compression by Deletion with LSTMs. **Emnlp**, lstmsen, n. September, p. 360–368, 2015.

FILIPPOVA, K.; ALTUN, Y. Overcoming the lack of parallel data in sentence compression. In: **Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing**. Seattle, Washington, USA: Association for Computational Linguistics, 2013. p. 1481–1491.

FILIPPOVA, K.; STRUBE, M. Dependency tree based sentence compression. In: **Proceedings of the Fifth International Natural Language Generation Conference**. Salt Fork, Ohio, USA: Association for Computational Linguistics, 2008. p. 25–32.

FU, J.; LIU, G.; ZHU, J.; TAKEUCHI, I. **A Joint Selective Mechanism for Abstractive Sentence Summarization**. [S.l.], 2018.

GALANIS, D.; ANDROUTSOPOULOS, I. An extractive supervised two-stage method for sentence compression. In: **Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics**. Los Angeles, California: Association for Computational Linguistics, 2010. p. 885–893. Disponível em: <https://www.aclweb.org/anthology/N10-1131>.

GALLEY, M.; MCKEOWN, K. Lexicalized Markov grammars for sentence compression. In: **Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference**. Rochester, New York: Association for Computational Linguistics, 2007. p. 180–187.

GAMBHIR, M.; GUPTA, V. Recent automatic text summarization techniques: a survey. **Artificial Intelligence Review**, Springer, v. 47, n. 1, p. 1–66, 2017.

GRAVES, A.; JAITLY, N.; MOHAMED, A.-r. Hybrid speech recognition with deep bidirectional lstm. **2013 IEEE Workshop on Automatic Speech Recognition and Understanding**, p. 273–278, 2013.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. **Neural computation**, v. 9, p. 1735–80, 12 1997.

JING, H. Sentence reduction for automatic text summarization. In: **Proceedings of the Sixth Conference on Applied Natural Language Processing**. Stroudsburg, PA, USA: Association for Computational Linguistics, 2000. (ANLC '00), p. 310–315.

KINGMA, D. P.; BA, J. Adam: A method for stochastic optimization. **arXiv preprint arXiv:1412.6980**, 2014.

KNIGHT, K.; MARCU, D. Statistics-based summarization - step one: Sentence compression. In: **Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence**. [S.l.]: AAAI Press, 2000. p. 703–710. ISBN 0-262-51112-6.

LIN, C.-Y. ROUGE: A package for automatic evaluation of summaries. In: **Text Summarization Branches Out**. Barcelona, Spain: Association for Computational Linguistics, 2004. p. 74–81.

LUONG, M.; PHAM, H.; MANNING, C. D. Effective approaches to attention-based neural machine translation. **CoRR**, abs/1508.04025, 2015.

MARCUS, M. P.; SANTORINI, B.; MARCINKIEWICZ, M. A. Building a large annotated corpus of English: The Penn Treebank. **Computational Linguistics**, v. 19, n. 2, p. 313–330, 1993.

MARNEFFE, M.-C. de; DOZAT, T.; SILVEIRA, N.; HAVERINEN, K.; GINTER, F.; NIVRE, J.; MANNING, C. D. Universal Stanford dependencies: A cross-linguistic typology. In: **Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)**. Reykjavik, Iceland: European Language Resources Association (ELRA), 2014. p. 4585–4592.

MCDONALD, R. T. Discriminative Sentence Compression with Soft Syntactic Evidence. **Proceedings of EACL**, p. 297–304, 2006. ISSN 09020063.

MIKOLOV, T.; CHEN, K.; CORRADO, G.; DEAN, J. Efficient estimation of word representations in vector space. **arXiv preprint arXiv:1301.3781**, 2013.

NOTHMAN, J.; RINGLAND, N.; RADFORD, W.; MURPHY, T.; CURRAN, J. R. Learning multilingual named entity recognition from wikipedia. **Artificial Intelligence**, v. 194, p. 151 – 175, 2013. ISSN 0004-3702. Artificial Intelligence, Wikipedia and Semi-Structured Resources.

PANCHENDRARAJAN, R.; AMARESAN, A. Bidirectional LSTM-CRF for named entity recognition. In: **Proceedings of the 32nd Pacific Asia Conference on Language, Information and Computation**. Hong Kong: Association for Computational Linguistics, 2018.

PAPINENI, K.; ROUKOS, S.; WARD, T.; ZHU, W.-J. Bleu: A method for automatic evaluation of machine translation. In: **Proceedings of the 40th Annual Meeting on Association for Computational Linguistics**. Stroudsburg, PA, USA: Association for Computational Linguistics, 2002. (ACL '02), p. 311–318.

PASCANU, R.; MIKOLOV, T.; BENGIO, Y. Understanding the exploding gradient problem. **CoRR**, abs/1211.5063, 2012.

PENNINGTON, J.; SOCHER, R.; MANNING, C. Glove: Global vectors for word representation. In: **Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)**. Doha, Qatar: Association for Computational Linguistics, 2014. p. 1532–1543.

POWERS, D. M. W. Applications and explanations of Zipf's law. In: **New Methods in Language Processing and Computational Natural Language Learning**. [S.l.: s.n.], 1998.

PRABHAVALKAR, R.; RAO, K.; SAINATH, T.; LI, B.; JOHNSON, L.; JAITLY, N. A comparison of sequence-to-sequence models for speech recognition. In: . [S.l.: s.n.], 2017.

RUSH, A. M.; CHOPRA, S.; WESTON, J. A neural attention model for abstractive sentence summarization. In: **Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing**. Lisbon, Portugal: Association for Computational Linguistics, 2015. p. 379–389.

SRIVASTAVA, N.; HINTON, G.; KRIZHEVSKY, A.; SUTSKEVER, I.; SALAKHUTDINOV, R. Dropout: a simple way to prevent neural networks from overfitting. **The journal of machine learning research**, JMLR. org, v. 15, n. 1, p. 1929–1958, 2014.

SUTSKEVER, I.; VINYALS, O.; LE, Q. V. **Sequence to Sequence Learning with Neural Networks**. [S.l.], 2014.

TAS, O.; KIYANI, F. A survey automatic text summarization. **PressAcademia Procedia**, v. 5, n. 1, p. 205–213, 2007.

VENUGOPALAN, S.; ROHRBACH, M.; DONAHUE, J.; MOONEY, R. J.; DARRELL, T.; SAENKO, K. Sequence to sequence - video to text. **CoRR**, abs/1505.00487, 2015.

WANG, L.; JIANG, J.; CHIEU, H. L.; ONG, C. H.; SONG, D.; LIAO, L. Can syntax help? improving an LSTM-based sentence compression model for new domains. In: **Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)**. Vancouver, Canada: Association for Computational Linguistics, 2017. p. 1385–1393.

WEISCHEDEL, R.; PALMER, M.; MARCUS, M.; HOVY, E.; PRADHAN, S.; RAMSHAW, L.; XUE, N.; TAYLOR, A.; KAUFMAN, J.; FRANCHINI, M.; EL-BACHOUTI, M.; BELVIN, R.; HOUSTON, A. **OntoNotes Release 5.0**. [S.l.], 2013.

ZHOU, Q.; YANG, N.; WEI, F.; ZHOU, M. Selective encoding for abstractive sentence summarization. **Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)**, Association for Computational Linguistics, 2017.