

JOÃO PAULO REIS ALVARENGA

Orientador: Anderson Almeida Ferreira

**AVALIAÇÃO DE MÉTODOS DE TRANSFERÊNCIA DE  
APRENDIZADO APLICADOS A PROBLEMAS DE  
PROCESSAMENTO DE LINGUAGEM NATURAL EM  
TEXTOS DA LÍNGUA PORTUGUESA**

Ouro Preto  
Julho de 2019

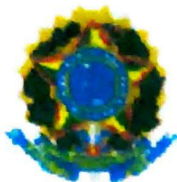
UNIVERSIDADE FEDERAL DE OURO PRETO  
INSTITUTO DE CIÊNCIAS EXATAS  
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

**AVALIAÇÃO DE MÉTODOS DE TRANSFERÊNCIA DE  
APRENDIZADO APLICADOS A PROBLEMAS DE  
PROCESSAMENTO DE LINGUAGEM NATURAL EM  
TEXTOS DA LÍNGUA PORTUGUESA**

Monografia apresentada ao Curso de Bacharelado em Ciência da Computação da Universidade Federal de Ouro Preto como requisito parcial para a obtenção do grau de Bacharel em Ciência da Computação.

JOÃO PAULO REIS ALVARENGA

Ouro Preto  
Julho de 2019




UNIVERSIDADE FEDERAL DE OURO PRETO

FOLHA DE APROVAÇÃO

Avaliação de Métodos de Transferência de Aprendizado Aplicados a Problemas de Processamento de Linguagem Natural em Textos da Língua Portuguesa

JOÃO PAULO REIS ALVARENGA

Monografia defendida e aprovada pela banca examinadora constituída por:

  
Dr. ANDERSON ALMEIDA FERREIRA – Orientador  
Universidade Federal de Ouro Preto

  
Dra. ANDREA GOMES CAMPOS BIANCHI  
Universidade Federal de Ouro Preto

  
Dr. GUILLERMO CÁMARA CHÁVEZ  
Universidade Federal de Ouro Preto

Ouro Preto, Julho de 2019

# Resumo

Este trabalho tem como objetivo avaliar estratégias baseadas em transferência de aprendizado e *fine-tuning*, para reduzir a quantidade de exemplos necessários para treinar modelos eficazes para análise de sentimentos na língua portuguesa, além de tentar entender o que é transferido entre os modelos. Os modelos avaliados são baseados em abordagens de pré-treinamento de modelos de língua em um corpus grande e genérico e o acoplamento de classificadores específicos a esses modelos de língua pré-treinados. Avaliando os modelos baseados em transferência de aprendizado, foi possível obter um F1 de 97,65% para a coleção Mercado Livre, 89,30% para a coleção Buscapé e 84,02% para a coleção de (Souza et al., 2016). Também foi avaliada a performance dos modelos quanto a redução dos conjuntos de dados, obtendo 97,50% de F1 utilizando cerca de 45% da coleção Mercado Livre e 88,29% de F1 usando cerca de 71% da coleção Buscapé. Foi realizada uma análise das camadas interna dos modelos baseados em atenção, em que é possível observar que alguns padrões que contribuem para análise de sentimento são herdados do modelo sem *fine-tuning*.

# Abstract

This work focus on evaluating strategies based on transfer learning and fine-tuning, searching for reducing the number of examples needed to train effective models for sentiment analysis in texts written in Portuguese language, as well as trying to understand what is transferred between the models. The models evaluated are based on pre-training approaches of language models in a large and generic corpus and the coupling of specific classifiers to these pre-trained language models. Evaluating the models based on learning transfer, we got an F1 of 97.65%, 89.30% and 84.02% for the Mercado Livre, Buscapé and Souza et al. (2016) datasets. It was also evaluated the performance of the models regarding the reduction of data sets, obtaining 97.50% F1 using about 45% of the Mercado Livre dataset and 88.29% F1 using about 71% of the Buscapé dataset. An analysis of the inner layers of attention-based models was performed, in which it is possible to observe that some patterns that contribute to sentiment analysis are inherited from the model without fine-tuning.

*Dedico esse trabalho a minha mãe e a minha irmã, e todos os que me apoiaram nesse árduo, porém valioso, percurso.*

# Agradecimentos

Agradeço a minha mãe e minha irmã pelo apoio incondicional.

Agradeço a todos os docentes e servidores da Universidade Federal de Ouro Preto por se esforçarem para manter a melhor infraestrutura possível.

Agradeço a todos os meus amigos pelos momentos de descontração e, principalmente, por serem apoio nos momentos mais complicados.

Agradeço ao meu orientador, Anderson, pela mentoria nesta fase final da minha graduação.

# Sumário

<b>1</b>	<b>Introdução</b>	<b>1</b>
1.1	Justificativa . . . . .	3
1.2	Objetivos Geral e Específicos . . . . .	3
<b>2</b>	<b>Fundamentação Teórica</b>	<b>4</b>
2.1	Processamento de Linguagem Natural . . . . .	4
2.2	Mineração de opinião . . . . .	5
2.3	Reconhecimento de entidades nomeadas . . . . .	6
2.4	Modelo de Língua . . . . .	6
2.5	Redes Neurais . . . . .	6
2.6	<i>Long Short-Term Memory</i> . . . . .	7
2.7	<i>Embeddings</i> . . . . .	10
2.8	<i>Discriminative fine-tuning</i> . . . . .	11
2.9	<i>Transformer</i> . . . . .	11
<b>3</b>	<b>Trabalhos Relacionados</b>	<b>14</b>
<b>4</b>	<b>Metodologia</b>	<b>18</b>
4.1	Performance . . . . .	18
4.1.1	Avaliação de Performance Geral . . . . .	18
4.1.2	Avaliação de Performance com Base Reduzida . . . . .	20
4.2	Transferência entre os modelos . . . . .	20
<b>5</b>	<b>Avaliação Experimental</b>	<b>22</b>
5.1	Recursos . . . . .	22
5.1.1	Córpus para Modelos de Língua . . . . .	22
5.1.2	Coleções de dados para Análise de Sentimento . . . . .	23
5.2	Avaliação dos Resultados . . . . .	24
5.3	Avaliação com <i>dataset</i> completo . . . . .	25
5.3.1	Abordagem ULMFiT . . . . .	25



5.3.2	Abordagem BERT . . . . .	26
5.3.3	Comparações com Resultados disponíveis na literatura . . . . .	28
5.4	Avaliação com redução do <i>dataset</i> . . . . .	30
5.4.1	Abordagem C-SVM . . . . .	30
5.4.2	Abordagem ULMPiT . . . . .	32
5.5	Análise dos padrões . . . . .	33
<b>6</b>	<b>Conclusão</b>	<b>37</b>
	<b>Referências Bibliográficas</b>	<b>39</b>

# Lista de Figuras

2.1	Esquema de pré-processamento de dados.(Mayo, 2017) . . . . .	5
2.2	Representação de rede neural de duas camadas (Dertat, 2017) . . . . .	7
2.3	Decidir qual informação do contexto deve ser descartada (Olah, 2015). . . . .	8
2.4	Definir qual nova informação será armazenada (Olah, 2015). . . . .	8
2.5	Atualizar os valores de estado da célula. (Olah, 2015). . . . .	9
2.6	Atualizar os valores de estado da célula. (Olah, 2015). . . . .	9
2.7	Representação do CBOW e do Skip-gram, respectivamente. (Mikolov et al., 2013a)	10
2.8	Ilustração da ideia do <i>encoder-decoder</i> (Cho et al., 2014). . . . .	12
2.9	Arquitetura Transformer, a esquerda o <i>encoder</i> a direita o <i>decoder</i> (Vaswani et al., 2017) . . . . .	12
3.1	Representação do ULMFiT, proposto por Howard e Ruder (2018) . . . . .	15
3.2	Taxa de aprendizado gerada pelo <i>Slanted Triangular Learning Rates</i> (Howard e Ruder, 2018). . . . .	15
4.1	Ferramenta para observar a conexão entre cada <i>token</i> da entrada, sendo possível também olhar cada camada de atenção e cada <i>head</i> de atenção (Vig, 2019). . . . .	21
5.1	Visualização das camadas de atenção dos modelo após <i>fine-tuning</i> . . . . .	34
5.2	Comparação da <i>head</i> 2 da camada 6 entre os modelos sem <i>fine-tuning</i> , com <i>fine-tuning</i> e sem pré-treino. . . . .	35
5.3	Visualização das camadas de atenção dos modelo após <i>fine-tuning</i> com um exemplo negativo. . . . .	35
5.4	Comparação da <i>head</i> 2 da camada 6 entre os modelos sem <i>fine-tuning</i> , com <i>fine-tuning</i> e sem pré-treino. . . . .	36

# Lista de Tabelas

5.1	Estatísticas da parcela pública do córpus (Hartmann et al., 2017). . . . .	23
5.2	Exemplo de matriz de confusão, em que VP é a quantidade verdadeiro positivos, FP a quantidade falso positivos, FN a quantidade de falso negativos e VN a quantidade de verdadeiro negativos. . . . .	24
5.3	Resultado médio da validação cruzada executada no <i>dataset</i> Mercado Livre . . . . .	25
5.4	Resultado médio da validação cruzada executada no <i>dataset</i> Buscapé . . . . .	26
5.5	Resultado médio da validação cruzada de 4 partições executada no <i>dataset</i> Souza et al. (2016) . . . . .	26
5.6	Resultado médio da validação cruzada executada no <i>dataset</i> Mercado Livre . . . . .	27
5.7	Resultado médio da validação cruzada executada no <i>dataset</i> Buscapé . . . . .	27
5.8	Resultado médio da validação cruzada executada no <i>dataset</i> (Souza et al., 2016). . . . .	27
5.9	Resultado médio da validação cruzada executada no <i>dataset</i> Mercado Livre . . . . .	28
5.10	Resultado médio da validação cruzada executada no <i>dataset</i> Buscapé . . . . .	28
5.11	Resultado médio da validação cruzada executada no <i>dataset</i> Souza et al. (2016). . . . .	28
5.12	Melhores resultados dos classificadores baseados em aprendizado de máquina (Avanço, 2015). . . . .	29
5.13	Resultado médio da validação cruzada no C-SVM (Avanço, 2015). . . . .	29
5.14	Comparação dos resultados entre as técnicas. . . . .	29
5.15	Resultado médio da validação cruzada com 4 partições executada no <i>dataset</i> Souza et al. (2016), utilizando C-SVM (Avanço, 2015). . . . .	30
5.16	Comparação dos resultados entre as aplicações de técnicas no <i>dataset</i> Souza et al. (2016). . . . .	30
5.17	Tamanhos das amostras por classe e tamanho total do conjunto para treino calculado usando tamanho da amostra multiplicado pela quantidade de classes. . . . .	31
5.18	Resultado médio das validações cruzadas para cada configuração de confiança e erro na técnica C-SVM (Avanço, 2015) . . . . .	31
5.19	Resultado do teste t para cada configuração no <i>dataset</i> Buscapé, usando o SVM (Avanço, 2015). . . . .	31
5.20	Resultado do teste t para cada configuração no <i>dataset</i> Mercado Livre, usando o SVM (Avanço, 2015). . . . .	32

5.21	Resultado médio das validações cruzadas para cada configuração de confiança e erro na técnica ULMFiT (Howard e Ruder, 2018). . . . .	32
5.22	Resultado do teste t para cada configuração no <i>dataset</i> Buscapé, usando o ULMFiT (Howard e Ruder, 2018). . . . .	33
5.23	Resultado do teste t para cada configuração no <i>dataset</i> Mercado Livre, usando o ULMFiT (Howard e Ruder, 2018). . . . .	33
5.24	Comparação para cada configuração de confiança e erro entre ULMFiT (Howard e Ruder, 2018). e C-SVM (Avanço, 2015). . . . .	33



# Capítulo 1

## Introdução

A computação surgiu como uma ferramenta para escalar a produtividade do ser humano e a cada ano vem aumentando sua capacidade e velocidade de processamento. Segundo Rodzvilla (2017), com o grande volume de dados, principalmente textual, 80% da informação de valor para um negócio está em textos não-estruturados, principalmente em redes sociais, com o grande crescimento destas. Além do cérebro humano, uma ferramenta importante para lidar com textos não-estruturados é a Análise de texto, do inglês *text analytics* (Rodzvilla, 2017). Essa tarefa é comumente relacionada com mineração de texto, do inglês *text mining*. Sendo que, mineração de texto trata de extrair dados de textos não-estruturados e análise de texto trata de descobrir e apresentar conhecimento.

Segundo Rodzvilla (2017), olhando para um conceito mais abrangente de análise de texto, extrair dados é uma parte importante para análise de texto, sendo necessário, para um bom trabalho na extração de dados, levar em conta elementos linguísticos e cognitivos do texto.

Extrair dados de textos exige uma série de tarefas provenientes do Processamento de Linguagem Natural (PLN), tais como, etiquetagem morfosintática (POS-Tagging), reconhecimento de entidades nomeadas (NER), categorização de sentenças, análise de sentimentos, dentre outras. Essas tarefas de PLN podem ser desenvolvidas utilizando uma abordagem baseada em regras e características escolhidas manualmente, ou em abordagens baseadas em aprendizado de máquina.

(Avanço, 2015) demonstrou que abordagens baseadas em aprendizado de máquina são mais eficazes do que abordagens baseadas em regras (léxico), quando se trata de análise de sentimento em redes sociais. Outro exemplo importante é o de (Màrquez et al., 2000), que experimentou uma abordagem baseada em aprendizado de máquina para etiquetagem morfosintática, demonstrando que, também para esta tarefa, abordagens baseadas em aprendizado de máquina superam abordagens baseadas em regras.

A construção de modelos baseados em aprendizado de máquina supervisionado pode ser resumida em três passos: a construção da base de dados para o aprendizado, em que é selecionada uma quantidade razoável de dados não-estruturados e estruturada manualmente uma

parcela desses dados; o treinamento da técnica de aprendizado de máquina, no qual, a partir dos dados manualmente estruturados é feito um pré-processamento e uma transformação desses dados, possibilitando que a técnica de aprendizado de máquina possa inferir um novo modelo preditivo; e a validação da técnica, em que medimos a qualidade da predição do modelo preditivo resultante em um novo conjunto de dados não-estruturados.

A estruturação de dados, ou simplesmente anotação, para aprendizado de máquina é uma tarefa extremamente custosa para o ser humano, dado que para obter bons resultados é necessária anotar uma grande quantidade de dados. Um exemplo é o conjunto de dados proposto em (Maas et al., 2011), que contém 50 mil exemplos anotados para análise de sentimentos, a partir de críticas de filmes. Nesse caso, cada crítica já possuía a polaridade previamente atribuída pelo próprio usuário escritor da crítica, um raro exemplo no universo de dados disponíveis. Quando é necessário construir um conjunto de dados para aprendizado de máquina a partir de dados complementamente não-estruturados, estruturando exemplo por exemplo, nos limitamos a produtividade normal do ser humano, tornando-se cada vez mais difícil alcançar grandes quantidades de dados anotados, como o exemplo do STS-Gold (Saif et al., 2013), um conjunto de dados gerado a partir de textos postados por usuários no Twitter, contendo, ao todo, 3 mil exemplos anotados manualmente. Para esse último conjunto de exemplos, usou-se um grupo de estudantes para o trabalho de anotação dos dados.

Um dos problemas chave para a eficácia dos modelos baseados em aprendizado de máquina, nas tarefas de mineração de opinião e reconhecimento de entidades nomeadas, é a necessidade de grandes volumes de dados. Uma técnica, apresentada por (Howard e Ruder, 2018), para reduzir a quantidade de dados estruturados e manter a performance do modelo, é a utilização de dados não-estruturados como uma base para o aprendizado de dados estruturados, dessa forma o modelo parte de uma base de conhecimento, ao invés de aprender todas as informações necessárias para a classificação desde o início.

Este trabalho propõe implementar e analisar os modelos e técnicas propostos por (Howard e Ruder, 2018; Devlin et al., 2018). Diferentemente de trabalhos anteriores relacionados a análise de sentimentos para língua portuguesa, este trabalho implementa e aplica modelos baseados em transferência de aprendizado e técnicas de *deep learning*, como modelos baseados em LSTM e modelos baseados em mecanismos de atenção. Mostrando que a transferência de aprendizado pode ajudar em cenários de conjunto de dados reduzidos, como avaliado neste trabalho na Subseção 5.4, em que reduzindo para 41% o conjunto de dados do Mercado Livre (Hartmann et al., 2014) é possível obter uma performance superior a de abordagens baseadas em aprendizado não profundo. Além disso, neste trabalho, também é mostrado que alguns padrões podem ser herdados da fase de pré-treinamento, ajudando o modelo na redução de tempo de treinamento e performance final.

## 1.1 Justificativa

A capacidade de aprender efetivamente a partir de um texto não-estruturado, em linguagem natural, é essencial para aliviar a dependência de aprendizado supervisionado para tarefas de PLN, segundo Howard e Ruder (2018). As técnicas atuais de *deep learning* geralmente requerem uma grande quantidade de dados anotados manualmente, o que restringe a performance desses modelos a qualidade desses poucos dados (Radford et al., 2018).

Segundo Yosinski et al. (2014), transferência de aprendizado vem se tornando essencial para modelos de imagem, melhorando a performance dos modelos, mostrando-se uma técnica viável para experimentação em outros problemas, como, por exemplo, classificação textual (Howard e Ruder, 2018).

Modelos chamados universais (Radford et al., 2018) têm sido explorados para resolver, por meio de aprendizado não-supervisionado e transferência de aprendizado, problemas como classificação de texto. Fixando o tamanho do dataset em 100 exemplos, para o problema de análise de sentimento utilizando a coleção de dados do IMDb, (Howard e Ruder, 2018) conseguiram alcançar a mesma performance comparando o modelo proposto ao modelo treinado utilizando 50 mil exemplos. Entretanto, a exploração de domínios textuais é dependente da língua, sendo necessário treinamento e novos conjuntos de dados para cada língua a aplicar o modelo.

A língua portuguesa apresenta uma reduzida quantidade de dados anotados manualmente, refletindo na baixa qualidade de ferramentas textuais, tais como tradutores, mineração de opinião, classificação automática de entidade nomeada, entre outros. Sendo assim, este trabalho pode ajudar a melhorar a qualidade dessas ferramentas e o entendimento do funcionamento de modelos, considerados “caixa-preta”, para a língua portuguesa.

## 1.2 Objetivos Geral e Específicos

De maneira geral, o objetivo deste trabalho é avaliar a performance de estratégias baseadas em transferência de aprendizado e *fine-tuning* para mineração de opinião em língua portuguesa, e analisar o que é transferido entre um aprendizado não-supervisionado e o supervisionado. São objetivos específicos:

1. Identificar análises necessárias para avaliar transferibilidade e as limitações das técnicas de *deep learning* baseadas em transferência de aprendizado e *fine-tuning* disponíveis na literatura;
2. Avaliar a performance das estratégias que realizam transferência de aprendizado de modelos de língua pré-treinados para mineração de opinião;
3. Analisar os padrões transferidos entre o modelo de língua e o modelo após o *fine-tuning*.



## Capítulo 2

# Fundamentação Teórica

Este capítulo apresenta a fundamentação teórica sobre transferência de aprendizado, processamento de linguagem natural e *deep learning*. Em cada seção seguinte é apresentada uma definição geral de cada conceito, bem como a aplicação do conceito para o tema principal deste trabalho.

### 2.1 Processamento de Linguagem Natural

Segundo (Goldberg e Hirst, 2017), Processamento de Linguagem Natural (PLN) é um termo coletivo que se refere ao processamento computacional automático de linguagens humanas, incluindo tanto algoritmos que recebem textos produzidos por humanos como entrada, bem como algoritmos que produzem textos parecidos com linguagem humana como saída. A capacidade de transformar textos em dados estruturados nos dá uma série de ferramentas que podem ser aplicadas em diversas áreas da tecnologia tais como: classificação textual, linguística aplicada, assistentes pessoais, análise de mídias, robótica e inteligência artificial genérica.

O principal desafio quando se trata de processar linguagens naturais, segundo Goldberg e Hirst (2017), é o alto nível de ambiguidade presente nessas linguagens. As melhores ferramentas computacionais conhecidas para ajudar a lidar com problemas de linguagem natural são algoritmos de aprendizado de máquina supervisionados, que trabalham buscando padrões a partir de um conjunto pré-annotado de dados.

Tratando-se de classificação de textos, além de ser um problema de PLN, um procedimento muito comum é utilizar de ferramentas de PLN, no pré-processamento dos dados utilizados, para construir um classificador baseado em aprendizado de máquina. Um esquema genérico deste procedimento pode ser visto na Figura 2.1, sendo: a transformação dos textos em sequências de *tokens* e segmentação responsáveis por identificar e separar o texto em unidades, seja esta unidade definida como, caractere, palavra, conjunto de palavras, sinais de pontuação, entre outros; A normalização, utilizada nesse caso para padronizar todo o texto, seja a ortografia

e gramática dos dados de entrada; e a remoção de ruídos, um método desenhado para remover *tokens* indesejados, exemplos fora do contexto da tarefa final de classificação.

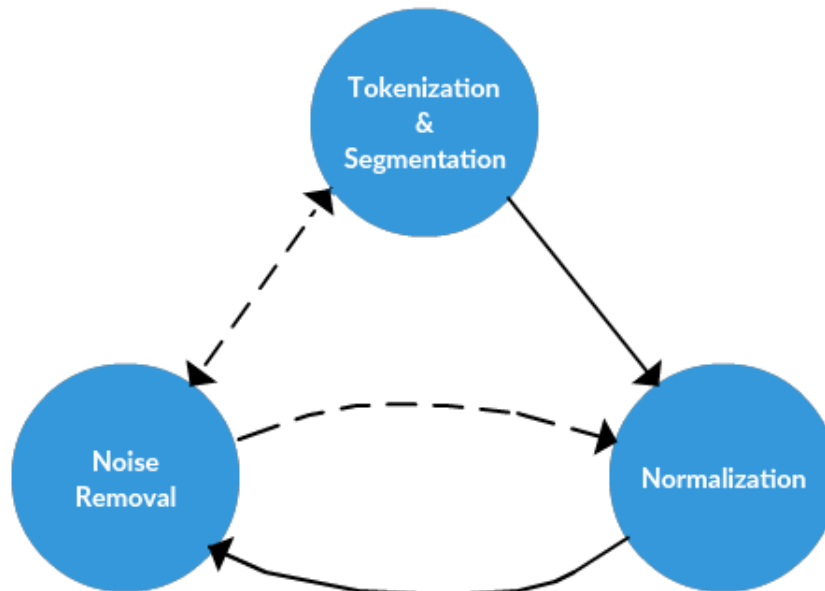


Figura 2.1: Esquema de pré-processamento de dados.(Mayo, 2017)

## 2.2 Mineração de opinião

Uma das tarefas avaliadas neste trabalho é mineração de opinião (MO), também conhecida como análise de sentimento. Segundo (Becker e Tuminan, 2013), ela pode ser dividida em 3 subtarefas:

- a) identificar as opiniões expressas sobre determinado assunto ou alvo em um conjunto de documentos;
- b) classificar a orientação ou polaridade dessa opinião, isto é, se tende a positiva ou negativa;
- c) apresentar os resultados de forma agregada e sumarizada.

Este trabalho analisa o desempenho da sub tarefa de classificar a polaridade de opiniões. Em alguns casos segmentando, as opiniões em positivas, neutras ou negativas, ou apenas entre positivas e negativas.

### 2.3 Reconhecimento de entidades nomeadas

Segundo (Tjong Kim Sang, 2002), entidades nomeadas são expressões que contêm nomes próprios, organizações, locais, datas e quantidades. Já (Jurafsky e Martin, 2009) dão uma definição mais genérica, definindo entidades nomeadas como qualquer coisa que possa ser referenciada por meio de um nome pertencente a uma categoria, sejam nomes de pessoas, de organizações ou de locais, expressões de tempo e data, quantidades, valores monetários, porcentagens, entre outros. Reconhecer entidades nomeadas, no contexto deste trabalho, é identificar em um texto quais *tokens* pertencem a algum tipo de entidade nomeada.

### 2.4 Modelo de Língua

Segundo Radford et al. (2018), um modelo de língua pode ser definido por, dado um *cópus* não-supervisionado, ou seja, contendo somente as sentenças, sem a necessidade de algum tipo de estruturação, contendo um conjunto de *tokens*  $U = \{u_1, \dots, u_n\}$ , tem-se como objetivo maximizar a probabilidade a seguir:

$$L_1(U) = \sum_i \log P(u_i | u_k, \dots, u_{i-1}; \theta) \quad (2.1)$$

em que  $k$  é o tamanho da janela de contexto, e a probabilidade condicional  $P$  é modelada usando redes neurais com parâmetros  $\theta$ , podendo estes parâmetros serem treinados com gradiente estocástico descendente, por exemplo. Em outras palavras, um modelo de língua pode ser descrito como um modelo capaz de reescrever o *cópus*, onde o objetivo é contextualizar os *tokens*, ou seja, dado um contexto, uma sequência de *tokens* de uma frase, qual é o *token* mais provável de ser o próximo *token* daquela sequência.

### 2.5 Redes Neurais

Redes Neurais é uma família de técnicas de aprendizado de máquina que são historicamente inspiradas no funcionamento do cérebro humano, que pode ser caracterizada como o aprendizado dos parâmetros de uma função diferenciável, segundo (Goldberg e Hirst, 2017). Comumente representada, em arquitetura, por uma rede de unidades, chamadas de neurônios, organizadas em camadas, que transformam uma entrada em uma saída determinada pela tarefa, como ilustrado na Figura 2.2.

Com o vasto uso de redes neurais e a crescente quantidade de dados disponíveis, uma série de novos tipos de neurônios foram criados para atender diversas demandas específicas, tais como: processamento de imagem (Lecun et al., 1998), problemas temporais (Azoff, 1994), processamento de texto (Socher et al., 2011), entre outros. Esses diversos tipos de neurônios podem ser combinados, afim de criar módulos responsáveis por resolver partes específicas dos

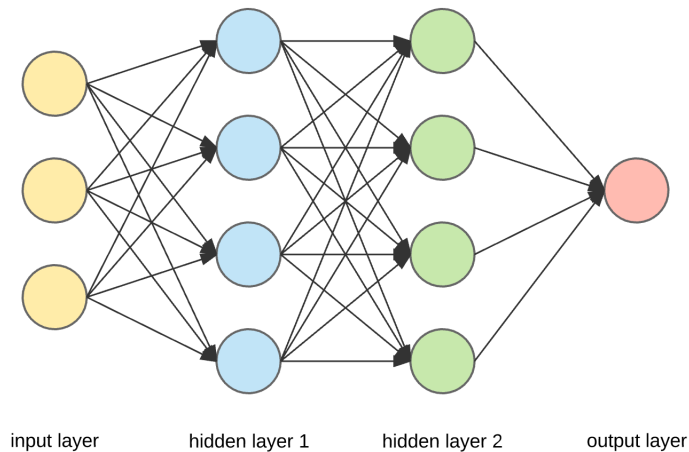


Figura 2.2: Representação de rede neural de duas camadas (Dertat, 2017)

problemas, a medida de que os dados são propagados pela rede, como é o caso de camadas de convolução em uma rede convolucional, inicialmente implementada para resolver problemas de processamento de imagem por (Lecun et al., 1998).

No contexto de PLN, uma arquitetura de rede muito utilizada, é a rede recorrente (RNN), uma rede especializada em dados sequenciais (Goldberg e Hirst, 2017). Essa arquitetura implementa um tipo de neurônio que tem como entrada uma sequência de itens e produz como saída uma sumarização dessa sequência de acordo com o que a unidade aprendeu no processo de treinamento. No contexto de PLN, essa sequência pode ser representada por uma sentença, sendo a entrada uma sequência de tokens desta sentença. Além disso, no processo de predição, os parâmetros aprendidos pela unidade são passados a unidades seguintes, mantendo o contexto em cada item da sequência.

## 2.6 *Long Short-Term Memory*

As RNN foram projetadas para armazenar o contexto durante o processo de predição. Entretanto, a janela de contexto varia de acordo com a tarefa, por exemplo, durante a predição de uma entrada textual, pode-se ter várias sentenças como entrada, no entanto, essas sentenças têm tamanhos diferentes, fazendo com que haja um problema de contextos muito longos. Uma solução para esse problema foi implementada por (Hochreiter e Schmidhuber, 1997), através de uma melhoria nas RNN, chamada de “*Long Short-Term Memory*“, LSTM. Uma LSTM tem como entrada a representação de um item  $x_t$  da sequência de entrada e o valor da saída da célula anterior  $h_{t-1}$ , que carrega o contexto da sequência,  $t$  representa o instante atual.

O primeiro passo, ilustrado na Figura 2.3, em uma célula de LSTM é decidir qual informação do contexto será descartada, utilizando uma camada sigmóide chamada de *forget gate layer*. Essa camada leva em consideração  $x_t$  e  $h_{t-1}$  e gera um número, entre 0 e 1, em

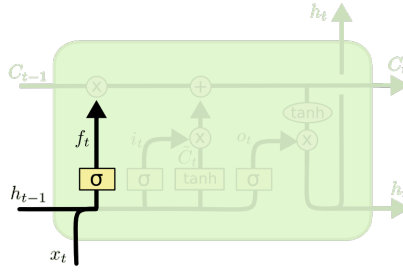


Figura 2.3: Decidir qual informação do contexto deve ser descartada (Olah, 2015).

que 0 indica que a informação de contexto deve ser totalmente descartada e 1 indica que a informação de contexto deve ser considerada em sua totalidade. Esse número é armazenado no estado da célula  $C_{t-1}$ . Representada na Equação 2.2.

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad (2.2)$$

O próximo passo é decidir qual nova informação de contexto será armazenada no estado da célula, esse passo é dividido em duas partes. Primeiro, um camada sigmoide chamada *input gate layer* decide quais valores serão atualizado, detalhado na Figura 2.4. Em seguida, uma camada de *tanh* cria um vetor de novos valores candidatos a estado de célula  $\tilde{C}_t$  (Equações 2.3 e 2.4).

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (2.3)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \quad (2.4)$$

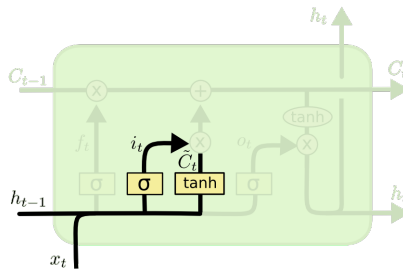


Figura 2.4: Definir qual nova informação será armazenada (Olah, 2015).

O passo seguinte, apresentado na Figura 2.5, atualiza o estado  $C_t$  para o estado  $\tilde{C}_t$ , gerado no passo, multiplicando o estado anterior  $C_{t-1}$  por  $f_t$ , definido na Figura 2.3, descartando do contexto o que foi decidido no primeiro passo (Equação 2.5). Então são adicionados os novos valores candidatos limitado pelo quanto foi decidido atualizar cada valor de estado,  $i_t * \tilde{C}_t$ , ilustrado na Figura 2.5.

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t \quad (2.5)$$

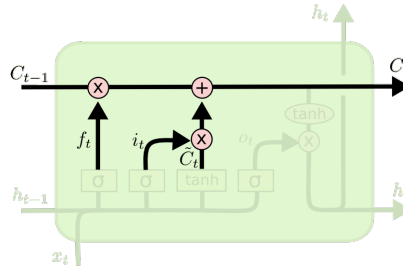


Figura 2.5: Atualizar os valores de estado da célula. (Olah, 2015).

O último passo é decidir a saída da célula (Figura 2.6), primeiramente uma camada sigmoide decide qual parte do estado de célula vai para saída, então é calculado o  $\tanh$  do estado de célula (Equações 2.6 e 2.7), afim de normalizar o valor para o intervalo entre -1 e 1, e multiplicado pela saída da camada sigmoide, fazendo com que a saída contenha apenas as partes do estado de célula escolhido.

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t + b_o]) \quad (2.6)$$

$$h_t = o_t \cdot \tanh(C_t) \quad (2.7)$$

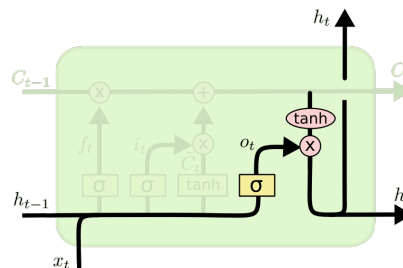


Figura 2.6: Atualizar os valores de estado da célula. (Olah, 2015).

O contexto dentro de uma LSTM flui como em uma linha de montagem, podendo sofrer pequenas modificações. O diferencial de uma LSTM é o poder de controlar o que será levado em consideração do contexto, podendo zerar completamente o contexto ou manter parte ou completamente o contexto. Controlando o fluxo do contexto e do dado de entrada dentro da LSTM é possível aprender quando um contexto deve ser substituído por um novo, ou mantido caso a sequência seja longa.

## 2.7 *Embeddings*

Textos são comumente representados por estruturas de dados baseados na frequência dos *tokens* pesados por um coeficiente de raridade no texto, como *Term Frequency–Inverse Document Frequency*, TF-IDF (Rajaraman e Ullman, 2011), entre outras. Entretanto, essas estruturas além de serem esparsas, geralmente não extraem características relacionadas a semântica do texto. O trabalho de (Mikolov et al., 2013b) implementa uma nova representação de textos, focando em gerar um espaço que mantém alguma propriedade dos dados, como, por exemplo, a morfologia dos textos, aproximando nesse espaço n-dimensional, tokens com a mesma função morfológica ou a semântica do textos, aproximando tokens semanticamente semelhantes.

A arquitetura de um modelo de *embeddings* é particularmente simples, onde se tem um *hierarchical softmax* (Mikolov et al., 2013b), treinado para, dado um *token*, predizer os *tokens* que compõem o contexto daquele *token*, numa janela  $N$  de *tokens*, no caso do modelo *Skip-gram* (Mikolov et al., 2013b). Também, largamente utilizado, implementa-se o *Continuous Bag of Words*, CBOW, (Mikolov et al., 2013a), diferindo-se do Skip-gram, o modelo prediz o token a partir do contexto, numa janela  $N$  de tokens.

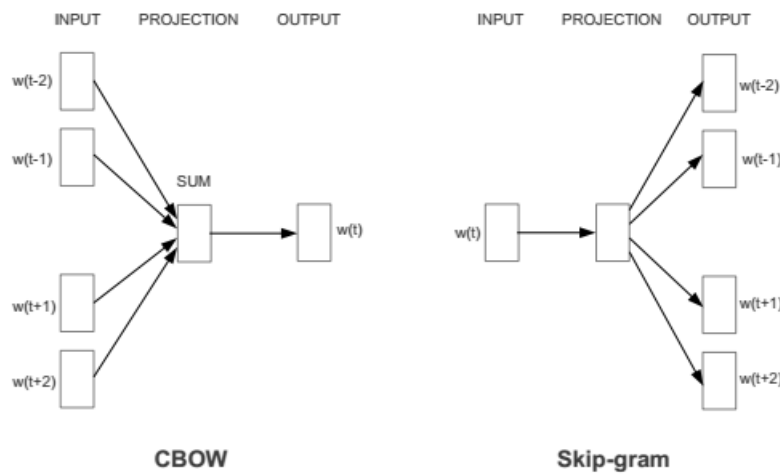


Figura 2.7: Representação do CBOW e do Skip-gram, respectivamente. (Mikolov et al., 2013a)

O grande diferencial dos modelos de *embeddings* comparados aos modelos clássicos TF e TF-IDF, é que além de *embeddings* ser uma representação menos esparsa, ela gera um universo de acordo com propriedades do texto. Por exemplo, em um modelo de *embeddings* treinado, dado os vetores representantes das palavras “Madrid“, “Espanha“ e “França“, a soma vetorial  $vetor(“Madrid”) - vetor(“Espanha”) + vetor(“França”)$ , resulta em um vetor mais próximo do vetor representante da palavra “Paris“ do que de qualquer outro vetor, segundo (Mikolov et al., 2013b), fazendo com que essa representação de dados extraia características importantes dos

dados além da quantidade de termos reguladas por um coeficiente de raridade, por exemplo.

Existe uma série de implementações e variações de modelos de *embeddings*, que consideram fatores diferentes para geração do universo como: word2vec (Mikolov et al., 2013b), a implementação original de *embeddings*, que toma como *token* as palavras; fastText (Joulin et al., 2016), uma implementação focada em agilizar o processamento de *embeddings* e que toma como token n-gramas de radicais de palavras, fazendo com que seja possível prever o vetor de uma palavra que não esteja presente no vocabulário original; GLoVe (Pennington et al., 2014), que a partir de uma matriz de co-ocorrência de palavras, implementa um modelo estatístico para os vetores de *embeddings*.

## 2.8 Discriminative fine-tuning

O *discriminative fine-tuning*, proposto em (Howard e Ruder, 2018), parte da análise que as diferentes camadas capturam diferentes tipos de afirmação (Yosinski et al., 2014). Logo, elas devem ser ajustadas de maneiras diferentes. Diferentemente do *fine-tuning* indutivo, a ideia do *discriminative fine-tuning* é ter uma taxa de aprendizado pra cada camada da rede ao invés de uma pra todos. Utilizando o mesmo exemplo de Howard e Ruder (2018), contextualizando, o gradiente descendente estocástico normal (SGD) atualiza os pesos  $\theta$  do modelo no passo  $t$ , como na equação a seguir:

$$\theta_t = \theta_{t-1} - \eta \cdot \nabla_{\theta} J(\theta) \quad (2.8)$$

em que  $\eta$  é a taxa de aprendizado e  $\nabla_{\theta} J(\theta)$  o gradiente em relação a função objetivo do modelo. O *discriminative fine-tuning* separa os pesos  $\theta$  entre as  $L$  camadas do modelo,  $\{\theta^1, \dots, \theta^L\}$ , em que  $\theta^l$  contém os pesos da  $l$ -ésima camada do modelo, separando também as taxas de aprendizado entre as  $L$  camadas do modelo,  $\{\eta^1, \dots, \eta^L\}$ , em que  $\eta^l$  é a taxa de aprendizado da  $l$ -ésima camada do modelo.

O passo de atualização do SGD, utilizando *discriminative fine-tuning*, se define como na equação a seguir:

$$\theta_t^l = \theta_{t-1}^l - \eta^l \cdot \nabla_{\theta^l} J(\theta) \quad (2.9)$$

## 2.9 Transformer

Uma alternativa às tradicionais abordagens, LSTM e convoluções, para problemas de sequência é a arquitetura Transformer. Proposta em (Vaswani et al., 2017), a *Transformer* também parte do princípio de dividir a arquitetura em duas partes, *encoder* e *decoder* (Cho et al., 2014), ilustrado na Figura 2.8. Conceitualmente, o *encoder* é responsável por extrair as características da entrada e codificá-las num vetor para, então, o *decoder* realizar a tarefa destinada, seja classificação, tradução, entre outros problemas.



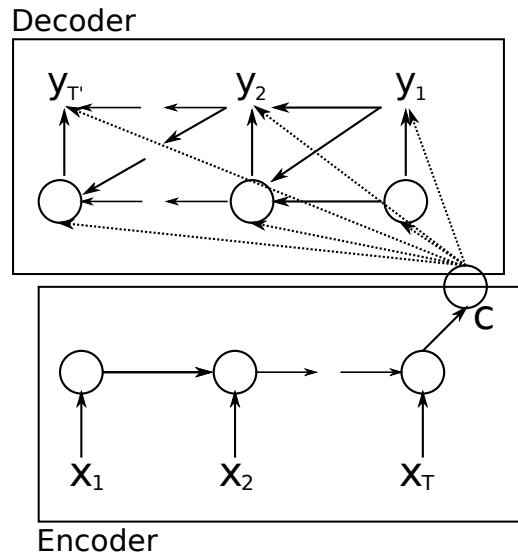


Figura 2.8: Ilustração da ideia do *encoder-decoder* (Cho et al., 2014).

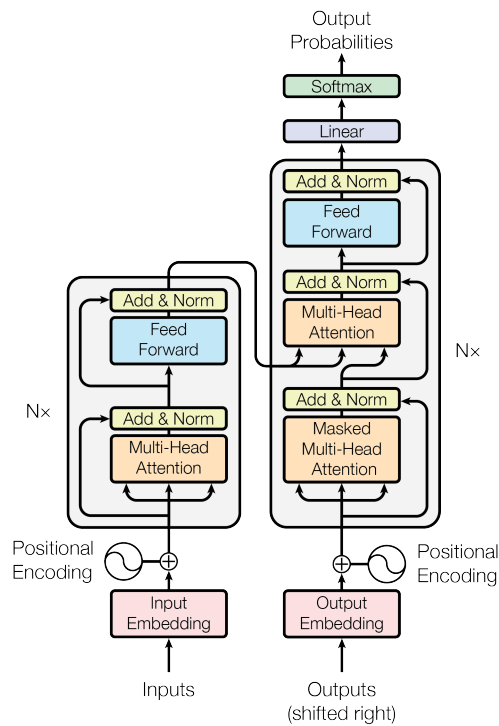


Figura 2.9: Arquitetura Transformer, a esquerda o encoder a direita o *decoder* (Vaswani et al., 2017)

A *Transformer* estrutura sua arquitetura implementando mecanismos de auto-atenção (do inglês *self-attention*), ilustrado na Figura 2.9. O mecanismo de auto-atenção proposto nessa arquitetura é implementado pela Atenção do Produto Escalar Escalonado (do inglês *Scaled Dot-Product Attention*). Que consiste em três elementos as *queries* (Q), as *keys* (K) e os *values*

(V). Cada um desses elementos são gerados através da multiplicação da entrada  $X$  com os parâmetros de cada um dos elementos, sendo  $W_Q$ ,  $W_K$  e  $W_V$ , parâmetros de *query* (Equação 2.10), *key* (Equação 2.11) e *value* (Equação 2.12), respectivamente:

$$Q = XW_Q \quad (2.10)$$

$$K = XW_K \quad (2.11)$$

$$V = XW_V \quad (2.12)$$

Em suma, essa camada computa o produto escalar de cada *query* com todas as *keys*, dividindo cada uma pelo  $\sqrt{d_k}$ , onde  $d_k$  é a dimensão dos vetores de *query* e, por fim, é aplicada a função de *softmax* para obter os pesos sobre os valores (Vaswani et al., 2017). A camada de auto-atenção se dá pela seguinte equação:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (2.13)$$

Na arquitetura *Transformer*, ao invés de implementar uma única função de atenção, Vaswani et al. (2017) propõem implementar o chamado *Multi-Head Attention*. A ideia é implementar várias funções de atenção e concatená-las ao final, sendo:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \dots, \text{head}_h)W^O$$

onde  $\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$

Segundo Vaswani et al. (2017), *Multi-head Attention* permite que o modelo consiga observar, conjuntamente, informações de diferentes subespaços de representação em diferentes posições.

## Capítulo 3

# Trabalhos Relacionados

Esse capítulo apresenta um compilado de trabalhos relacionados em duas abordagens, aprendizado profundo e aprendizado de máquina tradicional, sem o uso de redes neurais. Os trabalhos baseados em aprendizado profundo Howard e Ruder (2018); Radford et al. (2018); Devlin et al. (2018) estruturam e avaliam estratégias baseadas em transferência de aprendizado para texto, assim como essa monografia. Já os trabalhos Avançaço (2015); Teles et al. (2016); Cirqueira (2018) são baseados em aprendizado de máquina tradicional, não profundo, e avaliam análise de sentimento para o português, como proposto nesse trabalho.

Howard e Ruder (2018) observaram que, após implementar um modelo de língua (ML) pré-treinado em um contexto genérico, acoplar um classificador, que recebe como entrada o modelo de língua, é possível reduzir a quantidade de exemplos para um conjunto de dados 100 vezes menor que um conjunto de dados utilizado para treinar completamente classificadores, atingindo a mesma performance e, em alguns casos, melhorando a performance, reduzindo o erro de 18% a 24%, de tarefas como classificação mineração de opinião, classificação de perguntas e classificação de tópicos. O trabalho (Howard e Ruder, 2018) apresenta a ideia de treinar, com dados genéricos, um modelo de língua; aplicar um *fine-tuning* no modelo de língua com os dados da tarefa final; e acoplar nessa arquitetura um classificador e treiná-lo com os dados da tarefa final. Essa arquitetura se torna universal para qualquer tarefa textual, uma vez que o modelo é flexível quanto ao tipo do classificador final.

Além da arquitetura, o trabalho de (Howard e Ruder, 2018), introduz uma técnica de *slanted triangular learning rates*, taxas de aprendizado triangulares, que durante o treinamento, a taxa de aprendizado apresenta uma forma triangular, com um rápido crescimento no início do treinamento e um lento decréscimo até o final do treinamento, como ilustrado na Figura 3.2. Segundo Howard e Ruder (2018), essa técnica melhorou a performance do treinamento comparado ao treinamento com taxas de aprendizado uniforme.

O trabalho de Howard e Ruder (2018) implementa um novo modelo e técnica para minimizar a quantidade de exemplos e aumentar a performance, validando-o para conjuntos de dados da língua inglesa, diferentemente deste trabalho que foca em comparar e avaliar modelos de

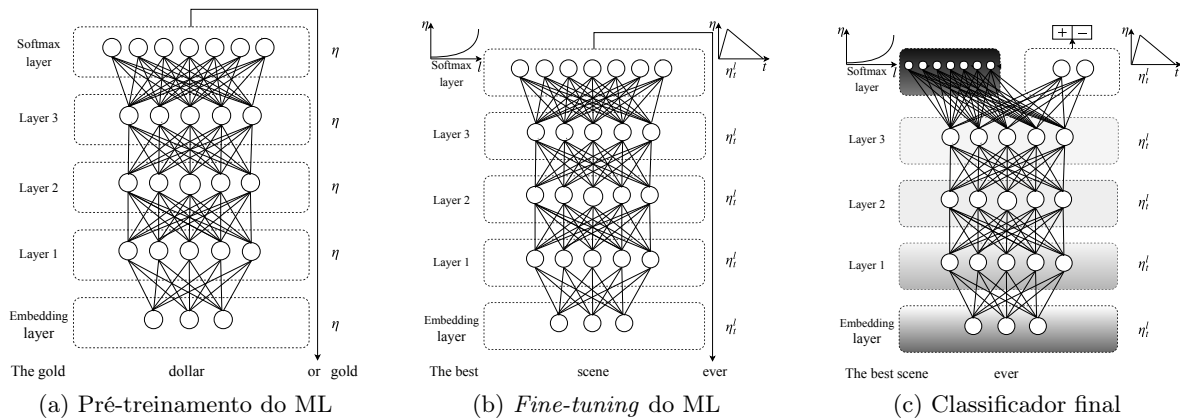


Figura 3.1: Representação do ULMFiT, proposto por Howard e Ruder (2018)

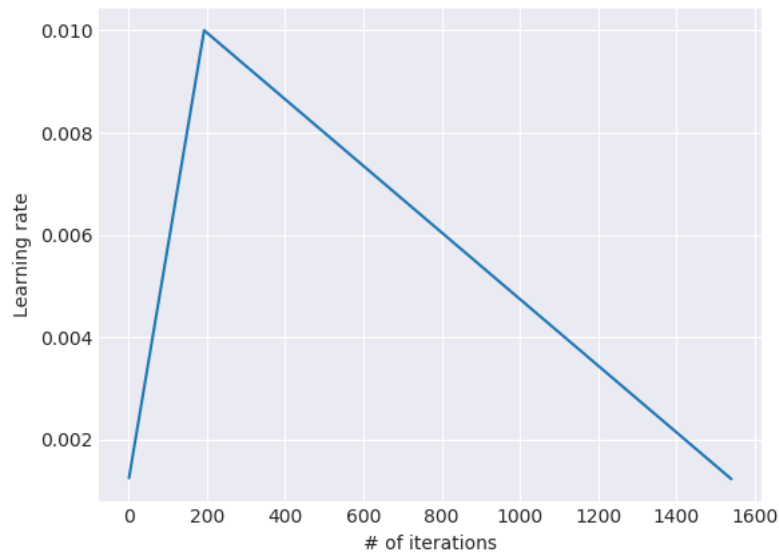


Figura 3.2: Taxa de aprendizado gerada pelo *Slanted Triangular Learning Rates* (Howard e Ruder, 2018).

transferência de aprendizado no limitado universo de recursos para a língua portuguesa, além de fazer uma análise sobre o que é transferido entre o modelo de língua em cada tarefa final.

Radford et al. (2018) avaliaram 4 tarefas de PLN, inferência de linguagem natural, sistemas de perguntas e respostas, similaridade de sentenças e classificações textuais, utilizando um modelo baseado em transferência de aprendizado. O trabalho implementou uma arquitetura de *Transformers*, comparando a performance com e sem auxílio de um modelo de língua pré-treinado. Este trabalho apresentou uma melhoria de 8.9% para tarefa de raciocínio de senso comum, 5.7% para tarefa de perguntas e respostas e 1.5% para classificação de dependência

textual. Diferentemente de Howard e Ruder (2018), Radford et al. (2018) implementaram uma arquitetura baseada em *Transformers*, além de avaliar 4 tarefas de PLN. Já este trabalho compara tanto o modelo de Howard e Ruder (2018) e o modelo de Radford et al. (2018) para tarefas de PLN em língua portuguesa.

Baseado na arquitetura *Transformers*, proposta por Vaswani et al. (2017), Devlin et al. (2018) propõem o BERT: *Bidirectional Encoder Representations from Transformers*, em tradução livre, Codificador Bidirecional de Representações de *Transformers*, uma abordagem não-supervisionada baseada em *fine-tuning*, em que é feito uma etapa pré-treino, com dados não-supervisionados, e uma segunda etapa com os dados supervisionados da tarefa final. Diferentemente das abordagens tradicionais de modelagem de língua, o BERT realiza duas tarefas em sua etapa não-supervisionada: Modelo de Língua com Máscara, do inglês *Masked LM* e Predição da Próxima Sentença, do inglês *Next Sentence Prediction* (NSP).

A ideia do Modelo de Língua com Máscara é, ao invés de predizer o próximo *token* dado os *tokens* anteriores, escolher aleatoriamente alguns *tokens* da sentença. Esses *tokens* são mascarados da sentença e a função do ML é predizer esses *tokens*, reconstruindo apenas os *tokens* mascarados da sentença. Essa tarefa é conhecida na literatura como tarefa de *Cloze* (Taylor, 1953). Nos experimentos, Devlin et al. (2018) mascararam apenas 15% de todos os *tokens*, aleatoriamente, em cada sequência. Entretanto, segundo Devlin et al. (2018), isso cria um desalinhamento entre o pré-treino e o *fine-tuning*, uma vez que a representação [MASK] do *token* não aparece durante o *fine-tuning*. Para resolver isso em apenas alguns casos o *token* é realmente substituído pelo *token* [MASK]. Os casos são selecionados da seguinte maneira: Em 80% das vezes, 15% das posições do *token* são escolhidas aleatoriamente, se a  $i$ -ésima posição for escolhida, o  $i$ -ésimo *token* é substituído pelo *token* [MASK]; Em 10% das vezes, o *token* é substituído por um *token* aleatório e 10% das vezes o *token* é mantido.

A NSP, segundo Devlin et al. (2018), parte da necessidade dos modelo de língua capturarem a relação entre duas sentenças, uma característica relevante para problemas como *Question Answering* (QA) e *Natural Language Inference* (NLI). Pensando em treinar um modelo que capture essa relação entre duas sentenças, basicamente, essa tarefa foi implementada de maneira que, para cada sentença do corpus e uma segunda sentença, o modelo foi treinado para classificar essa sentença como *IsNext*, caso fosse a sentença seguinte, ou *NotNext*, caso contrário. Em 50% das vezes, para cada sentença, a sentença para classificação era realmente a sentença e nas outras 50% era selecionada uma sentença aleatória que não era a sentença seguinte. Isso fez com que, nos experimentos de Devlin et al. (2018), os resultados melhorassem quando comparado a abordagem sem o NSP. O trabalho de Devlin et al. (2018) avalia a performance da arquitetura BERT para bases em inglês e com um corpus de pré-treino também em língua inglesa, diferentemente desse trabalho que avalia a performance da BERT para a língua portuguesa na tarefa de análise de sentimentos.

Avanço (2015) propôs uma estratégia baseada em normalização para análise de senti-

mento, utilizando o UGCNormal (Sanches Duran et al., 2015), um fluxo de processamento, com 4 principais partes, corretor ortográfico, utilizando dicionários para corrigir as palavras; mapeamento de acrônimos, identificando os acrônimos e colocando-os em caixa-alta; mapeamento do *internetês*, através de dicionários as palavras são transcritas da forma comumente apresentada na internet para a versão formal e o mapeamento de nomes próprios, as marcas e nomes citados nos exemplos são capitalizados no texto final.

A partir da normalização dos textos, Avanço (2015) avalia três tipos de técnicas para classificação, baseada em léxicos, baseada em algoritmos de aprendizado de máquina e uma abordagem híbrida. Em sua avaliação, utilizando a base de dados Buscapé e Mercado Livre construída por Hartmann et al. (2014), os melhores resultados estão compartilhados entre a abordagem baseada em aprendizado de máquina, comparando a abordagem bayesiana, *Naive Bayes* e o algoritmo linear Máquina de Vetores Suporte (SVM), sendo o SVM a melhor abordagem.

Teles et al. (2016) avalia estratégias baseadas em três técnicas de aprendizado de máquina, o Naive Bayes, SVM e Regressão Linear, avaliando dois tipos de variação de pré-processamento, a nível de caractere, eliminando e mantendo caracteres com acento e a nível de palavra, aplicando suavização de Lidstone (Chen e Goodman, 1996) e removendo *stopwords*. Alternando essas variações, ao final, Teles et al. (2016) avaliou 6 configurações de pré-processamento nas três técnicas de aprendizado de máquina na coleção de dados de Souza et al. (2016), essa coleção apresenta um conjunto de dados com três classes positivo, negativo e neutro e também apresenta um conjunto de dados apenas com duas classes positivo e negativo.

A técnica de melhor performance nos resultados de Teles et al. (2016) foi a SVM, para o conjunto com 2 classes, com a eliminação de acentos e suavização, obtendo 94,86% de F1 e para o conjunto com 3 classes, também eliminando acentos, entretanto aplicando a remoção de *stopwords*, obtendo 82,63% de F1. Os experimentos foram feitos utilizando validação cruzada de 4-partições.

Cirqueira (2018) avalia um *framework* de pré-processamento de texto, contendo 14 etapas: transformação das frases em sequência de *tokens*; normalização da capitalização, transformando os *tokens* com caracteres maiúsculos para minúsculos; remoção de acentuação; correção de pontuação, inserindo pontuação nas frases; remoção de pontuações; remoção de *stopwords*; tratamento de abreviações do *internetês*; correção ortográfica; correção ortográfica fonética; *stemming*, extração do radical das palavras; *POS Tagging*, identificação morfosintática; lematização, extração do radical através do rotulo dado pelo *POS tagger*; reconhecimento de entidades nomeadas; extração de aspectos. Após o pré-processamento esses exemplos são representados utilizando o *Bag-of-Words*. No trabalho é utiliza o SVM como técnica de classificação, experimentando 7 conjuntos de dados, alcançando para 90% para o conjunto de dados Souza et al. (2016) também avaliado nessa monografia. As avaliações de Cirqueira (2018) foram feitas utilizando validação cruzada de 10-partições.

## Capítulo 4

# Metodologia

Parte do problema de estratégias baseadas em técnicas de *deep learning* é analisar o comportamento dos modelos, muitas vezes chamados de “caixa-preta”, por serem difíceis de compreender o que de fato está armazenado em seus pesos. Este capítulo detalha os procedimentos e análises realizadas para avaliar a performance de estratégias implementadas e analisar o que é transferido entre os modelos de língua pré-treinados e o modelos finais.

### 4.1 Performance

Um dos pontos-chave de utilizar um modelo pré-treinado é aumentar a performance na tarefa e, segundo os experimentos de Howard e Ruder (2018), reduzir a quantidade de exemplos necessários para alcançar a mesma performance de um modelo treinado completamente. A proposta de Howard e Ruder (2018) inclui treinar um modelo de língua com um *cópus* genérico, realizar um *fine-tuning* deste modelo de língua, utilizando o *discriminative fine-tuning*, descrito na Sessão 2.8, utilizando os dados da tarefa objetivo e acoplar uma camada de classificação ao modelo específica para tarefa, treinando, agora, esta última camada do modelo com os rótulos do *dataset* da tarefa objetivo.

A análise de performance constitui de dois objetivos: avaliar de maneira geral, como as técnicas de transferência de aprendizado, baseadas em *Transformers*, BERT ou LSTM, ULMFiT, desempenham-se para a análise de sentimento; avaliar o comportamento do ULMFiT quanto a redução da quantidade de exemplos de treino.

#### 4.1.1 Avaliação de Performance Geral

Um dos objetivos desse trabalho é avaliar a performance de técnicas de transferência de aprendizado, comparando-as com os respectivos estados da arte. A avaliação desses modelos tem duas etapas:

1. Pré-treinar o modelo com dados não-supervisionados

2. Treinar os modelos realizando o *fine-tuning* para a análise de sentimento

O pré-treino do modelo é feito utilizando um corpus relativamente grande e genérico, comumente enciclopédias, livros e artigos (Howard e Ruder, 2018). É também necessária uma preparação do corpus para cada uma das técnicas. Tanto para o ULMFiT quanto para o BERT são realizados os seguintes passos:

1. Redução do corpus
2. Transformar as sentenças em sequência de *tokens*.
3. Gerar do vocabulário e transformar a sequência de *tokens* em uma sequência de índices, sendo que cada item da sequência remete ao índice da palavra correspondente através do vocabulário. Esse passo é necessário para gerar os *embeddings* de cada token.
4. Treinar o modelo de língua com os dados processados.

No caso do BERT, existe uma diferença na etapa de transformação das sentenças em sequência de *tokens*. É utilizada a ferramenta *SentencePiece*<sup>1</sup>, uma ferramenta que a partir dos dados não-supervisionado, utiliza, nesse trabalho, o *Unigram Language Model* (Kudo, 2018) para determinar quais serão as sub-palavras para o vocabulário final.

Na etapa de avaliação geral, o *fine-tuning* dos modelos é feito a partir do seguintes passos:

1. Limpeza dos dados nos mesmos moldes dos estados da arte.
2. Transformação das sentenças em sequências de *tokens*, dessa vez limitando os tokens ao vocabulário definido no pré-treino
3. Para o ULMFiT é feito o *fine-tuning* de todo o modelo com os dados da tarefa, sem considerar os rótulos. No caso do BERT, esse *fine-tuning* é feito durante o treinamento da camada de classificação do modelo.
4. Após o pré-treino, é acoplado uma camada de classificação nos modelos com pesos não treinados.
5. As camadas de classificação dos modelos são treinadas com os rótulos de sentimento.

Também para o *fine-tuning*, no caso do BERT, os modelos tem sua transformação de sentenças em *tokens* realizada pelo *SentencePiece*, dessa vez, também já treinado.

---

<sup>1</sup><https://github.com/google/sentencepiece>



### 4.1.2 Avaliação de Performance com Base Reduzida

Um segundo ponto na avaliação da performance é entender o comportamento dos modelos baseados em transferência de aprendizado quando reduzimos a quantidade de dados de treinamento. Para analisar a redução, os modelos são submetidos às mesmas condições de teste observados na literatura sobre as coleções de dados, seja validação cruzada ou *holdout*. Entretanto, os conjuntos de treinamento são reduzidos por meio de amostragens do conjuntos de treino. Os tamanhos escolhidos para a redução dos conjuntos de treino são selecionados a partir de um nível de confiança, uma margem de erro e a população, em que a quantidade da população é a quantidade de exemplos da maior classe. Dessa forma, é possível manter um conjunto de treino balanceado entre as classes.

$$\text{Tamanho da amostra} = \frac{\frac{z^2 \times p(1-p)}{e^2}}{1 + \left(\frac{z^2 \times p(1-p)}{e^2 N}\right)} \quad (4.1)$$

São realizadas as mesmas etapas de pré-treino e *fine-tuning* da avaliação geral (Subsessão 4.1.1). Entretanto, essa avaliação é feita somente para o ULMFiT por desempenhar uma performance melhor, para o treinamento, que o BERT.

## 4.2 Transferência entre os modelos

Um dos objetivos deste trabalho é analisar os padrões transferidos entre o modelo de língua e o modelo após o *fine-tuning*. Uma maneira de explorar as camadas internas do modelo é observando os parâmetros *queries* em conjunto com os parâmetros *keys*, cujo objetivo é estabelecer um relacionamento entre os *tokens* da sentença. A partir dessa percepção, Vig (2019) propõem uma ferramenta para visualização das camadas de atenção, chamada *BertViz*, sendo possível visualizar cada camada de atenção, cada *head* de atenção e os pesos entre os *tokens*.

Passando uma frase para a o modelo já treinado, a ferramenta plota 3 tipos de gráficos em que é possível comparar cada *token* da frase com ele mesmo, observando a intensidade das relações entre os *tokens*, que pode ser visualizado na ferramenta através da intensidade da cor das aretas entre os *tokens*, um exemplo pode ser visto na Figura 4.1, que mostra a relação aprendida entre cada *token* para cada camada e *head* do modelo. Essa intensidade é calculada a partir do produto *query* e *key*.

Utilizando a ferramenta de visualização, a análise dos padrões segue as seguintes etapas:

1. Pré-treino do modelo de língua. Esse modelo é reservado separadamente.
2. A partir do pré-treino é feito o *fine-tuning* com cada um dos *datasets* completos.
3. É selecionado um conjunto de sentenças dos próprios *datasets*.

4. Para cada uma dessas sentenças, são exploradas, através do *BertViz*, as relações de atenção, comparando o modelo *fine-tuning* com o modelo antes do *fine-tuning*, apenas com o treinamento não-supervisionado.

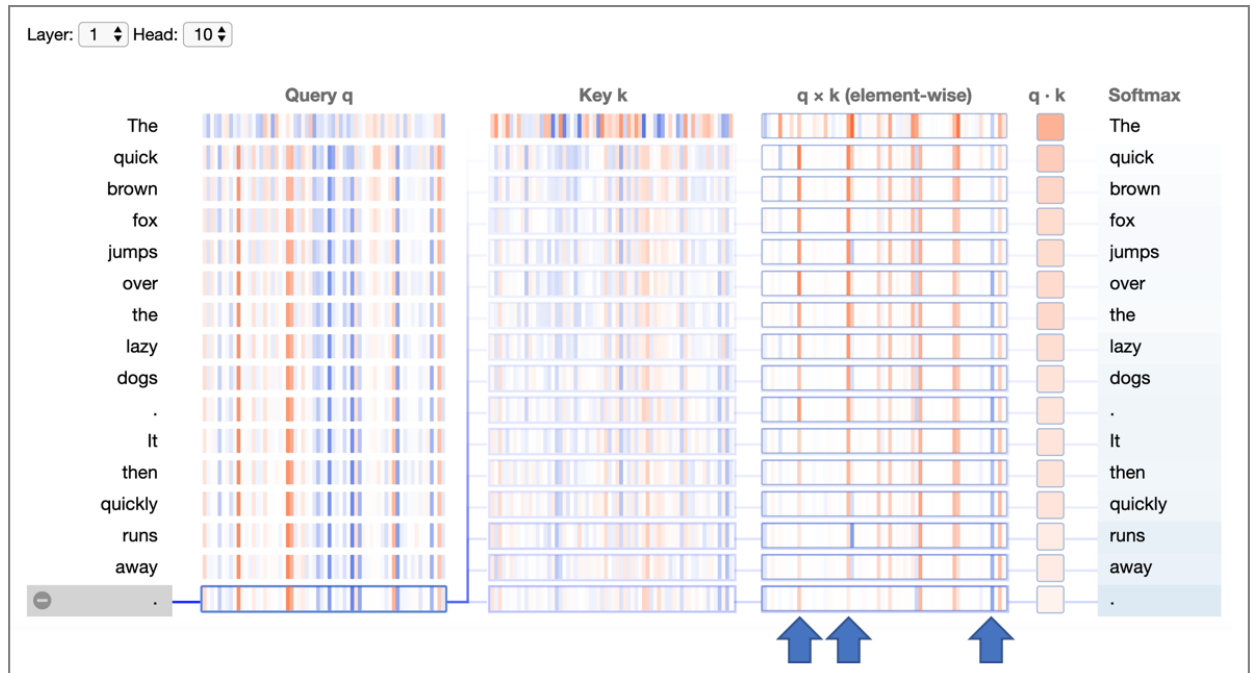


Figura 4.1: Ferramenta para observar a conexão entre cada *token* da entrada, sendo possível também olhar cada camada de atenção e cada *head* de atenção (Vig, 2019).

## Capítulo 5

# Avaliação Experimental

Este capítulo descreve os resultados dos experimentos realizados, bem como uma breve descrição dos recursos utilizados nos experimentos.

### 5.1 Recursos

Recursos para língua portuguesa são bem raros comparado ao vasto universo de recursos para língua inglesa. Um dos grandes desafios deste trabalho é levantar um corpus grande o suficiente para atender as necessidades do modelo de língua, bem como encontrar conjunto de dados para diferentes tarefas de classificação. Foi avaliada análise de sentimento, nesse trabalho, utilizando recursos disponibilizados pela literatura para cada uma das tarefas de classificação. Além dos recursos para as tarefas de classificação, para os modelos avaliados neste trabalho, também foi necessário recuperar um corpus genérico para treinamento dos modelos de língua universais.

#### 5.1.1 Córpus para Modelos de Língua

Um modelo de língua universal necessita de um córpus grande e diverso para conseguir atender as características que serão transferidas nas diversas tarefas de classificação. Para o treinamento do modelo de língua, foi utilizado a parcela pública do corpus disponibilizado por Hartmann et al. (2017). Este corpus contém artigos da Wikipédia, notícias, artigos de revistas, legendas em português, entre outros, detalhados na Tabela 5.1, totalizando cerca de 682 milhões de tokens. Foi aplicado ao corpus o mesmo pré-processamento utilizado por Hartmann et al. (2017), transformando palavras com menos de 5 ocorrências em um único token; normalizando os numerais para zero; URLs mapeadas para o token ‘URL’ e emails mapeados para o token ‘EMAIL’. Após as transformações, os exemplos foram representados em uma sequência de *tokens*, baseada em espaços e pontuações. Diferentemente da versão original do corpus de Hartmann et al. (2017), para o pré-treinamento dos modelos do nosso

trabalho, foi utilizada uma versão reduzida do corpus, totalizando 142.859.380 de tokens, cerca de 20% do total.

Tabela 5.1: Estatísticas da parcela pública do corpus (Hartmann et al., 2017).

Corpus	Tokens	Gênero	Descrição
Wikipedia	219293003	Enciclopédia	Dump do Wikipédia Português de 20/10/2016
GoogleNews	160396456	Notícias	Notícias coletadas do serviço GoogleNews
SubIMDB-PT	129975149	Linguagem falada	Legendas coletadas do site IMDb
G1	105341070	Informativo	Notícias coletadas do portal de notícias G1, entre 2014 e 2015
PLN-Br(Bruckschen et al., 2008)	31196395	Informativo	Grande corpus do projeto PLN-BR com textos amostrados de 1994 a 2005
Literatura de domínio público	23750521	Prosa	Uma coleção de 138268 trabalhos literários coletados do site Domínio Público
Lacio-web(Aluísio et al., 2003)	8962718	Gêneros variados	Textos de vários gêneros, por exemplo, literatura e suas subdivisões (prosa, poesia e romances), notícias, textos científicos, textos jurídicos e textos didáticos.
E-books em português	1299008	Prosa	Coleção de livros clássicos da ficção escritos em português brasileiro, coletados do site Literatura Brasileira
Mundo Estranho	1047108	Informativo	Textos coletados da revista Mundo Estranho
CHC	941032	Informativo	Textos coletados do site Ciência Hoje das Crianças (CHC)
FAPESP	499008	Divulgação Científica	Textos brasileiro de divulgação científica publicados na revista Pesquisa FAPESP
Livros didáticos	96209	Didático	Livros didáticos para crianças entre terceiro e sétimo ano do ensino fundamental
Folhinha	73575	Informativo	Notícias para crianças, coletado em 2015 da edição Folhinha do jornal Folha de São Paulo
Subcorpus NILC	32868	Informativo	Textos escritos para crianças entre terceiro e quarto ano do ensino fundamental
Para Sua Filho Ler	21224	Informativo	Notícias escritas para crianças do jornal Zero Hora
SARESP	13308	Didático	Questões de Matemática, Ciências Humanas, Ciências Naturais e redações.
<b>Total</b>	<b>682938652</b>		

### 5.1.2 Coleções de dados para Análise de Sentimento

Foram selecionadas duas coleções de dados disponíveis na literatura, uma disponibilizada pelo grupo de pesquisa MiningBR<sup>1</sup> no trabalho Souza et al. (2016), contendo *tweets*, na sua versão balanceada de duas classes, divididos em 166 exemplos rotulados positivos, 553 exemplos rotulados neutros e 1299 exemplos rotulados negativos, totalizando 2018 exemplos e a outra desenvolvida por Hartmann et al. (2014), constituída de análises de produtos publicados por usuários nas plataformas Buscapé<sup>2</sup> e Mercado Livre<sup>3</sup>, distribuído em, para o Buscapé, 6873 exemplos positivos e 6812 exemplos negativos, totalizando 13685 exemplos; para o Mercado Livre, 21819 exemplos positivos e 21499 exemplos negativos, totalizando 43318 exemplos.

<sup>1</sup><https://sites.google.com/site/miningbrgroup/home/resources>

<sup>2</sup><https://www.buscape.com.br>

<sup>3</sup><https://www.mercadolivre.com.br>

## 5.2 Avaliação dos Resultados

Comparar resultados de classificadores obtidos de maneiras distintas também é um problema. Segundo Avanço (2015), comparar sistemas de PLN tem sempre uma dificuldade imposta pela língua. Entretanto, observando tarefas mais consagradas que a mineração de opinião, os valores que representam o desempenho dos sistemas que solucionam essas tarefas mais consagradas, no estado da arte, não diferenciam muito entre línguas distintas.

Segundo Avanço (2015), para a mineração de opinião, avaliada na Seção 5.3, as principais métricas são: Precisão, Revocação, *F-Measure* e Acurácia. Essas métricas podem ser obtidas a partir de uma matriz de confusão, uma tabela que discrimina os tipos de erros e acertos cometidos pelo classificador, exemplificada na Tabela 5.2.

Tabela 5.2: Exemplo de matriz de confusão, em que VP é a quantidade verdadeiro positivos, FP a quantidade falso positivos, FN a quantidade de falso negativos e VN a quantidade de verdadeiro negativos.

		Real	
		Positivo	Negativo
Predito	Positivo	VP	FP
	Negativo	FN	VN

- Precisão: é a proporção de itens selecionados que o sistema acertou (Manning e Schütze, 1999). Precisão para uma classe  $c$  qualquer é dada por:

$$P_c = \frac{VP}{VP + FP} \quad (5.1)$$

- Revocação: é a proporção de de itens de uma classe que foram classificados como tal (Manning e Schütze, 1999). Revocação para uma classe  $c$  qualquer é dada por:

$$R_c = \frac{VP}{VP + FN} \quad (5.2)$$

- *F-Measure*: é a medida que combina a precisão e a revocação em uma única métrica, através da média harmônica entre as duas medidas (Manning e Schütze, 1999). *F-Measure* para uma classe  $c$  qualquer é dada por:

$$F_{\beta c} = (1 + \beta^2) \cdot \frac{P_c \cdot R_c}{\beta^2 \cdot P_c + R_c} \quad (5.3)$$

Comumente é utilizado  $\beta = 1$ , originando a medida chamada por *F1-Measure*, abreviada em F1. F1 para uma classe  $c$  qualquer é dada por:

$$F1_c = 2 \cdot \frac{P_c \cdot R_c}{P_c + R_c} \quad (5.4)$$

- Acurácia: é a proporção do total de acertos pela quantidade total de exemplos classificados. A acurácia de um classificador é dada por:

$$A = \frac{VP + VN}{VP + FP + VN + FN} \quad (5.5)$$

### 5.3 Avaliação com *dataset* completo

Os experimentos foram executados na plataforma Google Colab<sup>4</sup>, em máquina com processador Intel Xeon de 2.3 GHz, 12.6GB RAM e uma GPU Nvidia Tesla K80 ou TPUv2. Para este trabalho foram realizados experimentos com a tarefa objetivo mineração de opinião, utilizando a parcela pública do *corpus* do NILC, descrito na Subseção 5.1.1 e os *datasets* para mineração de opinião, Mercado Livre e Buscapé, descritos na Subseção 5.1.2. O código-fonte utilizado para o pré-treino e *fine-tuning* do ULMFiT está disponível no repositório do *FastAI*<sup>5</sup>. Para a execução dos experimentos com o BERT foi utilizada a implementação original em *Tensorflow*<sup>6</sup> e também a implementação em *PyTorch*<sup>7</sup>.

#### 5.3.1 Abordagem ULMFiT

Inicialmente, foi treinado o modelo de língua, proposto na arquitetura ULMFiT, utilizando os parâmetros: 0,001 de taxa de aprendizado, 12 épocas e 32 para o tamanho do lote. Esse treinamento foi realizado com os 20% da parcela pública do *corpus* do NILC (Subsessão 5.1.1).

Assim como Avanço (2015), foi realizada uma validação cruzada extratificada de 10 partes. Para cada parte, foi feito o *fine-tuning* do modelo de língua pré-treinado com o *corpus* do NILC. Durante o treinamento, foram mantidos os parâmetros padrões.

Por fim, após cada *fine-tuning* para cada parte da validação cruzada, foi treinada a última camada da arquitetura da ULMFiT, composta por uma rede neural totalmente conectada, resultando no classificador final, mantendo os parâmetros padrões, tamanho do lote 64, 50 épocas, taxa de aprendizado inicial 0,01, utilizando o *discriminative fine-tuning*.

Tabela 5.3: Resultado médio da validação cruzada executada no *dataset* Mercado Livre

Classe	Precisão	Revocação	F1	Suporte
<b>Positivo</b>	0,97483	0,97855	0,97667	21819
<b>Negativo</b>	0,97817	0,97432	0,97623	21499
<b>Média/Total</b>	0,97650	0,97643	0,97645	43318

Na Tabela 5.3, é possível observar os resultados da validação cruzada com o conjunto de dados Mercado Livre. Foi obtido 97,65% de F1 médio, e com as métricas Precisão e Revocação

<sup>4</sup><https://colab.research.google.com>

<sup>5</sup>[https://github.com/fastai/fastai/tree/master/courses/dl2/imdb\\_scripts](https://github.com/fastai/fastai/tree/master/courses/dl2/imdb_scripts)

<sup>6</sup><https://github.com/google-research/bert>

<sup>7</sup><https://github.com/huggingface/pytorch-pretrained-BERT>

de cada classe bem balanceadas em 97%. O mesmo balanceamento se repete para o conjunto de dados Buscapé, na Tabela 5.4 é possível observar um F1 médio de 88,40%. Diferentemente dos resultados pro Buscapé e Mercado Livre, na validação cruzada do conjunto de dados de Souza et al. (2016), os valores das métricas entre as classes estão desbalanceadas, comparando a precisão da classe Positivo, 69,72% com 75,58% da classe Negativo é possível notar uma diferença de 5,86%.

Tabela 5.4: Resultado médio da validação cruzada executada no *dataset* Buscapé

Classe	Precisão	Revocação	F1	Suporte
<b>Positivo</b>	0,88272	0,88742	0,88477	6866
<b>Negativo</b>	0,88645	0,88066	0,88325	6804
<b>Média/Total</b>	0,88459	0,88404	0,88401	13670

Tabela 5.5: Resultado médio da validação cruzada de 4 partições executada no *dataset* Souza et al. (2016)

Classe	Precisão	Revocação	F1	Suporte
<b>Positivo</b>	0,69722	0,74652	0,74334	166
<b>Negativo</b>	0,75838	0,68700	0,69967	166
<b>Média/Total</b>	0,73139	0,70833	0,70398	332

### 5.3.2 Abordagem BERT

Está disponível, junto com o código-fonte original do BERT, os modelos pré-treinados com corpus em inglês em seis versões, o BERT<sub>BASE</sub> contendo 12 camadas de *encoder-decoder* (6 *encoders* e 6 *decoders*) com 12 *heads* de *self-attention* cada e 768 camadas escondidas, somando 110 milhões parâmetros; O BERT<sub>LARGE</sub> contendo 24 camadas de *encoder-decoder* (12 *encoders* e 12 *decoders*) com 16 *heads* de *self-attention* cada e 1024 camadas escondidas, totalizando 340 milhões de parâmetros. A arquitetura BERT<sub>BASE</sub> está disponível em duas versões, com capitalização e sem capitalização. Já a BERT<sub>LARGE</sub> além da variação da capitalização possui versões com capitalização e sem capitalização aplicando as máscaras. Além das versões em inglês, também está disponível uma versão multilingual contendo as 100 mais frequentes línguas da Wikipédia, incluindo o português, disponível apenas para arquitetura BERT<sub>BASE</sub> variando entre com capitalização e sem capitalização.

Neste trabalho, foi avaliada a abordagem utilizando o modelo pré-treinado disponibilizado, utilizando a configuração multilingual com capitalização. A partir desse modelo foi feito o *fine-tuning* para os *datasets* Buscapé e Mercado Livre, também utilizando a mesma validação cruzada aplicada na Seção 5.3.1. O *fine-tuning* foi executado utilizando como configuração a taxa de aprendizado inicial 5e-6, tamanho máximo da sequência 512, tamanho do lote 16 e 4 épocas de treinamento. Os resultados da validação cruzada são mostrados nas Tabelas 5.6,

5.8 e 5.7. Esse treinamento utilizando a plataforma *Google Colab* levou cerca de 20 horas para cada um dos *datasets*, realizando o treinamento em GPU.

Tabela 5.6: Resultado médio da validação cruzada executada no *dataset* Mercado Livre

Classe	Precisão	Revocação	F1	Suporte
<b>Positivo</b>	0,97538	0,97796	0,97666	21819
<b>Negativo</b>	0,97759	0,97493	0,97625	21499
<b>Média/Total</b>	0,97649	0,97645	0,97646	43318

Analisando os resultados das métricas das validações cruzadas para o conjunto de dados Mercado Livre (Tabela 5.6, é possível notar o mesmo balanceamento entre as classes, apresentando um F1 médio de 97,66%. Já em relação ao conjunto de dados Buscapé, os resultados das métricas, apresentados na Tabela 5.7, mostram uma precisão da classe Negativo equivalente a revocação da classe Positivo, com cerca de 91% para as duas. Também é possível notar na Tabela 5.8, uma estabilização das métricas para 80% pro conjunto de dados (Souza et al., 2016).

Tabela 5.7: Resultado médio da validação cruzada executada no *dataset* Buscapé

Classe	Precisão	Revocação	F1	Suporte
<b>Positivo</b>	0,87579	0,91756	0,89605	6866
<b>Negativo</b>	0,91294	0,86846	0,88998	6804
<b>Média/Total</b>	0,89436	0,89301	0,89302	13670

Tabela 5.8: Resultado médio da validação cruzada executada no *dataset* (Souza et al., 2016).

Classe	Precisão	Revocação	F1	Suporte
<b>Positivo</b>	0,80159	0,84379	0,83766	166
<b>Negativo</b>	0,84347	0,79559	0,81974	166
<b>Média/Total</b>	0,83303	0,82927	0,82870	332

Além do modelo disponibilizado, foi pré-treinado um modelo utilizando os 20% da parcela pública do corpus do NILC (Subsessão 5.1.1). A configuração utilizada foi a mesma do BERT<sub>BASE</sub>, mantendo o mesmo pré-processamento original do corpus, resultando num modelo pré-treinado com capitalização. Diferentemente da configuração de (Devlin et al., 2018), foi utilizado um vocabulário com 32 mil *tokens*. Foram cerca de 140 horas de treinamento na plataforma *Google Colab* utilizando TPU.

Analisando os resultados das métricas para a validação cruzada do conjunto de dados Mercado Livre, apresentados na Tabela 5.9, é possível ver o mesmo comportamento balanceado entre métricas, comparando com os resultados das técnicas apresentados nas subseções anteriores, é possível observar que é um comportamento típico do conjunto de dados quando aplicado



Tabela 5.9: Resultado médio da validação cruzada executada no *dataset* Mercado Livre

Classe	Precisão	Revocação	F1	Suporte
<b>Positivo</b>	0,96933	0,97773	0,97351	21819
<b>Negativo</b>	0,97720	0,96860	0,97288	21499
<b>Média/Total</b>	0,97327	0,97317	0,97320	43318

a técnicas de aprendizado profundo. É possível observar também uma queda da precisão quando comparamos com o BERT pré-treinado com multilíngua, apresentado na Tabela 5.6.

Tabela 5.10: Resultado médio da validação cruzada executada no *dataset* Buscapé

Classe	Precisão	Revocação	F1	Suporte
<b>Positivo</b>	0,87757	0,91450	0,89555	6866
<b>Negativo</b>	0,91016	0,87110	0,89009	6804
<b>Média/Total</b>	0,89386	0,89280	0,89282	13670

Observando os resultados do Buscapé (Tabela 5.10 é possível ver a mesma equivalência entre a revocação da classe, Positivo e a precisão da classe Negativo, que os resultados do BERT multilíngua apresentados na Tabela 5.6. Tem um F1 médio de 89,28%.

Tabela 5.11: Resultado médio da validação cruzada executada no *dataset* Souza et al. (2016).

Classe	Precisão	Revocação	F1	Suporte
<b>Positivo</b>	0,88933	0,86731	0,85075	166
<b>Negativo</b>	0,87001	0,89112	0,82961	166
<b>Média/Total</b>	0,85188	0,84147	0,84018	332

O BERT treinado com o corpus do NILC (Hartmann et al., 2017) apresentou o melhor F1 médio, com 84,02%, com a revocação do Negativo chegando a 89,11%.

### 5.3.3 Comparações com Resultados disponíveis na literatura

Nos experimentos de Avanço (2015), os melhores resultados foram obtidos nas avaliações de classificadores baseados em aprendizado de máquina. A Tabela 5.12 mostra os resultados dos experimentos de classificadores baseados em aprendizado de máquina, alcançando, no seu melhor resultado, um F1 médio de 95,57% para o Mercado Livre e 89,10% para o Buscapé na técnica C-SVM, quando comparada ao Naive Bayes.

Os experimentos de Avanço (2015) foram realizados utilizando validação cruzada estratificada de 10 partes, com o código fonte disponibilizado na página do grupo de pesquisa Opinando<sup>8</sup>. Entretanto, na geração das partes, não foi disponibilizado a semente. Para comparação, o método C-SVM foi retreinado utilizando as mesmas partes utilizadas nos treinamentos da abordagem ULMFiT.

<sup>8</sup><https://sites.google.com/icmc.usp.br/opinando/>

Tabela 5.12: Melhores resultados dos classificadores baseados em aprendizado de máquina (Avanço, 2015).

Classificador	Dataset	F1-Positivo	F1-Negativo	F1-Média
C-SVM	Buscapé	<b>0,8935</b>	<b>0,8886</b>	<b>0,8910</b>
	Mercado Livre	<b>0,9564</b>	<b>0,9564</b>	<b>0,9557</b>
C-NB	Buscapé	0,8306	0,7925	0,8116
	Mercado Livre	0,9205	0,8989	0,9036

Retreinando o C-SVM foi possível obter 89,03% e 95,60% de F1 médio para os conjuntos Buscapé e Mercado Livre, respectivamente, apresentados na Tabela 5.13. Esses resultados estão bem próximos dos resultados reportados por Avanço (2015).

Tabela 5.13: Resultado médio da validação cruzada no C-SVM (Avanço, 2015).

Classe	Dataset	Precisão	Revocação	F1
Positivo	Buscapé	0,87794	0,90790	0,89260
	Mercado Livre	0,95418	0,95880	0,95647
Negativo	Buscapé	0,90790	0,87257	0,88791
	Mercado Livre	0,95802	0,95325	0,95561
Média	Buscapé	0,89292	0,89024	0,89026
	Mercado Livre	0,95610	0,95603	0,95604

Comparando os resultados da aplicação das técnicas nas coleções Buscapé e Mercado Livre, apresentado na Tabela 5.14 é possível observar que as técnicas baseadas em aprendizado profundo Howard e Ruder (2018); Devlin et al. (2018) se saíram melhor, em todas as métricas, quando comparadas ao trabalho original de Avanço (2015). Os resultados entre as técnicas de aprendizado profundo, são bem semelhantes. Entretanto, o BERT multilíngua, observando o F1 médio, apresenta um resultado superior para ambos os conjuntos de dados.

Tabela 5.14: Comparação dos resultados entre as técnicas.

Classificador	Dataset	F1-Positivo	F1-Negativo	F1-Média
C-SVM	Buscapé	0,89260	0,88791	0,89026
	Mercado Livre	0,95647	0,95561	0,95604
ULMFiT	Buscapé	0,88477	0,88325	0,88401
	Mercado Livre	<b>0,97667</b>	0,97623	0,97645
BERT <sub>Multi</sub>	Buscapé	<b>0,89605</b>	<b>0,88998</b>	<b>0,8930</b>
	Mercado Livre	0,97666	<b>0,97625</b>	<b>0,97646</b>
BERT <sub>PT</sub>	Buscapé	0,89555	<b>0,89009</b>	0,89282
	Mercado Livre	0,97327	0,97317	0,97320

Também foram avaliadas as técnicas com o conjunto de dados Souza et al. (2016). Dessa vez, para fins comparativos, foi adotada a validação cruzada com 4 partições, assim como Teles et al. (2016). Os resultados da técnica C-SVM com o conjunto de Souza et al. (2016)

obteve um F1 médio de 78,95% (Tabela 5.15), apresentando um resultado para classe Positivo melhor que a Negativo, 80,91% e 76,41% de F1 de cada classe, respectivamente.

Tabela 5.15: Resultado médio da validação cruzada com 4 partições executada no *dataset* Souza et al. (2016), utilizando C-SVM (Avanço, 2015).

Classe	Precisão	Revocação	F1	Suporte
<b>Positivo</b>	0,75923	0,86760	0,80909	166
<b>Negativo</b>	0,84530	0,72242	0,76410	166
<b>Média/Total</b>	0,80197	0,79269	0,78950	332

Comparando os resultados da aplicação das técnicas, apresentados na Tabela 5.16, é possível notar a melhor performance para a técnica BERT treinada com o corpus NILC (Hartmann et al., 2017). Entretanto, com resultados bem inferior comparado aos reportados por Teles et al. (2016), 94% de F1 médio.

Tabela 5.16: Comparação dos resultados entre as aplicações de técnicas no *dataset* Souza et al. (2016).

Classificador	F1-Positivo	F1-Negativo	F1-Médio
<b>C-SVM</b>	0,80909	0,76410	0,78950
<b>ULMFiT</b>	0,74334	0,69967	0,70398
<b>BERT<sub>Multi</sub></b>	0,83766	0,81974	0,82870
<b>BERT<sub>PT</sub></b>	0,85075	0,82961	<b>0,84018</b>

## 5.4 Avaliação com redução do *dataset*

A partir dos resultados obtidos por Howard e Ruder (2018), uma das hipóteses deste trabalho é avaliar se as técnicas baseadas em transferência de aprendizado a partir de treinamentos não-supervisionados podem ajudar a reduzir a quantidade de dados necessários para uma boa performance para tarefas de classificação textual. Para validar essa hipótese foram realizados experimentos reduzindo a quantidade de dados em cada parte da validação cruzada em seguida avaliando com o conjunto de teste completo de cada parte da validação cruzada.

Em cada *dataset*, foram escolhidos os níveis de confiança e margem de erro apresentados na Tabela 5.17. Para a validação cruzada, foi retirada uma amostra da base de treino de cada parte, mantendo integralmente as bases de teste. Foram utilizadas as mesmas partes da avaliação com o *dataset* completo.

### 5.4.1 Abordagem C-SVM

Utilizando a melhor técnica para os *datasets* Mercado Livre e Buscapé, encontrada no trabalho de Avanço (2015), foram realizados experimentos treinando o classificador com cada

Tabela 5.17: Tamanhos das amostras por classe e tamanho total do conjunto para treino calculado usando tamanho da amostra multiplicado pela quantidade de classes.

<i>Dataset</i>	População por classe	Nível de Confiança (%)	Margem de Erro (%)	Tamanho da amostra por classe	Tamanho do conjunto para treino
Buscapé	6866	99	1	4861	9722
			5	607	1214
		95	1	4004	8008
			5	364	728
Mercado Livre	21819	99	1	9441	18882
			5	646	1292
		95	1	6669	13338
			5	378	756

base de treino de cada parte da validação cruzada, adotando os mesmos procedimentos do trabalho original, utilizando o mesmo seletor de características e pré-processamento da base completa. Os resultados médio dos experimentos estão apresentados na Tabela 5.18. Também foram computados a significância entre os casos, utilizando o teste t de Student (Press et al., 1988), o valor p de cada comparação apresenta-se nas Tabelas 5.19 e 5.20. Observando que ao reduzirmos ao menor tamanho para o conjunto de dados Buscapé, chegamos a 79,23% de F1, cerca de 89% do F1 da validação cruzada com o conjunto de dados completo. Já para o conjunto de dados Mercado Livre, obtem 88,17% de F1, cerca de 92,23% do F1 da validação cruzada com o conjunto de dados completo.

Tabela 5.18: Resultado médio das validações cruzadas para cada configuração de confiança e erro na técnica C-SVM (Avanço, 2015)

<b>Dataset</b>	<b>Confiança(%) / Erro (%)</b>	<b>Precisão</b>	<b>Revocação</b>	<b>F1</b>
<b>Buscapé</b>	95 / 5	0,79275	0,79232	0,79230
	99 / 5	0,80780	0,80668	0,80661
	95 / 1	0,87524	0,87472	0,87472
	99 / 1	0,88260	0,88167	0,88169
<b>Mercado Livre</b>	95 / 5	0,87246	0,87241	0,87242
	99 / 5	0,87998	0,87994	0,87995
	95 / 1	0,93377	0,93378	0,93375
	99 / 1	0,94071	0,94066	0,94067

Analisando os teste, nas Tabelas 5.19 e 5.20, é possível observar que todas as hipóteses dos resultados serem iguais entre si, são rejeitadas utilizando um  $\alpha$  de 0,05.

Tabela 5.19: Resultado do teste t para cada configuração no *dataset* Buscapé, usando o SVM (Avanço, 2015).

<b>Confiança (%) / Erro (%)</b>	<b>99 / 1</b>	<b>95 / 1</b>	<b>99 / 5</b>
<b>95 / 5</b>	0,000000002108830386	0,00000003720209121	0,04922804206
<b>99 / 5</b>	0,00000003349974153	0,0000002134052889	
<b>95 / 1</b>	0,0003487953441		

Tabela 5.20: Resultado do teste t para cada configuração no *dataset* Mercado Livre, usando o SVM (Avanço, 2015).

Confiança (%) / Erro (%)	99 / 1	95 / 1	99 / 5
95 / 5	0,0001970675466	0,000360314656	0,0019942308
99 / 5	0,0002885451449	0,0008393404464	
95 / 1	0,01343557966		

#### 5.4.2 Abordagem ULMFiT

Aplicando a mesma metodologia para arquitetura ULMFiT, pré-treinando com 20% da parcela pública do corpus do NILC, foi realizado o *fine-tuning* e o treinamento do classificador para cada parte da validação cruzada com os dados de treino reduzidos. Os resultados estão apresentados na Tabela 5.21 e o teste t entre as configurações nas Tabelas 5.22 e 5.23.

Tabela 5.21: Resultado médio das validações cruzadas para cada configuração de confiança e erro na técnica ULMFiT (Howard e Ruder, 2018).

Dataset	Confiança(%) / Erro (%)	Precisão	Revocação	F1
Buscapé	95 / 5	0,84972	0,84919	0,84915
	99 / 5	0,85642	0,85603	0,85591
	95 / 1	0,88154	0,88155	0,88154
	99 / 1	0,88292	0,88290	0,88291
Mercado Livre	95 / 5	0,92464	0,92430	0,92416
	99 / 5	0,94343	0,94271	0,942615
	95 / 1	0,97283	0,97273	0,97276
	99 / 1	0,97496	0,97496	0,97495

O ULMFiT apresentou uma queda menor que a abordagem C-SVM quando compara-se o F1 do menor conjunto de treino do Buscapé, 84,92% com o F1 com o conjunto de dados completo, cerca de 95,07% do resultado com o conjunto de dados completo. Já os resultados do ULMFiT, quando compara-se o menor conjunto de treino com o conjunto de dados completos, apresenta um resultado de 94,67% do F1 com o conjunto de dados completo.

Analisando os resultados de testes, mostrados nas Tabelas 5.22 e 5.23, é possível observar que todas as hipóteses de os resultados serem iguais entre si, são rejeitadas utilizando um  $\alpha$  de 0,05, com a exceção da comparação entre 95/1 e 99/1, que possuem tamanhos bem próximos e um p-valor de 0,25.

A comparação das reduções realizadas para o ULMFiT e para o C-SVM, Tabela 5.24, mostra que o ULMFiT apresenta uma queda menor do F1, para o conjunto Buscapé, em média 1,13%, quando comparada a queda do C-SVM, em média 2,98%. O mesmo comportamento é notado no conjunto de dados do Mercado Livre, uma queda média de 1,69% e para o C-SVM 2,28%.

Tabela 5.22: Resultado do teste t para cada configuração no *dataset* Buscapé, usando o ULMFiT (Howard e Ruder, 2018).

Confiança (%) / Erro (%)	99 / 1	95 / 1	99 / 5
95 / 5	0,000001460821394	0,0000006616939155	0,008060369573
99 / 5	0,00003863409444	0,00003826951321	
95 / 1	0,24521357		

Tabela 5.23: Resultado do teste t para cada configuração no *dataset* Mercado Livre, usando o ULMFiT (Howard e Ruder, 2018).

Confiança (%) / Erro (%)	99 / 1	95 / 1	99 / 5
95 / 5	0,00001816078534	0,00002147074759	0,08279434063
99 / 5	0,006890534807	0,008918191243	
95 / 1	0,0931189375		

Tabela 5.24: Comparação para cada configuração de confiança e erro entre ULMFiT (Howard e Ruder, 2018). e C-SVM (Avanço, 2015).

Dataset	Confiança(%) / Erro (%)	F1 - ULMFiT	F1 - C-SVM
Buscapé	95 / 5	0,84915	0,79230
	99 / 5	0,85591	0,80661
	95 / 1	0,88154	0,87472
	99 / 1	0,88291	0,88169
Mercado Livre	95 / 5	0,92416	0,87242
	99 / 5	0,94262	0,87995
	95 / 1	0,97276	0,93375
	99 / 1	0,97495	0,94067

## 5.5 Análise dos padrões

Um dos objetivos desse trabalho é analisar os padrões transferidos entre o pré-treino do modelo e o *fine-tuning*. Seguindo a metodologia proposta neste trabalho, o modelo pré-treinado escolhido para as avaliações foi o BERT<sub>PT</sub>, pré-treinado com o 20% do corpus NILC (Hartmann et al., 2017). Para comparar, a partir do modelo pré-treinado, foi feito o *fine-tuning* com o conjunto de dados Mercado Livre, que demonstrou o melhor desempenho nas avaliações de validação cruzada. Entretanto, esse *fine-tuning* foi realizado com o conjunto de dados completo. Também foi realizado um treinamento do modelo sem o pré-treino.

Utilizando o *BertViz*, inserindo no modelo o exemplo positivo retirado do próprio conjunto de dados “Ótimo para empresa e para casa com crianças por possuir o sistema de bloqueio com chave.”, analisando o mapa de cada *head* de cada camada de atenção, é possível notar na *head* 0 da camada 0 (Figura 5.1), e na *head* 2 da camada 6 (Figura 5.1), uma relação bem forte entre a palavra “ótimo” e o restante dos *tokens*, analisando de maneira simplória, o ‘ótimo’ é

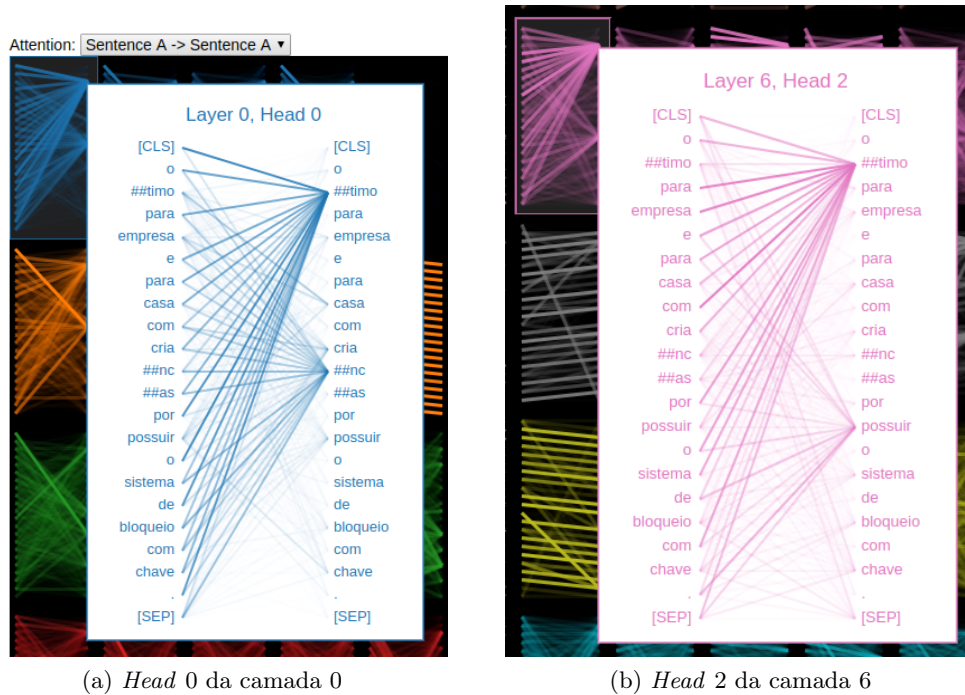


Figura 5.1: Visualização das camadas de atenção dos modelo após *fine-tuning*.

um dos fatores mais importantes para tornar a frase positiva. Comparando as camadas com o modelo apenas pré-treinado, é possível observar que essas relações já estavam presentes, entretanto foram reforçadas com o *fine-tuning*. Também comparando com o modelo treinado sem pré-treino, é possível observar que o modelo sem pré-treino não consegue aprender essas relações quando aplicamos a mesma quantidade de iterações no treinamento. Analisando a Figura 5.2 é possível ver que as cores das relações estão menos intensas, o que representa que os pesos aprendidos nas camadas de atenção não são relevantes o suficiente que essa *head* funcione como um filtro e aprenda alguma característica da tarefa.

Analisando a *head* 0 da camada 0 e a *head* 2 da camada 6 para um exemplo negativo do próprio conjunto de dados “É descartável, já comprei uns 2 e não passa de 3 semanas”, é possível ver novamente um destaque forte das relações com a palavra “descartável”, representados na Figura 5.3 pela intensificação das cores das aretas apontando para o início da palavra “descartável”, apresentando o mesmo comportamento de reforço quando comparado ao modelo sem *fine-tuning*.

Observando também a *head* 6 da camada 6, visto na Figura 5.4, é possível ver uma leve intensificação no relacionamento das palavras da frase com o início da palavra “descartável”.

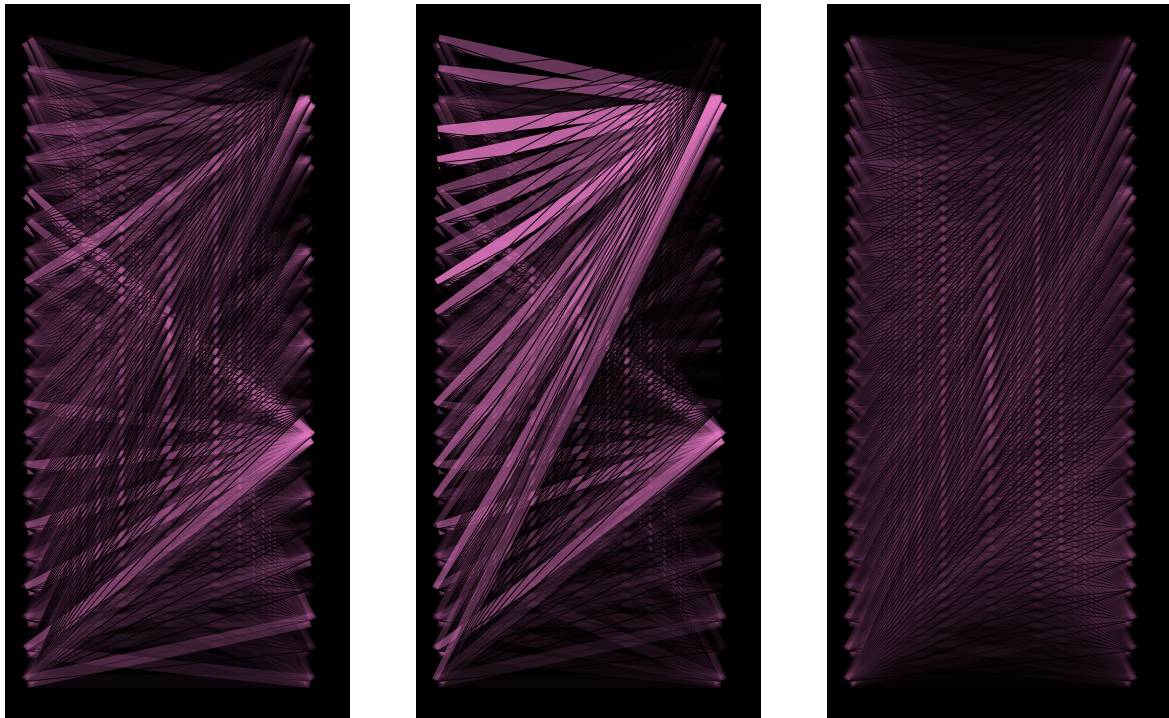
(a) *Head 2* da camada 6 do modelo sem *fine-tuning*(b) *Head 2* da camada 6 após *fine-tuning*(c) *Head 2* da camada 6 sem pré-treino

Figura 5.2: Comparação da *head 2* da camada 6 entre os modelos sem *fine-tuning*, com *fine-tuning* e sem pré-treino.

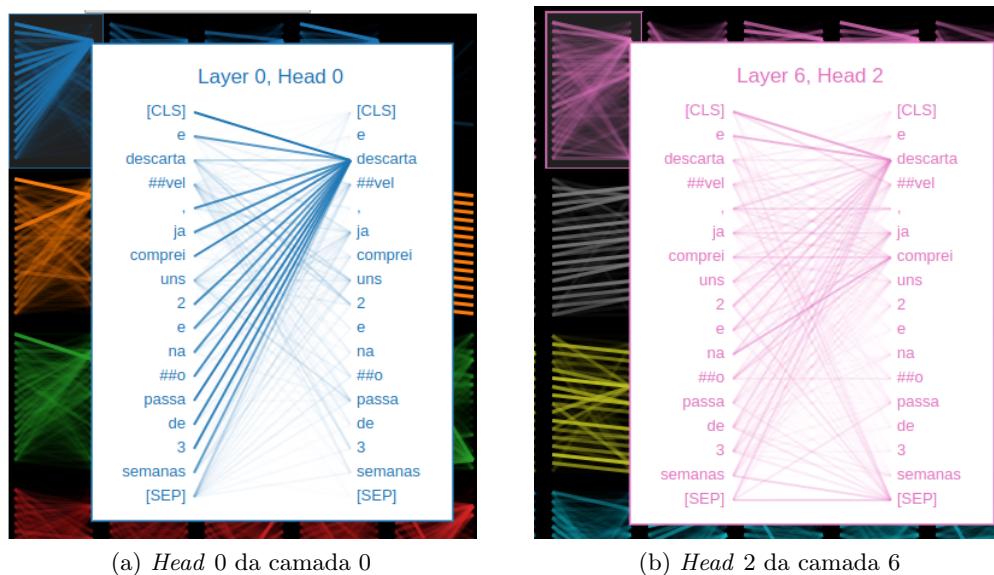
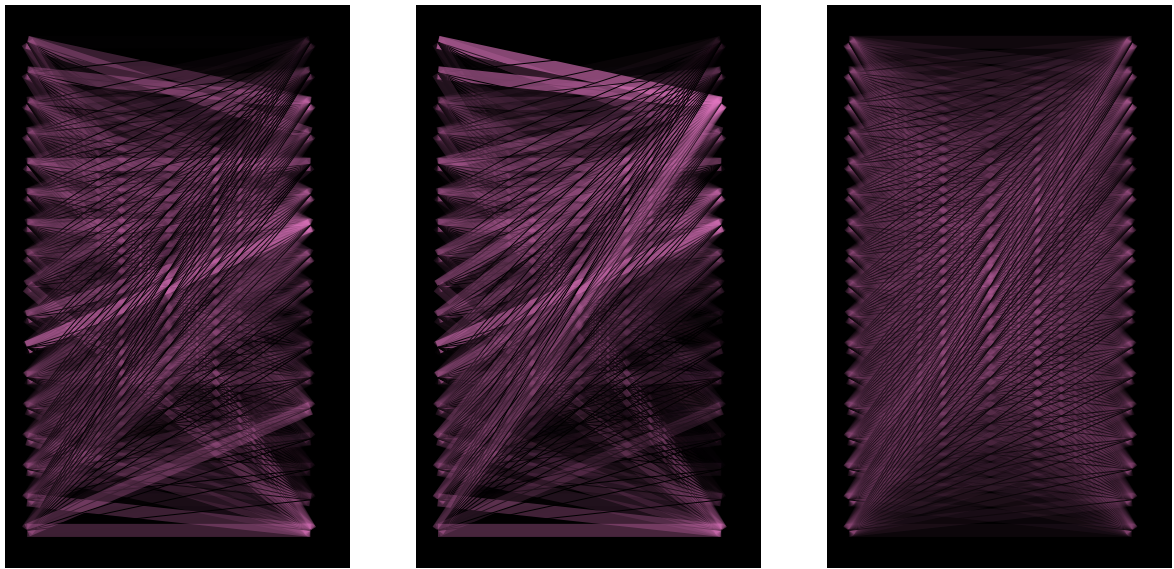
(a) *Head 0* da camada 0(b) *Head 2* da camada 6

Figura 5.3: Visualização das camadas de atenção dos modelo após *fine-tuning* com um exemplo negativo.





(a) *Head 2* da camada 6 do modelo sem *fine-tuning*

(b) *Head 2* da camada 6 após *fine-tuning*

(c) *Head 2* da camada 6 sem pré-treino

Figura 5.4: Comparação da *head 2* da camada 6 entre os modelos sem *fine-tuning*, com *fine-tuning* e sem pré-treino.

## Capítulo 6

# Conclusão

Soluções que utilizam aprendizado de máquina estão sendo amplamente usadas como ferramentas em diversos campos de atuação, desde a agricultura até fabricação de automóveis, e muitas dessas soluções lidam com texto e, principalmente, precisam lidar com o texto em língua portuguesa na hora de interpretar informações importantes para a solução, o que gera uma necessidade alta de córpus e coleções dados.

Este trabalho mostra que transferência de aprendizado pode ser uma ferramenta importante em problemas com conjuntos de dados restritos. Ao comparar os resultados diminuindo a quantidade de exemplos de um conjunto de dados, é possível observar que a diferença na métrica final é inferior quando comparado a abordagem sem transferência de aprendizado. Reduzindo cerca de 41% no conjunto Mercado Livre, é possível obter resultados melhores que a abordagem C-SVM (Avanço, 2015) treinada com o conjunto completo.

É possível observar que as técnicas baseadas em transferência de aprendizado se mostram, para o conjunto Mercado Livre, melhores quando comparadas a abordagem de aprendizado de máquina com foco em pré-processamento. Entretanto, se mostra menos eficaz quando há um conjunto de dados muito restrito, como é o caso do conjunto Souza et al. (2016). É importante ressaltar que o custo do treinamento das técnicas baseadas em transferência de aprendizado é alto quando comparados a abordagens clássicas, mas, o custo está concentrado no pré-treino do modelo, que uma vez realizado, pode ser utilizado para diversos outros tipos de treinamento.

Avaliando as camadas internas dos modelos, é possível observar que para as técnicas baseadas em atenção de aprendizado profundo, existe uma transferência de padrões entre o modelo de língua pré-treinado e o modelo *fine-tuning*. Isso se torna evidente quando compara-se o treinamento sem pré-treino e também quando observa-se alguns padrões que já existiam no modelo pré-treinado e foram ressaltados, como nos resultados apresentados na Sessão 5.5.

Como perspectiva de trabalhos futuros é interessante: explorar toda a potência do corpus (Hartmann et al., 2014), requisitando os recursos necessários para o pré-treino de modelos baseados em *Transformer* e transferência de aprendizado; e avaliar a performance.

---

Também, para trabalho futuro, há uma vasta possibilidade de exploração dos modelos baseados em *Transformer*, explorando melhorias no pré-treino, utilizando a base completa do NILC (Hartmann et al., 2017); Avaliar a performance da proposta de Yang et al. (2019), que também se baseia em transferência de aprendizado e avaliar a estratégia de transferência de aprendizado para as outras tarefas de PLN, como reconhecimento de entidade nomeada, sistemas de tradução e sistemas de pergunta e resposta.

# Referências Bibliográficas

- Aluísio, S. M.; Pinheiro, G. M.; Finger, M.; das Graças Volpe Nunes, M. e Tagnin, S. E. O. (2003). The lacio-web project : overview and issues in brazilian portuguese corpora creation. In *Proceedings of the Corpus Linguistics*, pp. 14–21. UCREL Technical Papers.
- Avanço, L. V. (2015). Sobre normalização e classificação de polaridade de textos opinativos na web. Master’s thesis, Universidade de São Paulo.
- Azoff, E. M. (1994). *Neural Network Time Series Forecasting of Financial Markets*. John Wiley & Sons, Inc., New York, NY, USA, 1st edição.
- Becker, K. e Tumitan, D. (2013). Introdução à mineração de opiniões: Conceitos, aplicações e desafios. *Palestras do Simpósio brasileiro de banco de dados*, 75:1–2.
- Bruckschen, M.; Muniz, F.; Souza, J.; Fuchs, J.; Infante, K.; Muniz, M.; Gonçalves, P.; Vieira, R. e Aluisio, S. (2008). Anotação lingüística em xml do corpus pln-br. *Série de relatórios do NILC, ICMC-USP*.
- Chen, S. F. e Goodman, J. (1996). An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th Annual Meeting on Association for Computational Linguistics*, ACL ’96, pp. 310–318, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Cho, K.; van Merriënboer, B.; Gülçehre, Ç.; Bougares, F.; Schwenk, H. e Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078.
- Cirqueira, D. d. R. (2018). Uma arquitetura de pré-processamento para análise de sentimento em mídias sociais em português brasileiro. Master’s thesis, Universidade Federal do Pará.
- Dertat, A. (2017). Applied deep learning - part 1: Artificial neural networks. <https://towardsdatascience.com/applied-deep-learning-part-1-artificial-neural-networks-d7834f67a4f6> [Acessado em: 08 de outubro de 2018].

- Devlin, J.; Chang, M.; Lee, K. e Toutanova, K. (2018). BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.
- Goldberg, Y. e Hirst, G. (2017). *Neural Network Methods in Natural Language Processing*. Morgan & Claypool Publishers.
- Hartmann, N.; Fonseca, E. R.; Shulby, C.; Treviso, M. V.; Rodrigues, J. e Aluisio, S. M. (2017). Portuguese word embeddings: Evaluating on word analogies and natural language tasks. *CoRR*, abs/1708.06025.
- Hartmann, N. S.; Avanço, L. V.; Balage Filho, P. P.; Duran, M. S.; Nunes, M. D. G. V.; Pardo, T. A. S.; Aluisio, S. M. et al. (2014). A large corpus of product reviews in portuguese: Tackling out-of-vocabulary words. In *International Conference on Language Resources and Evaluation, 9th*. European Language Resources Association-ELRA.
- Hochreiter, S. e Schmidhuber, J. (1997). Long short-term memory. *Neural Comput.*, 9(8):1735–1780.
- Howard, J. e Ruder, S. (2018). Fine-tuned language models for text classification. *CoRR*, abs/1801.06146.
- Joulin, A.; Grave, E.; Bojanowski, P. e Mikolov, T. (2016). Bag of tricks for efficient text classification. *CoRR*, abs/1607.01759.
- Jurafsky, D. e Martin, J. H. (2009). *Speech and Language Processing (2Nd Edition)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.
- Kudo, T. (2018). Subword regularization: Improving neural network translation models with multiple subword candidates. *CoRR*, abs/1804.10959.
- Lecun, Y.; Bottou, L.; Bengio, Y. e Haffner, P. (1998). Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pp. 2278–2324.
- Maas, A. L.; Daly, R. E.; Pham, P. T.; Huang, D.; Ng, A. Y. e Potts, C. (2011). Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 142–150, Portland, Oregon, USA. Association for Computational Linguistics.
- Manning, C. D. e Schütze, H. (1999). *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA, USA.
- Márquez, L.; Padró, L. e Rodríguez, H. (2000). A machine learning approach to pos tagging. *Mach. Learn.*, 39(1):59–91.

- Mayo, M. (2017). A general approach to preprocessing text data. <https://www.kdnuggets.com/2017/12/general-approach-preprocessing-text-data.html> [Accessado em: 08 de outubro de 2018].
- Mikolov, T.; Chen, K.; Corrado, G. e Dean, J. (2013a). Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.
- Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. e Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546.
- Olah, C. (2015). Understanding lstm networks. <http://colah.github.io/posts/2015-08-Understanding-LSTMs/> [Accessado em: 08 de outubro de 2018].
- Pennington, J.; Socher, R. e Manning, C. D. (2014). Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543.
- Press, W. H.; Flannery, B. P.; Teukolsky, S. A. e Vetterling, W. T. (1988). *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, New York, NY, USA.
- Radford, A.; Narasimhan, K.; Salimans, T. e Sutskever, I. (2018). Improving language understanding by generative pre-training.
- Rajaraman, A. e Ullman, J. D. (2011). *Data Mining*, p. 1-17. Cambridge University Press, New York, NY, USA.
- Rodzvilla, J. (2017). Deep text: Using text analytics to conquer information overload, get real value from social media, and add big(ger) text to big data. *Journal of Web Librarianship*, 11(2):148–149.
- Saif, H.; Fernández, M.; He, Y. e Alani, H. (2013). Evaluation datasets for twitter sentiment analysis: a survey and a new dataset, the sts-gold. In *1st Interantional Workshop on Emotion and Sentiment in Social and Expressive Media: Approaches and Perspectives from AI (ESSEM 2013)*.
- Sanches Duran, M.; Volpe Nunes, M. d. G. e Avanço, L. (2015). A normalizer for UGC in Brazilian Portuguese. In *Proceedings of the Workshop on Noisy User-generated Text*, pp. 38–47, Beijing, China. Association for Computational Linguistics.
- Socher, R.; Lin, C. C.-Y.; Ng, A. Y. e Manning, C. D. (2011). Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 28th International Conference on International Conference on Machine Learning, ICML'11*, pp. 129–136, USA. Omnipress.

- Souza, E.; Alves, T.; Teles, I.; Oliveira, A. L. I. e Gusmão, C. (2016). TOPIE: an open-source opinion mining pipeline to analyze consumers' sentiment in brazilian portuguese. In *Computational Processing of the Portuguese Language - 12th International Conference, PROPOR 2016, Tomar, Portugal, July 13-15, 2016, Proceedings*, pp. 95–105.
- Taylor, W. L. (1953). “cloze procedure”: A new tool for measuring readability. *Journalism Bulletin*, 30(4):415–433.
- Teles, V.; Santos, D. e Souza, E. (2016). Uma análise comparativa de técnicas supervisionadas para mineração de opinião de consumidores brasileiros no twitter. *Proceedings of the XIII Encontro Nacional de Inteligência Artificial e Computacional (ENIAC 2016)*, pp. 217–228.
- Tjong Kim Sang, E. F. (2002). Introduction to the conll-2002 shared task: Language-independent named entity recognition. In *Proceedings of the 6th Conference on Natural Language Learning - Volume 20, COLING-02*, pp. 1–4, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L. e Polosukhin, I. (2017). Attention is all you need. *CoRR*, abs/1706.03762.
- Vig, J. (2019). A multiscale visualization of attention in the transformer model. *arXiv preprint arXiv:1906.05714*.
- Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J. G.; Salakhutdinov, R. e Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for language understanding. *CoRR*, abs/1906.08237.
- Yosinski, J.; Clune, J.; Bengio, Y. e Lipson, H. (2014). How transferable are features in deep neural networks? *CoRR*, abs/1411.1792.