ANÁLISE DE SENTIMENTO POR MEIO DE REDES NEURAIS ARTIFICIAIS EM SISTEMAS DE AVALIAÇÃO DE CURSOS DE GRADUAÇÃO

Henry Maas[1], Bruno H.P. Pereira[1], Marcelo E. G. Barbosa[1], Luciano F. de Medeiros[2]

RESUMO

Sistemas de avaliações institucionais são ferramentas essenciais para viabilizar a melhoria das instituições de ensino, baseado de forma geral no *feedback* do aluno com relação a uma série de critérios. Na tentativa de compreender a grande quantidade de informações geradas por este processo, as técnicas de análise de sentimento podem auxiliar a extrair informações importantes em tais cenários. Baseando-se na literatura a partir das técnicas disponíveis para análise de sentimento, o projeto apresenta os principais conceitos utilizados para a construção de modelos de redes neurais capaz de classificar o sentimento com base em um corpus textual de domínio específico. Por meio de técnicas de experimentação, foram treinados 5 modelos de redes neurais artificiais a fim de evidenciar o melhor modelo para o corpus de avaliação institucional. Na avaliação dos modelos treinados, foi identificado que o BERT (*Bidirectional Encoder Representations From Transformers* - Representações Bidirecionais Codificadas por Transformadores) é uma técnica se mostrou como o melhor modelo para análise de sentimento com o corpus do questionário institucional aqui estudado, atingindo um *F1 score* de 0,912. Em adição a esta constatação, foi identificado também que, para futuras melhorias no modelo, o aumento do conjunto de dados pode levar a resultados mais precisos nas predições.

Palavras Chave: Inteligência Artificial; Análise de Sentimento; Processamento de Linguagem Natural; Classificação de Texto; BERT.

[1] Estudantes de graduação em Engenharia da Computação modalidade presencial da UNINTER.
[2] Professor orientador.
Artigo de Trabalho de Conclusão de Curso entregue como requisito parcial para obtenção do grau de bacharel em Engenharia da Computação ao Centro Universitário Internacional UNINTER, Curitiba – PR, 2019.

1 INTRODUÇÃO

Um dos problemas encontrados na automação de análise textual por inteligências artificiais é que, ao contrário de informações objetivas como equações matemáticas, sentimentos e opiniões tem um caráter altamente subjetivo. Pessoas têm experiências distintas e isso leva à formação de opiniões diferentes sobre um mesmo assunto (LIU, 2015). A humanidade se encontra em uma era onde há quantidades massivas de dados contendo opiniões, em lugares dispersos por toda a internet, gerando o interesse em corporações para a mineração destes dados e a automação da análise do sentimento contido nestas bases de dados (LIU, 2012).

Sendo a opinião de indivíduos, sobre assuntos relacionados a uma instituição ou corporação, de grande importância para o auxílio da tomada de decisão (LIU, 2012), a busca por soluções para tais problemas se encontra em áreas tais como o processamento de linguagem natural (*NLP*[3]) e a análise de sentimento. De forma geral, pode ser constatar que os algoritmos existentes para a automação da análise de sentimento, são treinados utilizados bases de dados registrados em uma escrita formal (como publicações em enciclopédias *on-line* e livros clássicos de literatura cujos direitos autorais se encontram expirados), e isso nem sempre é o caso de opiniões deixadas por usuários em publicações *on-line* e em questionários. Nestes, a tendência é encontrar uma linguagem informal e gramaticalmente incorreta. Portanto, o questionamento colocado aqui indaga se as métricas utilizadas para a análise de corpo textual generalizado são realmente as mais eficientes, ou se é possível o uso de um algoritmo específico para analisar um corpo textual em particular.

Considerando o segundo caso, onde opiniões fornecidas por usuários podem ser escritas de maneira informal, propõe-se o estudo comparativo de uma rede neural artificial treinada, utilizando um corpo textual genérico e um corpo específico, realizando-se o treinamento com uma base de dados selecionada, composta por três conjuntos de questionários voltados à avaliação institucional do Centro Universitário Internacional Uninter. Os questionários são: i) sobre a qualidade do ambiente acadêmico; ii) um questionário dividido em três partes sobre o módulo da disciplina que o aluno atualmente participava (A, B, C, os quais correspondem ao quadrimestre ou semestre de uma determinada turma de graduação); e iii) referente à infraestrutura

[3] Do inglês, Natural Language Processing.

dos polos de apoio ao estudante, estando estes localizados por todo território nacional. Cada um destes questionários possui um vocabulário específico, tratando de questões especiais. Os dados destes questionários foram pré-classificados manualmente, utilizando-se um critério binário (positivo simples ou negativo simples) para cada resposta individual, reunindo um conjunto com aproximadamente 12 mil entradas passiveis de serem utilizadas para o treinamento. Ao final de todos os treinamentos, foram considerados como adequados, os modelos que atingirem uma métrica de qualidade conhecida como *F1 Score* acima de 90%, podendo ser considerado como um modelo que satisfaz o que está sendo proposto.

No decorrer deste documento, buscou-se a descrição geral dos fundamentos envolvidos no sistema de processamento de linguagem natural. Para alcançar este objetivo, utilizou-se a metodologia de pesquisa experimental com auxílio da pesquisa bibliográfica, realizando-se o treinamento e comparação de sistemas de análise de sentimento existentes a um algoritmo criado especificamente para um conjunto classificado de informações textuais.

Este documento se encontra dividido em cinco partes distintas. Na primeira, é apresentada a introdução ao problema abordado e uma sumarização do contexto. A segunda parte refere-se à fundamentação teórica das técnicas e ciências referentes a pesquisa. Na terceira parte, é descrita a metodologia utilizada, sendo a quarta parte a apresentação da sistemática e resultados obtidos, e por fim as conclusões e considerações finais obtidas por meio desta pesquisa.

2 FUNDAMENTAÇÃO TEÓRICA

A fundamentação da pesquisa conta com uma série de conceitos relacionados de forma geral à Inteligência Artificial, paradigmas de aprendizado, redes neurais artificiais, *deep learning* e processamento de linguagem natural. A Inteligência Artificial pode ser compreendida como o ramo da ciência da computação que compreende a automação do comportamento inteligente. E que qualquer sistema, para que possa ser considerado como inteligente, necessita ter a capacidade de aprendizado (LUGER, 2013).

Técnicas utilizadas para *machine learning*[4] podem compreender o aprendizado supervisionado, onde o algoritmo recebe um conjunto de entradas e saídas pré-classificados, assim como o aprendizado não supervisionado, onde o algoritmo recebe as entradas e cabe ao algoritmo abstrair alguma utilidade destes dados. Também contempla o aprendizado por reforço, onde são mapeadas ações especificas a situações apresentadas, com o algoritmo buscando executar todas as possíveis ações a ele dispostas até encontrar uma que melhor atenda a solicitação esperada (SUTTON e BARTO, 2018).

Os recentes avanços nas tecnologias de capacidade computacional, ou o quão rápido computadores conseguem realizar instruções juntamente com a paralelização de códigos de instrução sendo executados, permitem a utilização de métodos de aprendizado denominados profundos, ou *Deep Learning*[5], compostas por Redes Neurais Artificiais, de nódulos (ou neurônios) interconectados utilizando camadas ocultas entre a entrada de dados e a saída encontrada, redes neurais de aprendizado profundo podem ser utilizadas para auxiliar na criação da Modelagem de Linguagem, ou o processo para criar um modelo de predição de palavras, essencial para criação de um sistema de Processamento de Linguagem Natural (OTTER, MEDINA e KALITA, 2018).

Um sistema de Processamento de Linguagem Natural ou *NLP* é uma das vertentes do estudo da inteligência artificial, sendo um sistema que tem como saída uma determinação a partir da entrada de um texto escrito em linguagem natural ou idiomática. Tais sistemas necessitam ser capazes de realizar a desambiguação sobre o sentido de uma palavra, sua categoria, seu escopo semântico e também sua estrutura sintática (MANNING e SCHÜTZE, 1999). Por meio do processamento de linguagem natural, é possível obter informações que possam ser consideradas como subjetivas, para então serem processadas por um sistema que realize a análise de sentimento destas informações, podendo ser classificadas como contendo uma opinião considerada como positiva, negativa ou mesmo neutra (LIU, 2012), podendo ainda ser explorada uma classificação em diferentes graus.

[4] Aprendizado de máquinas, tradução livre.
[5] Aprendizado profundo, tradução livre.

2.1 Inteligência Artificial

Inteligência artificial é uma ciência que pode ser resumida como "*o estudo da representação e da busca por meio do qual a atividade inteligente pode ser executada em um dispositivo mecânico"* (LUGER, 2013). O objetivo de estudo das tecnologias de inteligência artificial é algo que depende do objetivo que cada pesquisador busca, mas podem ser definidos como sendo relacionados a como funcionam os processos de pensamento e raciocínio, como emular o pensamento da forma humana, descrito em 1950 por Alan Turing, elencando que um computador inteligente necessitaria compreender o processamento de linguagens naturais, se comunicando de forma bem sucedida no idioma escolhido. Inteligência artificial também podem ser considerada como um processo que busca compreender o comportamento (LUGER, 2013), utilizando agentes computacionais que observam o ambiente em que se encontram utilizando entradas sensoriais podendo ser definidas como *percepts*[6] e então realizando uma ação neste ambiente por intermédio de atuadores (RUSSEL e NORVIG, 2010).

Para que um algoritmo de inteligência artificial possa ser considerado como inteligente, é necessário que tenha uma capacidade de modificar-se (aprender) ao decorrer de interações que tenha com seu ambiente, aprendendo com cada saída a reavaliar a tomada de decisão, ou generalizações que possam ser consideradas como confiáveis utilizando quantidades consideravelmente pequenas de treinamento, os paradigmas para *machine learning* podem ser considerados supervisionados, não supervisionados ou utilizando o aprendizado por reforço (LUGER, 2013).

2.2 Paradigmas de aprendizado

Machine learning pode ser categorizado em distintos grupos, com cada um deles podendo ser mais eficaz em particulares casos de análise de dados. Dentre os paradigmas mais utilizados, temos quatro (4) casos, como (**i**) **Aprendizado supervisionado**, com as metodologias aplicadas ao aprendizado supervisionado podem ser consideradas as mais comuns, utilizadas amplamente no estudo da estatística, como a previsão de um valor real, (regressão) previsão de categorização

[6] Percepção, tradução livre.

de dados (classificação) e previsão do ordenamento de dados (hierarquização) (SUGIYAMA, 2015).

O aprendizado supervisionado tem como objetivo a análise da relação entre dados de entrada e suas saídas (SUGIYAMA, 2015), utilizando-se de um conjunto de treinamento de N exemplos em pares de entradas e saídas do tipo: $(x_1, y_1), (x_2, y_2), \dots (x_N, y_N)$. Sendo x e y valores quaisquer, e y sendo gerado por uma função de x ($y = f(x)$) desconhecida, o algoritmo tem como objetivo descobrir uma função h que se aproxime da função f. A função h pode ser compreendida como uma hipótese levantada e o aprendizado ocorre onde o algoritmo realiza uma busca por um espaço de possíveis hipóteses, determinando qual é a que melhor se enquadra a um problema específico, inclusive em conjuntos de dados diferentes do conjunto de treinamento original. Uma hipótese pode ser considerada bem-sucedida se realizar a predição correta de valores de y em exemplos novos, podendo assim ser considerada como generalizada (LUGER, 2013).

O (**ii**) **Aprendizado Semi-Supervisionado** encontra-se entre o aprendizado supervisionado e o não supervisionado, mas ainda assim pertencendo ao grupo do aprendizado supervisionado. O algoritmo recebe como supervisão, um conjunto de dados que representa algumas possíveis saídas, mas que não necessariamente representem todos os exemplos (CHAPELLE, SCHÖLKOPF e ZIEN, 2006).

A utilização de algoritmos do tipo semi-supervisionados é útil quando existem mais dados não classificados do que classificados, sendo que os dados são de fácil aquisição, mas a classificação demanda muito tempo, esforço ou dinheiro como, por exemplo, a classificação de áudio de voz humana, sendo possível gravar horas de áudio de maneira rápida, mas sua classificação necessitaria de colaboradores humanos para escutar cada gravação e assim a classificar (CHAPELLE, SCHÖLKOPF e ZIEN, 2006).

Em contraste ao aprendizado supervisionado, o (**iii**) **Aprendizado Não Supervisionado** contém somente os dados de entrada, cabendo ao algoritmo encontrar alguma distinção nestes dados. Por esta ambiguidade ser dependente da saída ter valor ou não, o aprendizado não supervisionado é utilizado pensando-se caso a caso, (ao contrário da forma generalista do aprendizado supervisionado). Casos típicos da utilização do aprendizado não supervisionado incluem o

agrupamento de dados baseados em sua similaridade (*clustering*[7]), estimação da probabilidade de distribuição de dados (densidade), detecção de dados que se encontrem fora da curva de probabilidade (detecção de anormalidades), também podendo utilizar o aprendizado não supervisionado para a pré-classificação de dados que possam ser utilizados no aprendizado supervisionado (SUGIYAMA, 2015).

O (**iv**) **Aprendizado por Reforço** pode ser considerado um terceiro paradigma, no qual não é fornecida uma supervisão explicita (um conjunto de dados de saída), mas ainda assim espera-se que o algoritmo encontre uma função que descreva a relação das entradas com as saídas. O algoritmo recebe supervisão de maneira implícita, em forma de recompensas (SUGIYAMA, 2015).

Na utilização de algoritmos considerados como aprendizado por reforço, alguns elementos chave são considerados em sua construção, sendo eles: (a) Política – é a definição de como um agente em processo de aprendizado se comporta em um período, considerada o núcleo do aprendizado por reforço; (b) Sinal de recompensa – define o objetivo em um problema de aprendizado, é a métrica utilizada para retorno de cada ação realizada pelo algoritmo; (c) Função de valor – especifica se uma recompensa é melhor a curto ou longo prazo; (d) Modelagem – a modelagem é um caso especial, podendo ser utilizada ou não. A modelagem de um ambiente permite ao agente prever o resultado de ações futuras antes que elas sejam realizadas, o agente realiza simulações e decide quais ações devem ser tomadas de acordo com as projeções modeladas (SUTTON e BARTO, 2018).

2.3 Redes Neurais Artificiais e *Deep Learning*[8]

Redes neurais artificiais podem ser compreendidas como modelos computacionais paralelizados, com variados níveis de complexidade que dependem da quantidade de entradas e níveis ocultos utilizados (HASSOUN, 1995). Uma das principais características de redes neurais artificiais é sua capacidade de adaptação, onde o aprendizado pode ser obtido por intermédio da experimentação, ou aprender fazendo. Assim, são modelos passíveis de serem usados em cenários onde somente tem-se uma pequena, ou mesmo nenhuma compreensão do problema a ser resolvido,

[7] Agrupamento, tradução livre.
[8] Aprendizado Profundo, Tradução Livre.

mas que existe uma grande quantidade de dados disponível para o treinamento (HASSOUN, 1995).

Redes neurais artificiais podem ser utilizadas em problemas como a classificação de padrões, sintetização de vozes e problemas complexos de Física (HASSOUN, 1995), e são compostas de um número de nódulos interconectados entre si, denominados neurônios, cada neurônio recebe um número de entradas, processa e retorna uma saída, então analisa o resultado e se necessário, os pesos utilizados para os cálculos podem ser atualizados (OTTER, MEDINA e KALITA, 2018).

Os principais fatores de distinção entre os tipos de redes neurais artificias se referem a disposição dos nódulos e como estão conectados entre si, assim como o número de camadas que a rede possui, redes consideradas mais simples tem seus nódulos organizados em camadas sequenciais, já redes que contenham uma quantidade superior de camadas ocultas podem ser consideradas redes neurais artificiais profundas (OTTER, MEDINA e KALITA, 2018).

2.4 Quasi-Recurrent Neural Networks

As redes neurais quase recorrentes (QRNN's), definida como uma arquitetura de aprendizado profundo, utilizado em áreas como modelagem de linguagem, analise e classificação textual, entre outras propostas, a qual é desenvolvida por meio de componentes simples, sendo, uma camada de convolução e uma camada de *pooling*, ou concentração de dados em redes neurais (BRADBURY, MERITY, *et al.*, 2016)

A camada de convolução é utilizada para realizar o cálculo de representações intermediárias, sequencialmente, utilizando a camada de *pooling* para tratar as dependências sequenciais. A camada de *pooling* recebe cada saída do mapa de características da camada convolucional e prepara um mapa de características condensadas (HAYKIN, 2007).

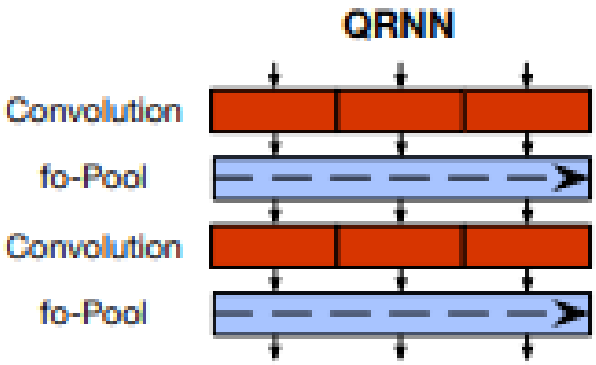

Figura 1 - Diagrama de blocos mostrando a estrutura computacional de uma QRNN. Fonte: *(BRADBURY, MERITY,* et al.*, 2016)*

Na figura 1, os blocos marrons são as convoluções, os quais realizam os cálculos, enquanto que os blocos azuis contínuos representam o paralelismo, onde as operações atuam em paralelo ao longo da dimensão do recurso.

Esta arquitetura apresenta algumas características, sendo, dentre elas, o tratamento de dependências utilizando o *pooling*, uma espécie de convolução mascarada, ou seja, as saídas não dependem de entradas futuras. Esta arquitetura pode obter bom desempenho quando comparada a outras, levando em consideração o custo de treinamento e processamento (BRADBURY, MERITY, *et al.*, 2016).

2.5 Deep Learning

Ao considerar que a saída encontrada por um neurônio pode gerar uma ação que modifique o ambiente alvo, o aprendizado pode ser descrito como encontrar os pesos corretos que tenham como resultado o comportamento esperado. Dependendo da complexidade do problema e como os neurônios estão conectados em uma rede, podem ser necessárias grandes cadeias de etapas computacionais. O *deep learning* pode ser compreendido como a obtenção de resultados positivos durante diversas destas etapas (SCHMIDHUBER, 2015). Redes neurais que sejam consideradas como profundas ou muito profundas são utilizadas no processamento de problemas complexos como o processamento de linguagem natural.

2.6 Processamento de Linguagem Natural

O processamento de linguagem natural pode ser definido como o campo da ciência da computação e linguística que estudas as interações entre seres humanos

e máquinas, utilizando a linguagem natural. Tanto na transformação de dados existentes em bases de dados em algo que possa ser lido por seres humanos, ou na conversão de amostras da linguagem natural em arvores de análise sintática, para que computadores possam as manipular mais facilmente (KUMAR, 2011). Processamento de Linguagem Natural está voltado aos aspectos da linguagem natural, sendo, a fonologia, morfologia e sintaxe, semântica e pragmática. Ou seja, consiste no desenvolvimento computacional para a compreensão desta linguagem, fazendo com que a máquina tenha a capacidade de se comunicar, interpretar e processar de forma semelhante ao ser humano (JURAFSKY e MARTIN, 2019).

Um sistema de processamento de linguagem natural pode ser considerado como funcional, quando for capaz de tratar as ambiguidades presentes em um texto, categorizar palavras e estruturas de sintaxe, assim como o espoco semântico (MANNING e SCHÜTZE, 1999).

A construção de algoritmos capazes de realizar o processamento de linguagem natural depende de que alguns fatores sejam considerados, relacionando o pensamento, a representação e significado das entradas e o conhecimento mundial. Um sistema de processamento de linguagem natural pode começar determinando a origem e estrutura de palavras, e então começar a analisar suas estruturas de construção, como gramatica e significado de uma frase, e assim chegar no contexto generalizado (KULTZAK, 2016). Um sistema computacional interpreta uma sentença em linguagem natural, por intermédio de uma análise de informações morfológicas (léxicas), sintáticas (regras gramaticais) e semânticas (significados), armazenadas em um dicionário, juntamente com as palavras que o sistema compreende (SILVA e LIMA, 2007).

Algumas ferramentas são utilizadas na classificação gramatical de textos, consistindo ainda de técnicas utilizadas por lexicógrafos na análise de palavras e também por máquinas no processamento de linguagem natural. Dentre elas, pode-se assinalar:

- A **classe gramatical**[9], sendo uma ou mais categorias de palavras ou itens lexicais que possuem propriedades gramaticais semelhantes. Ou seja, palavras que desempenham papéis semelhantes dentro da estrutura gramatical de sentenças (JURAFSKY e MARTIN, 2019).
- O **reconhecimento de entidade nomeada**[10] é classificado como uma subtarefa de extração de informações, o qual busca localizar dentro de um conjunto de sintaxes, ou texto, categorias predefinidas, como por exemplo, nomes, organizações, localização, códigos, expressões, entre outros (JURAFSKY e MARTIN, 2019).
- As **perguntas e respostas**[11]**,** definido como um campo do processamento de linguagem natural, esta busca construir sistemas, os quais objetivam responder automaticamente perguntas feitas por humanos, com base em um texto ou em uma estrutura de sentenças, dentro de um idioma natural (JURAFSKY e MARTIN, 2019).
- A **transferência de aprendizado**[12], tratando-se de *machine learning*, a transferência de aprendizagem define-se por armazenar o conhecimento adquirido ao resolver um determinado problema, utilizando este conhecimento para ajudar a resolver um problema diferente, porém, relacionado (MURPHY, 2012).
- A **classificação de texto**, definida como uma das tarefas fundamentais do processamento de linguagem natural, a classificação de texto é o processo no qual, atribui categorias a um texto de acordo com o seu conteúdo, buscando a rotulagem de tópicos, detecção de intenção e análise de sentimento. Dentro deste processo, estão envolvidas as tarefas de pré-processamento, como a *tokenização, stop-words, lematização* e *stemming* (JURAFSKY e MARTIN, 2019).

Um exemplo deste processo pode ser definido quando a máquina busca reconhecer um determinado objeto, no caso um carro. Desta forma, quando o objetivo é concluído, o conhecimento adquirido pelo treinamento desta rede, pode ser aplicado no reconhecimento de um novo objeto, desta vez sendo um caminhão. Ou seja, este

[9] *Part of Speech,* tradução livre.
[10] *Named entity recognition,* tradução livre
[11] *Question and answering*, tradução livre.
[12] *Transfer learning,* tradução livre.

processo colabora efetivamente na eficiência do processamento, seja no ganho de tempo, quanto na acurácia da amostra de um agente de aprendizado (MURPHY, 2012).

2.6.1 Análise De Sentimento

A análise de sentimento é o campo de estudo que analisa a opinião de pessoas, assim como os sentimentos, atitudes e emoções conectadas a um tópico ou produto. Historicamente, a análise de sentimento foi utilizada em variados níveis, começando pelo nível de documento, classificando o texto inteiro nele contido como tendo um sentimento positivo ou negativo; é utilizada também em nível de sentença, classificando se uma sentença em particular pode ser considerada como positiva, negativa ou neutra; e em nível de entidade e aspecto, realizando uma análise mais profunda buscando compreender se a opinião em si é positiva ou negativa, transformando texto não estruturado em dados estruturados (LIU, 2012).

2.6.2 Classificadores de texto.

Sendo a classificação de texto a atribuição de uma categoria a um texto, como descrito anteriormente, um classificador de texto é um algoritmo que realiza esta tarefa de maneira automatizada, dentro de possíveis soluções para a classificação de texto, em aprendizado de maquinas, podemos citar classificadores como o *naïve bayes* e as máquinas de vetor de suporte.

2.6.2.1 Naïve Bayes

Naïve Bayes é um classificador de *machine learning* probabilístico simples, porém, eficiente e especialmente popular na classificação textual (CARUANA e NICULESCU-MIZIL, 2006). Por ser um classificador probabilístico, tem como objetivo determinar a probabilidade dos recursos que ocorrem em cada classe retornando a classe mais provável (JURAFSKY e MARTIN, 2019), utilizando a regra de Bayes, regida pela equação apresentada na equação a seguir.

$$P(c|x) = \frac{P(x|c)P(c)}{P(x)} \quad (1)$$

Na equação 1, tem-se os seguintes elementos:

- $P(c \mid x)$ é a probabilidade posterior da classe (c, alvo) dada preditor (x, atributos).
- $P(c)$ é a probabilidade original da classe.
- $P(x \mid c)$ é a probabilidade que representa a probabilidade de preditor dada a classe.
- $P(x)$ é a probabilidade original do preditor.

O modelo Naïve Bayes envolve uma suposição de independência condicional simplificadora. Dada uma classe (positiva ou negativa), as palavras são condicionalmente independentes entre si. Essa suposição não afeta a precisão na classificação do texto por muito, mas tornam algoritmos de classificação realmente rápidos aplicáveis ao problema (MAAS, DALY, *et al.*, 2011).

Um exemplo seria uma fruta, a qual pode ser considerada uma maçã, se é vermelha, redonda e tem cerca de 10 cm de diâmetro. Mesmo que esses recursos dependam um do outro ou da existência das outras características, o classificador de Bayes considera que todas essas propriedades contribuem independentemente para a probabilidade dessa fruta ser uma maçã (CARUANA e NICULESCU-MIZIL, 2006).

Dependendo da natureza precisa do modelo de probabilidade, os classificadores de Naïve Bayes podem ser treinados com muita eficiência em um ambiente de aprendizado supervisionado, onde, uma vantagem deste classificador, é que, este requer apenas uma pequena quantidade de dados de treinamento para estimar parâmetros (medias e variações das variáveis) necessários para a classificação (CARUANA e NICULESCU-MIZIL, 2006).

2.6.2.2 Máquinas de Vetores de Suporte

Uma máquina de vetores de suporte (*Support Vector Machines* - SVM) pode ser compreendida como um conjunto de métodos do aprendizado supervisionado, os quais buscam analisar dados para reconhecer padrões (JURAFSKY e MARTIN, 2019). As SVM funcionam, classificando a entrada de um determinado conjunto de dados, predizendo para cada entrada, qual das duas possíveis classes este dado pertence, ou seja, o que torna esta, um classificador binário. Sua base está em definir

qual o vetor que melhor classifica os dados dentro destas classes, como ilustrado na figura 3.

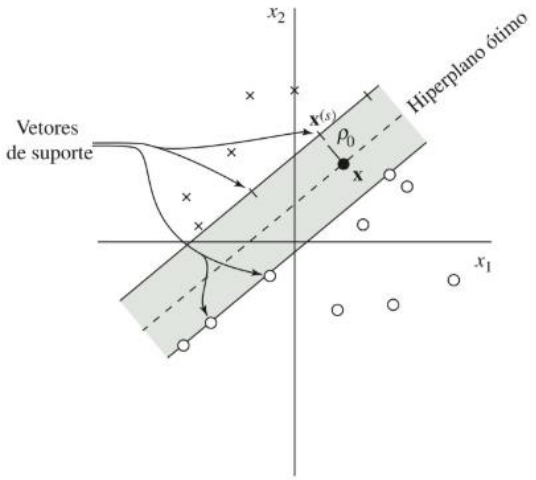

Figura 2 - Ilustração da ideia de um hiperplano ótimo para padrões linearmente separáveis. Fonte: *(HAYKIN, 2007)*

A equação que separa os padrões por intermédio de hiperplanos pode ser definida por: $w^t x + b = 0$, onde, $w^t x$ é o produto escalar entre os vetores w e x, em que x é um vetor de entrada que representa os padrões de entrada do conjunto de treinamento, w é o vetor de pesos ajustáveis e b é um limiar também conhecido como *bias* (CAMPBELL, 2001).

Para um dado vetor de peso w e bias com valor b, a separação entre o hiperplano e o ponto de dado mais próximo é denominada de *margem de separação,* representada por p. O objetivo de uma máquina de vetor de suporte é encontrar o hiperplano particular para o qual a margem de separação p é máxima, sendo sob esta condição, a superfície de decisão referida como o hiperplano ótimo (HAYKIN, 2007).

2.6.3 Modelos de Linguagem

Modelos de linguagem apresentam contextos para facilitar a interpretação de palavras ou frases com entonação similar. Em processamento de linguagem natural, são utilizados modelos de representação de linguagem para realizar o aprendizado autônomo da interpretação de palavras (BOSCO, PILATO e SCHICCHI, 2018). Tradicionalmente, modelos para o tratamento de linguagem natural eram específicos para uma única função, um modelo para classificação, um modelo para análise de

sentimento, um modelo para perguntas e respostas, atualmente, modelos como o BERT surgiram para realizar todas estas funções em um único algoritmo.

2.6.3.1 Modelo *BERT*

Dentre os modelos utilizados para o aprendizado de linguagens naturais, o modelo *BERT*[13] foi desenhado para o pré-treinamento de representações bidirecionais profundas. Ou seja, utilizando-se redes neurais, representações bidirecionais tem seu sinal se propagando tanto para frente como para trás no tempo, ao serem extraídas de textos não rotuladas ao unir os contextos encontrados em lados direito e esquerdo de maneira condicional em todas as camadas. Modelos pré-treinados pelo *BERT* podem ser melhorados por meio de adicionar uma camada de saída de dados adicional, criando modelos para possíveis aplicações como inferência de linguagem e prover respostas a perguntas de um questionário sem que sejam necessárias modificações especificas em sua arquitetura (DEVLIN, CHANG, *et al.*, 2019).

O pré-processamento de dados no modelo *BERT* acontece inicialmente na classificação de entradas e saídas em forma de acondicionamento em três etapas distintas. Primeiramente, o algoritmo aprende e utiliza acondicionamento posicional de palavras em uma frase, de maneira a facilitar a compreensão do algoritmo sobre a posição relativa ou posição absoluta de *tokens* em uma sequência. Em seguida, o acondicionamento segmentado pode ser utilizado para unir pares de sentenças no caso de utilização do *BERT* para questionários de perguntas e respostas, ajudando na diferenciação entre o que é uma pergunta e o que se qualifica como resposta. Por fim, o acondicionamento de *tokens* pode ser utilizado, onde um *token* qualquer seja a representação de aprendizado relacionada a técnica de segmentação de palavras para processamento de linguagem natural conhecida como *WordPiece*, algoritmo de segmentação introduzido por Schuster e Nakajima (2012), para a solução de busca de palavras nos idiomas japonês e coreano. Este algoritmo prepara um corpo de palavras de tamanho adequado, separa palavras em sequências de caracteres, monta um modelo de linguagem e então repete os passos até atingir um vocabulário que tenha sido definido como adequado (SCHUSTER e NAKAJIMA, 2012).

---

[13] *Bidirectional Encoder Representations From Transformers,* Representações bidireccionalmente codificadas por transformadores*,* tradução livre.

O modelo utilizado pelo *BERT* utiliza uma combinação dos três tipos de acondicionamentos, somando um *token* correspondente a um segmento específico e uma posição definida (DEVLIN, CHANG, et al., 2019). Com os dados pré-classificados, é possível realizar tarefas não supervisionadas para realizar um treinamento inicial, ou pré-treinamento do *BERT*. Primeiramente, uma porcentagem da palavra é mascarada com um *token*, de maneira que o algoritmo então necessite realizar uma predição de uma palavra de acordo com o contexto encontrado nas palavras não mascaradas. Em seguida, o algoritmo busca prever qual será a próxima frase em um texto, ao receber pares de sentenças como entrada, com o intuito de entender o relacionamento entre as duas sentenças, a sentença como um todo é alimentada ao modelo de transformação e tem como saída um vetor de utilizando uma camada de classificação, e então a probabilidade da próxima sentença é calculada utilizando uma função de normalização exponencial (DEVLIN, CHANG, et al., 2019).

O modelo *BERT* utiliza-se de um paradigma chamado de *Bidirecional Transformer*[14]. Transformadores são uma arquitetura que busca evitar a reincidência de conexões, típica de redes neurais recorrentes ou RNN. O paradigma transformador depende apenas do mecanismo de atenção utilizado em redes neurais convolucionais ou CNN, de maneira a produzir dependências globais entre a entrada e saída de dados, facilitando a paralelização computacional, e por consequência, diminuindo os tempos de treinamento (VASWANI, SHAZEER, *et al.*, 2017). Já o paradigma bidirecional de transformadores utilizado pelo modelo *BERT* busca melhorar este método, ao avaliar uma sentença não apenas da esquerda para direita (de forma unidirecional), mas também da direita para a esquerda, e analisando a sentença como um todo utilizando apenar o conceito de codificação apresentado no modelo transformador, ao empilhar camadas de mapeamento de sequencias e utilizando pedaços de palavras para expressar a posição de palavras dentro de uma sentença (DEVLIN, CHANG, *et al.*, 2019).

[14] Transformador bidirecional, tradução livre.

2.6.3.2 Modelo ULMFit

O modelo de linguagem universal, aperfeiçoado para classificação de texto[15] ou ULMFit, é um modelo de pré-treinamento de modelos de linguagem utilizando um vasto corpo textual de domínio geral, e então aperfeiçoa o ajuste ao se adaptar as idiossincrasias do corpo textual, podendo ser treinado mesmo com *datasets*[16] menores (HOWARD e RUDER, 2018).

O modelo consiste em três etapas distintas, primeiramente, um modelo de linguagem é treinado utilizando um corpo textual de domínio geral, de maneira a compreender os atributos de uma linguagem em diferentes camadas de uma rede neural, e então, o algoritmo ajusta o dado alvo utilizando um processo de ajustamento discriminativo e taxas de aprendizado triangular inclinadas, e por último, o classificador é aperfeiçoado ao “congelar” e “descongelar” camadas neurais para lidar com problemas recorrentes gerados pelo modelo de aprendizado por transferência (HOWARD e RUDER, 2018).

2.7.2.3 Modelo MultiFiT

Combinando o modelo ULMFit com QRNNs, o modelo MultiFiT surgiu para poder utilizar as funções do ULMFiT, mas sem a necessidade de treinar *datasets* de múltiplas linguagens, podendo o usuário treinar apenas uma linguagem alvo específica. O modelo MultiFit troca as funções de memória e taxas de aprendizado triangular utilizadas pelo ULMFit por camadas de convolução alternadas, e uma função de concentração (*pooling*) recorrente, combinados com uma função de tokenização de *subwords,*[17] e a utilização de n-gramas do tipo unigramas, ou sequências continuas de uma palavra de um texto de entrada (EISENSCHLOS, RUDER, *et al.*, 2019).

[15] *Universal Language Model Fine-tuning for Text Classification,* tradução livre.
[16] Conjunto de dados, tradução livre.
[17] Quase palavras, tradução livre.

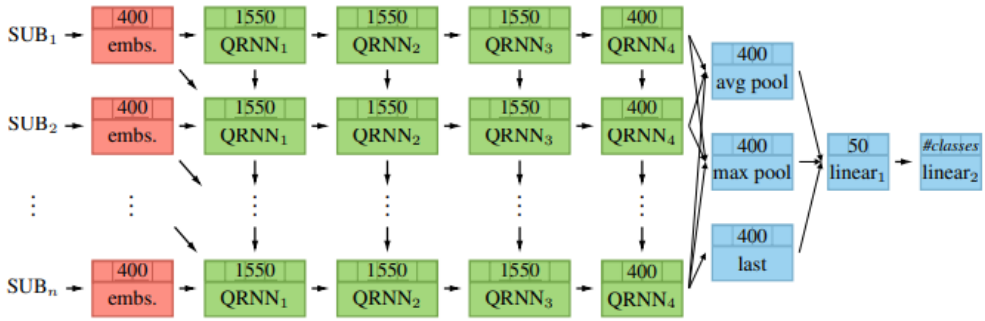

Figura 3 - O modelo de camadas completo MultiFit. Fonte: *(EISENSCHLOS, RUDER,* et al.*, 2019)*

A tokenização de *subwords* é realizada por intermédio de um *tokenizador* conhecido como *SentencePiece*. Este *tokenizador* foi desenhado para trabalhar com processamento de linguagens via redes neurais, sendo de código aberto. O *SentencePiece* trabalha com *subwords* sem a necessidade de pré-tokenização de palavras, podendo ser utilizado diretamente em textos de diversos idiomas, mesmo em idiomas ricas em morfologia (KUDO e RICHARDSON, 2018).

O modelo ULMFit utiliza-se de treinamento bidirecional, realizando um treinamento gradual para frente, e depois um treinamento gradual de trás para frente. Já o modelo MultiFit (Figura 4) descarta esta necessidade, treinando sentenças como são escritas, dependendo apenas do sistema de linguagem, já que o sistema de treinamento bidirecional ocupa grande quantidade de memória, e seu foco é o treinamento rápido e pouco dispendioso (RUDER e EISENSCHLOS, 2019).

3 ASPECTOS METODOLÓGICOS

A metodologia de base utilizada neste trabalho foi o método experimental, que consiste em submeter o objeto de estudo a influências de variáveis em condições controladas e conhecidas pelo investigador, a fim de observar os resultados que a variável produz no objeto (GIL, 2019). As etapas que seguem o processo de experimentação para a análise de sentimento deste projeto são compostas por: seleção dos dados, anotação de dados, pré-processamento de dados, treinamento dos modelos, validação e comparação entre modelos.

3.1 Seleção dos Dados

Para a experimentação, foram escolhidos para compor a base de dados para o treinamento da rede, questionários realizados pela Comissão Própria de Avaliação (CPA) do centro universitário, referentes a avaliação institucional, sendo eles: a) um questionário avaliando a instituição como um todo; b) um conjunto de questionários referentes aos módulos de cada turma de graduação; c) um questionário contendo perguntas sobre a infraestrutura de cada polo de apoio ao ensino a distância ou campus presencial.

O questionário de ambiente acadêmico (a) corresponde à avaliação do curso, tutoria local e à distância, secretarias e central de mediação acadêmica. A avaliação das disciplinas (b) é referente às disciplinas ao final de cada modulo atual, a qual envolve vídeo aulas, materiais disponíveis, avaliações, entre outros. E por fim, a infraestrutura (c), que por sua vez, trata da parte física e tecnológica do polo ao qual cada aluno está matriculado, sendo avaliado sobre a limpeza, disponibilidade de equipamentos, como computadores, livros, internet, cantina, etc.

3.2 Anotação dos Dados

Para a classificação destes dados, foram utilizados os critérios de positivo ou negativo quanto a cada resposta com relação aos questionários, com a figura 4 ilustrando exemplos do processo de anotação. Ao final do processo de anotação, foram reunidos um total de 12.422 respostas referentes aos 3 questionários.

texto	sentimento
As disciplinas estão a altura do curso, muito bom !	positivo
O texto dos livros didáticos, por vezes são redundantes.	negativo
Os temas são adequados, as aulas são bem diversificadas.	positivo
a acústica do polo deixa muito a desejar, barulhos externos comprometem a concentração dos alunos.	negativo
conteúdos cobrados em desacordo com os abordados.	negativo
Nota dez para toda equipe de apoio!	positivo

Figura 4 - Exemplos da anotação de dados. Fonte: Os Autores

3.3 Pré-Processamento de Dados

Tratando-se de processamento de linguagem natural, há algumas métricas que são utilizadas para a normalização, visando que este processo se torne funcional e eficiente, nos quais, estão envolvidos a Tokenização[18], Lematização e Stemização[19] e o conceito de *stop words*[20] (JURAFSKY e MARTIN, 2019) e (CHAUDIRON, 2007).

3.3.1 Normalização

Para que se possa modelar a linguagem e, como consequência possibilitar o entendimento desta pela máquina, são adotados pré-processamentos de normalização que abstraem a estrutura da língua, deixando apenas informações relevantes. Este processo reduz o vocabulário, tornando este, conveniente para o processamento computacional (JURAFSKY e MARTIN, 2019).

A normalização neste contexto, objetiva colocar palavras / *tokens*, em um formato padrão dentre os diferentes que podem existir, isto é, padronizar / normalizar uma base de dados de acordo com a necessidade do processamento. Esta padronização é muito valiosa apesar das percas de informações ortográficas que podem ocorrer. A normalização é importante por começar a estruturar um conjunto de dados, já que processamentos atuam em unidades sentenciais e lexicais (JURAFSKY e MARTIN, 2019).

3.3.2 Tokenização

A Tokenização objetiva separar palavras ou sentenças em unidades, dentro das quais, se tem a Tokenização lexical, onde, marca cada palavra como um token, identificando mesmo se estiver junto à alguma pontuação (JURAFSKY e MARTIN, 2019).

---

[18] *Tokenization*, tradução livre.
[19] *Stemming*, tradução livre.
[20] Palavra vazia, tradução livre.

Exemplo de Tokenização Lexical:

Esta é uma sentença – [‘Esta’, ‘é’, ‘uma’, ‘sentença’]

Já a Tokenização sentencial, identifica e marca sentenças, como por exemplo:

Esta é a primeira sentença. Esta é a segunda. Esta é a terceira.

Sendo:

[‘Esta é a primeira sentença’. ‘Esta é a segunda’. ‘Esta é a terceira’]

3.3.3 Lematização

A lematização busca determinar que duas ou mais palavras possuem a mesma raiz, ou seja, permite considerar que determinadas palavras possuem um radical comum, que significam aproximadamente a mesma coisa, mas que diferem quanto ao gênero, ao número ou ao fato de serem substantivos, adjetivos ou advérbios. Por exemplo: AMOR, AMORES, AMOROSOS, são substituídas no texto pela forma AMOR+. Este processo é muito importante também para o processamento de linguagens morfologicamente complexas. (JURAFSKY e MARTIN, 2019)

3.3.4 Stemização

A Stemização, apresenta uma referência de versão mais simples da lematização, na qual, principalmente, objetiva tirar o sufixo do final de uma palavra, reduzindo-a a seu radical, ou seja, reduzindo ao seu lema, que é a forma no masculino e singular. No caso de verbos, o lema é infinitivo. Como por exemplo, as palavras “gato”, “gata”, “gatos” e “gatas” são todas formas do mesmo lema: “gato”. Igualmente, as palavras “tiver”, “tenho”, “tinha”, “tem” são formas do mesmo lema “ter” (JURAFSKY e MARTIN, 2019).

3.3.5 Stop words

Stop words são palavras muito frequentes que venham a aparecer dentro de um conjunto de dados, que na maioria das vezes acabam não sendo relevantes para a construção do modelo, como por exemplo, palavras como, “a”, “de”, “o”, “que”, “e”, “do”. Podendo ser removidas ao classificar o vocabulário por frequência no conjunto de treinamento e definindo as 10 – 100 entradas principais do vocabulário como *stop words* ou simplesmente utilizar conjuntos de palavras predefinidas disponíveis. É

importante que seja feita uma implementação deste contexto de forma cautelosa, há palavras que indicam justamente o sentimento de um contexto, pois em diferentes modelos, a necessidade da tarefa pode variar (JURAFSKY e MARTIN, 2019).

3.4 Treinamento

Para a tarefa de análise de sentimento com o conjunto de dados do sistema de avaliação institucional, a proposta é achar o melhor modelo ou algoritmo classificador capaz identificar o sentimento da resposta. Foi selecionado então, dois modelos de linguagem, *BERT* e *ULMFiT*, e duas redes neurais classificadoras, SVM e Naïve Bayes. A escolha desses dois modelos de linguagem se deu por conta da disponibilidade de modelos pré-treinados em português disponíveis na internet e por apresentarem resultados próximos aos de estado-da-arte[21]. As redes neurais classificadoras foram escolhidas por ter um amplo acervo na literatura para dar suporte na construção dos algoritmos, onde esses classificadores atingem um bom resultado.

Entre os modelos de linguagem, serão alterados alguns dos hiperparâmetros da rede afim de entender suas finalidades e obter o melhor resultado possível no final do treinamento (*fine tunning*), sem alterar a arquitetura das redes. São eles: tamanho máximo da sequência, tamanho do lote (*batch size*), taxa de aprendizado (*learning rate*) e número de épocas.

O tamanho máximo da sequência (*maximum sequence length*) é um parâmetro definido para limitar o input da rede, sendo essa sequência máxima definida pela quantidade de tokens após o processo de tokenização.

O processo de atualização dos pesos da rede é realizado pelo algoritmo gradiente estocástico descente (JURAFSKY e MARTIN, 2019), onde este é um processo de correção iterativo sobre o conjunto de dados cujo objetivo é achar os pesos ideais para a rede onde a perda seja mínima, ou seja, em uma representação geométrica, deseja-se descer pela curva de erro.

[21] Disponível em: http://nlpprogress.com/english/sentiment_analysis.html, Acesso em 19 de novembro de 2019.

Neste processo, o número de amostras controlados pelo gradiente descente onde os pesos são atualizados, é chamado de tamanho do lote, ou *batch size* (JURAFSKY e MARTIN, 2019).

A taxa de aprendizado determina a rapidez com que o algoritmo irá atingir o mínimo de erro aceitável (JURAFSKY e MARTIN, 2019). Em uma representação geométrica, na curva do gradiente, a taxa de aprendizado representa os passos da descida dessa curva. O hiperparâmetro que define a quantidade de vezes que o algoritmo irá trabalhar sobre todo o conjunto de dados é chamado de número de épocas.

3.5 Validação

A avaliação dos modelos de classificação é realizada a partir da construção de uma matriz de confusão, a qual demonstra a quantidade de acertos e erros entre duas classes, demonstrados na tabela 1.

		Classe Atual	
		Positivo	**Negativo**
Classe Predita	**Positivo**	Verdadeiro Positivo (VP)	Falso Positivo (FP)
	Negativo	Falso Negativo (FN)	Verdadeiro Negativo (VN)

Tabela 1 – Matriz de confusão. Fonte: Os autores.

A partir da matriz de confusão pode-se estabelecer algumas métricas para calcular a eficácia desses modelos. Jurafsky e Martin (2019) apresentam conceitos matemáticos que ajudam a entender o resultado das predições.

3.5.1 Acurácia

A acurácia, também chamada de precisão geral do modelo, é uma medida que corresponde a uma porcentagem de todas as predições realizadas corretamente

por todos os valores obtidos. A acurácia pode ser calculada pela equação 2 (JURAFSKY e MARTIN, 2019).

$$acurácia = \frac{VP + VN}{VP + FP + VN + FN} \quad (2)$$

Jurafsky e Martin (2019) observam ainda que a acurácia geralmente não é utilizada quando as classes não são balanceadas em termos de frequência por ignorar os falsos positivos existentes nos resultados.

3.5.2 Precisão

A precisão é uma métrica que calcula a efetividade de uma predição positiva correta. A precisão pode ser obtida dividindo o total de verdadeiros positivos pela soma das predições verdadeiros positivos e falsos positivos, regida pela equação 3 (JURAFSKY e MARTIN, 2019).

$$precisão = \frac{TP}{TP + FP} \quad (3)$$

3.5.3 Recall

Utiliza-se o recall para entender a frequência com que o modelo consegue predizer os dados de uma classe corretamente. O cálculo do recall é (equação 4) obtido dividindo os verdadeiros positivos pela soma de verdadeiros positivos com falsos negativos (JURAFSKY e MARTIN, 2019).

$$recall = \frac{TP}{VP + FN} \quad (4)$$

Combinando as medidas de precisão e recall, é possível fazer uma melhor análise das classes que estão sendo respondidas, dando ênfase às predições verdadeiras, resultados que são de real interesse nas análises dos modelos.

Observando a partir das métricas de precisão e recall, os resultados esperados são quando essas medidas estão em equilíbrio. Quando o modelo não comete erros nas predições, o valor de recall e precisão são iguais, e o resultado dessa divisão é igual a 1.

3.5.4 F1 Score

O *F1 Score* é uma média ponderada harmônica que combina precisão e recall. A medida consegue trabalhar com classes desbalanceadas por atribuir maior peso ao menor valor. O resultado é utilizado para avaliar a qualidade do modelo onde o score máximo será uma pontuação igual a 1 (equação 5) (JURAFSKY e MARTIN, 2019).

$$F1 = \frac{2 * precisão + recall}{precisão + recall} \quad (5)$$

3.5.5 Função de *loss*[22]

A função de *loss*, tem como objetivo avaliar a distância de uma predição com relação à sua classe correspondente. A função de custo é uma métrica que também indica o aprendizado da rede. A função de custo ou perda pode ser definida pela equação 6 onde o erro pode ser medido calculando a diferença da predição realizada com o resultado esperado.

$$L(y, \hat{y}) = y - \hat{y} \quad (6)$$

4 RESULTADOS

Esta seção apresenta os resultados obtidos com base nas métricas de avaliação da seção anterior. Sendo esta dividida da seguinte forma: i) Investigação inicial em corpus genérico, para o entendimento das relações entre diferentes corpus utilizados e evidenciação de treinamento de um modelo com corpus de domínio específico; ii) Resultados dos classificadores de texto utilizando *SVM* clássico e Naïve Bayes, utilizando uma abordagem clássica e uma abordagem com utilizando técnicas de negação; iii) Resultados dos modelos de linguagem BERT e MultiFiT; (**iv**) Comparativo entre técnicas e evidenciação da melhor abordagem para análise de sentimento em domínio específico.

4.1 Investigação inicial em corpus genérico

Na busca pela melhor forma de solucionar a o problema proposto, a análise de sentimento para o sistema de avaliação institucional, foi investigado a possibilidade da utilização de um corpus genérico para essa tarefa. Entretanto, há poucos conjuntos

[22] Custo, Perda. Tradução livre.

de dados anotados em português disponíveis na internet, tornando a busca por conjuntos de dados anotados uma tarefa difícil para a língua portuguesa.

O *dataset* do IMDB é um importante conjunto de dados anotados para modelos de classificação, comumente visto em benchmarkings de tarefas de *NLP*. O *dataset* é composto de resenhas de filmes da plataforma IMDB, onde o usuário avalia o filme com uma nota de 1 a 10 e faz sua resenha justificando a nota. Dessa maneira, torna-se fácil o processo de anotação, pois os dados já estão previamente classificados por intermédio de uma nota.

Por este motivo, foi escolhido este *dataset* traduzido[23] para testar a eficácia de um *dataset* genérico na análise de sentimento de um corpus de domínio específico. O conjunto de dados possui 49.459 amostras, estando divididas em 24.765 consideradas como sentimento negativo e 24.694 com sentimento positivo. É válido destacar que o *dataset* foi traduzido com uma ferramenta de tradução automática, podendo assim, ter uma margem de erros inerentes da técnica de tradução automática, podendo levar a uma menor precisão no final do treinamento.

Utilizando esse conjunto de dados, foi treinado um modelo de linguagem utilizando o BERT com os parâmetros propostos pelos pesquisadores da arquitetura[24]. Foi utilizado uma sequência máxima de tokens de 128, *batch size* de 32 e *learning rate* de 2^{-5} para um treinamento com 3 épocas. Os resultados do treinamento são apresentados na tabela 2.

Dataset	Acurácia	F1	Precisão	Recall	Perda
IMDB PT-BR	0.861	0.863	0.859	0.868	0.583

Tabela 2 - Resultado do treinamento com BERT. Fonte: (Os autores)

A figura 5 apresenta a perda por a cada 500 iterações do treinamento. Observa-se que na última época, definida como mil iterações, o modelo já começa a convergir.

[23] IMDB PT-BR - Tradução do *dataset IMdb* para o português, Disponível em: https://www.kaggle.com/luisfredgs/imdb-ptbr/discussion/74851 Acesso em 12 de novembro de 2019.
[24] *Predicting Movie Review Sentiment with BERT on TF Hub*, Disponivel em: https://github.com/google-research/bert/blob/master/predicting_movie_reviews_with_bert_on_tf_hub.ipynb Acesso em 12 de novembro de 2019.

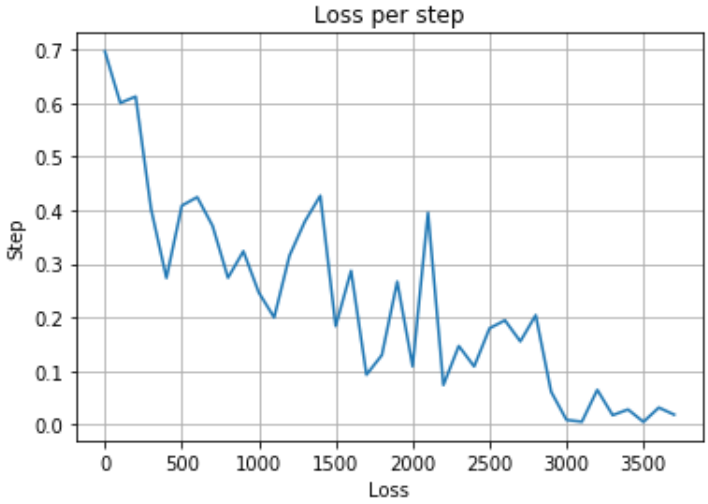

Figura 5 - Treinamento BERT IMDB. (Fonte: Os Autores)

		Classe Atual	
		Positivo	**Negativo**
Classe Predita	**Positivo**	4356	716
	Negativo	662	4158

Tabela 3 - Matriz de Confusão - IMDB. Fonte: Os Autores

Apesar do resultado mostrar uma precisão aceitável, o modelo treinado se mostra ineficaz quando utilizado para predizer sentenças do domínio do questionário. Para essa validação, foi selecionado 20% das amostras aleatórias com as classes balanceadas de todo o questionário CPA anotado, totalizando 1925 amostras para testes de inferência. Os resultados são demonstrados na tabela 4.

Dataset	Acurácia	F1	Precisão	Recall	Perda
CPA	0.705	0.707	0.666	0.757	1.147

Tabela 4 - Validação do modelo BERT-IMDB com dataset da CPA. Fonte: Os Autores

Observa-se que o modelo passa a errar mais sobre esse novo conjunto de dados. A matriz de confusão (tabela 5) deixa claro os erros nos resultados das predições.

		Classe Atual	
		Positivo	**Negativo**
Classe Predita	**Positivo**	900	451
	Negativo	288	846

Tabela 5 - Matriz de confusão – Questionário CPA. Fonte: Os Autores

Como exemplo, as sentenças a) melhorar a qualidade das videoaulas, anotada como negativa, foi classificada como positiva e b) não tenho reclamações, anotada como positiva, foi classificada como negativa pelo modelo. Assim, evidencia-se a necessidade de um conjunto de dados de próprio domínio para satisfazer a necessidade de previsões mais assertivas.

4.2 Support Vector Machines

O experimento com o classificador SVM foi iniciado separando um conjunto de dados do questionário utilizando 80% do total de anotações para o treinamento e 20% para testes e validação.

Para o pré-processamento dos dados, foi utilizado as técnicas de tokenização, *stop words* e *stemming*. Para a *tokenização*, foi utilizado a biblioteca *tokenize do Natural Language Toolkit*[25] (NLTK). A *tokenização* é seguida pelo processo de remoção das *stop words* de todo conjunto de dados, utilizando o recurso do NLTK. Seguindo os passos da literatura, foi realizado o processo de *stemização* das palavras utilizando a biblioteca *SnowballStemmer*, também do NLTK, que oferece suporte em português.

Para os algoritmos classificadores, foi utilizado a biblioteca *SKLEARN* que oferece vários recursos de alto nível para o pipeline de treinamento. Aproveitando dessa biblioteca, foi utilizado a técnica de *Grid Search*[26], processo que busca achar os parâmetros ideais para o modelo testando-os de maneira exaustiva. Utilizando uma validação cruzada *K-Fold* no conjunto de dados de treinamento, esse foi subdividido em 5 partes iguais para obter os melhores parâmetros. Para o *Grid Search*, foram oferecidos 4 valores para taxa de aprendizado: 0,002, 0,01, 0,1 e 1.

[25] NLTK, Disponivel em: https://www.nltk.org/ Acesso em 22/11/2019
[26] Pesquisa em grade, tradução livre.

O treinamento requer dois parâmetros importantes para se obter um resultado ótimo. São eles o kernel da rede e o tipo de balanceamento. De acordo com a documentação SVM da biblioteca[27], o parâmetro de kernel define como o hiperplano pode ser dividido utilizando uma expressão matemática. Por tratar de dados de classificação de texto, onde cada alfabeto é uma característica nova podendo ser separado por um vetor, foi utilizado a função “linear”, definida por: $linear = \langle x, x' \rangle$.

O parâmetro “*class weight*” define o balanceamento da classe, conforme a documentação da biblioteca, e foi utilizado o valor “*balanced*”, devido às amostras balanceadas entre as classes positivas e negativas.

Ao final da busca, o algoritmo retornou os valores ideais para a taxa de aprendizado e uma pontuação média da validação cruzada (*score*), sendo esses respectivamente 0.1 para a taxa de aprendizado e 0.931 para a média da melhor pontuação. O treinamento foi realizado utilizando o algoritmo C-Support Vector Classification, uma variação matemática e de implementação do SVM apresentado por Chang e Lin (2013), e validação cruzada para apresentação do melhor resultado. (CHANG e LIN, 2013)

Os resultados da validação cruzada com o conjunto de teste são apresentados na tabela 6.

SVM	Acurácia	F1	Precisão	Recall	Perda
	0.929	0.862	0.857	0.867	4.343

Tabela 6 - Resultados do treinamento com SVM. Fonte: Os Autores

Os resultados indicam uma acurácia alta, porém com uma precisão e recall relativamente menores, o que indica que de modo geral, as predições verdadeiras são menores. A matriz de confusão da tabela 7 mostra que o modelo acerta ligeiramente mais as classes negativas, evidenciando os dados de precisão.

		Classe Atual	
		Positivo	**Negativo**
Classe Predita	**Positivo**	971	149
	Negativo	162	1191

Tabela 7 - Matriz de Confusão SVM. Fonte: (Os Autores)

[27] Disponivel em: https://scikit-learn.org/stable/modules/svm.html#svm-kernels Acesso em 22/11/2019

Observando as predições para as sentenças: a) a disciplina contribuiu para o meu aprendizado, classificado como positivo, e b) a disciplina não contribuiu para meu aprendizado, classificado como positivo, é possível observar que, o modelo apesar das métricas mostrarem um resultado aceitável, não consegue entender o contexto por intermédio da negação da sentença.

4.3 Naïve Bayes

As etapas do treinamento com o classificador Naïve Bayes podem ser consideradas de menor complexidade por não utilizar parâmetros no treinamento, dispensando também a utilização de *grid search* para encontrar os melhores parâmetros. Para a abordagem utilizando Naïve Bayes, foi utilizado uma implementação clássica do algoritmo e uma implementação utilizando um método de negação de sentença, proposto por Narayanan et al (2013), com o intuito de resolver o problema de sentenças negadas, apresentado na utilização do modelo SVM.

4.3.1 Abordagem Clássica

Os dados são primeiramente separados proporcionalmente em 80% para o conjunto de treinamento e 20% para o conjunto de testes. Em seguida, a realização de etapas de pré-processamento utilizando técnicas de *tokenização* e remoção de *stop words*. Para esse algoritmo, não foi utilizado a técnica de *stemming*, pois notou-se que os resultados decaiam aproximados 3% a 5% nas métricas de avaliação.

Utilizando o recurso *CountVectorizer* da biblioteca *sklearn*, os tokens são transformados em uma matriz de contagem de tokens. De acordo com a documentação da biblioteca[28], essa implementação resulta em uma representação esparsa da contagem de tokens.

Após a vetorização do conjunto de dados, foi escolhido o algoritmo Multinomial Naïve Bayes da biblioteca *sklearn*. Em probabilidade e estatística, a distribuição multinomial é uma generalização da distribuição binomial. O algoritmo Multinomial Naïve Bayes é um classificador com recursos discretos, como contagem de palavras para a classificação de texto.

[28] Biblioteca CountVectorizer, Disponivel em: https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html acesso em 22/11/2019

O classificador Multinomial Naïve Bayes é uma das duas implementações do clássico Naïve Bayes, utilizado para classificação de texto, onde os dados são representados por contagem de vetores[29]. O utilizando o *K-Fold* para validação cruzada, os resultados do treinamento e a matriz de confusão são apresentados na tabela 8.

NB	Acurácia	F1	Precisão	Recall	Perda
	0.875	0.863	0.888	0.839	4.316

		Classe Atual	
		Positivo	**Negativo**
Classe Predita	**Positivo**	970	122
	Negativo	189	1193

Tabela 8 - Resultados e matriz de confusão. Fonte: Os Autores

Nota-se que os resultados estão bem próximos do classificador SVM, exceto pela acurácia que agora se mostra 5% abaixo do SVM.

4.3.2 *Enhanced* Naïve Bayes

Para tentar resolver o problema apresentado nos classificadores com relação às sentenças negativas como “a disciplina não contribuiu para meu aprendizado”, foram utilizadas as técnicas propostas por Narayanan et al (2013), chamada de *Enhanced Naïve Bayes* (E-NB).

A implementação utiliza o algoritmo Bernoulli Naïve Bayes, outra variação do clássico Naïve Bayes, com suavização laplaciana, que ajuda a normalizar novas palavras que não foram vistas no set de treinamento, negação de sentença e *n-gramas*, que captura pares ou triplas de palavras para extrair conteúdo (NARAYANAN, ARORA e BHATIA, 2013).

[29] Biblioteca de dados Naïve Bayes, Disponivel em: https://nlp.stanford.edu/IR-book/html/htmledition/naive-bayes-text-classification-1.html acesso em 22/11/2019

Para o pré-processamento, apenas a tokenização é realizada em conjunto com a negação das palavras antes de realizar a contagem. O processo não envolve a utilização das técnicas de *stemming*, *lematização* e *stop words*.

Os dados são separados em 80% para treinamento e 20% para testes, randomizando todo o conjunto de amostras. Com a finalidade de preservar os resultados próximos aos apresentados por Narayanan et al. (NARAYANAN, ARORA e BHATIA, 2013), a implementação de todo o algoritmo foi realizada sem utilização de bibliotecas de suporte, utilizando os mesmos códigos do autor, sem implementação de funções de perda. A validação dos resultados é apresentada na tabela 9 utilizando a média de 5 treinamentos randomizando as amostras.

E-NB	Acurácia	F1	Precisão	Recall
	0.791	0.712	0.983	0.558

Tabela 9 - Validação dos resultados ENB. Fonte: Os Autores

Observando os resultados da tabela 9, é possível perceber que a precisão é desbalanceada com relação ao recall, o que significa que o classificador é especialista em predizer uma classe, mas retorna poucos resultados, ou os deixa passar, evidenciados na matriz de confusão da tabela 10.

		Classe Atual	
		Positivo	**Negativo**
Classe Predita	**Positivo**	640	11
	Negativo	506	1317

Tabela 10 - Matrix de confusão ENB. Fonte: Os Autores

Com relação ao problema de contextos de negação, agora o classificador consegue predizer corretamente uma sentença negada. Os exemplos a) a disciplina não contribuiu para meu aprendizado, e b) a disciplina contribuiu para meu aprendizado, desta vez foram classificados corretamente com o auxílio da negação de sentenças e *n-gramas*. Entretanto, o classificador tem um problema grande para predizer classes negativas, em sua maioria retornando falsos negativos.

4.4 BERT

A etapa que precede o pré-processamento é a escolha do modelo pré-treinado a ser utilizado. Tratando-se de um modelo de linguagem pré-treinado, Devlin, Chang, et al compartilharam modelos[30] utilizando conjuntos de dados variados, dependendo da escolha da língua a ser utilizada. Para o caso conjunto em Português, foi escolhido o ***multilingual cased***[31]. O modelo disponível foi treinado por Devlin, Chang et al a partir de um corpus textual da Wikipédia[32], onde os dados selecionados compõem artigos de diferentes línguas, essas escolhidas por estarem entre as 100 maiores Wikipédias, estando o português incluído.

O conjunto de dados do questionário foi separado 80% para o treinamento, e 20% para o conjunto de testes. Conforme a abordagem utilizando modelos de linguagem com BERT, as etapas de pré-processamento se diferem dos demais classificadores clássicos pela ausência das técnicas de transformações de palavras, como *lematização*, *stemming* ou remoção de *stop words*.

Os dados de entrada necessitam ser adaptados para a arquitetura da rede, onde as amostras são colocadas no formato: texto_a, texto_b e rótulo. Para a tarefa de classificação, o texto_b e rótulos são deixados em branco, pois estes são utilizados para a tarefa de *sentence pair*.

No pré-processamento, considerando o modelo *multilingual cased*, não é necessário transformar os dados para caixa baixa, uma vez que o modelo foi treinado utilizando um vocabulário com tokens de caixa alta. Nesta etapa, a tokenização acontece utilizando o *WordPiece* que mascara a palavra para a predição. É adicionado os sufixos [CLS] e [SEP] em cada sentença para que a rede identifique o começo e o fim destas para a tarefa de classificação.

Foi então definido o tamanho máximo da sequência de 256, truncando toda sentença que chegar nesse limite. As sentenças são então transformadas em *features*, de acordo com o tamanho do vocabulário, resultando em uma matriz vetorial onde cada token têm um valor numérico que o representa (*token embedding*), um

---

[30] Bert Multilingual models, Disponivel em: https://github.com/google-research/bert/blob/master/multilingual.md acesso em 20/11/2019
[31] Estrutura bicameral tipográfica, maiúsculas e minúsculas. Tradução Livre.
[32] Wikipédia, A enciclopédia livre. Disponivel em: https://pt.wikipedia.org/ acesso em 19/11/2019

valor numérico para toda a sentença (*sentence embedding*), e um valor de posição do token dentro da sentença (*transformer positional embedding*).

Foram definidos os parâmetros de batch size como 16, taxa de aprendizado de 2^{-5}, e sequência máxima de tokens de 256. Esses valores obtiveram os melhores resultados entre as outras tentativas para o treinamento utilizando outros valores. Os resultados apresentados na tabela 11 são os de melhor pontuação obtida dos 5 treinamentos realizados com o conjunto de testes com estes parâmetros, onde a variação dos resultados foi menor do que 0.01%.

BERT	Acurácia	F1	Precisão	Recall	Perda
	0.919	0.912	0.916	0.909	0.461

Tabela 11 - Resultados BERT. Fonte: Os Autores

Observando os resultados, para a tarefa de classificação de texto, o BERT demonstrou-se superior aos algoritmos classificadores SVM e Naïve Bayes, por uma grande margem nas métricas de avaliação dos modelos. Esse resultado é obtido em 5 épocas de treinamento, podendo-se notar que o modelo já começa a convergir na terceira época, como mostra a imagem 6.

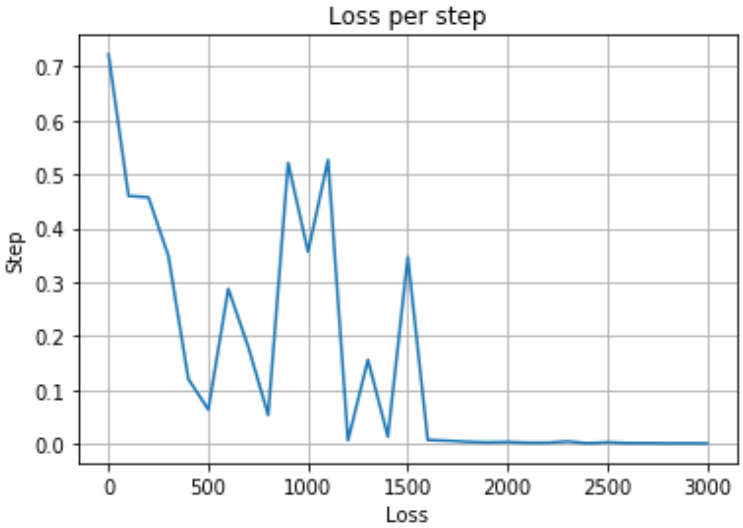

Figura 6 - Perda por etapas – BERT

Como resultado das predições (tabela 12), as sentenças **a)** a disciplina contribuiu para o meu aprendizado, e **b)** a disciplina não contribuiu para o meu aprendizado, foram classificadas corretamente como **a)** positiva e **b)** negativa,

evidenciando que o modelo bidirecional baseado em atenção compreende o contexto das sentenças. A matriz de confiança mostra os resultados das predições no conjunto de teste.

		Classe Atual	
		Positivo	**Negativo**
Classe Predita	**Positivo**	1041	96
	Negativo	105	1232

Tabela 12 - Matriz de confiança BERT. Fonte: Os Autores

As variações de parâmetros do modelo não apresentaram resultados distantes do melhor modelo treinado, sendo essas variações atingindo aproximados 0.02% nos resultados finais e, por este motivo, não foram apresentadas as comparações desses parâmetros. Entretanto, ao longo da pesquisa, observou-se que a quantidade de amostras revelou ser a principal causa na busca por melhores resultados. Inicialmente foi definido um número máximo de anotações em um determinado prazo de tempo estipulado para a pesquisa, juntando assim um total de 7.500 anotações. Deste conjunto de amostras, os resultados não satisfaziam a condição de pesquisa, cujo objetivo era chegar em um F1 score de 0.90. Ao avaliar os parâmetros utilizados, foi observado que a quantidade de amostras poderia ser pequena demais para um melhor aprendizado, e assim foi realizado outras duas etapas de anotações, reunindo um total de 10 mil amostras e 12 amostras consecutivamente. A tabela 13 apresenta os resultados do F1 score por quantidade de amostras, utilizando os mesmos parâmetros do melhor treinamento.

	7.500	**10.000**	**12.000**
F1 Score	0.891	0.906	0.912

Tabela 13 - F1 Score por Quantidade Total de Amostras. Fonte: Os Autores

4.5 MultiFiT

As sequências de etapas para o treinamento do modelo MultiFiT são: fine-tunning unidirecional do modelo de linguagem e em seguida, fine-tunning unidirecional do classificador. Para essas etapas que se seguem, foram seguidos os passos

apresentados[33] por Guillou (2019), porém sem utilizar o treinamento na direção reversa da rede. Para todas as etapas, o conjunto de dados utilizado foram os do questionário do sistema de avaliação institucional, e a separação foi mantida com 80% para treino e 20% para validação. A arquitetura da rede é construída utilizando os parâmetros de número de camadas como 4 e tamanho das redes QRNN de 1550 conforme os passos apresentados por Guillou (2019), e utilizando uma de regularização *Label Smooth*, proposto por (EISENSCHLOS, RUDER, *et al.*, 2019).

4.5.1 Fine-Tunning do Modelo de Linguagem

Como primeira etapa utilizando o modelo pré-treinado em português, foi realizado o processo de *forward* fine-tunning, iniciado buscando valores ideais para o parâmetro de taxa de aprendizado utilizando *Cyclical Learning Rates* (SMITH, 2017), O resultado da curva de perda por taxa de aprendizado é demonstrado na figura 7.

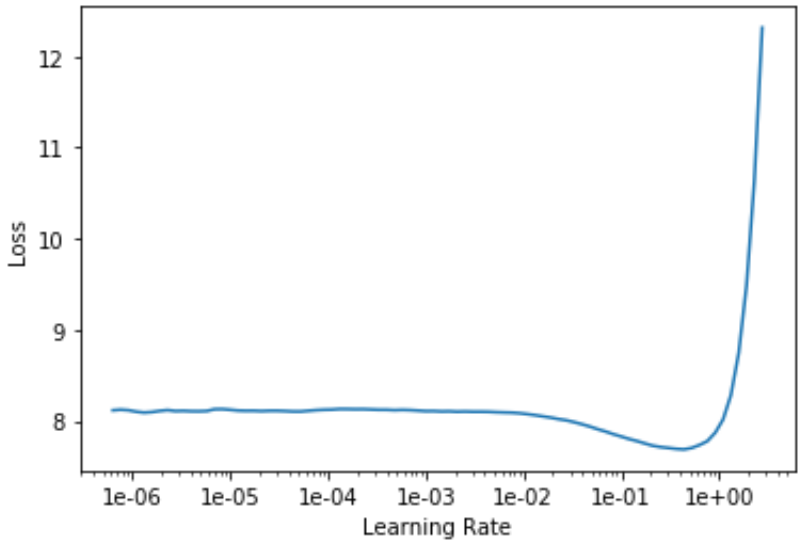

Figura 7 - Curva de perda sobre a taxa de aprendizado. Fonte: Os Autores

A partir do estudo apresentado por Smith (2017), os valores ideais de taxa de aprendizado para uma investigação inicial, pode ser obtida no ponto íngreme da curva, resultado assim em valores próximos à 2^{-2}, o qual foi adotado para esta etapa. Em seguida, foi utilizado um batch size de 48 e 18 épocas de treinamento e os resultados são apresentados na figura 8.

[33] Classificador MultiFit em português. Disponível em https://github.com/piegu/language-models/blob/master/lm3-portuguese-classifier-TCU-jurisprudencia.ipynb Acesso em: 22/11/2019

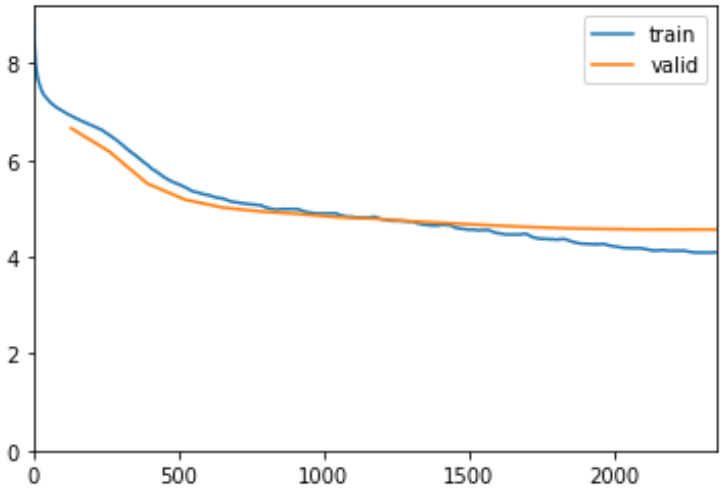

Figura 8 - Perda por época de treinamento – Forward Fine Tunning. Fonte: Os Autores

O resultado do treinamento apresenta uma curva de perda que se estabiliza, para o conjunto de validação, em aproximadamente 2000 passos. O resultado final é apresentado na tabela 14.

Época	Perda (Treino)	Perda (Validação)	Acurácia
18	4.098	4.568	0.305

Tabela 14 - Forward Fine-Tunning do Modelo de Linguagem. Fonte: Os Autores

É observado que após o treinamento, a acurácia do modelo de linguagem mostra valores abaixo de 50%, valores inferiores aos apresentados por Guillou (2019), indicando que o modelo atual é pouco eficiente para fazer predições com esse conjunto de dados em particular.

4.5.2 Fine-Tunning do Classificador

Para a tarefa de análise de sentimento, é treinado então, o modelo classificador. Os melhores resultados foram obtidos utilizando um *batch size* de 18, contrário ao *batch size* anterior.

Na busca por melhores parâmetros de taxa de aprendizado utilizando *Cyclical Learning Rates*, a curva de perda é apresentada na figura 10.

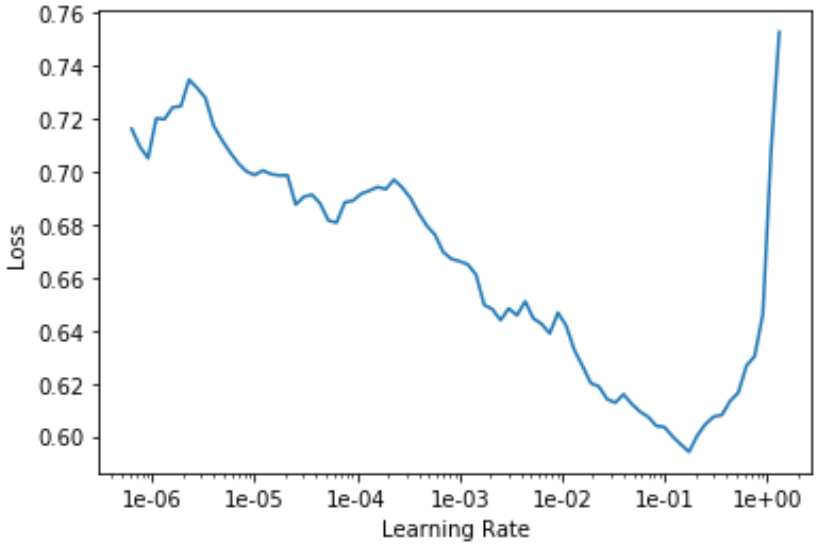

Figura 9 - Curva de Perda do Classificador - Forward Fine-tunning. Fonte: Os Autores

Nas etapas de treinamento foram utilizadas as técnicas descongelamento gradativo de camadas e taxas de aprendizado discriminativas (SMITH, 2017), utilizando os parâmetros de treinamento apresentados por Guillou (2019). Os resultados do forward fine-tunning são demonstrados na tabela 15.

F-MULTIFIT	Acurácia	F1	Precisão	Recall	Perda
	0.901	0.900	0.876	0.916	0.371

Tabela 15 - Resultados do Forward Fine-tunning do Classificador. Fonte: Os Autores

Apesar do modelo de linguagem apresentar baixa acurácia, o classificador apresenta bons resultados para as classificações (tabela 16).

		Classe Atual	
		Positivo	**Negativo**
Classe Predita	**Positivo**	1060	97
	Negativo	150	1165

Tabela 16 - Matriz de Confusão - Forward Fine-tunning do Classificador. Fonte: Os Autores

Ainda, como exemplo as sentenças a) a disciplina contribuiu para o meu aprendizado, e b) a disciplina não contribuiu para o meu aprendizado, foram corretamente classificadas pelo modelo direcional. Utilizando as ferramentas para visualização de atenção da biblioteca fast.ai, o resultado demonstra maior peso nas

palavras “contribuiu”, para a) e “contribuiu” e “não” para b), evidenciando o aprendizado dos contextos entre as sentenças.

4.6 Comparativo dos Modelos

Ao final do treinamento de todos os modelos, os resultados são apresentados na tabela 17.

Modelo	**Acurácia**	**F1**	**Precisão**	**Recall**	**Perda**
SVM	**0.929**	0.862	0.857	0.867	4.343
NB	0.875	0.863	0.888	0.839	4.316
E-NB	0.791	0.712	**0.983**	0.558	n/a
BERT	0.919	**0.912**	0.916	0.909	0.461
F-MultiFiT	0.901	0.900	0.876	**0.916**	**0.371**

Tabela 17 - Comparativos entre modelos. Fonte: Os Autores

Observa-se que, entre todos os modelos, individualmente cada técnica apresenta uma qualidade específica para predições. O modelo SVM obteve a melhor acurácia entre o conjunto de dados, entretanto a métrica de acurácia acaba sendo trivial se o modelo realmente não realizar predições. O *Enhanced Naïve Bayes* se mostra o modelo com maior precisão, onde o modelo é especialista para predições verdadeiras, mas faz poucas predições em geral. Nos modelos de linguagem, o MultiFiT unidirecional apresenta maior recall, acertando predições com a maior frequência e menor perda, evidenciando que as predições estão mais perto das classes anotadas. Por fim, o BERT se mostra em geral, o melhor modelo para predições considerando a métrica de F1, por ter o melhor balanceamento entre as métricas de validação, podendo ser considerado o modelo ideal a ser utilizado para a tarefa de análise de sentimento com o conjunto de dados do questionário.

5 CONCLUSÃO

A dificuldade de modelos treinados em português na internet traz a necessidade de treinamento de novos modelos capazes de resolver tarefas específicas. Linguagens específicas requerem corpus compatíveis para realizar o treinamento de tarefas específicas, descartando o uso de corpus textuais genéricos.

No âmbito do questionário de avaliação institucional da UNINTER, para o conjunto de dados anotados, o modelo que melhor atende à tarefa de análise de sentimento, dentre o conjunto de modelos utilizados, obtendo um F1 score de 0.912, o modelo BERT superou os objetivos iniciais propostos no projeto, podendo esse ser utilizado em sistemas de avaliação institucional como ferramenta de análise de sentimento. Assim, seguido pelo modelo MultiFiT com F1 de 0.9, é possível afirmar que modelos de linguagem têm melhores desempenhos quando comparados aos algoritmos de classificação clássicos, como SVM e Naïve Bayes, por melhor entender as relações entre as palavras e o contexto nas quais essas estão inseridas.

É válido observar que, na medida que foi aumentado o conjunto de dados anotados utilizando o BERT, foram-se obtendo melhores resultados, como apontado por Devlin, Chang et al (2018) na investigação sobre o treinamento dos modelos. Assim, para uma futura melhoria nos resultados do BERT utilizando o conjunto de dados do questionário, o aumento da quantidade de dados anotados pode levar à melhores resultados.

Já para o modelo MultiFiT, não houve significativa mudança nos resultados conforme os dados foram aumentando. Como o modelo de língua obteve uma baixa acurácia como resultado do treinamento e também no classificador, melhores resultados poderiam ser obtidos utilizando-se de parâmetros diferentes nas etapas de fine-tunning.

Observando o conjunto de dados do questionário, desconfia-se que os erros gramaticais produzem um vocabulário maior e, consequentemente, novas *features*. A relação entre as diferentes formas de escrever uma palavra pode acabar atrapalhando o modelo ao aprender o contexto dessas. Nesse caso, o número de amostras anotadas pode ajudar a melhorar esse processo afim de mostrar mais dados para os modelos de linguagem.

6 REFERÊNCIAS

BOSCO, G. L.; PILATO, G.; SCHICCHI, D. A Neural Network model for the Evaluation of Text Complexity in Italian Language: a Representation Point of View. **Procedia Computer Science**, v. 145, p. 464-470, 2018.

BRADBURY, J. et al. Quasi-Recurrent Neural Networks. **arXiv**, 2016. ISSN 1611.01576. Disponivel em: <https://arxiv.org/abs/1611.01576>.

CAMPBELL, C. An introduction to kernel methods. In: _____ **Radial basis function networks**. Vienna: Physica Verlag Rudolf Liebing KG, v. 1, 2001. p. 155-192.

CARUANA, R.; NICULESCU-MIZIL, A. **An empirical comparison of supervised learning algorithms**. ICML '06 Proceedings of the 23rd international conference on Machine learning. Pittsburg: Association for Computational Linguistics. 2006. p. 161-168.

CHANG, C.-C.; LIN, C.-J. LIBSVM: A library for support vector machines. **ACM Transactions on Intelligent Systems and Technology**, v. 2, Setembro 2013.

CHAPELLE, O.; SCHÖLKOPF, B.; ZIEN, A. **Semi-Supervised Learning**. [S.l.]: The MIT Press, 2006.

CHAUDIRON, S. Technologies linguistiques et modes de représentation de l'information textuelle. **Documentaliste-Sciences de l'Information**, v. 44, n. 1, p. 30-39, 2007. Disponivel em: <https://www.cairn.info/revue-documentaliste-sciences-de-l-information-2007-1-page-30.htm>.

DEVLIN, J. et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. **NAACL 2019**, 2019.

EISENSCHLOS, J. et al. MultiFiT: Efficient Multi-lingual Language Model Fine-tuning. **Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)**, p. 5706-5711, Janeiro 2019.

GIL, A. C. **Métodos e Técnicas de Pesquisa Social**. 7a. ed. [S.l.]: Editora Atlas, 2019.

HASSOUN, M. H. **Fundamentals of Artificial Neural Networks**. [S.l.]: MIT Press, 1995.

HAYKIN, S. **Redes Neurais:** Princípios e Prática. 2a. ed. [S.l.]: Ed. Bookman, 2007.

HOWARD, J.; RUDER, S. **Universal Language Model Fine-tuning for Text Classification**. Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics. [S.l.]: [s.n.]. 2018. p. 328-339.

JURAFSKY, D.; MARTIN, J. H. **Speech and Language Processing:** An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. 3rd (Draft). ed. [S.l.]: Prentice Hall, 2019. Disponivel em: <https://web.stanford.edu/~jurafsky/slp3/ed3book.pdf>.

KUDO, T.; RICHARDSON, J. SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing. **Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations**, Bruxelas, p. 66-71, Novembro 2018.

KULTZAK, A. F. **CATEGORIZAÇÃO DE TEXTOS UTILIZANDO ALGORITMOS DE APRENDIZAGEM DE MÁQUINA COM WEKA**. UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ. Ponta Grossa. 2016.

KUMAR, E. **Natural Language Processing**. [S.l.]: I. K. International Pvt Ltd, 2011.

LIU, B. **Synthesis Lectures on Human Language Technologies**. [S.l.]: Morgan & Claypool, 2012.

LIU, B. **Opinions, Sentiment, and Emotion in Text**. [S.l.]: Cambridge University Press, 2015.

LUGER, G. F. **Inteligencia Artificial**. 6a. ed. São Paulo: Pearson Education do Brasil Ltda, 2013.

MAAS, A. L. et al. **Learning Word Vectors for Sentiment Analysis**. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. Portland: Association for Computational Linguistics. 2011. p. 142-150.

MANNING, C. D.; SCHÜTZE, H. **Foundations of Statistical Natural Language Processing**. [S.l.]: MIT Press, 1999.

MURPHY, K. P. **Machine Learning:** A Probabilistic Perspective (Adaptive Computation and Machine Learning series). [S.l.]: The MIT Press, 2012.

NARAYANAN, V.; ARORA, I.; BHATIA, A. **Fast and Accurate Sentiment Classification Using an Enhanced Naive Bayes Model**. [S.l.]: Intelligent Data Engineering and Automated Learning, Proceedings, 2013.

OTTER, D. W.; MEDINA, J. R.; KALITA, J. K. A Survey of the Usages of Deep Learning in Natural Language Processing, v. arXiv:1807.10854 , 2018.

RUDER, S.; EISENSCHLOS, J. Efficient multi-lingual language model fine-tuning. **NPL FastAI**, 2019. Disponivel em: <http://nlp.fast.ai/classification/2019/09/10/multifit.html>. Acesso em: 22 Novembro 2019.

RUSSEL, S.; NORVIG, P. **Artificial Intelligence, A Modern Approach**. 3a. ed. New Jersey: Prentice Hall, 2010.

SCHMIDHUBER, J. Deep learning in neural networks: An overview. **Neural Networks**, v. 61, p. 85-117, 2015.

SCHUSTER, M.; NAKAJIMA, K. Japanese and Korean Voice Search. **1988 International Conference on Acoustics, Speech, and Signal Processing, 1988. ICASSP-88**, p. 5149-5152, Março 2012.

SILVA, R. R.; LIMA, S. M. B. Consultas em Bancos de Dados Utilizando Linguagem Natural. **Revista Eletrônica da Faculdade Metodista Granbery**, v. 002, p. 1-20, 2007.

SMITH, L. N. **Cyclical Learning Rates for Training Neural Networks**. [S.l.]: arXiv, 2017. Disponivel em: <https://arxiv.org/abs/1506.01186>. Acesso em: 22 Novembro 2019.

SUGIYAMA, M. **Statistical Reinforcement Learning, Modern Machine Learning Approaches**. [S.l.]: CRC Press, Taylor & Francis Group, LLC, 2015.

SUTTON, R. S.; BARTO, A. G. **Reinforcement Learning:** An Introduction. [S.l.]: MIT Press, 2018.

VASWANI, A. et al. Attention Is All You Need. **Advances in Neural Information Processing Systems**, p. 6000-6010, 2017.