UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

CARLOS HENRIQUE DO NASCIMENTO MELO

Análise de sentimentos de postagens em português na pandemia de COVID-19 utilizando redes de codificadores automáticos

Recife
2023

CARLOS HENRIQUE DO NASCIMENTO MELO

Análise de sentimentos de postagens em português na pandemia de COVID-19 utilizando redes de codificadores automáticos

Trabalho apresentado ao Programa de Pós Graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

Área de Concentração: Inteligência Computacional
Orientador: Leandro Maciel Almeida

Recife
2023

Catalogação na fonte
Bibliotecária Monick Raquel Silvestre da S. Portes, CRB4-1217

M528a Melo, Carlos Henrique do Nascimento
Análise de sentimentos de postagens em português na pandemia de COVID-19 utilizando redes de codificadores automáticos / Carlos Henrique do Nascimento Melo. – 2023.
184 f.: il., fig., tab.

Orientador: Leandro Maciel Almeida.
Dissertação (Mestrado) – Universidade Federal de Pernambuco. CIn, Ciência da Computação, Recife, 2023.
Inclui referências e apêndices.

1. Inteligência computacional. 2. Codificadores automáticos. I. Almeida, Leandro Maciel (orientador). II. Título.

006.31 CDD (23. ed.) UFPE - CCEN 2023-174

CARLOS HENRIQUE DO NASCIMENTO MELO

"**Análise de sentimentos de postagens em português na pandemia de COVID-19 utilizando redes de codificadores automáticos**"

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

Aprovado em: 19/07/2023.

Orientador: Leandro Maciel Almeida

BANCA EXAMINADORA

Prof. Dr. Luciano de Andrade Barbosa
Centro de Informática / UFPE

Prof. Dr. João Fausto Lorenzato de Oliveira
Universidade de Pernambuco / UPE

Decido este trabalho a minha família e minha namorada que foram porto seguro perante as dificuldades durante este percurso.

AGRADECIMENTOS

Agradeço primeiramente a minha mãe Maria Diana do Nascimento Melo, porque sem ela eu não conseguiria chegar até onde cheguei, nem conquistar nada que alcancei. Ao meu orientador, Leandro Almeida, que me instruiu de modo tão dedicado para que eu obtivesse este grau. Minha esposa Izabelly Maria da Silva Mota que me deu suporte na etapa final, tanto como apoio emocional (me estimulando a sempre continuar e finalizar) como com suporte de correções. Agradeço a todas as pessoas que torceram por mim e que me incentivaram a sempre continuar e nunca desistir de um sonho que sempre foi tão meu, mas que em momentos de nossas vidas tendemos a tropeçar. Minha irmã Rayssa que sempre foi uma grande encorajadora e me ajudou em momentos que nem eu saberia como passar por eles. Agradeço também ao CAPES que através de suas bolsas de incentivo a edução me deu suporte a conseguir focar na educação.

"Depois de escalar uma grande montanha se descobre que existem muitas outras montanhas para escalar"(MANDELA, 2016).

RESUMO

A pandemia da Corona Virus Disease 2019 (COVID-19) impulsionou um aumento no número de interações em redes sociais, através de postagens, em virtude das medidas não farmacológicas implementadas durante o período de 2020 e 2021. Essa maior conexão da população com as diversas plataformas propiciaram uma grande quantidade de conteúdos textuais relacionados à vivência dos usuários nos períodos de surto da doença. Muitas dessas postagens apresentam um caráter opinativo, no qual indica a possibilidade de um estudo acerca dos sentimentos expressados pelos usuários das redes sociais. Desse modo, a utilização de técnicas da área de Processamento de Linguagem Natural (PLN) em conjunto com modelos da Aprendizagem de Máquina (AM) fornecem uma análise de sentimentos através de classificadores automáticos. Porém, é visto em estudos anteriores que a tarefa de Análise de Sentimentos (AS) sofre da maldição da dimensionalidade (CHEN, 2009), pois os métodos principais de transformar conteúdo textual em informação útil recaem sobre vetores de grande dimensionalidade. Em pesquisas mais recentes, o uso de técnicas de redes neurais têm sido utilizadas como método de redução da dimensionalidade para a classificação de sentimentos (GHOSH; RAVI; RAVI, 2016; KIM; LEE, 2020; YILDIRIM, 2020). Dentre as técnicas, os Codificadores Automáticos (CA) (do inglês *autoencoders*) surgem como uma proposta já utilizada para redução de dados de imagem e áudio, pois processa vetores desses conteúdos e os reduz para diferentes propósitos. A utilização das RNN para redução possibilita construir um novo vetor contendo uma grande proporção da informação contida no vetor original para realizar o treinamento dos modelos. Portanto, este trabalho apresenta como objetivo explorar a técnica de CA para redução da dimensionalidade de vetores produzidos por técnicas de incorporação de palavras sobre dois *corpus* textuais na língua portuguesa, coletados através da rede social *Twitter*. Baseado nisso foi visto que os codificadores conseguem manter até 90% da informação e qualidade contida no treinamento, podendo ser observado uma diferença de pouco menos de 10% na acurácia dos modelos treinados sem a técnica. Além disso, é observado que custo computacional envolvido no treinamento dos modelos apresentaram uma redução em comparação aos classificadores treinados com o vetor original e aos modelos mais recentes, como LSTM e BERT, apresentando uma diferença de tempo de até 96%. Assim, mostra que a partir dos resultados obtidos através da técnica de redução por codificadores automáticos são produzidas qualidades equiparáveis aos modelos mais utilizados que realizam essa codificação de forma conjunta para a língua portuguesa. Desse modo, possibilita o uso de modelos mais custosos para a validação de resultados e uso de predição.

Palavras-chaves: análise de sentimentos; codificadores automáticos; comitê de classificadores; classificação de múltiplas classes; COVID-19.

ABSTRACT

The pandemic of Corona Virus Disease 2019 (COVID-19) stimulates an increase in the interactions on social media through posts, due to measures not pharmacological that were implemented during the period 2020 and 2021. This increase in the connection of the population with the various platforms provided a large amount of textual content related to the experience of users in periods of outbreak of the disease. Many of these posts have a tone of opinion, which indicates the possibility of a study about the feelings expressed by users of social media. Thus, the use of techniques of Natural Language Processing (NLP) in conjunction with Machine Learning (ML) models provide a sentiment analysis through automatic classifiers. However, it is seen in previous studies that the Sentiment Analysis (SA) task suffers from the curse of dimensionality (CHEN, 2009), as the main methods of transforming textual content into useful information are attributed to high-dimensional vectors. In the latest researches, the use of neural network techniques has been used as a dimensionality reduction method for classifying feelings (GHOSH; RAVI; RAVI, 2016; KIM; LEE, 2020; YILDIRIM, 2020). Among the techniques, the Automatic Encoders (AC), also known as autoencoders, emerge as a proposal already used for image and audio data reduction, as it processes vectors of these contents and reduces them for different purposes. The use of RNN for reduction makes it possible to build a new vector containing a large proportion of the information found in the original vector to perform the training of the models. Therefore, this work aims to explore the CA technique to reduce the dimensionality of vectors produced by techniques of word incorporation on two textual corpus in Portuguese, collected through the social network Twitter. Based on this, it was seen that the coders manage to maintain up to 90% of the information and quality contained in the training, and a difference of just under 10% can be observed in the accuracy of the models trained without the technique. Beyond that, it is observed that the computational cost involved in training the models showed a reduction in comparison to the classifiers trained with the original vector and to the most recent models, such as LSTM and BERT, presenting a time difference of up to 96%. Thus, this demonstrates that from the results obtained through the technique of reduction by automatic coders, the qualities comparable to the most used models that carry out this coding together for the Portuguese language are produced. Thus, it enables the use of more costly models for validating results and using prediction.

Keywords: sentiment analysis; autoencoders; ensemble classifier; muticlass classification; COVID-19.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

AE	*Autoencoders*
AM	Aprendizagem de Máquina
AP	Aprendizagem Profunda
API	Interface de programação de aplicações
AS	Análise de Sentimentos
BERT	*Bidirectional Encoder Representations for Transformers*
BoW	Bag of Word
CA	Codificadores Automáticos
CNN	Rede Neural Convolucional
COVID-19	Corona Virus Disease 2019
CSV	Comma Separated Values
DL	*Deep Learning*
ensemble	Comitê de Classificadores
FNN	*Feedforward neural network*
GNNs	*Graph Neural Networks*
IA	Inteligência Artificial
ICICT	Instituto de Comunicação e Informação Científica e Tecnológica em Saúde
IPD	Índice de Permanência Domiciliar
KNN	*K — Nearest Neighbors*
LDA	*Linear discriminant analysis*
LSA	*Latent Semantic Analysis*
LSTM	*Long short-term memory*
ML	*Machine Learning*
MLP	*multilayer perceptron*
NB	*naive bayes*
NMF	*Non-negative matrix factorization*
OMS	Organização Mundial da Saúde
PCA	*Principal component analysis*
PLN	Processamento de Linguagem Natural

PMI	*Pointwisemutual-information*
QDA	Quadratic Discriminant Analysis
RCA	rede de codificadores automáticos
RNN	Rede Neural Recorrente
RSLP	Removedor de Sufixos da Lingua Portuguesa
SNS	Social Networking Services
SQL	Structured Query Language
STF	Supremo Tribunal Federal
SVM	máquina de vetores de suporte
t-SNE	*t-distributed stochastic neighbor embedding*
TBCOV	two billion multilingual COVID-19 tweets with sentiment, entity, geo, and gender labels
TFIDF	*Term Frequency - Inverse Document Frequency*
UMAP	*uniform manifold approximation and projection*
UnB	Universidade de Brasília

SUMÁRIO

1 INTRODUÇÃO

O início do ano de 2020 foi marcado pelo surto de uma doença da família de vírus que causam problemas respiratórios conhecidos como síndrome respiratória aguda grave. Essa doença, mais tarde foi denominada síndrome respiratória aguda grave coronavírus 2 ou sars-Cov-2, assim como popularmente chamado de COVID-19. De acordo com Platto, Xue e Carafoli (2020) foi proposto que, inicialmente, o vírus surgiu de forma natural em Wuhan, capital da província de Hubei, na China, infectando parte da população que eram clientes no mercado local.

Porem, esta abordagem não consegue explicar totalmente suas características como visto em Aria et al. (2022). O primeiro caso relatado dessa doença ocorreu em Wuhan, na China, no começo de dezembro de 2019 (BARBERIA et al., 2021; CANDIDO et al., 2020; IGLECIAS, 2022; JACQUES et al., 2022). Após poucos meses do primeiro caso, em março de 2020, a doença já estava presente em 117 países (CANDIDO et al., 2020) e devido a isto, a Organização Mundial da Saúde (OMS) por meio do Diretor-geral Tedros Adhanom, declarou oficialmente que o mundo vivenciava um estado de pandemia (IGLECIAS, 2022).

A alta propagação da COVID-19 ocorreu por conta das suas características de transmissão, no qual as pessoas infectadas poderiam contaminar outras pessoas, por meio de espirros, tosse ou até mesmo de sua saliva, visto que, nelas contém gotículas com o vírus. Além disso, se essas mesmas gotículas estiverem presentes nas mãos do contaminado, um simples aperto de mão, levar as mãos à região do rosto e passar nos olhos, boca ou nariz sem lavá-las antes, pode contagiar outra pessoa.

A primeira morte em decorrência da COVID-19 no território brasileiro foi registrado em 12 de março de 2020, no estado de São Paulo, em sua cidade capital. O restante dos estados brasileiros também apresentaram óbitos em decorrência de complicações do vírus a partir de março do mesmo ano. Criando assim, um estado de atenção em todo o território nacional, no qual cada governo estadual adotou medidas para combater a doença.

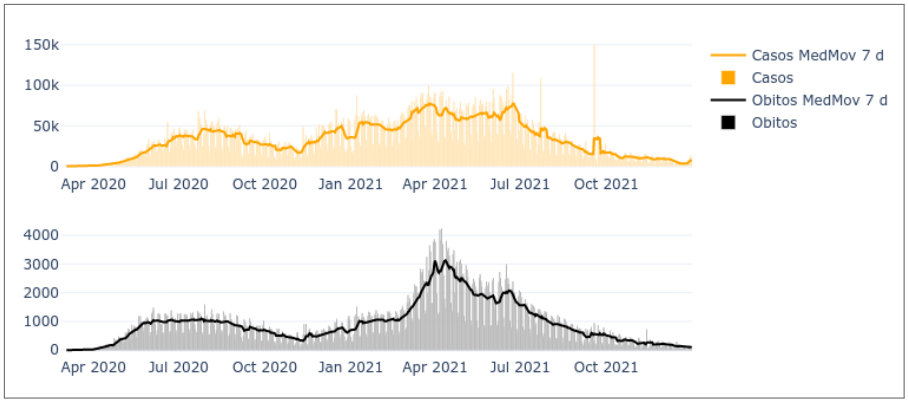
Para podermos observar como a COVID-19 se propagou no país, a figura 1 apresenta um gráfico que mostra a tendência dos casos e óbitos para os anos de 2020 e 2021. No gráfico de linha apresenta uma média móvel de 7 dias, e o de barras é mostrado a quantidade real por dia.

A partir desse gráfico na figura 1, é exibido dois grandes picos da doença nos anos de 2020 e 2021. O primeiro, visto logo após a entrada do vírus no país, em março de 2020, no qual os casos e óbitos apresentam um grande aumento.

Já o segundo pico, é visto no ano de 2021, após o mês de fevereiro, mesmo havendo a vacina no país, ocorre um grande crescimento nos casos e óbitos em decorrência da

[0] Disponível em: <https://bigdata-covid19.icict.fiocruz.br/>, Acesso em: 02 Out. 2022

Figura 1 – Evolução de casos e óbitos no período 2020 a 2021



Fonte: Elaboração própria, dados provenientes da ICICT.

COVID-19. Esse segundo pico, visto como maior que o ocorrido no ano anterior, foi relatado como decorrência da segunda onda.

É visto que, entre os meses de setembro e início de dezembro de 2020, os casos e óbitos apresentaram uma queda em seus índices. Após esse período, é visto um novo aumento, o que mostra que, devido a um relaxamento das medidas aplicadas inicialmente pelos governos estaduais e prefeituras.

A partir desse novo aumento dos índices no início do ano de 2021, só é observado uma nova queda após julho de 2021, após novas implementações de medidas restritivas em alguns estados e municípios e também do crescimento do número de pessoas vacinadas com a vacina de combate ao COVID-19.

As medidas acima citadas são denominadas "não farmacológicas"por serem ações tomadas sem suporte de remédios, vacinas, etc. Elas foram importantes para tentar achatar a curva de crescimento dos casos e óbitos, assim, como também influenciaram no dia a dia de usuários nas redes sociais.

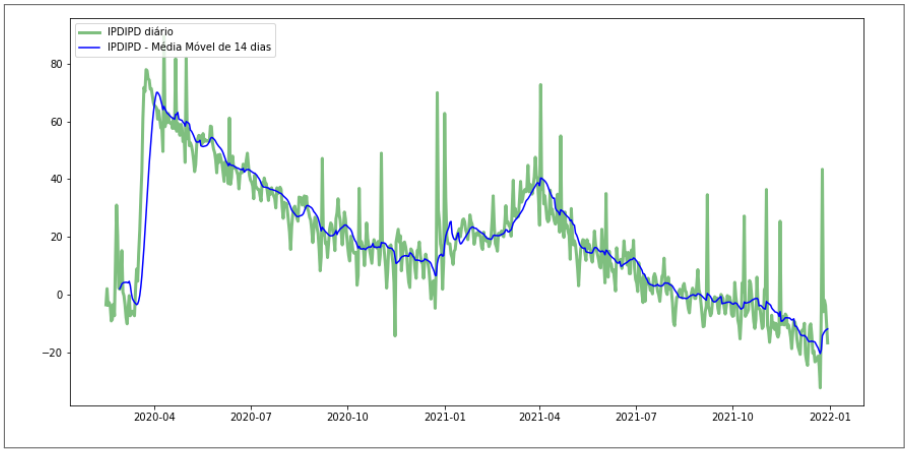
Por causa delas, muitos dados foram gerados e deram a possibilidade de realizar o estudo proposto neste trabalho. O próximo subcapítulo apresenta um detalhamento maior acerca dessas medidas não farmacológicas, bem como as políticas implementadas durante o período de pandemia.

1.1 POLÍTICAS PÚBLICAS

Ao longo dos primeiros meses da pandemia de COVID-19, foram empregadas diversas políticas de combate a propagação do vírus, muitas eram medidas não farmacológicas, que em outras regiões do mundo se mostraram eficazes e também foram recomendadas pela OMS. De acordo com Barreto et al. (2021), cada medida utilizada tinha um grau de rigidez em sua aplicação. Entre ações mais realizadas estão: distanciamento social, restrição de multidões, uso de máscaras, rodízio de fluxo de veículos, restrição de deslocamento ao trabalho (Somente profissões essenciais poderiam manter-se em trabalho) e *lockdown*.

O grau de rigidez para cada medida foi chamado de Índice de Permanência Domiciliar (IPD), que determinava o percentual de locomoção nas cidades. Nos quais valores próximos a 100 determinavam que poucos dispositivos analisados se locomoviam pelas cidades. Para calcular a rigidez ou efetividade de cada política foram utilizados os dados disponibilizados pelo *Google Mobility Report* que expõe as informações de deslocação dos usuários de Android e aplicativos Google. O gráfico apresentado na figura 2 apresenta a tendência de permanência durante os anos de 2020 e 2021 no Brasil.

Figura 2 – Índice de permanência domiciliar.



Fonte: Elaboração própria, dados provenientes da ICICT.[1]

A partir do gráfico da figura 2 foi possível determinar em quais períodos dos anos é apresentado uma maior índice de permanência, ou não locomoção nas regiões das cidades. Desse modo, baseando-se no IPD, é visto períodos de medidas restritivas mais rígidas e realizar estudos através das postagens durante estes períodos, para verificar a opinião dos usuários de determinadas redes sociais. Assim, o *twitter* será uma ferramenta principal para a preparação de uma base de dados (ou *corpus*) textual para tal estudo.

Sabe-se que o IPD pode influenciar, também, na quantidade de dados compartilhados através destas redes sociais, pois se entende que elas foram recorridas com os mais diversos motivos. Além da busca incessante por informações acerca do que estava ocorrendo afora, também foram um meio de compartilhar sentimentos e opiniões do que ocorria durante a pandemia. Essas opiniões também eram sobre as políticas públicas e medidas não farmacêuticas aplicadas.

Diante das discussões empreendidas, foi observado a oportunidade de realizar estudo nessa fonte de informação valiosa que se criava durante o período pandêmico. No qual será necessário realizar uma coleta dessas informações, para realizar procedimentos necessários, visando transformar esse dado não estruturado em informação útil.

Porém, para isso, é preciso entender acerca das redes sociais, então, no subcapítulo seguinte é destacado o valor e os detalhes sobre a rede social aqui estudada. Além disso,

[1] Disponível em: <https://bigdata-covid19.icict.fiocruz.br/>. Acesso em: 10 Out. 2022

também será discutido o uso de técnicas de processamento de textos necessários para realizar o estudo proposto.

1.2 AS REDES SOCIAIS E O USO DA PLN

Como foi observado, a pandemia impulsionou um grande aumento no consumo e compartilhamento de informações por meio das redes sociais. Esse aumento também foi observado em outras mídias da internet como *streamings* de jogos, filmes, series, entre outros (PIMENTEL et al., 2021; MARTINS, 2023). Porém, o aumento do consumo dessas mídias não pode ser somente associado à pandemia, visto que, já se tinha uma tendência de aumento antes vista. Por outro lado, as redes sociais demonstraram ter uma relação do aumento dos dados relacionados à pandemia (RONCERO; GARCÍA, 2014; ROMAN, 2017; ANTÓN, 2018; MATSUMOTO, 2019).

Mesmo antes do cenário pandêmico, as redes sociais já eram um meio de comunicação amplamente utilizado em todo o mundo, acessível a quase toda pessoa que tenha acesso à internet. Desse modo, são responsáveis, hoje, por gerarem uma imensa quantidade de dados estruturados e não estruturados, contribuindo com a maior base de dados hoje disponível, chamada *Big Data* (MELLO; CAMILLO; SANTOS, 2019).

Atualmente, existem diversas redes sociais, e cada uma com uma forma específica de divulgação de informação por meio de postagens, sendo por meio de textos, imagens, vídeos entre outras. Segundo Maia et al. (2021), tais plataformas fornecem um rica base de informações a respeito de fenômenos sociais que podem ser estudados. As redes sociais mais utilizadas para compartilhamento de informações, opiniões e entretenimento são: *Facebook*, *Instagram*, *Twitter*, *TikTok* e outras.

O *Twitter* é uma fonte de dados onde muitas pesquisas acadêmicas têm buscado informações de sentimentos e opiniões dos seus usuários (RAHMAN et al., 2021; KOH; LIEW, 2020; MORSHED et al., 2021). Isso ocorre por ser uma rede social mais focada em postagens de textos que outro tipo de mídia. O *Twitter* é baseado em um microblog, no qual propõe que sua principal forma de publicações sejam textos de até 280 caracteres, para que seus usuários expressem suas opiniões acerca de um determinado assunto (GADINI, 2020).

Essa rede social possui mais de 1 bilhão de contas criadas, 190 milhões de usuários ativos por dia e 500 milhões de novas postagens (os *tweets*). Para categorizar os *tweets*, existem as *hashtags*, palavras iniciadas pelo símbolo cerquilha, que inserem esses textos em um tópico ou tema especifico. Esse método de incluir um texto em um tema do *Twitter* é importante, pois permite filtrar de forma mais fácil textos relevantes para uma determinada pesquisa ou estudo. Assim como é apresentado por (FARIAS; OLIVEIRA, 2022), o *Twitter* foi escolhido por ter grande parte de suas publicações de acesso publico através de API da própria plataforma, ou *crawling* da pagina da rede social.

Essas pesquisas muitas vezes utilizam métodos da computação para alcançar conhecimento de padrões e assim poder classificar textos em rótulos específicos. Desse modo,

a classificação muitas vezes recai sobre algoritmos de Aprendizagem de Máquina (AM) que buscam ensinar os computadores a extraírem características que definam um atributo (ou texto) pertencente a tal rótulo. Porém, para ser possível utilizar esses algoritmos em dados não estruturados do tipo textual é necessário realizar diversos tratamentos sobre eles para os converterem em algo que o computador possa entender, e para isso é utilizado técnicas da Processamento de Linguagem Natural (PLN).

De acordo com Barbosa e Campelo (2020) a PLN é uma area da IA que busca da poder aos computadores de entender textos escritos e falados. McShane e Nirenburg (2021) descreve a PLN como uma subárea da computação e da linguística que busca desenvolver modelos através de técnicas da AM e da linguística para processar a linguagem humana.

Desse modo, nela se que busca encontrar padrões, processar e compreender a linguagem humana (falada ou escrita). Para atingir esse objetivo, diversas técnicas como limpeza de dados, vetorização de *corpus*, remoção de palavras vazias (*stop word*) entre outras são dadas por este campo.

Entre as técnicas da PLN, a Análise de Sentimentos (AS) busca deduzir o sentimento expressado no texto, no qual varia em duas polaridades: Negativa e Positiva. Benevenuto, Ribeiro e Araújo (2015, p.32) define a AS como "definir técnicas automáticas capazes de extrair informações subjetivas de textos em linguagem natural, como opiniões e sentimentos, a fim de criar conhecimento estruturado que possa ser utilizado por um sistema de apoio ou tomador de decisão". Porém, quando se trabalha com textos no twitter, sabe-se que também muitas dessas mídias têm caráter informativo e não expressam nenhum desses dois sentimentos, assim sendo classificados como Neutros.

Desse modo, o objeto de estudo deste trabalho são dados textuais filtrados para estarem dentro do tema da pandemia, entende-se que existe a possibilidade de milhares de textos de cunho informativo entre os demais coletados neste período. Então, para o processo de classificação foi escolhido esses três rótulos para determinarem a opinião dos usuários do *Twitter* durante a pandemia em negativo, positivo e neutro. Assim, possibilita a compreensão das opiniões importantes acerca das políticas públicas aplicadas, excluindo os textos neutros.

Portanto, para realizar as classificações propostas para esta tarefa, foi escolhido o uso de algumas abordagens da AM. Dentre elas, classificadores tradicionais como *Naive Bayes*, máquina de vetores de suporte (SVM) e *multilayer perceptron* (MLP) buscaram padrões nos vetores produzidos pelas técnicas da PLN entre as classes propostas. Sendo elas, técnicas mais utilizadas na literatura, irão apresente métricas de base para a escolha de parâmetros na metodologia proposta. Além dos algoritmos destacados, também foi proposto a utilização de algumas técnicas de Comitê de Classificadores (ensemble) através da escolha de algoritmos para compor o conjunto classificador por meio de métricas obtidas.

De acordo com Han, Kamber e Pei (2012, p.315, Tradução nossa) Os ensemble são

"uma série de k modelos de aprendizado (ou classificadores de base), com o objetivo de criar um modelo de classificação composto aprimorado". Desse modo, essa técnica produz um modelo classificador que em muitos casos ultrapassam a qualidade obtida entre os classificadores únicos utilizados. Sendo assim, possibilita a incorporação de técnicas mais recentes dentro desse conjunto de algoritmos, visando a produção de resultados maiores. Por meio dessas abordagens propostas, busca-se avaliação dos resultados da metodologia sugerida para este trabalho.

1.3 OBJETIVOS

Este trabalho pretende realizar uma classificação de sentimentos de uma base criada a partir da rede social twitter e coletada por meio de palavras-chave denominadas como *hashtag's*[2]. Essas *tag's* determinaram os textos que estarão contidos no contexto da pandemia de COVID-19, sendo os termos ou palavras mais utilizadas na rede social com o intuito de classificar o texto como pertencente ao tema. A escolha do uso das *hashtag's* como parâmetro de busca se dá devido a ter sido observado, através de pesquisas realizadas durante o ano inicial da COVID-19 (PESSANHA et al., 2020; POKHAREL, 2020; MANGURI; RAMADHAN; AMIN, 2020), que o uso foi amplamente difundido na rede social para discorrer sobre o tema. Desse modo, o *corpus* criado apresentará um cenário baseado nas opiniões expressadas pelos usuários acerca da pandemia.

Para isso, este trabalho se propõe a utilizar dois *corpus* pré-rotulados com os três sentimentos (Negativo, Positivo e Neutro) visando realizar a classificação do *corpus* criado. Além disso, será realizado uma análise da junção de diversas técnicas de processamento de linguagem natural como: redução das características a termos principais, vetorização, dicionario léxico que corrige termos informais e outros. No intuito de melhorar os resultados e reduzir o custo computacional que a tarefa de análise de sentimentos se mostra necessária.

Por fim, como ponto principal deste trabalho, a análise será realizada sobre um *corpus* de representação de dimensionalidade reduzida, produzido pela técnica de redes de codificadores automáticos que será utilizada sem estar vinculado a um algoritmo específico, como ocorre em diversos modelos (Exemplo o *Long short-term memory* (LSTM)). Todo esse estudo será com base em um *corpus* de língua portuguesa. Os modelos serão treinados com base em duas bases de dados textuais, também na língua portuguesa pré-rotulados. No qual, um é disponibilizado através da plataforma *Kaggle* e o outro foi disponibilizado pelo trabalho de Imran, Qazi e Ofli (2022a).

- desenvolver e analisar um modelo pré-treinado de rede de codificadores automáticos (RCA) para redução da dimensionalidade a partir de vetores de textos no idioma

[2] A *hashtag* é uma palavra escrita com o símbolo # em seu início, possibilitando determinar que um texto está contido em um tópico, ou tópicos, especifico.

português contextualizado sobre o período de pandemia mundial da COVID-19.

Objetivos específicos:

- Testar e treinar uma rede de codificadores automáticos, contendo uma diversificação de parâmetros, com base nos dois conjuntos de dados de treino estipulados para este trabalho.
- Comparar resultados de acurácia dos modelos tradicionais com modelos mais recentes da AM para a PLN.
- Utilizar a redução da dimensionalidade, através das RCA, sobre os corpus de treino e comparar resultados obtidos pelos modelos utilizando o vetor original e o vetor de representação (Reduzido).
- Elaborar um comitê de classificadores a partir dos modelos pré-visualizados, comparando-o com os resultados obtidos através dos classificadores tradicionais (únicos) com melhor qualidade.

Diante da execução dos objetivos gerais e específicos, é esperado responder às seguintes perguntas:

1. Foi observada uma grande diferença entre os modelos treinados com o *corpus* mais generalista e o específico (kaggle e TOBCV, respectivamente)?
2. Quais algoritmos de aprendizado de máquina apresentam melhores resultados com base nos dois *corpus* utilizados (disponibilizada pelo kaggle e TOBCV)?
3. Comitês classificadores apresentam uma melhora significativa nos resultados das bases?
4. O uso de *autoencoders* melhoraram significativamente ou se equipararam aos resultados e custos computacionais nos modelos utilizados?
5. Quais emoções predominam em cada período da pandemia no Brasil?
6. Quais emoções predominam durante o período restrito nos estados que aplicaram o *lockdwon*?

1.4 JUSTIFICATIVA

A literatura mostra que o uso de algoritmos de aprendizado de máquina para descoberta de sentimentos em textos disponíveis nas redes sociais é, hoje, bastante difundido e pretende enfocar o entendimento de opiniões acerca de um determinado tópico específico. Porém, esta é uma tarefa custosa quando se trata do seu uso em base de dados de alta

dimensionalidade, ou até mesmo em base da *Big data*. O uso de *autoencoders* para redução da dimensionalidade tem sido também difundido para esta tarefa, mas, poucos estudos abordam em fontes da língua portuguesa, e também uma aplicação prática com uma análise critica dos resultados em comparado com dados contextualizados em um problema real.

Esse estudo tem embasamento nos trabalhos publicados nos últimos cinco anos. Nos quais são vistos, um crescimento do uso de *autoencoders* com um processo da tarefa de análise de sentimentos. A partir do crescimento de publicações em rede sociais ocasionado pelo período pandêmico da COVID-19, foi aberta uma oportunidade de um estudo prático da análise de sentimentos, com uso de *autoencoders* em uma base de dados da língua portuguesa e obter uma análise critica com o vasto dado em relação à doença publicada hoje.

O método proposto neste trabalho é importante pois busca explorar e consolidar técnicas de Rede Neural Recorrente (RNN) (*autoenconders*) para redução de dimensionalidade em tarefas de classificação de sentimentos para o idioma português da região do Brasil, sob o contexto da pandemia de COVID-19. Se diferindo de outras pesquisas em sua exploração de resultados através da validação de diversas técnicas da AM consolidadas para a AS e também técnicas recentes, além do uso de comitês. Realizando uma comparação entre a qualidade através do uso da redução e também comparar seus custos computacionais.

A partir dessa pesquisa é esperado obter uma análise do uso de *autoencoders* para redução de dimensionalidade na tarefa da AS com viés de melhoramento do custo computacional, envolvido. Além disso, os resultados obtidos através dos vetores reduzidos serão comparados aos métodos tradicionais de classificação dos sentimentos. A presente pesquisa também visa avaliar como será seu comportamento quando aplicado a um problema real, de nível pandêmico, e comparado com dados deste contexto específico (Casos, Óbitos, Vacinas, etc.).

Ao fim deste trabalho, espera-se que a pesquisa contribua em estudos da área da PLN em textos na língua portuguesa, de fonte da rede social *Twitter*. Uma base de dados será gerada e disponibilizada com textos rotulados com os quatro melhores modelos obtidos (treinados com duas bases de dados diferentes, e para o conteudo normal e o codificado pelo *autoencoder*). Além disso, sera gerado um modelo de *autoencoder* pré treinado nas bases de dados de treino para seu uso em base de dados reais.

1.5 ESTRUTURA DO TRABALHO

Os demais capítulos, desse trabalho estão organizados da seguinte maneira:

- Capítulo 2: apresenta uma noção superficial sobre o campo da inteligência artificial e suas sub-áreas utilizadas como ferramentas neste estudo, sendo eles: aprendizagem de máquina, PLN e AS.

- Capítulo 3: traz uma revisão da literatura acerca de trabalhos realizados que são relacionados a este estudo.
- Capítulo 4: apresenta os materiais e os métodos utilizados para realizar o trabalho.
- Capítulo 5: apresenta os experimentos aplicados, a base de dados construída, os resultados desses experimentos e uma visualização baseada no contexto do estudo aplicado.
- Capítulo 6: é apresentado uma conclusão com analise critica de todos os resultados e resolução das perguntas aqui formulas, e também pontos futuros para melhorar os resultados.

2 ANÁLISE DE SENTIMENTOS

A análise de sentimentos é uma tarefa da PLN que estuda o sentimento expressado em textos com opiniões. Assim como os demais campos do processamento de linguagem natural, ele utiliza ferramentas da aprendizagem de máquina para tratar, aprender e classificar os dados utilizados. Muitas pesquisas no campo de análise de sentimentos crescem com o alto volume de dados textuais não estruturados gerados por meio do uso massivo das redes sociais. Com o fator do período de pandemia, decorrido por conta da COVID-19, a geração de dados não estruturados aumentou fortemente.

Desse modo, devido a estes fatores, muitas pesquisas têm sido realizadas com o interesse de entender como os algoritmos de aprendizado de máquina têm obtido êxito para o contexto de pandemia. As pesquisas mais recentes, publicadas durante o período pandêmico da COVID-19, estão voltadas à validação e compreensão dos sentimentos apresentados por meio das redes sociais (KAUR; KAUL; ZADEH, 2020; CHAKRABORTY et al., 2020; BEHL et al., 2021; KOH; LIEW, 2022). Muitos pesquisadores validam o uso de ferramentas já conhecidas de PLN, também se preocupando em formatar e criar bases de dados para uso da classificação de sentimentos, opiniões, etc.

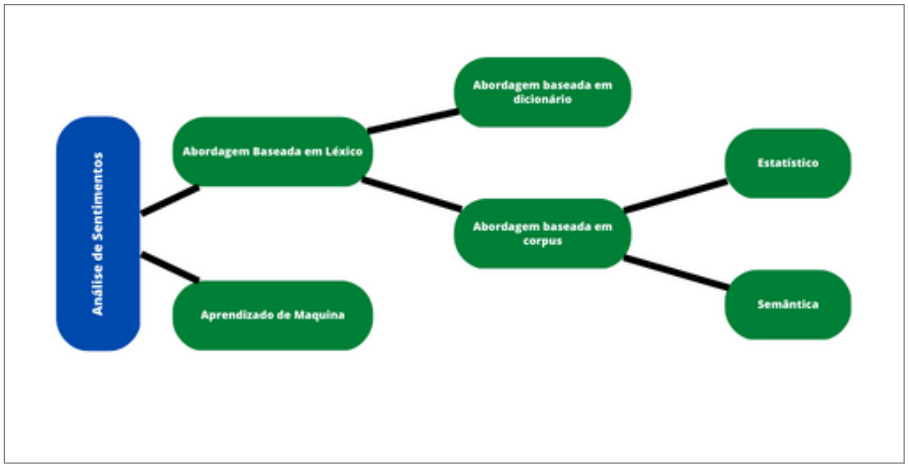
A tarefa de analisar de sentimentos (ou minerar opiniões) é hoje uma das subdivisões da PLN que demonstra um grande crescimento na realização de estudos acadêmicos. Zhang, Wang e Liu (2018) estudaram a AS em três níveis de granularidade que são: nível de documento, nível de frase e nível de aspecto. Esses autores as definem como:

> [...] A classificação de sentimento no nível do documento categoriza um documento opinativo (por exemplo, uma revisão do produto) como expressão de uma opinião geral positiva ou negativa. Ele considera todo o documento como a unidade básica de informação e assume que o documento é conhecido por ser opinativo e conter opiniões sobre uma única entidade (por exemplo, um telefone específico). A classificação de sentimento em nível de sentença categoriza sentenças individuais em um documento. No entanto, cada frase não pode ser considerada opinativa. Tradicionalmente, é comum primeiro classificar uma sentença como opinativa ou não opinativa, o que é chamado de classificação de subjetividade. Em seguida, as sentenças opinativas resultantes são classificadas como expressando opiniões positivas ou negativas. A classificação do sentimento no nível da sentença também pode ser formulada como um problema de classificação de três classes, ou seja, classificar uma sentença como neutra, positiva ou negativa. [...] Sua tarefa é extrair e resumir as opiniões das pessoas expressas sobre entidades e aspectos/características de entidades, que também são chamadas de alvos [1] (ZHANG; WANG; LIU, 2018, p.10, tradução nossa).

[1] [...] Document level sentiment classification categorizes an opinionated document (e.g., a product review) as expressing an overall positive or negative opinion. It considers the whole document as the basic information unit and assumes that the document is known to be opinionated and contain opinions about a single entity (e.g., a particular phone). Sentence level sentiment classification categorizes individual sentences in a document. However, each sentence cannot be assumed to be opinionated. Traditionally, one often first classifies a sentence as opinionated or not opinionated, which is called

A partir desses níveis de granularidade, existem dois meios utilizados para classificar os sentimentos, sendo a abordagem fundamentada por em léxicos e em algoritmos de aprendizagem de máquina. Segundo Bonta e Janardhan (2019, p.2, tradução nossa) "A abordagem baseada em léxico requer léxico predefinido, enquanto a abordagem de aprendizado de máquina classifica automaticamente a revisão que requer dados de treinamento". Na figura 3 é apresentado um diagrama com os tipos de abordagens baseadas em léxico, utilizadas na AS.

Figura 3 – Tipos de abordagem baseada em léxico.



Fonte: Gupta e Agrawal (2020)

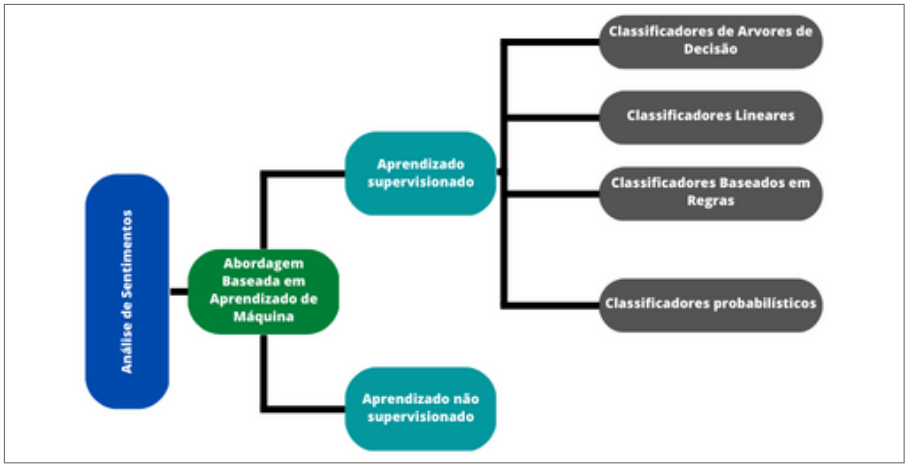
Conforme apresentado na figura 3, os diferentes tipos da abordagem léxica são a abordagem baseada em dicionário e em corpus. Na abordagem baseada em dicionário, é criado um agrupamento de palavras inicialmente baseando-se em seus sinônimos e antônimos. Então, esse dicionario é expandido, até que não seja possível inserir alguma nova palavra. Já na abordagem baseada em corpus, esta se subdivide em semântica e estatística.

No método estatístico, as palavras são determinadas como de uma polaridade de acordo com sua ocorrência nos textos negativos/positivos. Em caso de demonstrarem um comportamento positivo, são determinadas como da polaridade positiva, e na ocorrência dos textos negativos em frases negativas como da polaridade negativa. Por fim, no método semântico, o sentimento é atribuído às palavras com a semântica próxima a esta, podendo ser realizado por utilizar os sinônimos e antônimos em relação a essa palavra (GUPTA; AGRAWAL, 2020).

Por outro lado, a AS com abordagem baseada na Aprendizagem de Máquina (AM), utiliza os algoritmos já muito difundidos no meio, como fator de classificação desses textos. Na figura 4 é apresentado um diagrama, mostrando os tipos de métodos de classificação

subjectivity classification. Then the resulting opinionated sentences are classified as expressing positive or negative opinions. Sentence level sentiment classification can also be formulated as a three-class classification problem, that is, to classify a sentence as neutral, positive or negative. [...] Its task is to extract and summarize people's opinions expressed on entities and aspects/features of entities, which are also called targets.

Figura 4 – Tipos de abordagem baseada em aprendizado de máquina.



Fonte: Gupta e Agrawal (2020)

de sentimentos, na abordagem baseada em algoritmos de AM. Nessa abordagem há dois modos de aprendizado, o supervisionado (que necessita de um "supervisor" para realizar a classificação), e o não supervisionado (que aborda algoritmos de DL).

No aprendizado supervisionado, assim como é apresentado na figura 4, os classificadores mais difundidos na AS são: classificadores lineares, árvores de decisões, baseados em regras e os probabilísticos. Em muitos trabalhos, hoje, também se aplicam os comitês de classificadores, nos quais são utilizados um conjunto de classificadores simultaneamente para fazer uma predição composta. Esse método apresenta uma alta quantidade de trabalhos e aplicações no mercado, pois utiliza um conjunto de dados pré-rotulados (por isso a necessidade de um "supervisor", algo/alguém que rotule os dados) como base de treino para gerar os modelos preditores.

No método não supervisionado não há a necessidade dos dados (corpus) serem rotulado antes do treinamento dos modelos, os algoritmos se encarregam de realizar a predição com base na similaridade de suas características. Nesse aprendizado são utilizados diversos algoritmos da *Deep Learning*, visando obter resultados de acurácia melhores. Esse modo de realizar a classificação de sentimentos, muitas vezes sofre com a alta dimensionalidade atrelada a esta tarefa, tornando assim o custo e tempo computacional para o treino extremamente alto.

Além desses dois métodos, ainda na AS se aplica o uso de algoritmos semi-supervisionados, que apresenta uma parte do conjunto de dados com rótulos, o algoritmo aprende com esse conjunto de dados, classifica novas entradas e as re-treina. Desse modo, estes algoritmos buscam melhorar a acurácia de suas predições.

Assim como outros estudos, o uso dos sentimentos apresentados por usuários em uma rede social, podem estar correlacionados à diversos fatores, como saúde, políticos, sociais, entre outros. Desse modo, a AS se torna uma ferramenta de bastante utilidade, quando há uma abundância de dados gerados. Nesse sentido, o uso da análise de sentimentos neste

trabalho terá um enfoque em como as tarefas de PLN e da AS podem dar resultados específicos em diferentes base de dados. Assim, como também, avaliar um longo período pandêmico e suas implicações em usuários de redes sociais.

2.1 CONCLUSÃO

Diante das discussões empreendidas, é possível compreender, que a AM e suas técnicas produzem ferramentas capazes de desenvolver estudos nos mais diferentes campos. No presente estudo, a avaliação de técnicas de redução de processamento computacional, bem como de entendimento do impacto de uma pandemia sobre usuários de uma rede social. Através dos algoritmos evidenciados neste capitulo, será realizado um fluxo de processamentos sobre o corpus a ser estudado e, por fim, classificar-los nos três sentimentos para a tarefa de AS.

As técnicas apresentadas no presente capítulo serão utilizadas durante a etapa de pré-processamento do corpus, no qual será realizado um conjunto de diferentes agrupamentos delas. Esse método buscará encontrar os melhores resultados para essa etapa inicial do estudo dos dados. Só após essa validação com tais técnicas, que será realizada a tarefa de treinamento do modelo classificador e a classificação do corpus criado para o estudo.

3 REVISÃO DE LITERATURA

O custo computacional envolvido na tarefa de análise de sentimentos está, em grande parte, atrelado à dimensionalidade dos vetores de características produzidas para a classificação/treinamento. Este trabalho buscou utilizar os Codificadores Automáticos como uma etapa do pré-processamento para reduzir essa dimensionalidade e assim diminuir o custo computacional envolvido na etapa de treinamento e classificação dos dados. O presente capítulo visa explorar o que foi pesquisado em artigos acadêmicos nos últimos anos que envolvam a redução de dimensionalidade e análise de sentimentos. Para dessa forma, especificar as contribuições deste trabalho e o seu diferencial em relação ao estado da arte.

Os trabalhos com redução de dimensionalidade utilizando rede de codificadores automáticos são muito comuns em estudos com imagens, nos quais também são utilizados para remoção de ruídos. Nesse campo os AE são amplamente difundidos, apresentando diversas pesquisas, e por isso tem desfrutado de grandes avanços. Atualmente, com a necessidade de diminuir os custos e agilizar o processo de classificação de textos, estão surgindo numerosos modelos que incorporam o AE em seu escopo. Nesse sentido, também serão discutidas as recentes evoluções na área da PLN e da AS nos subcapítulos seguintes, para dessa forma, apresentar todo o domínio que este trabalho tem fundamento.

3.1 PROCESSAMENTO DE LINGUAGEM NATURAL (PLN)

Os últimos anos para a PLN têm sido extremamente benéficos com os avanços atrelados ao crescimento da área da Inteligência Artificial (IA). Os modelos computacionais produzidos no campo da IA são, muitas vezes, utilizados como etapa final de tarefas da PLN. Dessa forma, são utilizados para produzirem predições, agrupamentos e dar a máquina poder de entendimento da linguagem humana. Esse último, na IA, é um subcampo conhecido como Aprendizagem de Máquina (AM), cujo objetivo é encontrar e aprender padrões. Além de extrair e realizar previsões sobre um conjunto de dados (MAIA; SALTON, 2022). Assim, concedendo ao computador, o poder de aprendizado e reconhecimento de dados.

Diante das discussões realizadas, os métodos da IA estão sendo amplamente difundidos em sua subárea PLN. Nela, existem muitos estudos que exploram a linguagem humana, seja a falada, escrita e a escrita à mão. Em estudos recentes, a PLN é usada para investigar diversos campos de estudos como: jurídico, saúde, educação, empresarial, e outros. Foi visto que nos últimos anos houve uma grande concentração de trabalhos acadêmicas em volta do tema de detecção de *fake news* por meio da PLN.

A partir do crescimento das redes sociais e do consumo de notícias digitais, as *fake news*

têm se tornando um problema presente entre os mais diversos tipos de mídias sociais. Em função disso, acadêmicos, organizações e empresas têm buscado soluções que as detectem antes ou pouco depois de serem disseminadas para consumo. As pesquisas sobre *fake news* não são tão recentes, porém os modelos mais antigos, que se baseavam nos aspectos linguísticos oriundos da análise do discurso, estudado em linguística, caíam em vieses. Assim, muitas vezes, tais tendências os faziam classificar notícias verdadeiras como falsas (ZHOU et al., 2019).

O uso de *Deep Learning* (DL) é visto como auxílio na detecção das *fake news*, por meio de modelos baseados em *Long short-term memory* (LSTM) e *Feedforward neural network* (FNN). Segundo o trabalho de Deepak e Chitturi (2020) somente o uso da PLN não supera as complexidades da detecção das notícias falsas. Como resultado, obtiveram uma melhoria significativa nos resultados da classificação, utilizando o LSTM em combinação com a representação *word2vec*.

Por outro lado, Merryton e Augasta (2022) realizaram uma análise empírica, a partir de cinco conjuntos de dados de referência, com modelos clássicos de *Machine Learning* (ML) e diferentes tipos de vetorizadores de textos. Os pesquisadores utilizaram uma combinação de estudos entre as bases de dados textuais, vetorizadores e classificadores. Os resultados do seu trabalho mostraram que o classificador *Passive Aggressive*, em conjunto com o vetorizador TFIDF, foi a melhor combinação para classificar noticias falsas dentre os modelos de estado da arte em AM.

Assim como visto acima, Hamid et al. (2020) realizou a análise baseada em textos relacionados ao COVID-19, buscando detectar propagadores de desinformação. Para este problema, subdividiram-o em duas sub-tarefas de solução: detecção de notícias falsas (i) baseada em texto e (ii) baseada em estrutura. Como solução, utilizaram diferentes métodos, para a primeira subtarefa, foi escolhido o uso de Bag of Word (BoW) e *BERT embedding*.

Já na segunda tarefa de detecção de notícias falsas, por meio de estruturas, utilizaram *Graph Neural Networks* (GNNs). Foram aplicadas diferentes técnicas de classificação em conjuntos com o método escolhido para cada problema. Desse modo, foi observado que com BoW e *BERT*, alcançaram um melhor resultado, quando foi utilizado o comitê de classificadores por voto majoritário. Já em outro domínio, Sousa et al. (2022), em seu trabalho, apresenta um estudo no qual buscou responder se o uso das tecnologias da IA e da PLN poderiam fornecer celeridade ao processo judiciário.

A partir dos atrasos processuais que ocorrem nos órgãos de justiças há décadas, produziram rotinas que automatizavam tarefas relacionadas à triagem de processos em um julgamento. Essas rotinas recebem casos de julgamentos, realizam investigações, observando o último marco do processo, que o define como terminado ou obstruído, para desse modo realizar a triagem. Os dados utilizados foram oriundos do Supremo Tribunal Federal

(STF) brasileiro, obtidos através do sistema Victor[1].

Diante dos resultados obtidos, foi concluído que ainda há necessidade de uma análise maior e com um olhar mais amplo e multidisciplinar para poder concluir se o uso da IA tem um impacto realmente positivo na celeridade dos processos judiciários. Entretanto, foi observado também que, há uma perspectiva de benefícios do seu uso no sistema judicial. Ainda que não seja observado uma celeridade clara, é afirmado que a incorporação dos recursos da IA em rotinas jurídicas tornaram-se necessários.

Além disso, Engelmann et al. (2020) realizou uma mineração de argumentos em processos digitais judiciários. Ele monta uma ontologia onde se busca mediante processos similares, aplicar essa tarefa da PLN e relacionar com a sua teoria para encontrar padrões de argumentos e assim gerar novos processos judiciais, bem como executar processos já existentes. Assim, é realizado como um meio de tomada de decisão e não de resolução final. Essa ontologia foi concebida através do software *Protége*, que tem o objetivo de organizar conhecimentos (ou seja, formular ontologias). Já a mineração de argumento utilizada por eles, foi aplicada para reconhecer os padrões, a partir dos processos semelhantes e aferir resultados considerando a ontologia.

Por fim, os autores buscavam por meio de outra tarefa da PLN, os *chatbots*, apresentar um sistema capaz de comunicar-se de forma eficiente, detalhando como chegou a uma conclusão específica de um processo judiciário (ENGELMANN et al., 2020). Assim, os autores concluem que, o sistema idealizado e em produção por eles, é um meio de auxílio ao profissional da área de direito, e não uma ferramenta de substituição. Desse modo, é esperado que a partir dele e de suas conclusões, sejam tomadas decisões mais rápidas, sem negligenciar a incumbência do profissional, apresentando um sistema com interface fácil e com poder de comunicação em linguagem natural.

Foi visto que no judiciário, o seu foco está em criar sistemas com o uso da PLN para encontrar padrões e gerar classificações. Em muitos casos, também conseguirem explicar de uma forma que os humanos entendam, visando tornar um ambiente de fácil entendimento para o usuário final. Já no campo financeiro, a PLN é, também, uma ferramenta de auxílio na tomada de decisão sobre previsão de mercado de ações, otimização de portfólio e estratégias de execução comercial nos trabalhos mais recentes.

Dentre os algoritmos da PLN, o que difere entre as soluções buscadas, é o método de pré-processar, processar e analisar os dados. Na literatura é visto que a principal tarefa da PLN utilizada nos estudos do seu uso no campo financeiro, é a AS (DAS et al., 2022; KHALIL; PIPA, 2022; PATEL et al., 2021; QIAN et al., 2022; SHARMA et al.,). Pois, muitos trabalhos processam em conjunto com algoritmos de *machine learning* e *deep learning* para otimizar a acurácia dos resultados obtidos. O *Long short-term memory* (LSTM) é um tipo de Rede Neural Recorrente (RNN), um algoritmo com grande incidência entre os

[1] O sistema Victor é um projeto fruto da pareceria do Supremo Tribunal Federal (STF) com a Universidade de Brasília (UnB), tendo seu início no ano de 2017 durante a gestão da ministra Cármen Lúcia.

trabalhos lidos. Nas pesquisas com o LSTM foi de fato observado otimização nos resultados obtidos e dados de fontes de mídias sociais, notícias financeiras e redes sociais.

No campo da saúde, a utilização de tarefas da PLN é amplamente utilizada em prontuários eletrônicos, que fornecem dados sobre diagnósticos, sintomas, doenças, entre outros. No trabalho de Roehrs et al. (2018) foi desenvolvido um modelo chamado *OmniPHR* (Onde o *"Omni"* vem do inglês *"omnipresent"*) que visava criar um padrão de interoperabilidade entre os prontuários pessoais. Desse modo, foi alcançado um sucesso de 87,9% de *F1-score*, demonstrando uma promissora possibilidade de criar regras de inferência sobre problemas de saúde ou prevenção de problemas futuros.

Além do uso sobre prontuários médicos e extração de seus padrões (AKEN; HERRMANN; LÖSER, 2021), a PLN também é empregada para outros fins na saúde. O uso de *chatbots*, uma de suas tarefa tem sido aplicado em conjunto com muitas formas de interação com paciente, como a telemedicina (houve uma grande utilização durante o período pandêmico) (BHARTI et al., 2020; KHADIJA; ZAHRA; NACEUR, 2021; SOUFYANE; ABDELHAKIM; AHMED, 2021; VASILEIOU; MAGLOGIANNIS et al., 2022). Os prognósticos que também estão presentes nos prontuários, são muito pesquisados na busca de informações que entreguem poder de tomada de decisão. Muitas pesquisas por tarefas da PLN e modelos de ML buscam extrair decisões confiáveis a serem tomadas com relação ao prognóstico obtido (BAR et al., 2021; DESHMUKH; PHALNIKAR, 2021; FRONDELIUS et al., 2022).

Ainda nesse contexto, o uso da AS é observado em estudos recentes relacionados à saúde mental. O poder da tarefa de predizer o sentimento em estudos sobre saúde mental é bastante evidenciado por atribuir uma polaridade ao texto classificado (OYEBODE; ORJI, 2020b; OYEBODE; ORJI, 2020a). Aliado à essa tarefa, também é visto o uso de *chatbots* para criar uma interação entre a máquina e o usuário e assim tentar prever a polaridade da conversa, assim como é visto no trabalho de Abedin et al. (2021).

No campo da saúde, a PLN tem se tornado cada vez mais presente em trabalhos e pesquisas acadêmicas nos anos mais recentes.A partir do surgimento da COVID-19 e a pandemia, foi gerada uma abundância de dados relacionados à saúde, como: sintomas, diagnósticos, resultados de testes, entre outros. Esse conjunto de dados disponibilizados, em sua maioria, de forma digital, devido às medidas e recomendações empregadas na pandemia.

A AS é uma subdivisão da PLN que tem apresentado uma ampla quantidade de trabalhos acadêmicos em virtude da pandemia que ocorreu nos últimos anos de 2020 e 2021. Desse modo, há um crescimento na criação de dados textuais em redes sociais de tipo micro blog, devido às políticas/medidas aplicadas e seguidas.

O presente trabalho utiliza a tarefa da AS como meio de classificar textos por meio de técnicas de aprendizado profundo que visam reduzir o alto custo computacional envolvido na abordagem. Desse modo, explorando uma gama de técnicas da PLN comparando seus resultados para encontrar uma maior qualidade com a combinação dessa metodologia

apresentada. O subcapítulo seguinte trata sobre o estado da arte de trabalhos que enfocam o uso dessa tarefa da PLN.

3.2 ANÁLISE DE SENTIMENTOS (AS)

Diversos trabalhos, durante os anos de 2020 e 2021, definiram um enfoque na situação que o mundo vivia, do surto da doença COVID-19, das medidas tomadas para tentar contê-la, da situação política e social que cada país se deparou em razão dela. Sendo assim, foi explorado no meio acadêmico as metodologias mais importantes para os últimos cinco anos de estudo da área.

É bastante evidenciado o uso do twitter como fonte para criação de corpus (EDARA et al., 2019; SHI et al., 2020), por ser uma rede-social de tipo microblog, em que seus usuários expõem suas opiniões em textos curtos. Além desses fatores, ainda é apresentada a oportunidade de captura de dados em tempo real, por meio de suas *api's*, se tornando muito populares entre pesquisas que utilizam esta tarefa. Desse modo, é possível atacar diferentes problemas propostos, como a detecção de *bots* (Atores automatizados com tarefa predefinida) (SHI et al., 2020), análises clínicas e mentais (Como identificação de polarização dos sentimentos e estresse)(EDARA et al., 2019; ELAYAN et al., 2020; LEVIS et al., 2021), estudo de opiniões públicas (NA et al., 2021; NASEEM et al., 2021) que tem sido muito mais explorado nos últimos dois anos, a partir da evolução da pandemia de COVID-19.

Para alcançar tais análises, foram utilizadas diversos modelos com diferenças entre suas arquiteturas e métodos. Muitos trabalhos utilizaram bibliotecas disponíveis no *python* para realizar o treinamento ou a classificação dos sentimentos (POLIGNANO et al., 2019; BELLODI; BERTAGNON; GAVANELLI, 2022; MOHAN et al., 2022). Além disso, com o intutito de obter níveis de acurácia maiores, muitos trabalhos exploraram o uso de *Deep Learning* (CAMBRIA et al., 2020), para auxiliar nas etapas de processamento, pré-processamento ou treinamentos dos dados.

Entre as técnicas de DL utilizadas em conjunto com a AS, o LSTM tem demonstrado grande eficácia nos trabalhos que o pesquisaram (EDARA et al., 2019; CAMBRIA et al., 2020; NASEEM et al., 2021; BELLODI; BERTAGNON; GAVANELLI, 2022). Como Edara et al. (2019) apresenta, o LSTM é uma Rede Neural Recorrente, com etapa de longo e curto prazo, no qual aprende as informações recebidas inicialmente as retém, possibilitando recuperá-las, dando poder de predição mais eficaz que as RNN comuns.

Porém, as tarefas da AS já sofriam de grande custo operacionais, a partir da implementação da DL como etapa, esse custo aumentou em alguns casos. Muitas técnicas são vistas para tentar reduzi-los, desde utilizar máquinas com mais capacidade de processamento, até aumentar a atenção na etapa de pré-processamento dos textos. A redução de dimensionalidade e dados não estruturados demonstrou ser útil para reduzir esses custos. No subcapítulo seguinte serão exploradas mais questões a respeito do estado da arte da

redução de dimensionalidade para tarefas de análise de sentimentos e qual a sua eficácia observada entre os trabalhos.

3.3 REDUÇÃO DE DIMENSIONALIDADE

Na Análise de Sentimentos existe a maldição da dimensionalidade (*curse of dimensionality*), a qual é muito citada em pesquisas que utilizam esta tarefa da PLN (VEIGA, 2016; PINHEIRO, 2017; PONTES, 2021). Isso ocorre, devido ao tipo de dados utilizados, que geralmente, são textos não estruturados. O uso do texto como dados de teste e treino requer a tradução dele para algo que os modelos entendam, e os métodos mais comuns são a conversão deles em um vetor de *tokens*. *Tokens* são basicamente palavras, caracteres ou sub-palavras, e quando se observa um corpus muito grande, esse vetor tende a altas dimensionalidades. Treinar e testar modelos com um conjunto de dados de alta dimensionalidade resulta em um grande custo computacional e também muitos ruídos em seus resultados.

Este fenômeno da dimensionalidade ocorre quando o tamanho do espaço de recursos (no vetor) se torna cada vez maior. Assim, o seu volume aumenta de forma rápida a medida que os dados de treinamento disponíveis (do *word embbending*) se tornam esparsos, e isso acaba incorrendo sobre uma piora na qualidade dos algoritmos de aprendizagem da analise de sentimentos (RONG et al., 2014, p.841). Omuya, Okeyo e Kimwele (2023) também demonstra que esse fenômeno (da maldição da dimensionalidade) afeta diretamente no desempenho (e precisão) dos modelos de analise de sentimentos. Desse modo, na pesquisa destes autores é desenvolvido um modelo no qual combina a redução da dimensionalidade com a PLN para realizar a analise de sentimentos.

A redução dessa dimensionalidade demonstra necessária e o modo como é realizada, difere entre os trabalhos. Assim como é visto no trabalho de Kim (2018), a redução é aplicada como uma etapa do pré-processamento, na qual é utilizada, uma matriz de frequência dos termos, em seguida, aplicam um método semi-supervisionado para calcular os pesos desta frequência e então extraem a característica. Outros trabalhos mais recentes também utilizaram a extração de característica como uma etapa do pré-processamento para reduzir a dimensionalidade antes do treinamento e classificação dos sentimentos (DARWIS et al., 2019; OMUYA; OKEYO; KIMWELE, 2022).

Diversos métodos de extração de características têm sido implementados nos trabalhos recentes. Por exemplo, Huai e Voorde (2022) realizam a redução de dimensionalidade através do *t-distributed stochastic neighbor embedding* (t-SNE), uma técnica que utiliza a extração de características. É argumentado que o t-SNE "pode efetivamente lidar com dados não lineares e preservar as estruturas locais e globais dos dados originais"[2] (HUAI; VOORDE, 2022, p.5, tradução nossa). O uso da t-SNE também é bastante difundido como

[2] "can effectively handle nonlinear data and preserve the local and global structures of the original data"(HUAI; VOORDE, 2022, p.5)

redutor de dimensionalidade para tarefas da PLN (HAJIBABAEE et al., 2021; LI et al., 2022), e através dele foi possível alcançar melhorias consideráveis em trabalhos recentes.

Outras técnicas que também são muito utilizadas, visando diminuir os custos computacionais das tarefas de AS, através da redução da dimensionalidade do corpus são: *Principal component analysis* (PCA) (ZHANG et al., 2019; UMER et al., 2020), *Latent Semantic Analysis* (LSA) (SOUSA et al., 2018; STEVENSON; MUES; BRAVO, 2021), *Linear discriminant analysis* (LDA) (LEE et al., 2019), *Non-negative matrix factorization* (NMF) (TRUICĂ et al., 2021) e o *uniform manifold approximation and projection* (UMAP) (GENCOGLU, 2020).

Os métodos de extração de características são os mais vistos em trabalhos que utilizam a AS, como fator de redução de custos de treino e testes desta tarefa. Entretanto, outras técnicas da DL surgiram e estão sendo exploradas para executar essa tarefa de redução, por meio das técnicas semi-supervisionada e não supervisionada.

Entre as técnicas semi-supervisionadas que estão demonstrando crescimento em seu uso, os Codificadores Automáticos (CA) (do ingles *autoencoders*) têm ganhado interesse entre os pesquisadores. É muito comum serem vistos como parte do pré-processamento ou etapa principal para determinado tipo de dados, como a imagem e o áudio. Os CA tem diferentes objetivos quando utilizados, porém, também tem o proposito de reduzir a dimensionalidade desses dados. Sendo assim, surge a oportunidade de utilizá-los também para a tarefa da AS. Alguns trabalhos demonstram utilizá-los em conjunto com técnicas frequentes para ampliar os resultados desta redução (PALANI; RAJAGOPAL; PANCHOLI, 2021).

É observado por Tissier, Gravier e Habrard (2019), que o vetor binário codificado, do conjunto de *word embedding*, através do uso dos *autoencoders* exibiram quase os mesmos desempenhos que os vetores reais. Além disso, também foi relatado ser possível realizar as consultas top-K, 30 vezes mais rápido que o habitual, mostrando benefícios do uso da técnica. Ameur, Jamoussi e Hamadou (2018) também observam em seu trabalho que, o uso dos *autoencoders* com *Pointwisemutual-information* (PMI) "captura fielmente as características semânticas e sentimentais das palavras"[3] (AMEUR; JAMOUSSI; HAMADOU, 2018, p.1317, tradução nossa). Em outras palavras, é observado a relevância que o uso dos *autoencoders* têm na tarefa da AS.

Esta dissertação utilizará as redes de codificadores automáticos (do inglês *autoencoders*) como uma etapa de pré-processamento, sobre um corpus no idioma português, para avaliar o impacto na qualidade dos resultados equiparando-o com modelos mais recentes (e comitês classificadores) na analise de sentimentos e também comparar os custos computacionais entre as metodologias analisada. Os trabalhos anteriores são predominantemente em corpus da língua inglesa, nos quais o *autoencoder* faz parte do modelo de

[3] "faithfully captures the words semantic and sentimental characteristic"(AMEUR; JAMOUSSI; HAMADOU, 2018, p.1317)

treino, e não da etapa de pré-processamento. Além disso, o *autoencoder* utilizado aqui, tem seu estado salvo por ferramentas da linguagem *python* para poder evitar o custo de treiná-lo em todas as interações. Ainda em sua camada comprimida, na qual há a redução da dimensionalidade, é salva no conjunto de dados para ser realizada a classificação dos sentimentos.

3.3.1 Conclusão

Conforme apresentado neste capítulo, devido ao estado pandêmico dos anos de 2020 e 2021, muitos trabalhos foram realizados utilizando ferramentas de AS. Alguns desses, focando somente em discutir questões de saúde e/ou opiniões a respeito das políticas utilizadas em todo o mundo. Essas pesquisas focaram no uso de corpus na língua inglesa para realizar toda a tarefa. Alguns estudos utilizaram modelos multilinguísticos visando produzir a classificação de sentimentos para diversas regionalidades.

Porém, poucos tiveram a atenção para custos computacionais envolvidos nessa tarefa. Pois, seu único objetivo estava sobre os resultados, para realizarem as discussões propostas. Além disso, em alguns desses trabalhos, é observada a utilização de técnicas de pré-processamento de estado da arte, não visando uma experimentação completa de um agrupamento desses métodos para avaliar seus resultados.

Além disso, também foi discutido neste capítulo, neste capitulo, um dos métodos de redução do custo computacional da tarefa de classificação de sentimentos, que consiste na diminuição do uso da dimensionalidade. Existem muitos métodos para alcançar esse objetivo e muitos algoritmos de análise de textos já apresentam em seu escopo a tarefa de redução. Porém, este trabalho se propõe a utilizar a técnica de rede de codificadores automáticos, para treinar, reduzir e criar um modelo pré-treinado na etapa de pré-processamento. Assim, não responsabilizando os modelos a fazerem isso, e também utilizando as melhores técnicas de *word embbending*, caso não seja a *word2vec*.

Então, os trabalhos aqui apresentados não tiveram ênfase no estudo da qualidade do pré-processamento, do custo computacional e máquinas de porte fraco e também de corpus na língua portuguesa. Assim, o presente estudo busca por meio dessas tarefas, realizar uma avaliação de diversos agrupamentos de técnicas da PLN, assim como a aplicação de redes de codificadores automáticos para reduzir a dimensionalidade dos vetores. Além de avaliar os resultados baseados nos novos vetores e assim escolher o melhor conjunto para realizar a classificação de um corpus na língua portuguesa, oriundo da rede-social *twitter*.

Por fim, a presente pesquisa irá contribuir com um modelo pré-treinado sobre um cenário generalista e contextualizado sobre uma pandemia real. Este trabalho também produzirá um corpus rotulado com mais de 700 mil textos em português, por meio de modelos da AS. Além de uma avaliação completa das técnicas, classificador e comitês classificadores de estado da arte da AM. Assim, enriquecendo o campo de estudo de inteligência computacional focado em PLN.

4 MATERIAIS E MÉTODOS

4.1 CARACTERIZAÇÃO DO DOMÍNIO

A fase de coleta de dados é o ponto inicial de todo trabalho em análise de dados. Essa etapa visa conseguir recursos necessários para realizar estudos correlacionados à área de pesquisa. Muitas fontes de dados são utilizadas para tais estudos, dentre elas as redes sociais e mini *blogs* tem grande demanda na composição de base de dados. Essas redes sociais têm se atualizado e se adequado para seu uso em pesquisas de dados.

Dentre essas plataformas sociais, o *twitter* é uma rede social e mini-blog com grande popularidade em todo o mundo. No estudo de análise de sentimentos essa rede social tem sido amplamente estudada por acadêmicos e empresas. Os seus usos têm os mais diversos objetivos nessas áreas, porém todas consideram que o sentimento expresso pelos usuários do *Twitter* podem refletir informação para tomada de decisão. Para realizar a coleta dos dados foi utilizado a API disponibilizada pela rede social e ferramentas desenvolvidas e publicadas em *python*.

O pré-processamento dos dados é uma tarefa de bastante importância no PLN, pois esta etapa consegue definir quão otimizados os dados estarão para serem treinados pelo modelo e alcançar melhores acurácias. É uma etapa após a coleta dos dados, e transforma esse conjunto em algo viável para que a maquina possa entender.

No PLN os dados de treino são em muitas vezes textuais não estruturado, para a máquina essa categoria de dado não tem nenhum sentido, e para poder utiliza-lo torna-se necessário transformar esses dados em algo que seja possível de ser entendido. Para isso, muitas tarefas de pré-processamento de dados englobam algoritmos matemáticos que têm a capacidade de pegar esses dados não estruturados e convertê-los em dados entendíveis pela máquina.

4.2 COLETA DE DADOS

O *tweepy*, uma ferramenta disponível em *python*, é uma biblioteca que vincula a API do *twitter* com a facilidade da linguagem de programação para pegar os dados textuais e não textuais da rede social. Através dessa biblioteca foi coletado o conteúdo publicado pelos usuários desde maio de 2020. Para isso, foi necessário obter através da plataforma de desenvolvimento do *twitter* uma licença de desenvolvimento e acesso aos recursos da rede social.

A licença obtida para esse estudo foi a acadêmica, que possui a obrigatoriedade de não ter viés de utilização para o meio empresarial. Desse modo, possibilitou o acesso à plataforma de desenvolvimento que dá o poder de obter elementos da rede social como *tweet's*, *direct's* (Mensagens diretas), listas, *spaces*, usuários, etc. Inicialmente, o período

de coleta estipulado foi durante o horário considerado de maior uso dos usuários, entre as 12 horas e as 18 horas, durante todo os anos de 2020 e 2021 (inicio da pandemia).

O *corpus* reunido para esse ano, foi coletado através da ferramenta e obteve mais de 300 mil *tweets*, através das *hashtag's* utilizadas durante o periodo pandêmico. Essa base foi salva em MySql, sendo um sistema de gerenciamento de banco de dados em Structured Query Language (SQL) para dados relacionais, e foi escolhida por ser vista como uma ferramenta de fácil uso e também de exportação gerando um arquivo no formato Comma Separated Values (CSV). Nessa coleção foram salvos a data e hora, id do *tweet*, texto publicado e usuário que publicou.

Os dados de horário e data eram necessários para poder observar as mudanças de sentimentos dos usuários do *twitter* durante o período de estudo. O mesmo método foi utilizado na coleta de dados para o ano de 2021, no qual a quantidade de dados coletados foi mais de 1 milhão de textos. Isso ocorreu, porque em 2021 a doença já era um assunto bastante conhecido e comentado em todas as mídias sociais. A tabela 1 mostra a quantidade total de conteúdos coletadas da rede social utilizando a ferramenta *tweepy* do *python*.

Tabela 1 – Quantidade de conteúdo do *Twitter* sobre COVID19 coletado por Ano.

Ano	**N° *tweet's***
2020	1496687
2021	1628815

Fonte: Elaboração própria.

Apesar dessa quantidade de dados coletados com o *tweepy*, foi possível complementar através de outra ferramenta disponibilizada pelo *python*. O snscraper é um *scraper* (coletor) para Social Networking Services (SNS) que possibilita a coleta dos dados de redes sociais de maneira mais facilitada e sem limitações de data. A partir dessa biblioteca foi possível acrescentar a quantidade de dados para o ano de 2020. Além disso, também possibilitou a captura de dados do início do ano relativo ao assunto COVID-19.

Foi necessário passar para essa ferramenta um conjunto de palavras-chave e *hashtags*, vistos na tabela 2, correlacionados ao assunto de busca, e também definir por parâmetro a região de localização da busca no *twitter* através do atributo *near*(Proximo) fornecido por sua API. Desse modo, a base de dados de *tweets* do Brasil foi bastante incrementada para cada ano.

A primeira ferramenta, o *tweepy*, foi necessário executá-la durante 6 horas a cada dia, nos anos de 2020 e 2021. Isso ocorre, pois, a biblioteca (do *python*) foi utilizada capturando os *tweets* em tempo real, e produzindo assim uma base de dados salva no gerenciador de dados MySQL. Por essa ferramenta funcionar somente com os recursos disponibilizados pela API do *twitter*, que só permite recuperar dados retroativos de até uma semana da data, foi necessário realizar essa rotina de coleta. Desse modo, foi produzida uma base

Tabela 2 – Palavras-chave (*hashtags*) utilizadas para coleta de dados do *corpus* Brasil-Covid-Pandemia.

#ficaemcasa #covid-19 #combateaocorona #fiqueemcasa #quarentena #isolamentosocial #fiqueemcasacovid19 #quarentena #CovidVacina #pandemia #coronavirus #covid19brasil #coronavirusnobrasil #vacinabr #coronavirusbrasil #escolasfechadasvidaspreservadas #viruscorona #virus #viruscovid #VacinaSim #VacinaJa coronavirus covid covid19

Fonte: Elaboração própria.

de dados com mais de 300 mil *tweets* e esses dados foram salvos em CSV para serem rotulados pelo classificador escolhido com base em seus resultados.

Embora os resultados desta coleta foi bem-sucedido, ainda assim havia necessidade de coletar dados anteriores ao início da coleta, que foi em maio de 2020. O *tweepy* não fornecia suporte para coleta de dados retroativos de longas datas, então, foi necessária a utilização do *snscrape*, que fornecia recursos para esta coleta. Com essa ferramenta foi possível coletar dados do início do ano de 2020, estipulando uma quantidade máxima de *tweet's* por dia.

Assim, a mesma foi utilizada para complementar a base de dados final de estudo dos sentimentos dos usuários da rede social. A coleta com o *snscrape* começou no início de fevereiro de 2022, sendo estipulado uma quantidade máxima de cinco mil *tweet's* por dia, para que não se prolongasse muito. Desse modo, a tabela 3 apresenta as quantidade de *tweet's* para cada ano no *corpus* final.

Tabela 3 – Quantidade de conteúdo do *Twitter* sobre COVID19 coletado por Ano com as ferramentas *Tweepy* e *SNScrape*.

Ano	**N° *tweet's***
2020	510487
2021	258786

Fonte: Elaboração própria.

Também foi observado que, existia a possibilidade de filtrar em uma região menor que a do país Brasil e foram criados um conjunto de textos para regiões Brasileiras que aplicaram de forma mais rígida medidas não farmacológicas com intuito de prevenir a doença (como *Lockdown*). Assim, foram coletados *tweets* das cidades e estados brasileiros: Fortaleza (CE), Niterói (RJ) e Curitiba (PR). Essas cidades foram escolhidas através do Índice de Permanência Domiciliar (IPD), escolhendo as que mostraram ter maior rigor em períodos específicos. Isso foi feito, para buscar entender como os usuários reagiram à tais medidas durante a pandemia. A quantidade de tweets coletados para esse estudo é apresentada na tabela 4.

Conforme é visto na tabela 4, das regiões com medidas mais rígidas, foi coletado um total de 10440 *tweets* relacionados à medida *lockdown*. O período no qual houve maior

Tabela 4 – Quantidade de *tweet's* coletado por cidade e período de medida rígida através do IPD.

Cidade (Estado)	**N° *tweet's***	**Periodo**
Fortaleza (CE)	4892	3 de maio a 6 de junho de 2020
Niterói (RJ)	3156	6 a 20 de maio de 2020
Curitiba (PR)	2392	8 a 26 de maio de 2020

Fonte: Elaboração própria.

ocorrência da aplicação, como visto na tabela, foi durante o mês de maio de 2020. Esse *corpus* agrega ao estudo da avaliação dos sentimentos dos usuários de uma rede social durante um período pandêmico. Além disso, possibilita a investigação de suas reações a políticas públicas e medidas utilizadas durante o período.

Essa ferramenta também permite descartar da coleta os *retweet's*, que são o compartilhamento de um *tweet* por outros usuários. Isso evita que haja textos repetidos no *corpus* de estudo e diminui uma tarefa na etapa de pré processamento da base de dados criada. Além disso, é possível determinar a quantidade de *tweets* que serão incorporados no arquivo de saída e o período de coleta. Os textos que integram o *corpus* são coletados por palavras-chave e *hashtags* assim como na ferramenta *tweepy*. Essas ações são necessárias para poder definir o tema em que seja desejado que os textos estejam. Assim, cada ferramenta necessitou de um determinado tempo para produzir resultados das coletas.

O algoritmo de coleta das postagens foi executado por dois dias, os textos e informações foram salvos no formato csv com os seguintes atributos: *datetime*, *tweet id*, *text* e *username*. É importante destacar que o atributo *tweet id* é necessário para caso haja a necessidade de pegar novamente seus dados. Além disso, foi definida na *query* de busca o atributo *near* como *Brazil*, para garantir que os *tweet's* tivessem localizados no país e também o parâmetro idioma como pt-br (fornecido pela própria plataforma). Dessa forma, garante que os sentimentos apresentados e estudados fossem oriundos de usuários localizados no Brasil. As palavras-chave e *hashtag's* usadas na coleta de dados foram escolhidas por serem tendências de uso em postagens durante a época da pandemia e podem ser observadas na tabela 2.

Assim, com a base de estudo formada, era necessário um *corpus* rotulado, para realizar o treino e o teste dos modelos que classificariam os sentimentos dos *tweet's* coletados.Foram encontrados dois *corpus* disponíveis para estudo,, o primeiro sendo disponibilizado na plataforma *Kaggle*[1] e o segundo procedente de um estudo realizado durante o período inicial da pandemia de 2020. O *corpus* do *Kaggle* continha uma quantidade de 800 mil postagens oriundas do *Twitter* rotuladas em positivo, negativo e neutro.

Essas postagens foram coletadas no período de 1 de agosto de 2018 a 20 de outubro de 2018. O conjunto de dados foi dividido entre *tweet's* com tema, sem tema, neutros com tema e neutros sem tema. Desse modo, foram criados subconjuntos de treino e teste de

[1] Disponível em: https://www.kaggle.com/datasets/augustop/portuguese-tweets-for-sentiment-analysis

modelos de classificação. A tabela 5 mostra como foi subdividido o conjunto de dados inicial em um subconjunto de treino.

Tabela 5 – Divisão de subconjuntos de treino.

Tema	**N° de amostras (mil)**	**Classes**
tweet's sem tema	50, 100, 200, 300, 400, 500	Negativo, Positivo
tweet's com tema político	50	Negativo, Positivo
tweet's sem tema	100	Negativo, Positivo, Neutro

Fonte: Elaboração própria.

Assim, através desses subconjuntos foi possível realizar o treino e teste dos modelos usados neste trabalho. A utilização desse *corpus* ocorreu, inicialmente, por ser o melhor encontrado entre os que estavam disponíveis, por sua organização, subdivisão de conjuntos e também pela fonte de seus dados. A utilização de textos oriundos do *Twitter* para o treino demonstrou de melhor eficácia, devido ao dialeto utilizado. Como a base de dados também tem esse mesmo formato de escrita, então é possível que os modelos classifiquem com melhor efetividade a partir desta fonte de dados de treino.

O segundo *corpus* usado para treino dos modelos também utilizou textos procedentes do twitter para compor a base de dados. Imran, Qazi e Ofli (2022b) construíram um conjunto de dados de larga escala, compreendendo 2 bilhões de *tweet's* em múltiplas linguagens, oriundos de postagens de 218 países, por 87 milhões de usuários, em 67 idiomas. O conjunto de dados nomeado de TBCOV[2] A tabela 6 mostra os números estatísticos gerais que o *corpus* apresenta.

Tabela 6 – Estatísticas gerais do *corpus* TBCOV.

Tipo	**Quantidade**
Total de *tweet's*	2,014,792,896
Usuários únicos	87,771,834
Países abrangidos	218
Cidades cobertas	24,424
Idiomas cobertos	67

Fonte: adaptado de Imran, Qazi e Ofli (2022b)

Além disso, é considerado que o TBCOV possui várias perspectivas diferentes e opiniões sobre posicionamentos e decisões políticas com relação aos efeitos da pandemia da COVID-19, ocorrida durante a época da coleta dos *tweet's*. Nesse sentido, isso ocorre pelo amplo período de coleta e também por conter uma quantidade de palavras-chave abrangentes, tornando o tema mais vasto.

[2] two billion multilingual COVID-19 tweets with sentiment, entity, geo, and gender labels é proposto como um *corpus* de larga escala, com maior alcance geográfico e linguístico, coletado em um período de 14 meses, com seus sentimentos rotulados através de algoritmos multilinguísticos.

Certamente, a utilização desse *corpus* para o treinamento de modelos classificadores obteriam classificações mais próximas do tema, devido a que foi inserida no tema em que o presente trabalho se dispõe a estudar. Ao visar o treino de modelos, foi necessário realizar a coleta através dos *tweet id* disponibilizados no conjunto de dados, apenas dos textos rotulados como da linguagem português. Para isso foi novamente utilizada a ferramenta do *python tweepy*, que disponibiliza recurso de coleta através do atributo *tweet id*.

Apesar do *corpus* ser separado como postagens oriundas do Brasil através do atributo *near* disponibilizado pelo *Twitter* através de sua *api*, foi identificado que, entre os conteúdos também haviam textos em outras linguagens. Por esse motivo, foi necessário realizar a exclusão de textos em outras linguagens do conjunto de dados, para trabalhar somente com textos em português. A tabela 7 apresenta as estatísticas finais do conjunto de dados com o filtro de textos em português.

Tabela 7 – *corpus* TBCOV com textos em Português.

Tipo	**Quantidade**
Total de *tweet's*	32,427,446
Usuários únicos	2,621,036
Sentimento Positivo	2,507,038
Sentimento Neutro	11,803,816
Sentimento Negativo	18,116,592

Fonte: Elaboração própria.

Na tabela 7 é apresentado um desbalanceamento entre os sentimentos, no qual pode afetar o aprendizado dos modelos de analise de sentimentos. Para evitar que esse problema impacte os resultados dos classificadores treinados nesse trabalho foi utilizado a abordagem de re-amostragem *under-sampling*. Os dados das classes são reduzidas de forma aleatória ao tamanho da classe de menor quantidade de amostras. Assim, espera-se que se reduza o impacto do desbalanceamento sobre os algoritmos e técnicas utilizadas na dissertação.

Além disso, esse *corpus* teve os sentimentos classificados com base em técnicas de AM e PLN, assim, foram calculados o coeficiente para cada texto e rotulados com base em seu maior resultado. Para isso, foi usado o modelo multilinguagem, baseado no transformador XLM-T por ser observado que consegue alcançar os rótulos com um melhor desempenho. Desse modo, os rótulos são formulados com o seu coeficiente resultante e inseridos no conjunto de dados, para aferir a precisão da classificação.

Assim, é possível utilizar com certa acurácia a classificação dos objetos estudados. Para possibilitar a avaliação dos resultados dos modelos usando os Codificadores Automáticos e sem utilizá-los, comparando os seus resultados. Na próxima seção será realizada uma abordagem de como foi feito o pré-processamento desses dados, para possibilitar o cumprimento das etapas de treinamento dos modelos.

4.3 PRÉ PROCESSAMENTO

A etapa de pré-processamento é essencial para ser possível realizar o treinamento dos modelos classificadores em textos. Pois, sem essa etapa, a quantidade de ruídos nos classificadores pode atenuar os resultados ruins. Outro ponto, é a necessidade de tornar os textos no conjunto de dados em elementos que a máquina e os algoritmos possam entender, pois, o texto (a linguagem humana) não é compreensível para os computadores.

Para isso, alguns métodos disponíveis no PLN são utilizados nos conjuntos de dados de treino dos modelos. Entretanto, existem etapas necessárias a serem feitas antes de transformar o conteúdo textual em algo manuseável pela máquina. Desse modo, foram utilizadas técnicas de pré processamento na linguagem *python* com as bibliotecas disponibilizadas por desenvolvedores que serão descritas na próxima seção.

4.3.1 Formalização de textos informais

Os textos provenientes da rede social *Twitter* apresentam uma espécie de dialeto próprio. Essa linguagem é entendida entre os pesquisadores como o *internetês* ou no inglês *netspeak*. Por exemplo, muitas palavras são abreviadas ou modificadas, são utilizados símbolos entre as conversas e postagens nas redes sociais, questões que geram ruído no modelo a ser usado. Para poder melhorar os resultados com esse conteúdo, foram desenvolvidos alguns métodos, visando tratar essas palavras e gírias.

Desse modo, Nascimento et al. (2021) produziu um dicionário léxico de termos em português, que traduz a palavra do modo informal para a norma padrão. Essa ferramenta foi baseada em um trabalho já realizado em inglês, utilizado como uma fase do pré processamento, assim construíram um semelhante para a língua portuguesa. Esse dicionario léxico foi disponibilizado através de repositórios e foi utilizado neste trabalho, para evitar a perda de dados ou torná-los ruídos.

4.3.2 Padronização

Todas as palavras presentes no *corpus* passaram por um processo que as deixou em letras minúsculas, para não diferirem quando forem repetidas. Além disso, foram removidas acentuações, pontuações e emojis. Após essa padronização, alguns recursos disponíveis nos conteúdos do *Twitter*, que não são a opinião expressas dos usuários, foram removidas, como: links de websites, citações de outros usuários e *hashtags*.

4.3.3 Tokenização

A tokenização é uma técnica de remoção de separadores das frases, capturando seus termos e transformando-os em um vetor de palavras. É um método importante para os trabalhos em PLN pois a partir dele, é possível aplicar outros processos de forma fácil para os métodos, abaixo segue um exemplo da tonkenização.

$$"hoje\ é\ um\ belo\ dia \rightarrow [”hoje”, ”é”, ”um”, ”belo”, ”dia”]$$

4.3.4 Remoção de *Stopwords*

A remoção de *stopword's* é uma tarefa necessária quando é perceptível que essas palavras são termos de ligações que não apresentam valor algum para o aprendizado. Por serem esses termos, demonstram grande incidência nas frases construídas e se tornam fonte de grande ruído, por estarem em termos considerados negativos, positivos e neutros. Por exemplo, algumas dessas palavras são: de, a, o, em, um, você, etc.

Assim, a remoção desses termos são necessários para obter melhores resultados na tarefa de classificação dos modelos. Para isso, foi utilizada a biblioteca do *python* NLTK[3] que dispõe de um dicionário em português de *stopword's*. Porém, além dessas, foram incluídas formas abreviadas muito comuns em textos da web, para que não passem, caso estejam presentes no texto.

4.3.5 Stemização

A *stemização* ou do inglês *stemming* é uma tarefa da PLN que visa reduzir uma palavra à sua raiz. Essa tarefa tem por objetivo reduzir a quantidade de termos no *corpus*, tornando todas as formas flexionadas, de uma palavra a um único vocábulo. Desse modo, a ocorrência da raiz mostra o quanto uma palavra e suas formas flexionadas são importante para determinada classe/*label*. A seguir, são apresentados alguns exemplos de palavras convertidas à sua raiz através do processo de *stemização*:

$$”gato”, ”gata”, ”gatinho” \rightarrow \mathbf{gat}$$

$$”meninas”, ”meninos”, ”menininhos” \rightarrow \mathbf{menin}$$

Por fim, para realizar essa tarefa foi utilizada a ferramenta Removedor de Sufixos da Lingua Portuguesa (RSLP) Stemmer, disponibilizada pela biblioteca NLTK. Esse recurso foi desenvolvido para ser um método de transformação de palavras em português para sua raiz por Orengo e Huyck (2001).

4.3.6 Vetorização

A vetorização dos documentos no *corpus* é uma etapa essencial para realizar o passo seguinte de treinar os modelos. Através dessa etapa, a máquina começa a entender o que o documento significa, ou seja, uma frase é convertida em algo claro para os algoritmos. Existem muitos métodos para serem realizados, porém, ao fim do processo, o *corpus* se

[3] Disponível em: https://www.nltk.org/

tornará um vetor n dimensional, contendo a ocorrência de palavras por documento/todas as palavras no *corpus*.

Para este trabalho alguns desses métodos foram utilizados com o intuito de validar os melhores resultados. Primeiramente, foi utilizado o *CountVector*, através da biblioteca *Sklearn*[4] que converte uma coleção de documentos de texto em uma matriz de contagens de *tokens*. Desse modo, demonstra ser o método mais simples entre os disponíveis que utilizam a técnica Bag of Word (BoW), ou saco de palavras. Nessa ferramenta existe o parâmetro N-Gram, que define como o *CountVector* irá realizar a contagem dos termos.

Por convenção esse parâmetro tem como valor o (1,1), indicando que esta contagem será feita a cada palavra, chamado uni-grama. Além disso, é possível determinar o valor como (2,2) para bigramas, (3,3) para trigramas, etc. A exemplo disso, se o trabalho for realizado com um bigrama, então a contagem é feita em uma sequência de duas palavras como: "Muito bom", "Muito ruim", "Sua Pesquisa", etc. A figura 5 ilustra uma matriz resultante da vetorização através do método *CountVector*.

Figura 5 – Matriz de contagem de termos por documento, apresenta quantas vezes cada token ocorre em cada frase, na qual cada frase é uma linha dessa matriz e cada token uma coluna.

token 1	token 2	token 3	...	token n	token m	token w
0	1	0	...	1	0	1
1	0	0	...	1	0	0
0	2	0	...	0	3	0
⋮	⋮	⋮	...	⋮	⋮	⋮
1	0	1	...	0	0	1
0	0	0	...	0	0	0
0	0	1	...	0	0	0

Fonte: Elaboração própria.

Após o uso do *CountVector*, e através da mesma biblioteca *Sklearn*, foi testado também o *Term Frequency - Inverse Document Frequency* (TFIDF), que diferente do anterior, que somente contabilizava a ocorrência, calcula a importância da frequência das palavras. O TFIDF é uma medida estatística que afere a importância de uma palavra em um documento para todo um *corpus* de documentos. O cálculo da importância dos tokens é definido pelas equações 4.1, 4.2, 4.3, 4.4 .

$$\mathbf{tf}(t,d) = \frac{f_d(t)}{\underset{w \in d}{max} f_d(w)} \tag{4.1}$$

$$\mathbf{idf}(t,D) = \ln\left(\frac{|D|}{|d \in D : t \in d|}\right) \tag{4.2}$$

[4] Disponível em: https://scikit-learn.org/

$$\mathbf{tfidf}(t, d, D) = \mathbf{tf}(t, d) \cdot \mathbf{idf}(t, D) \tag{4.3}$$

$$\mathbf{tfidf}'(t, d, D) = \frac{\mathbf{idf}(t, D)}{|D|} + \mathbf{tfidf}(t, d, D) \tag{4.4}$$

Desse modo, D é o *corpus* de documentos e $f_d(t)$ é a frequência do termo t, no documento d. O $\mathbf{tf}(t, d)$ calcula quantas vezes um token aparece no documento pelo total de palavras no documento. O $\mathbf{idf}(t, D)$ calcula o *log* da quantidade de documentos pela quantidade de documentos que o token aparece. Por fim, o $\mathbf{tfidf}(t, d, D)$ é a multiplicação dos resultados em $\mathbf{tf}(t, d)$ e $\mathbf{idf}(t, D)$ é a importância do token para o *corpus*.

O TFIDF foi escolhido por obter o melhor resultado na métrica da área sobre a curva PR, que determina quão bem o modelo classifica entre as classes. O TFIDF tem parâmetros que especificam como estes vetores irão ser formados. Um desses parâmetros, o *n-gram*, foi avaliado em diferentes testes, dividindo assim o TFIDF em duas diferentes avaliações.

Assim como é observado por Suzuki et al. (2010), um *word n-gram* se refere a menor unidade a ser declarada para um texto/frase. Quando esse parâmetro é definido como 1 (também chamado de *unigram*), significa que está trabalhando com a menor unidade do texto. Ao ser aumentado para dois, por exemplo, significa que foi definido que a menor unidade do texto serão duas palavras consecutivas e assim por diante.

No *TfidfVectorizer*[5], o *n-gram* é definido pela *tupla (min_n, max_n)*, que define os n-valores a serem extraídos pelo algoritmo. Desse modo, quando se define como (1-2), significa que a operação irá retornar um vetor com índice para 1 palavra e 2 palavras consecutivas. Na tabela 8 é possível observar que o TFIDF foi executado com os valores de *n-gram* (1,1), (1,2) e (1,3). A diferença notada entre as três execuções do algoritmo é baixa, porém, é visto que, para algoritmos mais simples, como o classificador *naive bayes* (NB), existe uma melhora nas métricas, quando aumenta a quantidade de palavras como unidade mínima. Porém, no caso do SVM, é notado uma piora dos seus resultados.

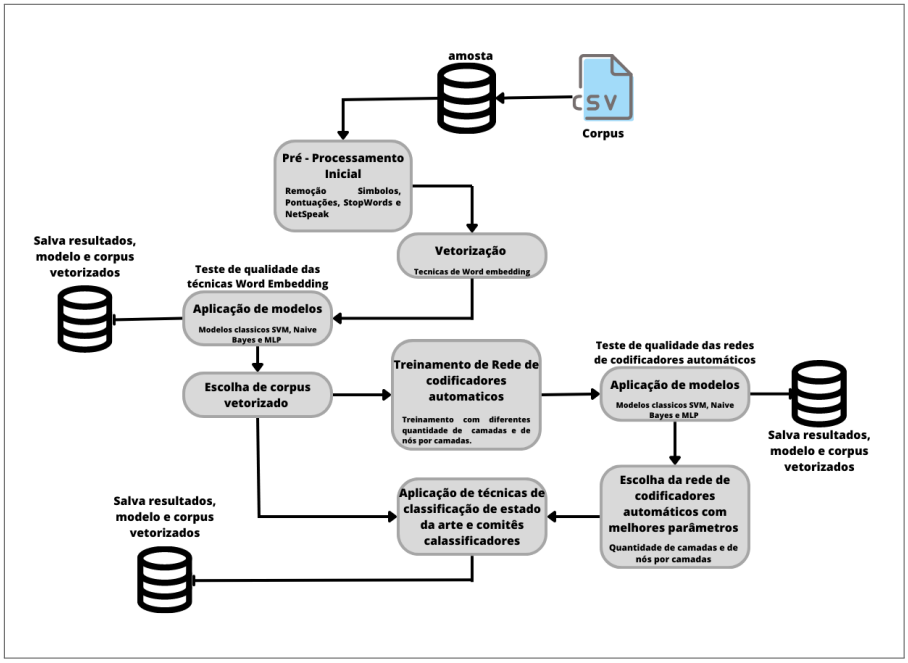
Além disso, quando se determina um maior número máximo de *n-gram*, significa dizer que os vetores criados pelo algoritmo TFIDF serão maiores, pois irão conter todo o conjunto de palavras entre o valor mínimo e o máximo. Por esse motivo, para este estudo, foi definido para o parâmetro *n-gram* o valor de (1,1), compreendendo a somente 1 palavra consecutiva como unidade do texto. O intuito da presente pesquisa é averiguar os resultados e custos computacionais, então, como a diferença entre as melhoras dos algoritmos observados é bem próxima, pode ser compreendida de uma forma coerente a utilização do que produza o menor vetor.

[5] Função de vetorização da linguagem *python* compartilhada pela biblioteca *scikit-learn*

4.4 CONCLUSÕES

Conforme todos os métodos e materiais expostos neste capítulo, a figura 6 apresenta um fluxograma de como serão realizados todos os experimentos, até a escolha da técnica de *word embedding*, rede de codificadores automáticos e modelo classificador. O diagrama apresenta a ordem cronológica de pré-processamento, testes, validação de resultados e aplicação de técnicas, até alcançar o objetivo final.

Figura 6 – Fluxograma com as etapas que os experimentos serão realizados, os métodos que vão ser aplicados e as validações de seus resultados



Fonte: Elaboração própria.

Como é possível ver na figura 6, inicialmente, de cada *corpus* de treino e teste será retirada uma amostra (de 7500 textos rotulados, divididos entre as três classes). Então, esse novo *corpus* será pré-processado com técnicas iniciais de remoção de textos vazios(Stop Words), pontuações, acentos, símbolos e tudo que é considerado irrelevante para o ganho de aprendizado. Ainda nessa etapa, será utilizado um dicionário léxico que transformará textos informais da internet em seu modo formal.

Após a padronização dos textos no corpus, serão transformados em uma lista de tokens, para, então, utilizar as técnicas de *lematização* e *stemização* nesta lista e, assim, criar duas novas listas. As listas resultantes dos procedimentos citados são salvas para a execução dos métodos de *word embbeding*. Assim, os resultados obtidos de acordo com os classificadores SVM, MLP e *naive bayes* serão comparados com base em métricas como precisão, acurácia, revocação e outros.

A partir da avaliação de resultados, a técnica de incorporação de palavras com melhor qualidade é escolhida para dar prosseguimento aos experimentos. Desse modo, baseado na melhor incorporação de palavras, é treinado um conjunto de RCA, fundamentado pelo

estudo de Ameur, Jamoussi e Hamadou (2018). Fatores como as diferentes quantidades de camadas e nós por camadas, serão avaliadas nessas redes, observando a tendência dos seus resultados.

Assim, a rede de codificadores automáticos será escolhida a que demonstrar uma melhor qualidade, menores custos computacionais e tempo de processamento durante o treinamento dos modelos de teste. Além disso, será validada a observação de Ameur, Jamoussi e Hamadou (2018) que expõem ter obtido os melhores resultados, através dos codificadores, com uma redução da dimensão em 20% da original. Ao fim, será escolhida a rede de codificadores com melhor qualidade e custo computacional, para então, realizar o treinamento de modelos de estado da arte, comitês classificadores e, posteriormente, o classificador final com o *corpus* total.

Em suma, os *corpus* estão formatados e definidos para treinar e validar os modelos. Todos os resultados do pré-processamento foram salvos através da biblioteca *Pickle*[6] do *python* e também em formato CSV para os dados em *dataframe*. Assim, é possível carregar seus resultados e utilizá-los para diferentes modelos e diminuir o tempo de processamento dos dados para o treinamento dos classificadores. O próximo capítulo abordará o treinamento, os experimentos realizados e seus resultados, também a escolha do modelo classificador que será usado no *corpus* criado nesta dissertação[7].

[6] Disponível em: https://docs.python.org/3/library/pickle.html

[7] Todos os dados produzidos nesta dissertação estão disponibilizadas através do repositorio no link: https://github.com/chnmelo/mestrado-covid-sentiment-analysis

5 MODELOS E RESULTADOS EXPERIMENTAIS

Neste capítulo serão abordados os modelos treinados, as escolhas de parâmetros, seus custos computacionais e resultados. Todos os modelos utilizados neste trabalho serão apresentados, os que demonstrarem uma melhor precisão e também os métodos diferentes abordados. Por fim, será exposto graficamente, quão bem os modelos escolhidos tiveram seus resultados através de métricas muito vistas no estado da arte.

5.1 MODELOS EXPERIMENTAIS

Para este estudo foi realizada uma análise exploratória dos modelos de AM e técnicas de tratamento e pré-processamento de dados da PLN, como detalhado no capítulo anterior. Esse levantamento dos modelos a serem treinados foi crucial para a avaliação dos resultados e custos computacionais envolvidos na tarefa de Análise de Sentimentos.Além da utilização dos Codificadores Automáticos como redutores de dimensionalidade afetam na precisão e no consumo computacional envolvido no treinamento de tais modelos.

A escolha dos modelos foi embasada em estudos que apresentaram tais algoritmos como de melhor resultado de precisão. Então, esses algoritmos foram separados entre as técnicas supervisionada e não supervisionada, e suas abordagens como aprendizado profundo ou não. Para este estudo foi necessário, explorar as mais diversas técnicas de AM, visando a validação do redutor de dimensionalidade RCA, suas vantagens e desvantagens.

5.1.1 Incorporação de palavras (*Word embedding*)

Diante das discussões empreendidas, foi gerado primeiramente duas bases de estudos, a partir de cada conjunto de dados aqui utilizado. Em cada base de estudo foi aplicada diferentes técnicas de pré-processamento, para avaliar os resultados obtidos com as RCA e as suas classificações. Assim, uma base de estudo era a que se aplicaria o redutor de dimensionalidade e na outra não o aplicaria, para comparar os resultados dos modelos de cada base. Primeiramente, foram observadas as técnicas iniciais do PLN como: TFIDF e *word2vec*, nas quais foram percebidas uma melhoria na utilização do TFIDF como técnica de incorporação de palavras[1].

Conforme observado na tabela 8, foram realizados dois treinos e teste de modelos para cada técnica de *Word Embedding* e cada função de incorporação de palavras. Desse modo, o estudo obteve 15 resultados, cada um com dois classificadores treinados a serem avaliados. Os resultados dessas tarefas foram avaliadas através das métricas de medidas

[1] **Incorporação de palavras** (Do inglês *Word Embedding*) - Técnica que gera representações das palavras em formatos vetoriais (e outros) para que a máquina (Computador) possa entender e realizar o treinamento de modelos e aplicação de outras técnicas.

Tabela 8 – Métricas obtidas através do uso de *Word embedding* com a base no *corpus kaggle*, treinadas com os classificadores Naive Bayes, MLP e SVC. Para comparar a qualidade obtida por meio de cada técnica utilizada subdividida em três métodos de representação das frases token, lematização e stemização.

Tec. Incorporação	**Tec. Vetorização**	**Modelo**	**Precisão**	**Recall**	**F1**	**Acurácia**	**Precisão Ponderada**
CountVector	Token	naive_bayes	0.682520	0.676133	0.673404	0.676133	0.989051
		svc	0.664099	0.647600	0.643265	0.647600	0.992537
		mlp	0.658319	0.653333	0.652895	0.653333	0.999547
	stem	naive_bayes	0.595366	0.562400	0.557145	0.562400	0.946073
		svc	0.583168	0.540133	0.533264	0.540133	0.965958
		mlp	0.580701	0.540800	0.535281	0.540800	0.985606
	lemma	naive_bayes	0.676102	0.673333	0.670281	0.673333	0.986142
		svc	0.653298	0.646267	0.643305	0.646267	0.989871
		mlp	0.634805	0.625467	0.625529	0.625467	0.999112
tfidf (1-1)	Token	naive_bayes	0.686530	0.681067	0.676898	0.681067	0.963662
		svc	0.708098	0.689333	0.691141	0.689333	0.989542
		mlp	0.661522	0.654800	0.657344	0.654800	0.999691
	stem	naive_bayes	0.681639	0.680133	0.677496	0.680133	0.969712
		svc	0.695789	0.687333	0.688219	0.687333	0.988892
		mlp	0.634812	0.630133	0.630423	0.630133	0.999772
	lemma	naive_bayes	0.686078	0.681867	0.679801	0.681867	0.966783
		svc	0.696896	0.688267	0.688832	0.688267	0.987450
		mlp	0.639799	0.634800	0.636089	0.634800	0.999463
word2vec	Token	naive_bayes	0.389325	0.373333	0.316534	0.373333	0.977227
		svc	0.562646	0.563467	0.555759	0.563467	0.984735
		mlp	0.577208	0.568133	0.567159	0.568133	0.992483
	stem	naive_bayes	0.361688	0.378400	0.324854	0.378400	0.956777
		svc	0.517996	0.513067	0.497664	0.513067	0.984232
		mlp	0.547362	0.542000	0.531094	0.542000	0.974179
	lemma	naive_bayes	0.369956	0.396667	0.349269	0.396667	0.945855
		svc	0.516049	0.520533	0.504103	0.520533	0.975175
		mlp	0.542180	0.537067	0.521348	0.537067	0.974230

Fonte: Elaboração própria.

mais utilizadas no estado da arte, que são a precisão do modelo, o *recall*, o F1 e a Curva PR (Curva baseada na precisão e no retorno do modelo).

A precisão do modelo é indicada quando a avaliação dos seus resultados são mais afetadas por falsos positivos que os falsos negativos. Onde é visto que nesse trabalho, sentimento que são rotulados erroneamente afetam diretamente a analise das reações dos usuários. Já o *recall* apresenta o quanto os modelos classificaram como falso negativo uma determinada classe, podendo assim mensurar quais classes sofrem mais com ruídos na classificação. Por outro lado, o F1-score traz uma média aritmética entre as duas apresentadas, assim podendo comparar o impacto conjunto entre a relação dos falsos positivos e negativos.

A curva PR é importante, pois demonstra quão sensível e específico o modelo consegue ser conforme a classe, é demonstra melhores resultados quando se trabalha com dados desbalanceados. Através da biblioteca scikit-learn essa medida é calculada com a função *Averaged Precision* (Precisão ponderada), que calcula o valor baseado na precisão, retorno e nos limites da classe (rotulo). Foi observado que o corpus TBCOV apresentou grande desbalanceamento entre as classes proposta, e desse modo foi optado por visualizar a especificidade das classes através dessa métrica. A partir dessa função e através de sua

relação com as outras medidas, é factível escolher quais conjuntos de incorporação de palavra e vetorização detêm melhor resultado para a base de dados *Kaggle*. Os resultados mostram que utilizar *stemming* ou *lemmatization* não resultou em uma melhora para esse corpus.

Os resultados com os *tokens* em si, mostraram que com a técnica de *word2vec*, não obteve boas medidas em relação às demais. Por outro lado, o TFIDF, como mencionado, foi a técnica que obteve maiores resultados entre os avaliados. Chegando a apresentar medidas um pouco melhores que as do método *CountVector*, que somente calcula a ocorrência das palavras (*Tokens*) na frase.

A técnica de *word2vec* demonstrou uma pior qualidade quando comparada com as outras utilizadas, poisa mesma foi realizada a través da media dos valores no vetor produzido através do posicionamento de cada palavra sobre a frase observada. Desse modo, é visto que essa abordagem utilizada pode ser submetida a ruídos de treinamento da técnica. Entretanto, foi o modo mais popularmente difundida entre as pesquisas com esta metodologia de vetorização de palavras (SOUZA; MAIA, 2019; MASSONI, 2021).

Diferente das métricas obtidas com o *corpus* da *kaggle*, o TBCOV alcançou números superiores e mais atraentes para seu uso como *corpus* de estudo principal. Assim como realizado para o *corpus* anterior, os mesmos testes foram aplicados sobre esse. A tabela 9 apresenta uma melhor precisão e característica de operação do receptor. Além disso, é visto que os resultados utilizando a incorporação de palavras *stemming* e somente utilizando *tokens*, obtiveram resultados muito próximos ou iguais.

Assim como nos testes anteriores, a técnica de vetorização TFIDF alcançou resultados pouco maior que às demais. O *Word2Vec* demonstrou o pior resultado entre as técnicas. Por outro lado, o parâmetro *n-gram* definido como (1,2) teve os resultados superiores aos demais, porém próximos. Desse modo, para os testes seguintes nesse *corpus* serão realizados experimentos com os dois primeiros valores, (1,1) e (1,2), para o parâmetro *n-gram*.

Desse modo, para entender como os resultados se diferenciavam entre eles para cada técnica de textitword-embbending foi realizado um teste ANOVA, onde a hipótese nula é que não diferença (ou há pouca diferença) entre os resultados. Já a hipótese alternativa é que existe diferença entre a media das métricas dos modelos aplicados. Os resultados do teste ANOVA entre todas as técnicas é apresentado na tabela 10.

Assim, se mostra, com um *p-value* abaixo de 0.05 em todas as técnicas de vetorização que há uma variância entre as três técnicas de incorporação de palavras (TFIDF, *Word2Vec* e *CountVector*). Porém, através da tabela 9 se vê que os resultados de métricas entre as técnicas *CountVector* e TFIDF se aproximam mais que as da técnicas *Word2Vec*. Desse modo, foi realizado um teste ANOVA somente com a media dos resultados dos modelos das técnicas mais próximas e é apresentado o resultado na tabela 11.

Com um *p-value* próximo a 1, o resultado do teste indica uma aproximação muito

Tabela 9 – Métricas obtidas através do uso de *Word embedding* com a base no *corpus* TBCOV, treinadas com os classificadores Naive Bayes, MLP e SVC. Para comparar a qualidade obtida por meio de cada técnica utilizada subdividida em três métodos de representação das frases token, lematização e stemização.

Tec. Incorporação	**Tec. Vetorização**	**Modelo**	**Precisão**	**Recall**	**F1**	**Acurácia**	**Precisão Ponderada**
CountVector	Token	naive_bayes	0.781339	0.772800	0.774606	0.772800	0.998356
		svc	0.790820	0.789867	0.790031	0.789867	0.996850
		mlp	0.799303	0.796667	0.797333	0.796667	0.999952
	Stemização	naive_bayes	0.580377	0.575600	0.573498	0.575600	0.952736
		svc	0.659378	0.628800	0.634712	0.628800	0.981166
		mlp	0.655944	0.625600	0.625609	0.625600	0.994718
	Lemmatização	naive_bayes	0.772229	0.760267	0.762166	0.760267	0.997555
		svc	0.798814	0.797600	0.797771	0.797600	0.997486
		mlp	0.795699	0.794800	0.794800	0.794800	0.999966
tfidf (1-1)	Token	naive_bayes	0.785384	0.776933	0.778757	0.776933	0.992303
		svc	0.818158	0.802800	0.805041	0.802800	0.992408
		mlp	0.799245	0.798133	0.797909	0.798133	0.999925
	Stemização	naive_bayes	0.784386	0.774000	0.775935	0.774000	0.994392
		svc	0.826695	0.816000	0.817931	0.816000	0.995317
		mlp	0.797587	0.797733	0.797318	0.797733	0.999955
	Lematização	naive_bayes	0.785100	0.775867	0.777666	0.775867	0.993406
		svc	0.825115	0.812933	0.814962	0.812933	0.994414
		mlp	0.825115	0.812933	0.814962	0.812933	0.994414
word2vec	Token	naive_bayes	0.471506	0.454800	0.446636	0.454800	0.903679
		svc	0.642946	0.605333	0.605154	0.605333	0.987638
		mlp	0.650083	0.643200	0.638404	0.643200	0.987599
	multirow3*Stemização	naive_bayes	0.469254	0.447200	0.439089	0.447200	0.890291
		svc	0.651759	0.611733	0.611833	0.611733	0.985096
		mlp	0.662439	0.645200	0.641724	0.645200	0.986702
	Lematização	naive_bayes	0.466063	0.445200	0.432665	0.445200	0.921738
		svc	0.651228	0.603200	0.603249	0.603200	0.983228
		mlp	0.657910	0.636667	0.633805	0.636667	0.984765

Fonte: Elaboração própria.

Tabela 10 – Resultados do teste ANOVA com métricas obtidas pelas técnicas *Word-Embbendind*.

Corpus	**Tec. Vetorização**	**P-Value**	**statistic**
KAGGLE	TOKEN	0.0000	59.6082
	STEMATIZAÇÃO	0.0000	6.9483
	LEMMATIZACÃO	0.0011	0.7071
TBCOV	TOKEN	0.0000	27.0655
	STEMATIZAÇÃO	0.0000	59.1920
	LEMMATIZACÃO	0.0005	7.9578

Fonte: Elaboração própria.

grande (e falta de variância) entre as incorporação de palavras *CountVector* e TFIDF. Mostrando assim, que por serem tecnicas que possuem o mesmo principio elas resultaram em métricas de validação pouco variantes, tendo então uma grande proximidade. Sendo assim, sem grande variação entre as tecnicas não seria possivel definir qual se superava, então na busca de entender mais sobre essa diferença foi realizado também o teste t pareado, que segundo Gaspar et al. (2023, p. 86) é:

> "[...] também chamado de teste-t de amostra dependente ou emparelhada, é um teste estatístico usado para comparar a média de duas

Tabela 11 – Resultados do teste ANOVA com métricas obtidas pelas técnicas *Word-Embbendind* TFIDF e *CountVector*.

Corpus	**Tec. Vetorização**	**P-Value**	**statistic**
KAGGLE	TOKEN	0.2168	1.7976
	STEMATIZAÇÃO	0.2374	1.6311
	LEMMATIZACÃO	0.4702	0.5745
TBCOV	TOKEN	0.7508	0.1081
	STEMATIZAÇÃO	0.6873	0.1743
	LEMMATIZACÃO	0.9183	0.0112

Fonte: Elaboração própria.

amostras relacionadas (emparelhadas). Ele é baseado na hipótese de que a diferença entre as médias das amostras é igual a zero. Se a diferença entre as médias foi estatisticamente significativa, conclui-se que a variável independente afeta a variável dependente."

Tabela 12 – Resultados do teste T pareado com a media das métricas obtidas pelas técnicas *Word-Embbendind* TFIDF e *CountVector*.

Corpus	**Tec. Vetorização**	**P-Value**	**statistic**
KAGGLE	TOKEN	0.0000	31.8070
	STEMATIZAÇÃO	0.0007	-9.3992
	LEMMATIZACÃO	0.0713	-2.4386
TBCOV	TOKEN	0.0087	-4.7996
	STEMATIZAÇÃO	0.0122	-4.3497
	LEMMATIZACÃO	0.5635	-0.6289

Fonte: Elaboração própria.

Com os resultados apresentados na tabela 12, do uso do teste T pareado, é visto que o p-value do uso das técnicas de incorporação de palavras com a *tokenização* simples e a *stemização* foram abaixo de 0,05. Desse modo, entende-se que existe uma variância na média das métricas obtidas entre as duas técnicas *CountVector* e TFIDF. Somente a lematização obteve um p-value acima de 0.5, vendo que há pouca variação entre as técnicas com esse método de vetorização.

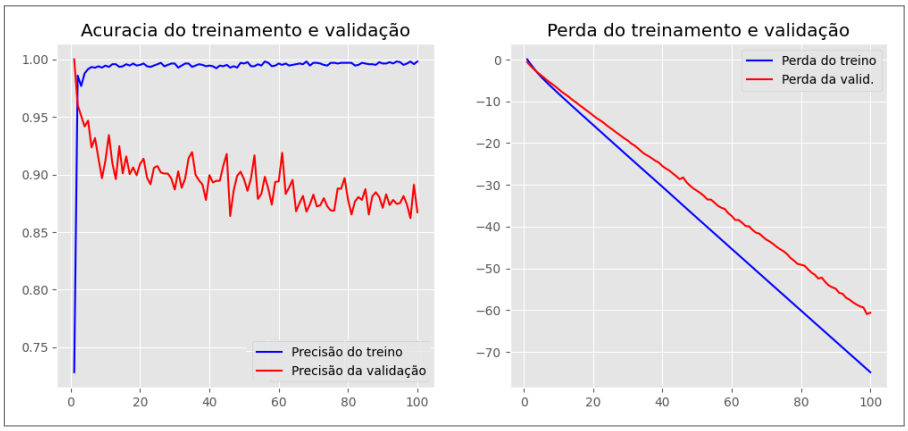
Então, com o *corpus* do *kaggle*, a técnica de vetorização TFIDF com (1,1) de parâmetro *n-gram* e com a *tokenização* simples, será escolhida para utilização deste estudo. Já no *corpus* TBCOV, foi escolhido o método de incorporação da palavra *stemming* juntamente com o TFIDF e os parâmetros de *n-gram* (1,1) e (1,2). Pois, esses que foram citados, apresentaram os melhores resultados avaliados nos testes realizados.

Como foi observado desbalanceamento dos dados no corpus TBCOV, então foi escolhido o uso da precisão como métrica principal de escolha da qualidade, pois a acurácia sofre uma drástica queda de qualidade sobre dados desbalanceados. E assim essa métrica

será utilizada para o treino e validação dos demais modelos. Então, para complementar o estudo das técnicas de vetorização também foram analisados o uso de duas técnicas de DL, para comparar as qualidades obtidas. Para isso, foram treinadas duas redes neurais, baseadas na precisão do modelo, para cada método de incorporação de palavras, as quais são: Rede Neural Recorrente (RNN) e Rede Neural Convolucional (CNN).

Por meio dos experimentos realizados com os algoritmos de DL foi visto que o seu uso na etapa de validação das técnicas não apresentou bons resultados. Isso é mostrado na figura 7 que expõe um gráfico de tendência da precisão e perda do modelo RNN treinado com 100 épocas para a técnica de incorporação de palavras TFIDF com $n-gram = (1, 1)$.

Figura 7 – Gráfico de tendência de precisão e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e teste. Realizado com o vetor TFIDF produzido a partir do *corpus* TBCOV



Fonte: Elaboração própria.

Através da figura 7 é visto que sua qualidade alcançada fica abaixo do modelo NB, mesmo sendo realizado com a técnica e parâmetro no qual foi observado melhores resultados entre os utilizados. Além disso, é evidenciado que o modelo não encontra uma tendência na perda, porém isso ocorre devido ao fato da rede ter sido treinada com base na melhoria da precisão. Desse modo, é observado uma precisão próxima de 90% na validação, nos modelos tradicionais utilizados são contempladas qualidades abaixo de 80% com o uso do *stemming* e TFIDF com $n - gram = (1, 1)$.

Porém, através dos resultados obtidos para as demais técnicas apresentados no apêndice D, é visto que a precisão obtida através dos algoritmos RNN produziram resultados muito próximos, a distinção entre esses elementos é somente destacado através das oscilações entre as épocas do treinamento do modelo. Em todos os gráficos disponíveis para a rede neural recorrente com 4 e 2 camadas, sua precisão na validação ultrapassou 90%. Por outro lado, foi visto alcançar 100% na precisão do treino, podendo indicar que os modelos apresentam boa qualidade, mas que podem sofrer, porém, pode estar indicando um *overfitting* dos modelos, necessitando de uma investigação mais avançada.

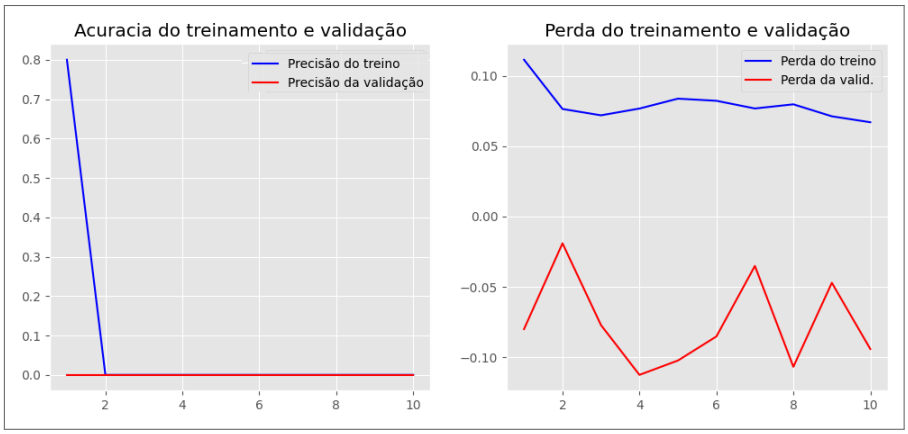
Entretanto, quando comparamos os resultados do modelo de DL do corpus TBCOV com o *kaggle* é visto que a precisão da validação mostra uma piora com a coletânea *kaggle* como visto nas figuras 62, 65 e 67 do apêndice D. Porém, esse evento já

era observado nos outros experimentos, mostrando que o corpus TBCOV apresenta uma melhor especificação entre seus atributos na tarefa de classificação.

Por outro lado, os experimentos realizados com a CNN apresentaram resultados ainda mais próximos, sendo treinados somente com 5 épocas, as redes apresentaram resultados muito baixos, quando comparados aos que foram vistos com o RNN. Também foi observado que a precisão dos resultados para validação e treino da CNN foram bastante semelhantes entre os modelos treinados com os dois corpus avaliado neste trabalho.

Assim, nenhuma das técnicas avaliadas apresentaram resultados no treino e teste acima de 40% de precisão nos modelos que utilizaram a tecnica CNN. A figura 8 mostra a tendência do treinamento das redes convolucionais com a incorporação de palavras TFIDF parâmetro $n - gram = (1, 1)$ e técnica *stemming*.

Figura 8 – Gráfico de tendência de precisão e perda obtida durante o aprendizado da Rede Neural Convolucional para treino e teste. Realizado com o vetor TFIDF $n - gram = (1, 1)$ produzido a partir do *corpus* TBCOV com técnica *stemming*



Fonte: Elaboração própria.

A partir da figura 8 é observado que o modelo convolucional inicia e termina o treinamento do seu modelo sem alteração no índice de precisão, atingindo o mínimo, não apresentando uma curva de aprendizado com esta métrica. Porém, quando se observa o gráfico de tendência da perda, é vista uma queda, demonstrando que o algoritmo está conseguindo melhorar essa métrica, mas não encontra uma tendência em 5 épocas, pois ainda apresenta uma diminuição. Além disso, foi observado que o treinamento, mesmo com uma menor amostragem, revela uma alta demanda de processamento computacional, no qual foi visto a necessidade de uma maior quantidade de tempo para a conclusão das 5 épocas de treinamento.

Entres as redes convolucionais treinadas, a que apresentou menor quantidade de tempo foi a que utilizou *CountVector* com a técnica *stemming* sobre uma amostragem do corpus com 7500 *tweets*. Além disso, esses algoritmos não apresentaram diferenças em sua precisão que pudesse dar suporte na tomada de decisão das técnicas a serem utilizadas neste trabalho, conforme visto no apêndice D. Por esse motivo, para os próximos experimentos

foi escolhido não realizar o treinamento das redes neurais como etapa de validação dos métodos a serem utilizados no modelo final.

Assim, foram definidas as técnicas com os melhores resultados a serem avaliados na próxima etapa. A fase seguinte a ser descrita no próximo subcapítulo é a de validação dos codificadores automáticos. Nessa etapa serão avaliados os resultados, custos e tempo de execução do treino e teste da rede de codificadores. Além da quantidade de camadas, número de dimensão a ser reduzida e a mensuração dos classificadores.

5.1.2 Codificadores automáticos (*Autoencoders*)

Diante da escolha do vetorizador de palavras (técnica de incorporação de textos), foi iniciado o processo de codificação dos vetores, para reduzir a dimensionalidade resultante do vetor final.As RCA foram aplicadas, seguindo os métodos mais utilizados por pesquisadores para a linguagem *python*. Desse modo, foi utilizada a biblioteca de funções *tensorflow*, que dispõe para a linguagem diversos algoritmos e ferramentas da Aprendizagem Profunda.

Os codificadores automáticos têm seu aprendizado baseado na perda mínima (da função de perda aplicada) da tarefa de comprimir e reconstruir os vetores. Para este estudo, com base na ferramenta disponibilizada no *python*, os parâmetros escolhidos foram fundamentados em experimentação dos valores, além de aplicação e validação de estudos recentes com as mesmas ferramentas.

Os parâmetros de maior importância a serem observados são os de dimensionalidade, função de ativação e perda para cada camada que íntegra o codificador. De uma camada para outra, essa dimensão é reduzida para forçar o codificador a restringir as características mais importantes e também forçar o aprendizado.

Como a dimensão inicial do vetor original é grande, então, primeiramente, é iniciada a entrada com o tamanho total da dimensão do vetor. Em seguida, o treinamento é iniciado, reduzindo a dimensionalidade em camadas densas[2], sendo o método que os codificadores buscam aprender sobre as características do vetor.Assim, é imposto um gargalo, redução da dimensão, para assim, obrigar o codificador a buscar uma representação com menor perda ou aumento de métrica imposta ao vetor reduzido.

Foram realizados experimentos para os dois corpus inicialmente aplicados a um codificador com 6 níveis, uma escala de redução dos nós após a entrada de 1/2 do valor anterior. Esse codificador apresentava para cada camada os seguintes nós: 4096, 2048, 1024, 512, 256 e 128. Desse modo, buscava reduzir o vetor original a uma representação de menor perda possível com uma dimensão de 128 atributos. Para assim, possibilitar resultados mais rápidos e com menor custo computacional.

[2] No aprendizado de máquina, uma camada densa compreende a uma camada que esta totalmente conectada à camada anterior, ou seja, todos os neurônios apresentam conexão com os neurônios da camada anterior. (NANDINI; KUMAR; CHIDANANDA, 2021)

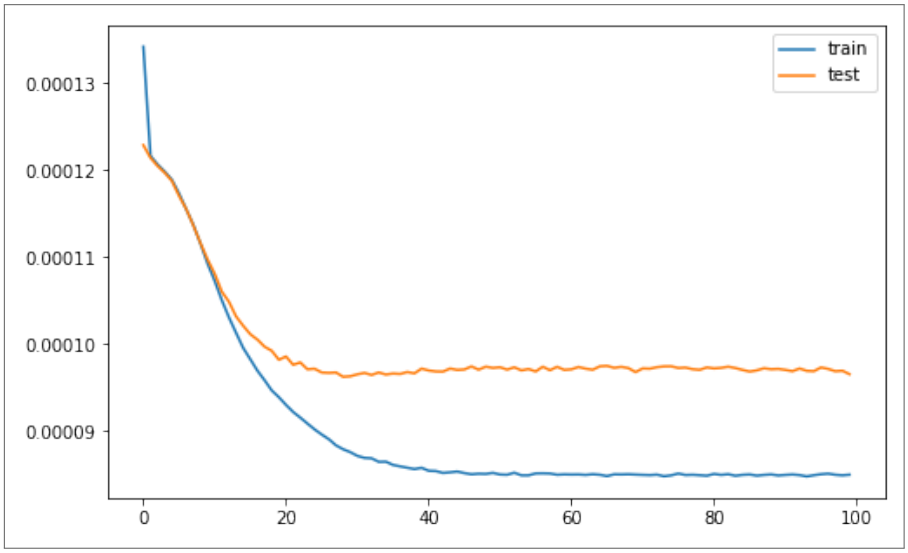
5.1.2.1 Corpus *Kaggle*

A partir dos valores vistos nesse experimento evidenciado acima, foi observada uma alta demanda de custo computacional necessária para realizar a operação. Foi utilizada a mesma amostragem do experimento com as técnicas de incorporação de palavras, para assim, cruzar os resultados. Entre as técnicas de incorporação, foi escolhido o TFIDF, que apresentou os maiores resultados, com o parâmetro *n-gram* igual a (1, 1) e (1, 2). Esses dois valores de parâmetros foram os que demonstraram estar próximos nas métricas e nos melhores resultados.

Esses dois valores para o parâmetro *n-gram* foram escolhidos para comparar seu custo computacional e tempo de processamento, visto que, possuem dimensionalidades diferentes. Os vetores produzidos pela técnica, resultam em uma dimensionalidade de 39340 e 7973 para (1,2) e (1,1) respectivamente. Desse modo, o resultado dimensional do vetor seria menor e, por conseguinte, foi observado a redução do tempo de execução de mais de 3 horas (em média) para pouco mais de 1 hora de processamento.

Então, foi iniciado o treinamento dos codificadores, no qual foi definido como método de aprendizado a redução do índice de perda *log-loss*. Assim, por padrão, é definido no algoritmo concedido pela biblioteca *Keras* do *python*. O gráfico apresentado na figura 119 mostra esse aprendizado relacionado a queda da perda até o codificador atingir a tendência.

Figura 9 – Gráfico de tendência de perda obtida durante o aprendizado da rede de codificadores automáticos para treino e teste. Realizado com a redução de dimensionalidade até 128 com as funções de validação Linear e ReLu, treinadas com o vetor TFIDF produzido a partir do *corpus kaggle*



Fonte: Elaboração própria.

O gráfico de aprendizado da rede de codificadores com o vetorizador TFIDF com parâmetro *n-gram* $= (1, 1)$ mostra que ao atingir $\approx 8.5^{-5}$, o treino alcança sua tendência. Isso é visto a partir da época[3] (Período) 40 na RCA e na validação, após a 20, já se

[3] Uma época, é quando o modelo utiliza todos os dados de treinamentos e validação e assim conclui

demonstra esta inclinação.O custo computacional para essa tarefa foi admissível para o seu uso, necessitando somente de 1 hora e 5 minutos para realizar todo o treinamento e validação.

Como os experimentos realizados com as técnicas de incorporação de palavras mostraram que o TFIDF com o parâmetro *n-gram* igual (1, 1) e (1, 2) apresentou resultados muito próximos, foi realizado, então, um experimento para comparar seus custos computacionais no treinamento de codificadores. Esse experimento é apresentado na tabela 13, que exibe quantas horas, minutos e segundos levaram para ser concluído o aprendizado. Assim, como também nos apresenta o menor perda do treino (MLT) e a menor perda da validação(MLV).

Tabela 13 – Tabela de comparação de resultados a partir do *log-loss* dos *Autoencoders* com os *n-gram's* (1,1) e (1,2).

Qtd. de camadas	**Camadas**	**n-gram**	**hora**	**min**	**seg**	**MLT**	**MLV**
6	4096,2048,1024,512,256,128	1,1	1	5	23	0.000085	0.000096
6	4096,2048,1024,512,256,128	1,2	3	40	05	0.000021	0.000022

Fonte: Elaboração própria.

O treinamento da rede de codificadores com o vetor produzido pelo TFIDF com parâmetro *n-gram* (1, 2) necessitou de mais tempo, por ter uma dimensionalidade ainda maior que o vetor comparado. Além disso, as medidas de perdas são bastante próximas e não indicam superioridade ao outro vetor que pudesse incentivar seu uso. Por outro lado, o gráfico da inclinação do aprendizado mostra que a rede de codificadores não chegou a encontrar uma tendência, continua em queda até sua última época, este gráfico pode ser visualizado no apêndice E.

Além disso, a minimização da função de perda para os dois parâmetros são bem próximos e torna aceitável a escolha pelo que produz o menor tempo de treino. Ainda assim, com o viés de comparar e concretizar os resultados, os codificadores foram aplicados à base de dados de teste. O vetor resultante que é sua representação de dimensionalidade reduzida, foi analisado com base nas métricas vistas no subcapítulo anterior e também em seu tempo de execução e custo computacional.

Na tabela 31 é apresentada as medidas obtidas com a utilização do codificador treinado com o *corpus kaggle*. Nota-se que há uma piora nos resultados, quando é comparado com os classificadores treinados com o vetor original. Porém, o tempo de execução dos classificadores exibe uma redução que é o intuito do seu uso. A piora da precisão, métrica escolhida para embasar a escolha dos parâmetros nos vetores reduzidos, é em média 0.087, que não representa uma grande perda.

Conforme visto na tabela, o modelo que alcançou maior qualidade com a redução foi o SVC, que também obteve um tempo de processamento 98.76% menor que o visto com

uma iteração. Ou seja, cada época é um ciclo no modelo a ser treinado.

Tabela 14 – Resultados do teste dos classificadores a partir do vetor de representação produzido pelo codificador automático com 128 nós na camada de saída comparados com os resultados de teste obtidos partir do vetor TFIDF[*n-gram*= (1,1)]. Rede de codificadores automáticos com 6 camadas (corpus *kaggle*)

Reduzido	**Classificador**	**Precisão**	**Precisão Ponderada**	**Treino Tempo (Segundos)**
Sim	NB	0.511	0.555	0.009
Não	NB	0.576	0.482	0.842
Sim	SVM	0.630	0.666	18.685
Não	SVM	0.699	0.742	1508.023
Sim	MLP	0.563	0.593	22.291
Não	MLP	0.638	0.685	176.792

Fonte: Elaboração própria.

o vetor original. Os outros dois modelos apresentam uma precisão próxima aos resultados com o vetor completo, e também necessitam de menor quantidade de tempo para realizar todo o processamento. Desse modo, é visto para os modelos NB e MLP uma redução de 98.93% e 87.39% respectivamente. Para executar algoritmos de AP que requerem, diversas vezes, muito custo computacional, essa redução é vista como propícia para realizar testes de algoritmos e técnicas. Além disso, a avaliação da utilização de codificadores automáticos como redutor em bases de dados da língua portuguesa não tem muitas pesquisas exploratórias.

Diante do exposto, é observado que a precisão resultante pelos classificadores tradicionais com a redução proposta neste trabalho se aproxima da qualidade demonstrada nos algoritmos mais recentes no estado da arte, como apresentado na tabela 15. No qual, os algoritmos LSTM e *Bidirectional Encoder Representations for Transformers* (BERT) foram treinados através do *corpus kaggle* com a técnica *stemming*, como foi feito para os classificadores tradicionais.

Tabela 15 – Resultados do teste dos classificadores mais recentes, e com grande popularidade, LSTM e BERT treinados através do corpus *kaggle*

Classificador	**Tec. Inc**	**Precisão**	**Precisão Ponderada**	**Treino Tempo (Segundos)**
LSTM	Token Index	0.682	0.754	1841.404
BERT	BERTimbau	0.692	0.762	4699.119

Fonte: Elaboração própria.

Entre os algoritmos tradicionais, o que mais se aproximou dos resultados obtidos no LSTM com vetorização por índex[4] foi o modelo SVM, com a redução e também com o vetor original. Outro ponto que vale ressaltar, é a qualidade vista na técnica BERT, que

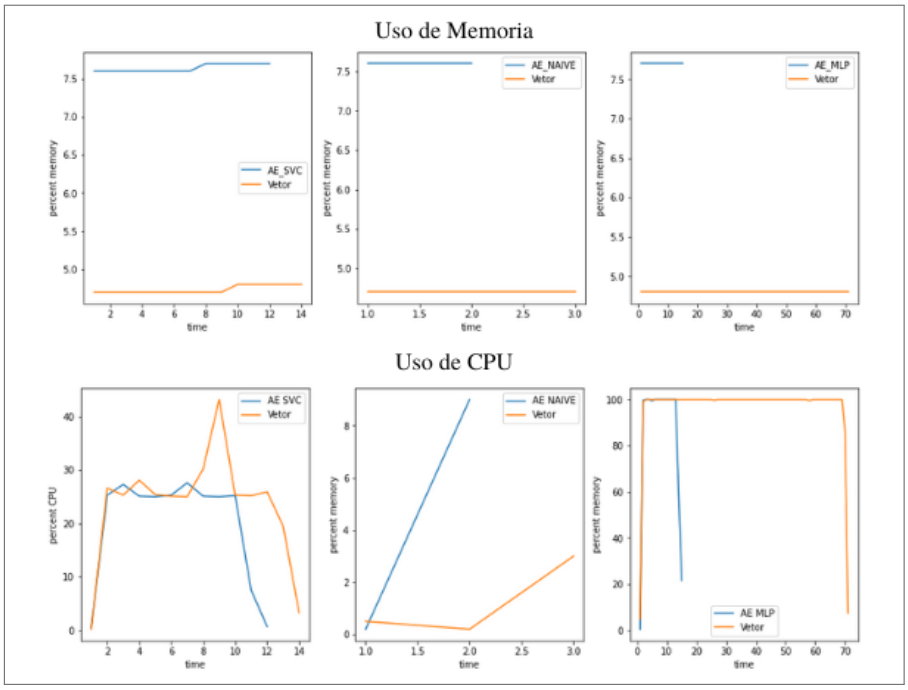
[4] A técnica *Tokenizer* disponibilizada através da biblioteca *tensorflow keras* que produz uma sequência de vetores indexados ou vetorizados. É uma técnica proposta no uso de LSTM com a linguagem python.

apesar de se sobressair ao LSTM necessitou de muito mais tempo de processamento para executar seu treinamento com 5 épocas.

Já os algoritmos tradicionais com a redução através do método proposto obtiveram precisões próximas as destes modelos e com menor quantidade de tempo de processamento necessário. É importante evidenciar que o ambiente de treinamento e teste de todas as técnicas e processos utilizados nesse trabalho foi o *kaggle* que é uma plataforma de estudo de dados compartilhado. No qual, o uso de memória e processador é cedido conforme a disponibilidade.

Além dos resultados vistos na tabela anterior, também foram armazenados os dados de custos de memória e uso da CPU durante o treinamento dos modelos. O mesmo também foi realizado no treinamento desses mesmos modelos para o vetor original. Para desse modo, poderem ser comparados os custos de ambas as tarefas nas duas perspectivas. A figura 10 apresenta o aumento do consumo de memória e CPU para os modelos treinados com o vetor e sua representação reduzida.

Figura 10 – Gráficos de comparação do custo computacional durante o treinamento dos classificadores com o vetor de representação produzido pela rede de codificadores automáticos e o vetor TFIDF. Medidas apresentadas em porcentagem de uso da memória e CPU. *corpus kaggle*



Fonte: Elaboração própria.

Assim, a figura 10 apresenta em seu gráfico a manutenção do alto uso de memoria e CPU com a redução da dimensão nos três modelos, porém o tempo necessário para realizar todo o treinamento é menor. Desse modo, é deduzido que, com a redução os algoritmos puderam utilizar-se de mais processamento nos dados estudados e assim ocupando mais recursos.

Isso, porquê é vista uma redução de cerca de 79% em seu tempo, porém se for com-

parado ao tempo necessário de treinamento dos codificadores automáticos com o tempo de realização do treinamento para o vetor completo, então não há grande vantagem para este processo. Somente ocorre uma vantagem, caso haja esse codificador já pré-treinado, sem a necessidade de realizar seu treino antes de aplicar os modelos.

É preciso estar atento a estes dados que foram capturados através da execução realizada pela plataforma *Kaggle*, que disponibiliza uma certa quantidade de memória e processamento de CPU por projeto. Para tentar capturar somente os valores referentes aos custos do processamento dos modelos, foi deletado do projeto, após sua utilização, as tabelas, *dataframes* e os codificadores.

Ameur, Jamoussi e Hamadou (2018) observaram, por experimentação, que a rede de codificadores automáticos com 1/5 da dimensionalidade produziu resultados melhores dentre os vistos. O resultado foi um vetor de 200 dimensões que alcançou um *f-score* de 74,27%. Considerando isso, também foi realizada uma experimentação com base na quantidade de nós por camada final, com o intuito de escolher o melhor dimensionamento que produzisse bons resultados. Os autores também afirmaram que:

> A partir de nossos resultados experimentais, podemos concluir que a taxa de redução não deve ter um valor alto, porque poucos nós na camada oculta levam a um alto erro de reconstrução. Além disso, em nosso caso, o número ideal de níveis de *autoencoder* varia entre 3 e 5.[5](AMEUR; JAMOUSSI; HAMADOU, 2018, p.1315, tradução nossa).

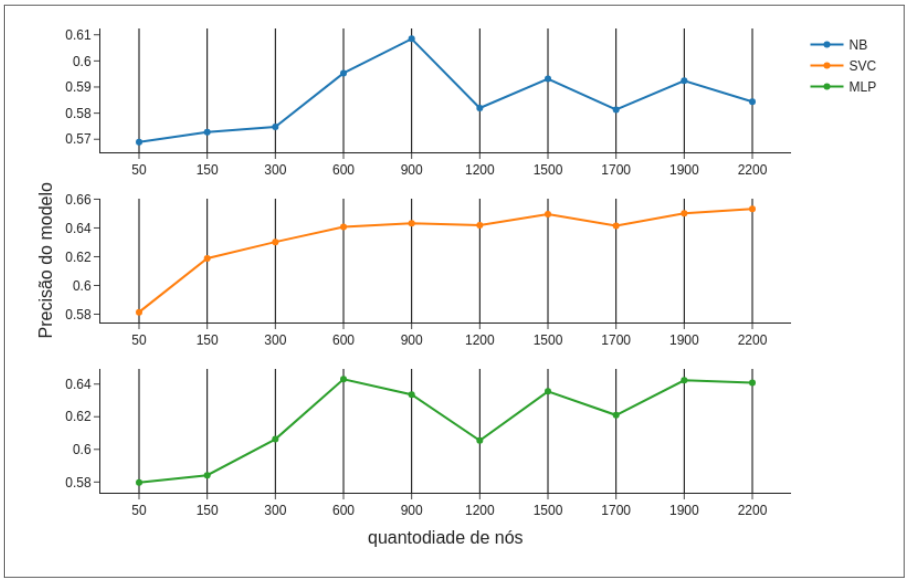
Então, com o intuito de validar esse argumento, e também obter o melhor codificador possível entre os experimentos, foram realizados testes com diferentes nós para 3, 4 e 5 camadas de codificadores. As RCA, com os parâmetros expostos, foram treinadas com os mesmos conjuntos de *corpus* de treino e teste utilizando a mesma técnica de vetorização. Para comparar os resultados entre os diferentes treinos nas diferentes camadas. Por fim, são aplicados 2 modelos, NB e SVM, para obter as métricas a serem comparadas.

O primeiro experimento foi realizado com as RCA de 5 níveis de camadas, e seus resultados são apresentados na figura 11. A partir deles, é visto que, existe uma melhora conforme os nós são aumentados. Porém, não é possível definir que com 1/5 dos nós (> 2200 nós) atinge o melhor resultado observado. Pois, é visto que há uma oscilação até 1200 nós, após se ver uma queda nos resultados e uma melhora nos dois últimos com maior quantidade de nós (1900 e 2200).

O codificador com 1500 nós no nível de saída apresenta um pico de 0.649 para o modelo SVM, porém os outros modelos, em relação aos demais resultados, apresenta uma queda de qualidade. O modelo NB apresenta seu pico com 900 nós na camada de saída, uma precisão de 0.608. Já o MLP apresenta sua melhor qualidade com 600 nós e uma precisão de 0.6429, no qual a imagem também apresenta um pico próximo a esse para o modelo com 1900 nós.

[5] From our experimental results, we can conclude that the reduction rate should not have a high value, because too few nodes in the hidden layer lead to a high reconstruction error. Besides, in our case the optimal number of auto-encoder levels ranges between 3 and 5. (AMEUR; JAMOUSSI; HAMADOU, 2018, p.1315)

Figura 11 – Gráficos com curvas da precisão do teste, dos classificadores escolhidos, para os vetores de representação produzidos por redes de codificadores automáticos com diferentes números de nós na camada de saída. Rede com 5 Camadas treinadas através do *corpus kaggle*



Fonte: Elaboração própria.

Entretanto, os resultados vistos no NB sofrem bastante oscilação, e apresentam (para alguns modelos) um aumento na qualidade a medida que se acresce a quantidade de nós. O modelo SVM apresenta uma curva sem oscilações, porém, os modelos não atingem boas qualidades com as mesmas quantidades de nós. Desse modo, para analisar um ponto em comum nos três modelos, será visto 900 nós na tabela 16 comparado com o valor total do vetor.

Ao escolher o codificador com 900 nós na camada final, apresentado na tabela 16, a diferença entre os seus resultados nos modelos testados com os obtidos no vetor original. É visto que a diferença entre as precisões obtidas estão entre 0.006 e 0.06, o modelo NB com a redução ultrapassa os resultados com o vetor original e o MLP se aproxima com uma diferença de 0.006.

Tabela 16 – Resultados do teste dos classificadores a partir do vetor de representação produzido pelo codificador automático com 900 nós na camada de saída comparados com os resultados de teste obtidos partir do vetor TFIDF[*n-gram*= (1,1)]. Rede de codificadores automáticos com 5 camadas (corpus *kaggle*)

Codificado	Dimensão	**Classificador**	**Precisão**	**Precisão Ponderada**	**Treino Tempo (Segundos)**
Sim	900	NB	0.608	0.641	0.035
Não	13049	NB	0.576	0.482	0.842
Sim	900	SVM	0.643	0.690	117.129
Não	13049	SVM	0.699	0.742	1508.023
Sim	900	MLP	0.633	0.674	27.312
Não	13049	MLP	0.638	0.685	176.792

Fonte: Elaboração própria.

Os resultados nesse experimento não atingiram uma qualidade superior as que foram vistas na tabela 15 com os modelos mais recentes. Porém, permaneceram próximos aos

obtidos por esses modelos, e superior aos do experimento com 128 nós na camada de saída do codificador. Assim, é percebida uma melhora com relação ao experimento anterior, e uma pequena aproximação dos resultados obtidos com o algoritmo LSTM.

Mesmo que com bons resultados, o intuito deste estudo é a redução dos custos e a avaliação da classificação pelo vetor reduzido comparado com o original. Diante do exposto, é observado que a medida que se aumenta a quantidade de nós na camada final, o vetor final se aproxima de seu estado inicial. Observar a viabilidade de reduções com menores quantidades é importante para dar a possibilidade de máquinas com menores poderes computacionais poderem executar tais modelos com menor custo. Além disso, o estudo do impacto na redução em um *corpus* da língua portuguesa, viabiliza estudos maiores para esse idioma.

Diante das discussões realizadas, é visto que o modelo continuar a melhorar a medida que aumenta a quantidade de nós e se aproxima de 1/5 da dimensionalidade para o modelo SVM e os outros modelos apresentam oscilações em seus resultados. Também é visto que não foram atingidos os melhores resultados com 5 níveis nas RCA, como observado na pesquisa que gerou esse experimento. Dois dos três modelos atingiram, mais de uma vez, resultados acima de 0.6 de precisão, antes e depois de 1/5 da dimensionalidade. Somente o modelo MLP teve uma melhora na métrica Precisão Ponderada.

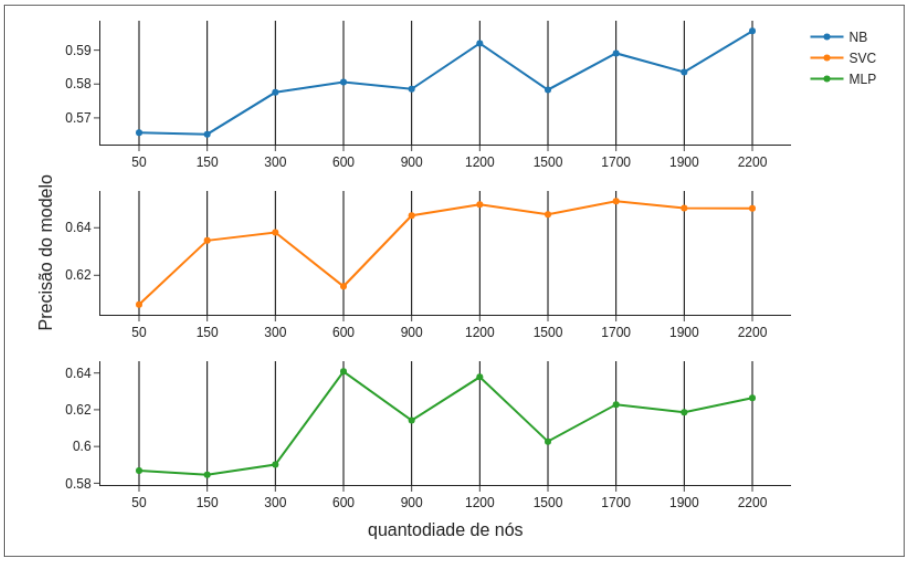
Assim como citado acima por Ameur, Jamoussi e Hamadou (2018), os melhores resultados observados estavam nos codificadores com 3 a 5 níveis com 1/5 da dimensionalidade como saída. Então, para validar nossos resultados, e o pressuposto por eles, foram realizados também experimentos para 3 e 4 níveis, visando obter o melhor codificador para o *corpus kaggle*.

Com 4 níveis na rede de codificadores, também é vista uma oscilação nos resultados da precisão. E, diferente dos experimentos anteriores, essa oscilação foi observada também no modelo NB. Assim como nos resultados com 5 níveis, também não foi visto o melhor resultado próximos a 1/5 da dimensão. Além disso, a rede de codificadores com maior quantidade de nós no nível de saída obtém os melhores resultados em 2 modelos, e alcança um alto resultado no modelo restante. Todas essas características são apresentadas no gráfico da figura 12.

Ao observar o gráfico, é visto que os resultados com 4 níveis atingiram precisão próxima as observados com 5 níveis. A oscilação foi menor que a vista com o experimento anterior, e o modelo SVM que antes apresentou uma melhor precisão com 600 nós, atingiu sua pior qualidade com essa quantidade. O modelo MLP repetiu sua melhor precisão 1200 nós, então, de modo a escolher um ponto em comum entre os três modelos, foi escolhido essa quantidade.

Desse modo, será analisado a quantidade de 1200 nós, entre os modelos, para se comparar aos resultados obtidos com o vetor original, e validar se há uma melhora ou viabilidade de seu uso. A tabela 17 nos apresenta esses valores a serem comparados.

Figura 12 – Gráficos com curvas da precisão do teste, dos classificadores escolhidos, para os vetores de representação produzidos por redes de codificadores automáticos com diferentes números de nós na camada de saída. Rede com 4 Camadas treinadas através do *corpus kaggle*



Fonte: Elaboração própria.

Tabela 17 – Resultados do teste dos classificadores a partir do vetor de representação produzido pelo codificador automático com 1200 nós na camada de saída comparados com os resultados de teste obtidos partir do vetor TFIDF[*n-gram*= (1,1)]. Rede de codificadores automáticos com 4 camadas (corpus *kaggle*)

Codificado	Dimensão	**Classificador**	**Precisão**	**Precisão Ponderada**	**Treino Tempo (Segundos)**
Sim	1200	NB	0.591	0.646	0.047
Não	13049	NB	0.576	0.482	0.842
Sim	1200	SVM	0.649	0.698	203.557
Não	13049	SVM	0.699	0.742	1508.023
Sim	1200	MLP	0.637	0.680	52.237
Não	13049	MLP	0.638	0.685	176.792

Fonte: Elaboração própria.

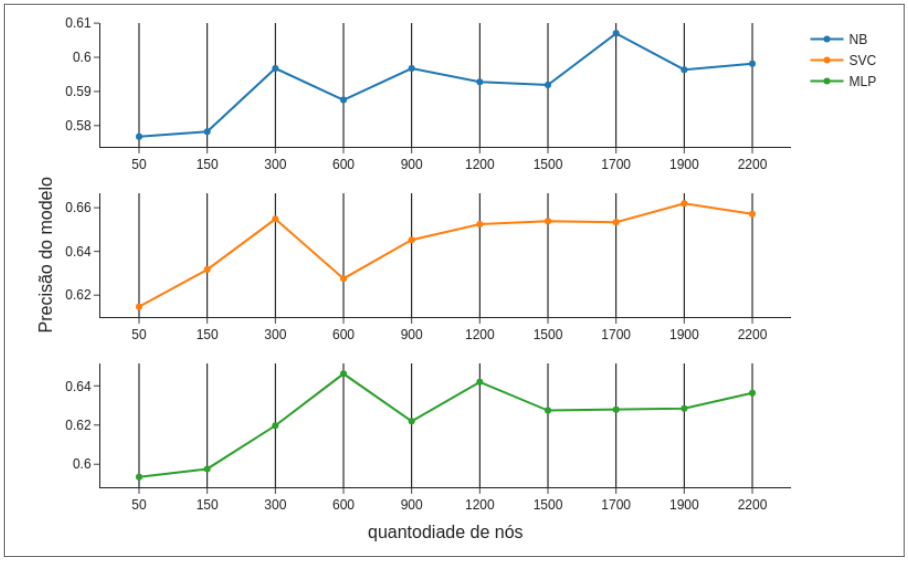
Os resultados dos modelos MLP e SVM foram melhores que os vistos no experimento com 5 níveis e também atingiu valores próximos aos que estavam utilizando o vetor original. A diferença entre os vetores foi de menos de 0.01 (para o MLP) e a precisão ponderada permaneceu superior à do vetor original. Por outro lado, o modelo NB e SVM demonstraram serem equiparáveis e com tempo computacional menor necessário para aprender. Porém, esse utilizado, não foi o de melhor resultado nesse experimento com 1200 nós, atinge uma precisão de 0.591 que é superior à do modelo treinado com o vetor original.

Entretanto, esses experimentos buscam encontrar uma rede de codificadores com melhor resultado igual para todos os modelos. Ainda assim, os resultados entre os dois modelos são bons, pois alcançam valores acima de 0.6 e se aproximam dos resultados vistos no vetor original. A distância entre esses vetores é de 0.01 e 0.05 para os modelos NB e SVM, respectivamente. O modelo NB, apesar de não ultrapassar 0.6 na precisão, teve uma melhora no seu resultado com o vetor reduzido. Porém, o modelo que mais reduziu o

seu tempo necessário para o aprendizado foi o NB com apenas 5.582% do tempo original, no SVM houve uma redução de aproximadamente 86.50% do seu tempo.

A partir desses resultados, os experimentos com 4 níveis mostram melhor aproximação que o de 5 níveis para os modelos MLP e SVM, porém, comparando com o modelo LSTM, é observado uma maior aproximação dos seus resultados, mas, ainda assim, não superando sua qualidade no teste. Para concluir os experimentos, foram executados os treinamentos dos modelos com rede de codificadores de 3 níveis. O gráfico de precisão é apresentado na figura 13, no qual mostra novamente oscilações nos resultados, sem uma tendência clara.

Figura 13 – Gráficos com curvas da precisão do teste, dos classificadores escolhidos, para os vetores de representação produzidos por redes de codificadores automáticos com diferentes números de nós na camada de saída. Rede com 3 Camadas treinadas através do *corpus kaggle*



Fonte: Elaboração própria.

Conforme o gráfico, o modelo MLP alcança o melhor resultado com 600 nós na camada de saída, porém o NB e SVM apresentas seus melhores resultados com 1700 e 1900 respectivamente. Os valores alcançados pelos outros dois modelos são bastante próximos aos obtidos no experimento com 4 camadas. Desse modo, demonstra que o único modelo que apresentou uma real diferenciação no seu aprendizado foi o MLP. Para comparar os resultados foi escolhido 1700 nós na camada de saída por ser a quantidade em que os 3 modelos apresentaram uma precisão próxima sem uma queda.

A maior divergência entre esses experimentos, é o seu tempo computacional. Á medida que se diminui as camadas, é menor o tempo necessário para os algoritmos alcançarem o aprendizado. Assim, demonstra ter uma influência direta sobre esta métrica, toda a mensuração é exibida na tabela 18.

Conforme a tabela 18, esse experimento alcançou resultados muito próximos aos que foram vistos no experimento de 4 níveis. Porém, além do NB produzir qualidade melhor com a redução, nesse também o MLP alcança esse resultado, ultrapassando a qualidade obtida com o vetor original. O tempo computacional foi reduzido, sendo a

Tabela 18 – Resultados do teste dos classificadores a partir do vetor de representação produzido pelo codificador automático com 1700 nós na camada de saída comparados com os resultados de teste obtidos partir do vetor TFIDF[*n-gram*= (1,1)]. Rede de codificadores automáticos com 3 camadas (corpus *kaggle*)

Codificado	Dimensão	**Classificador**	**Precisão**	**Precisão Ponderada**	**Treino Tempo (Segundos)**
Sim	1700	NB	0.607	0.660	0.117
Não	13049	NB	0.576	0.482	0.842
Sim	1700	SVM	0.653	0.704	337.910
Não	13049	SVM	0.699	0.742	1508.023
Sim	1700	MLP	0.628	0.684	44.417
Não	13049	MLP	0.638	0.685	176.792

Fonte: Elaboração própria.

principal diferença entre os experimentos. No experimento anterior, a redução no tempo do modelo SVM era de 86.50%, nesse experimento a redução foi de aproximadamente 96.48990%.

Portanto, com esses resultados, se compreende que a sugestão dos nós e níveis para o *corpus* e formato de aplicação dos codificadores automáticos não se aplica. Além de um aumento na qualidade dos modelos, se comparado à primeira rede de codificadores proposta com 6 níveis, não apresentaram menor custo de tempo e processamento computacional. A partir dos 3 experimentos realizados, foi escolhido utilizar 4 níveis no codificador, e a quantidade de nós na camada de saída escolhida foi de 600.

Essa escolha foi feita com base na questão da redução dos custos, pois utilizar a maior quantidade de nós, por ter o melhor resultado, não irá atingir o objetivo deste trabalho. Visto que, o intuito da presente pesquisa, é apresentar uma proposta que tenha qualidade e melhore os custos computacionais, avaliando seus resultados baseado em um *corpus* formulado com o tema da COVID-19 no Brasil. Todas as tabelas e gráficos com as métricas obtidas nos experimentos com o *corpus kaggle*, estão disponíveis no apêndice G.

Desse modo, os parâmetros dos codificadores e técnicas de incorporação de palavras, para a base de dados procedente do *kaggle* foi definida. Mesmo com os resultados para a base anterior, foi necessário realizar para o segundo *corpus*, pois foi notado que, para a base oriunda do *kaggle*, os resultados eram muito abaixo, próximos a 70% de precisão, com os melhores classificadores sem a redução. Então, para examinar se essas eram características somente dessa base, os mesmos testes foram aplicados ao *corpus* TBCOV.

5.1.2.2 Corpus *Two Billion Multilingual COVID-19* (TBCOV)

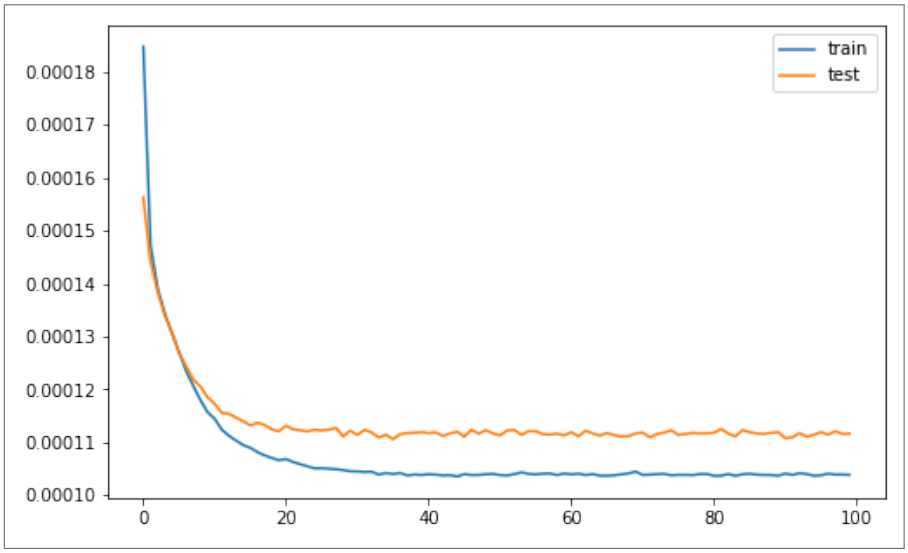
Assim como no *corpus* anterior, foi iniciada a aplicação da rede de codificadores com 6 níveis e 128 nós na camada de saída. Vale destacar que, com o TBCOV, os modelos treinados com o vetor original tiveram resultados acima dos que foram vistos no *dataset kaggle*. No qual, suas métricas vistas obtiveram, em sua máxima, 0.829 de precisão e 0.949 de precisão ponderada, para o vetorizado TFIDF com $n\text{-}gram = (1, 1)$ e utilizando

a técnica de redução *stemming*. Pois a técnica com os parâmetros citados, produz um vetor com uma dimensionalidade de 5591 características por texto. Essas características são as palavras reduzidas à raiz, tratadas como colunas no vetor.

Desse modo, esse vetor foi o escolhido para produzir os experimentos deste subcapítulo. O treino da rede de 6 níveis, alcançou sua tendência antes da época 40. Pois, em seus resultados há uma perda próxima de 0.000104 e 0.000111 no treino e teste, respectivamente. Essas RCA possuem como ativação a função Linear e a ReLu, por serem as mais utilizadas em estudos envolvendo redes neurais.

A figura 14 apresenta o gráfico da curva de aprendizado do codificador de 6 níveis. No qual seu treino começa com uma perda no aprendizado maior que a validação, após algumas épocas esses papeis se invertem. Porém, os valores atingidos permanecem bastante próximos, demonstrando o aprendizado da rede. Para realizar todo o treinamento dela, foi necessário 1 hora e 5 minutos de execução. Entretanto, quando treinada com o vetor gerado com o $n\text{-}gram = (1, 2)$, foi observada a necessidade de 3 horas e 40 minutos para concluir todo o processamento, recebendo um acréscimo de mais de 300%.

Figura 14 – Gráfico de tendência de perda obtida durante o aprendizado da rede de codificadores automáticos para treino e teste. Realizado com a redução de dimensionalidade até 128 com as funções de validação Linear e ReLu, treinadas com o vetor TFIDF, produzido a partir do *corpus* TBCOV



Fonte: Elaboração própria.

Além da diferença entre os tempos de processamento, o gráfico de aprendizado da rede de codificadores treinado com o vetor com $n\text{-}gram = (1, 2)$, no apêndice H, não apresenta que foi atingido a inclinação de aprendizado. É visto que, após 100 épocas, a rede ainda poderia alcançar menores perdas na geração de representações do vetor. Desse modo, a utilização da resolução obtida com $n\text{-}gram = (1, 1)$ foi escolhida para avaliar comparativamente com as métricas observadas, utilizando o vetor original.

Essa comparação é mostrada na tabela 19, no qual é observado a redução da dimensionalidade impacta fortemente no tempo de processamento dos algoritmos. No qual, há

uma redução no tempo necessário para todo o processamento de todos os modelos. Porém, em relação à precisão alcançada, existe uma diferença baixa entre o uso da redução e o vetor original. Sendo assim, todos os modelos treinados com o vetor reduzido possui uma precisão bastante próxima às alcançadas com o vetor original, com menos de 1% de diferença, e com um menor tempo de processamento.

Tabela 19 – Resultados do teste dos classificadores a partir do vetor de representação produzido pelo codificador automático com 128 nós na camada de saída comparados com os resultados de teste obtidos partir do vetor TFIDF[*n-gram*= (1,1)]. Rede de codificadores automáticos com 6 camadas (corpus TBCOV)

Reduzido	**Classificador**	**Precisão**	**Precisão Ponderada**	**Treino Tempo (Segundos)**
Sim	NB	0.654	0.672	0.008
Não	NB	0.653	0.525	0.815
Sim	SVM	0.754	0.830	15.103
Não	SVM	0.824	0.909	798.485
Sim	MLP	0.749	0.830	17.835
Não	MLP	0.787	0.889	104.683

Fonte: Elaboração própria.

Outra métrica que se mantém próxima é a Precisão Média(*Averaged Precision*), no qual, o modelo NB apresentou um maior valor com a redução, e o SVM e MLP uma diferença de menos de 2%. Desse modo, se observa através das métricas apresentadas na tabela 19 que o uso da redução ocasionou um impacto positivo em relação ao custo processual e também ao índice de precisão das classes.

Ao comparar os resultados dos modelos tradicionais com os que foram vistos acima, no LSTM e BERT através da tabela 20, é apresentado o uso da técnica de redução proposta que expõe uma qualidade próxima aos resultados alcançados através do teste dos dois algoritmos. Apesar das técnicas de incorporação serem diferentes das utilizadas, a qualidade dos resultados se equiparam quando é observado o modelo MLP e SVM com a redução, já quando é analisado com o vetor original, os modelos tradicionais se igualam.

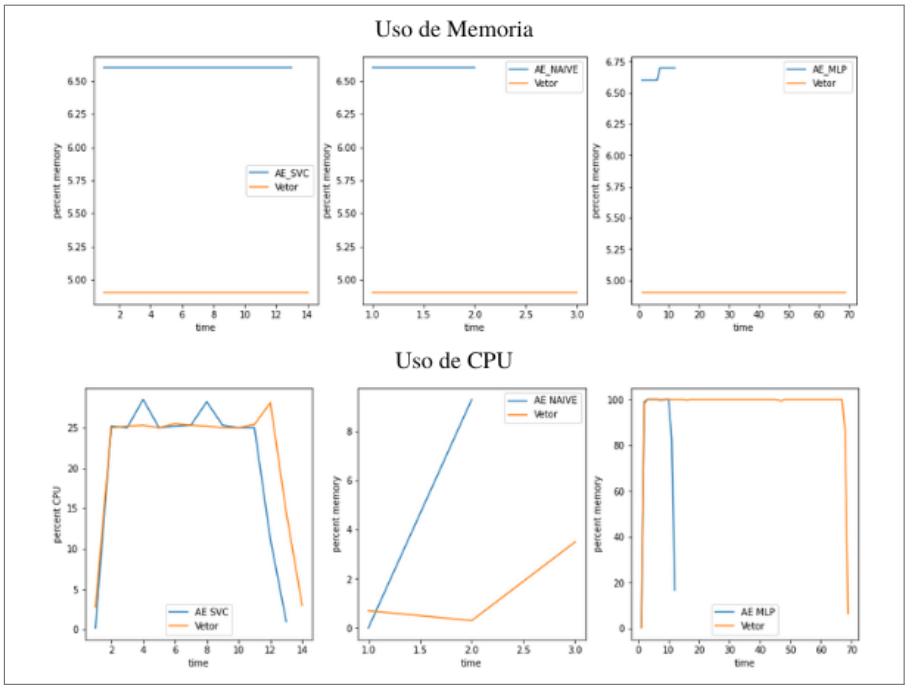
Tabela 20 – Resultados do teste dos classificadores mais recentes, e com grande popularidade, LSTM e BERT treinados através do corpus TBCOV

Classificador	**Tec. Inc**	**Precisão**	**Precisão Ponderada**	**Treino Tempo (Segundos)**
LSTM	Token Index	0.804	0.890	231.635
BERT	BERTimbau	0.810	0.906	4768

Fonte: Elaboração própria.

Quando analisamos os custos computacionais durante o aprendizado dos modelos, nos gráficos da figura 15, é exposto que apesar da redução do tempo computacional, o custo de memória com a representação reduzida do vetor é maior que com o vetor original. Por outro lado, quando é observado o uso de CPU durante esse período de aprendizado, é visto que se mantêm na mesma média de uso.

Figura 15 – Gráficos de comparação do custo computacional durante o treinamento dos classificadores com o vetor de representação produzido pela rede de codificadores automáticos e o vetor TFIDF. Medidas apresentadas em porcentagem de uso da memória e CPU. *corpus* TBCOV



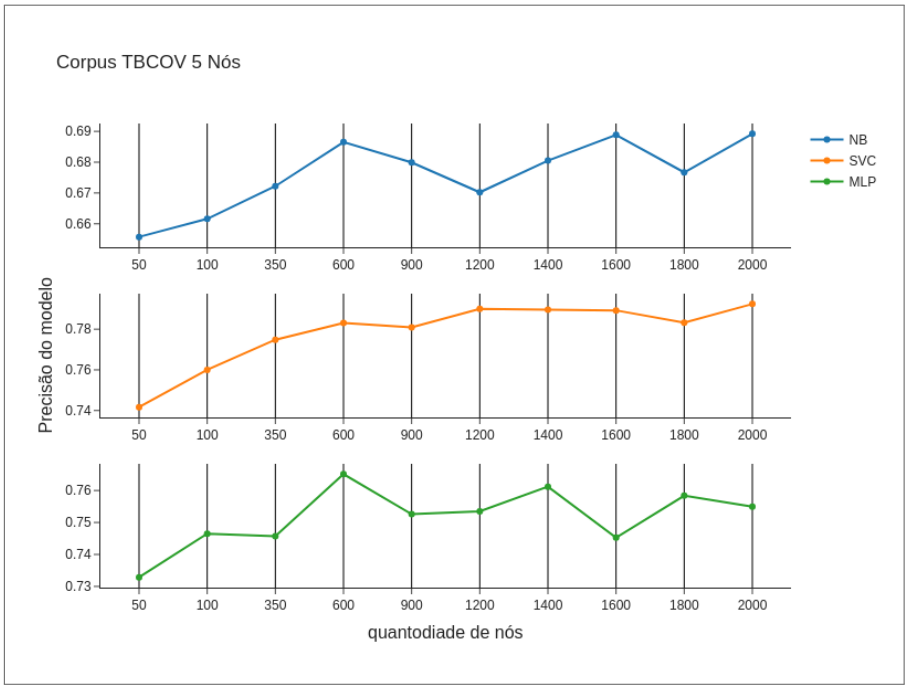
Fonte: Elaboração própria.

Em vista disso, a utilização desses codificadores se torna factível por demonstrar a redução esperada do tempo de processamento e também se manter com a qualidade próxima às apresentadas no vetor original. Dessa forma, não reduzindo o poder do *corpus* de aprendizado e demonstrando a utilidade e benefícios da redução mesmo se aplicados a um *corpus* na língua portuguesa. Porém, assim como no *corpus* anterior, foram realizados experimentos com diferentes níveis de codificadores para diferentes quantidades de nós de saída.

Os experimentos serão três conjuntos de treinamento dos modelos (com três diferentes níveis em cada) com as saídas variando em quantidades de 50 nós até 2000. Esse limite é definido com base na menor quantidade de nó antes da camada de saída, que era de 2500 nós. A partir desses valores foram gerados gráficos de tendência e tabela de resultados contendo as métricas precisão, *recall*, f1, acurácia e precisão ponderada e sendo disponibilizados no apêndice H (A tabela de todos os experimentos com esse corpus).

O primeiro experimento foi realizado com 5 níveis de camadas e as 10 diferentes quantidades de nós na camada final. Primeiramente, serão observados os seus resultados através do gráfico na figura 16, que apresenta os resultados para os testes com os nós baseados na precisão obtida para os três modelos escolhidos para a validação. Assim como visto nos experimentos do *corpus* anterior, esses resultados não apresentaram melhoria ao atingir 1/5 da dimensão do vetor original. Os valores com maior qualidade também foram vistos no vetor com maior quantidade de nós na saída.

Figura 16 – Gráficos com curvas da precisão do teste, dos classificadores escolhidos, para os vetores de representação produzidos por redes de codificadores automáticos com diferentes números de nós na camada de saída. Rede com 5 Camadas treinadas através do *corpus* TBCOV



Fonte: Elaboração própria.

Além disso, é mostrado, através do gráfico, diversas oscilações nos resultados, apenas o modelo SVM produz uma curva mais clara. Entretanto, essa curva não chega a encontrar um pico e então volta a decrescer, como nos experimentos de Ameur, Jamoussi e Hamadou (2018). Desse modo, demonstra que, como no *corpus* anterior, a redução do vetor a 1/5 da sua dimensionalidade, com os processos adotados neste trabalho, ainda não apresentam o melhor resultado.

Assim, para poder validar os resultados, foram comparadas a precisão, PA e o tempo de processamento na tabela 21, como feito nos experimentos do subcapítulo anterior. Essa comparação é feita com base na rede de codificadores que demonstrou os melhores resultados em predominância dos modelos. Diante das discussões realizadas, a rede de codificadores escolhida foi a que possui na camada de saída 600 nós de camadas, pois é apresentam boa qualidade de precisão nos três modelos. A outra rede também alcançou bons resultados, porém a que possuiu maior predominância de bons resultados nos modelos, foi a escolhida para essa comparação.

A partir dessa tabela, é visível que, assim como no *corpus* kaggle, o modelo NB teve um maior aumento de sua qualidade. Assim, o acréscimo na precisão foi superior a 0.01, já nos outros dois modelos houve uma perda de menos de 0.05. O modelo que mais se aproximou com a redução da dimensionalidade dos modelos LSTM e BERT foi o MLP, atingindo uma qualidade de 0.783. Ao observar a especifidade dos modelos através da precisão media, o modelo NB é o que representa uma melhor melhoria em seu resultado.

Tabela 21 – Resultados do teste dos classificadores a partir do vetor de representação produzido pelo codificador automático com 1600 nós na camada de saída comparados com os resultados de teste obtidos partir do vetor TFIDF[*n-gram*= (1,1)]. Rede de codificadores automáticos com 5 camadas (corpus TBCOV)

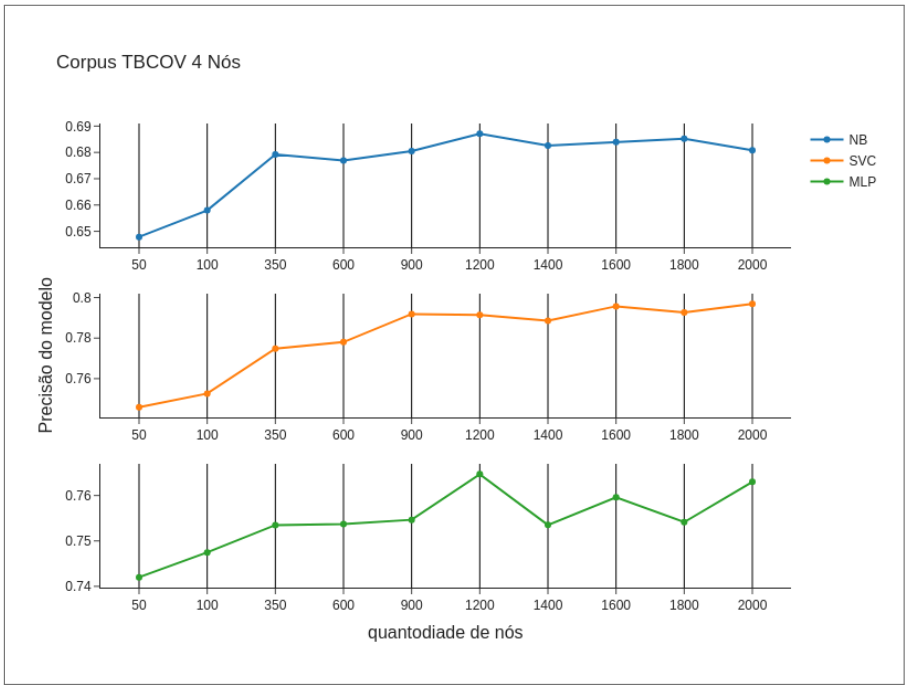
Codificado	Dimensão	**Classificador**	**Precisao**	**Precisão Ponderada**	**Treino Tempo (Segundos)**
Sim	600	NB	0.686	0.594	0.023
Não	9741	NB	0.653	0.525	0.815
Sim	600	SVM	0.783	0.779	46.951
Não	9741	SVM	0.824	0.909	798.485
Sim	600	MLP	0.765	0.752	63.208
Não	9741	MLP	0.787	0.889	104.683

Fonte: Elaboração própria.

O modelo SVM apresentou uma maior redução no tempo de processamento comparado ao obtido pelo mesmo com o vetor original. O gráfico de custos computacionais (como uso de memória e CPU) estão todos expostos no apêndice H, e demonstram que, à medida que utilizamos maiores quantidades de nós na camada de saída, esse custo aumenta.

O experimento seguinte é o que utiliza as redes de codificadores com 4 níveis, a mesma faixa de quantidade de nós na camada de saída. No gráfico da figura 17 é apresentado a curva da precisão alcançada a partir de cada um dos modelos aplicados às representações de menor dimensionalidade geradas com as dez redes de codificadores.

Figura 17 – Gráficos com curvas da precisão do teste, dos classificadores escolhidos, para os vetores de representação produzidos por redes de codificadores automáticos com diferentes números de nós na camada de saída. Rede com 4 Camadas treinadas através do *corpus* TBCOV



Fonte: Elaboração própria.

A partir desse gráfico é exposto, que o algoritmo MLP produz a maior oscilação entre os utilizados. Ao gerar uma tendência inicial e logo após 1200 nós, apresenta uma grande

oscilação com dois picos próximos ao primeiro, visto na tendência inicial. Já os dois outros algoritmos, apresentam uma curva mais clara que, porém, não encontra um pico, produzindo o resultado de maior precisão na rede de codificadores com maior quantidade de nós na camada de saída.

Como já foi discutido, ao alcançar 1200 nós, os modelos NB e MLP não produzem os seus melhores resultados, visto que, se trata de 1/5 da dimensão do corpus. Após esse valor, é observada uma melhora nos resultados do modelo SVM, mostrando a existência da possibilidade de que valores ainda maiores, alcançariam melhores resultados. Porém, aumentar a dimensionalidade iria tornar a tarefa de redução desnecessária, visto que, o objetivo não é encontrar resultados melhores que o vetor original, e, sim, manter uma precisão razoável e reduzir os custos computacionais.

Através da imagem é apresentado que o pico dos valores experimentados foi em 1200 nós na camada de saída para os modelos NB e MLP. O MLP produz uma grande oscilação, no qual sua precisão com 2000 nós, está abaixo dos vistos com 1200 nós. Nos modelos SVM e NB também é alcançado precisão de qualidade com essa quantidade de nós, tornando essa rede de codificadores, a escolhida para a tabela de comparação.

A tabela 22 detalha comparadamente o melhor codificador com o vetor TFIDF. Assim é identificado o acréscimo no tempo de treinamento dos modelos. Todos os modelos realizaram todo o processamento com menor tempo de processamento, porém o MLP foi o que se aproximou mais em tempo. O NB foi o único modelo que ultrapassou em qualidade de precisão, já os outros ficaram próximos aos resultados alcançados com o vetor original.

Tabela 22 – Resultados do teste dos classificadores a partir do vetor de representação produzido pelo codificador automático com 1200 nós na camada de saída comparados com os resultados de teste obtidos partir do vetor TFIDF[*n-gram*= (1,1)]. Rede de codificadores automáticos com 4 camadas (corpus TBCOV)

Codificado	Dimensão	**Classificador**	**Precisão**	**Precisão Ponderada**	**Treino Tempo (Segundos)**
Sim	1200	NB	0.687	0.587	0.060
Não	9741	NB	0.653	0.525	0.815
Sim	1200	SVM	0.791	0.785	123.949
Não	9741	SVM	0.824	0.909	798.485
Sim	1200	MLP	0.764	0.751	54.708
Não	9741	MLP	0.787	0.889	104.683

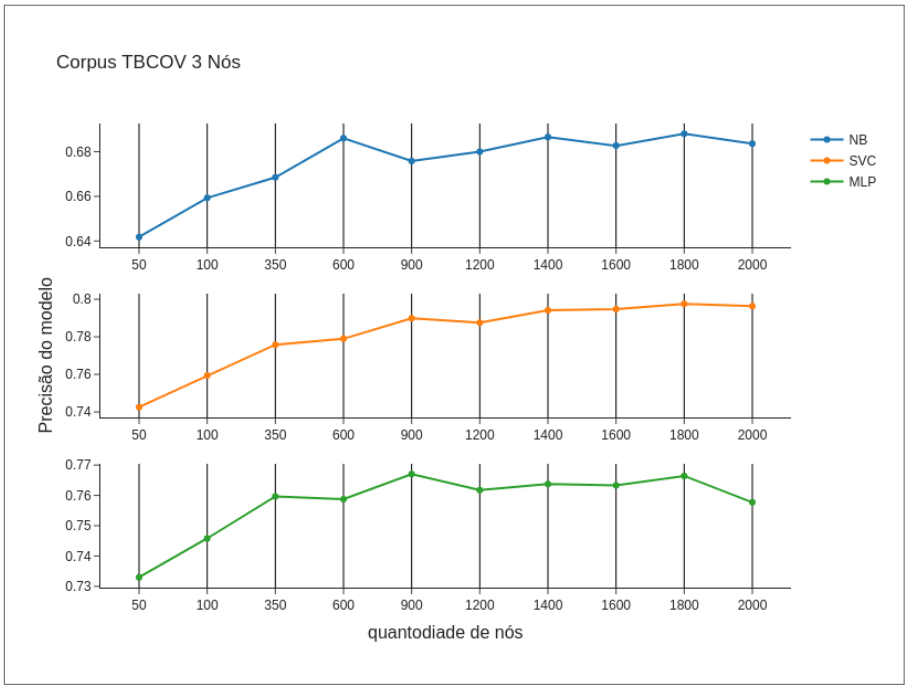
Fonte: Elaboração própria.

Conforme a tabela 22 é observado que nessa rede de codificadores os valores foram abaixo dos que foram vistos no experimento anterior. Além disso, houve um aumento no tempo de processamento do algoritmo MLP, que antes precisou de 24 segundos e nesse experimento necessitou de 27. A especificidade dos modelos também resultou em valores próximos. Assim como no experimento anterior, foi observado que somente o modelo SVM mais se aproximou dos algoritmo LSTM e BERT da tabela 20. Desse modo, sendo concluído, não haver grande vantagem ou desvantagem em relação ao experimento realizado com 5 camadas.

Diante das discussões realizadas, é visto que, a rede de codificadores que produziu os melhores resultados nesse experimento demonstrou resultados melhores de precisão que os observados com 5 camadas para o modelo SVM. Já os outros modelos resultaram em uma precisão bastante próxima do experimento com 5 níveis. Porém, é visto um acréscimo de tempo no modelo por se ter uma quantidade de nós maior. Desse modo, esse experimento ultrapassa em tempo computacional e se iguala em qualidade aos observados no da tabela 21, cabendo então, realizar os experimentos do último conjunto de codificadores com 3 níveis, para poder analisar, se superam os resultados vistos com 5 níveis de camada na rede de codificadores.

No experimento com 3 camadas ocultas os resultados foram próximos aos vistos com 4 e 5 níveis, mas, ainda assim, alcançaram valores maiores nas métricas observadas. A evolução da faixa de nós testada, demonstram uma inclinação no modelo SVM, no qual encontra seu pico com 1800 nós, e uma oscilação nos demais modelos. Os gráficos que contêm a tendência dos modelos é apresentado na figura 18.

Figura 18 – Gráficos com curvas da precisão do teste, dos classificadores escolhidos, para os vetores de representação produzidos por redes de codificadores automáticos com diferentes números de nós na camada de saída. Rede com 3 Camadas treinadas através do *corpus* TBCOV



Fonte: Elaboração própria.

Assim como nos demais experimentos, o modelo MLP apresenta grande oscilação, não podendo ser visto um único pico ou uma tendência clara. O NB mostra uma tendencia após 600 nós na camada de saída, apresentando valores de qualidade abaixo das vistas com esta quantidade. Porém, no SVM é visto um pico em 1600 e uma tendencia de queda que se repete entre 600/1400 e 1400/2000. Buscando a quantidade de nós que apresenta maior qualidade em comum, é escolhido o valor de 1800 para ser analisado na tabela 23.

A comparação é exibida na tabela 23, é visível que houve uma pequena melhora na precisão que os aproxima do obtido com o vetor original. Assim como nos experimentos anteriores, somente o modelo SVM alcança e ultrapassa o algoritmo LSTM, e também o MLP se aproxima ainda mais do resultado. Porém, o tempo de processamento dos algoritmos tradicionais com o vetor representação se aproxima do que foi visto no LSTM, no qual, inicialmente, o tempo era menor e a qualidade era próxima.

Tabela 23 – Resultados do teste dos classificadores a partir do vetor de representação produzido pelo codificador automático com 1800 nós na camada de saída comparados com os resultados de teste obtidos partir do vetor TFIDF[*n-gram*= (1,1)]. Rede de codificadores automáticos com 3 camadas (corpus TBCOV)

Codificado	Dimensão	**Classificador**	**Precisao**	**Precisão Ponderada**	**Treino Tempo (Segundos)**
Sim	1800	NB	0.688	0.596	0.094
Não	9741	NB	0.653	0.525	0.815
Sim	1800	SVM	0.797	0.791	208.033
Não	9741	SVM	0.824	0.909	798.485
Sim	1800	MLP	0.766	0.762	49.531
Não	9741	MLP	0.787	0.889	104.683

Fonte: Elaboração própria.

Além disso, é visto um aumento no tempo de treinamento para o modelo SVM e MLP de 1246% e 257%, respectivamente. Esse aumento no tempo de treinamento demonstra que o modelo precisou de mais processos necessários, até culminar no seu resultado. Ainda assim, os resultados se mantêm próximos e viáveis para o uso e como são melhores que os de 4 e 5 camadas, poderão ser utilizados para a classificação dos modelos.

A partir dos resultados obtidos nos quatro experimentos realizados acima, foi decidido prosseguir com a rede de codificadores com 6 camadas e 128 nós na dimensionalidade da representação do vetor. Essa escolha foi realizada, pois os resultados obtidos não apresentam grandes vantagens em sua qualidade e ocorre um acréscimo no tempo de processamento. Visto que, o intuito deste trabalho é reduzir os custos computacionais e validar os resultados sobre um *corpus* real, então, não seria útil manter uma representação de maior dimensionalidade.

Diante da representação do vetor com 128 de dimensionalidade, serão realizados treinos de modelos mais recentes, que também serão comparados os custos necessários entre a representação do vetor e o original. Além disso, será criado um comitê de classificadores para buscar uma melhor qualidade e, dessa forma, utilizar para classificar o *corpus* formulado para este trabalho, proveniente do *Twitter*.

Esse comitê será composto pelos modelos que representam as melhores qualidades baseadas na precisão e que tenha os melhores custos computacionais, de tempo e alocação de recursos. Por fim, esse comitê será comparado com os resultados de modelos mais recentes (como BERT, LSTM, *CatBoost*, *XGBoost*, etc) que tenham demonstrado bons resultados em outros estudos. Assim, uma análise será realizada para apresentar as vantagens e

desvantagens do uso dos codificadores automáticos na etapa de pré-processamento, com o intuito de reduzir a dimensionalidade de vetores de textos.

5.1.3 Algoritmos e Comitês classificadores

Com base nos resultados alcançados pelos experimentos de nós e camadas, o *corpus* que produziu maiores qualidades foi o TBCOV, e também alcançou resultados próximos quando foram utilizados os codificadores sob o vetor. Desse modo, a análise dos modelos aqui serão todas baseadas nesse *corpus*, buscando comparar o vetor original e a representação de menor dimensionalidade gerada pela rede de codificadores.

Os algoritmos escolhidos serão os principais utilizados para problemas multi-classes, visto que, as classes deste trabalho serão três: positivo, neutro e negativo. Dentre os mais utilizados que foram observados no estado da arte, estão a máquina de vetores de suporte (SVM), a rede neural perceptron multicamadas (do inglês *Multilayer perceptron*) e os mais clássicos como NB e árvores de decisões. Alguns outros algoritmos, mais atuais, que também serão tratados nessa etapa são o *CatBoost* e o *XGBoost*, por demonstrarem bons resultados em trabalhos mais atuais.

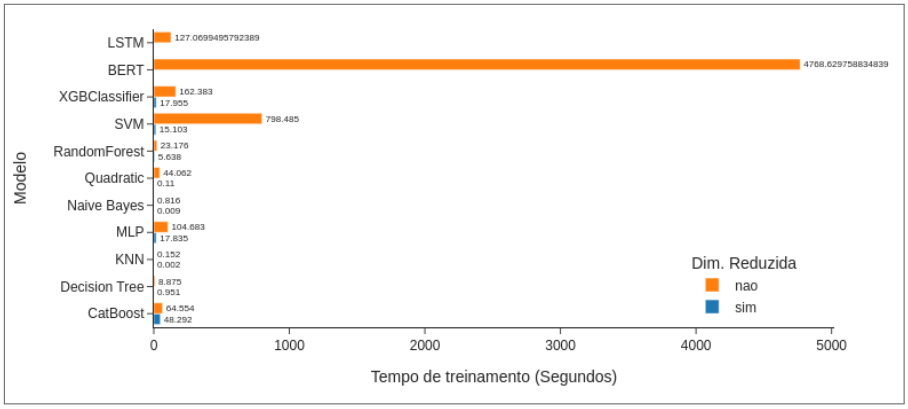
Inicialmente, foi realizada uma análise do tempo necessário de treinamento de cada algoritmos, utilizando o vetor original e sua representação reduzida. Para, assim, poder identificar o impacto dessa redução sobre os custos computacionais dos modelos. A partir disso, será discutido a respeito da qualidade alcançada e avaliado se a técnica proposta oferece vantagem ou desvantagem ao ser utilizada.

O treinamento e teste foi realizado a partir da ferramenta *cross_validade*, disponibilizada pela biblioteca *python scikit-learn*. Esse algoritmo possibilita a realização de validações cruzadas, que realizam a re-amostragem dos dados, para gerar métricas escolhidas baseadas no número de *k-folds* que determina quantas vezes o modelo será treinado com diferentes amostras da mesma base de dados. Entre as métricas resultantes do processamento dessa ferramenta, há o *fit_time* (Tempo de treinamento) que é todo o tempo necessário para cada treinamento com as diferentes amostras. A partir dessa métrica foi gerado o gráfico da figura 19 que informa o tempo médio necessário para cada modelo ser treinado.

A partir do gráfico de barra exposto na figura 19, é visto o tempo médio dos algoritmos treinados com o vetor original e a sua representação de dimensionalidade reduzida. As barras apresentadas na cor azul representam os modelos treinados com o vetor codificado. Também, é exibido no gráfico de barras alguns modelos que não demonstraram tanta vantagem com o uso dos codificadores, porém esses já eram classificadores que demonstram necessitar de pouco custo computacional para realizar seus treinamentos.

Por outro lado, modelos como SVM, *XGboost* e MLP que apresentavam uma necessidade de uma maior quantidade de processamento para realizar todo o treino, demonstrou uma redução drástica. Assim, o SVM precisou de 13 minutos para classificar os modelos

Figura 19 – Tempo de Treinamento dos classificadores, Vetor Original X Vetor de Representação produzido pelo codificador automático



Fonte: Elaboração própria.

com o vetor original, teve uma redução de aproximadamente 97% do tempo médio com sua representação reduzida. Já o *XGBoost* que apresentou um tempo médio de 3 minutos por amostra, teve uma redução de aproximadamente 90% de seu tempo.

Alguns outros algoritmos como a *Random Forest*, conhecido por ser um exemplo de comitê de classificador, produziu tempos baixos, porém como foi executado com os parâmetros padrões, sendo observado que isso influenciou diretamente no tempo. Essa interferência pode ser vista tanto para aumentar o tempo necessário como para diminuí-lo. Ainda assim, como visto nos outros modelos, houve uma redução de 85% do tempo, quando é utilizado com a representação de menor dimensão.

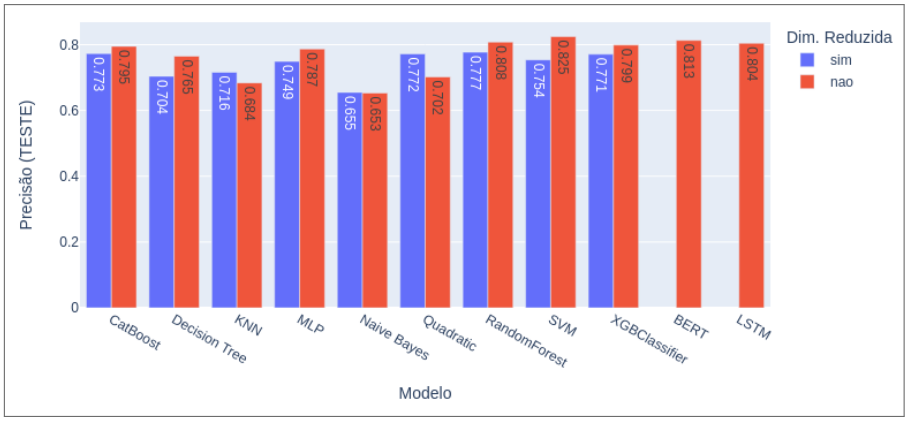
Já os algoritmos NB, KNN e as árvores de decisões, que já mostraram tempos baixos de processamento com o vetor original, tiveram uma redução proporcional. Assim, alcançaram uma diminuição de 98%, 97% e 84%, respectivamente, os tempos vistos já eram de menos de 1 segundo de processamento. Desse modo, é visto que através dessa representação reduzida, os algoritmos demonstram necessitar de menor quantidade de tempo para obter o aprendizado. Então, é necessário validar a qualidade que alcançam em comparação com os que foram vistos juntamente ao vetor original.

Ao comparar os classificadores treinados com o vetor reduzido com as RNN BERT (executado com cinco épocas) e LSTM treinados com o vetor original, é visto que a redução do tempo é mais significativa, pois o que mais se aproxima do tempo dos modelos recentes possui 3% e 14% de seus tempos, respectivamente. Assim, demonstra a proporcionalidade da técnica aos modelos recentes, porém somente quando é observado o tempo de treinamento deles, sendo necessário validar a qualidade dos resultados.

A precisão dos modelos treinados, que denotam em sua qualidade, é apresentada no gráfico da figura 20. Além disso, esse gráfico também mostra uma comparação, através dessa métrica de qualidade, obtidas com o modelo treinado juntamente ao vetor original e com a sua representação de dimensionalidade reduzida. Desse modo, são identificados como vermelho para os modelos treinados com o vetor original e o azul para os que

utilizam a representação gerada a partir dos codificadores automáticos.

Figura 20 – Precisão dos testes dos classificadores, Vetor Original X Vetor de Representação produzido pelo codificador automático



Fonte: Elaboração própria.

A partir desse gráfico, é visível que, o resultado dos algoritmos com os dois vetores possuem valores próximos. O algoritmo que demonstra maior diferença entre o treino com o vetor original e a representação, denota em uma distância de 0.065 no algoritmo SVM. Dos nove algoritmos testados, sete possuíram o melhor resultado com o vetor original e dois apresentaram uma melhora na precisão com a representação de menor dimensionalidade.

Assim como exposto acima, os algoritmos que apresentaram uma redução quando treinados com o vetor de dimensionalidade reduzida, não tiveram uma perda drástica. Os seus resultados se mantiveram a uma perda de menos de 0.1, isso também é visto quando comparado com os modelos mais recente LSTM e BERT, que no experimento anterior demonstra uma necessidade de muito tempo computacional. Desse modo, mostra que, utilizar a rede de codificadores automáticos nesse contexto gerou uma vantagem, a redução do tempo sem a perda drástica da qualidade.

Além disso, ainda para dois modelos proporcionou uma melhor qualidade avaliada com o uso do vetor resultante dos codificadores. Os algoritmos que alcançaram uma melhor qualidade foi o *Quadratic Discriminant Analysis* e o *K — Nearest Neighbors* (KNN), com um aperfeiçoamento de 0.06 e 0.04, respectivamente. Isso demonstra que existe a possibilidade que com o uso de codificadores automáticos na etapa de pré-processamento, ocorra a melhora da qualidade dos modelos, realizando essa redução. Mesmo que esse não seja o objetivo deste trabalho, representa uma oportunidade de pesquisa, estudar os parâmetros e funções de ativação da rede de codificadores, para além de reduzir a dimensionalidade, produzir qualidades melhores nos algoritmos.

A partir desses resultados, foi avaliado o uso dos algoritmos que produziram as melhores métricas para constituir o treinamento de alguns *ensembles* disponíveis pela ferramenta *scikit-learn* do *python*. Entre os comitês classificadores foram utilizados o voto majoritário, empilhamento (do inglês *stacked generalization*) e também *ensembles* que utilizam apenas um classificador para gerar um comitê com diversas amostras de treinamento. O

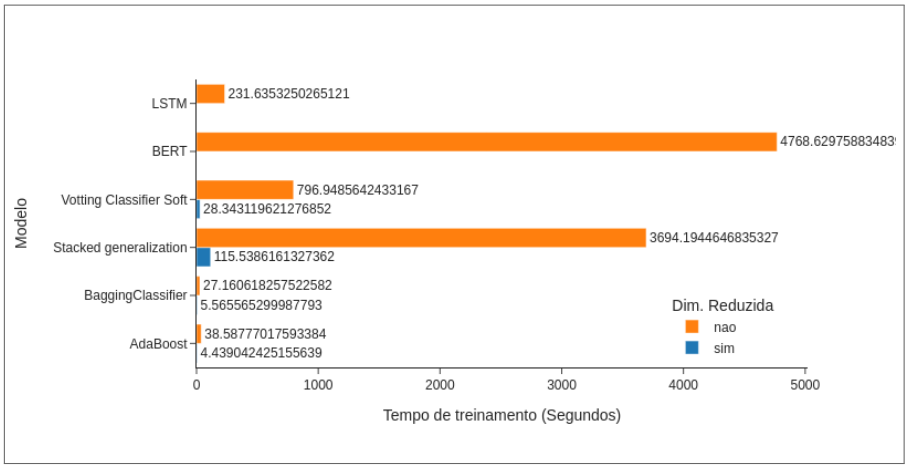
classificador utilizado foi as árvores de decisão, pois além de serem o classificador padrão na ferramenta, também produzem um menor tempo de treinamento e bons resultados como vistos no gráfico da figura 20.

Os comitês classificadores possuem a capacidade de gerar modelos preditivos com ainda mais qualidade. Assim, o objetivo desses agrupamento de classificadores é, através do uso de diversos modelos ou diversas amostragens, gerar um aprendizado que correlacione os resultados de todo o conjunto de classificadores ou amostra. Pois cada classificador (ou amostra) irá gerar um conjunto diferente de predições, no qual, isto será utilizado para constituir uma predição composta.

Desse modo, neste trabalho os experimentos foram realizados com cinco comitês diferentes: *AdaBoost*, *Bagging* e *Random Forest* e os dois já citados acima. As árvores aleatórias também foram avaliadas ainda na etapa de teste dos classificadores. Os experimentos foram realizados para o vetor original e a representação do vetor de dimensionalidade reduzida. Para assim, realizar uma comparação dos custos envolvidos na tarefa de aprendizado, a qualidade obtida e as vantagens ou desvantagens do seu uso com os codificadores automáticos.

Primeiramente, foi analisado o impacto da redução sobre o tempo de treinamento dos comitês, assim como realizado para os classificadores. A figura 21 apresenta o gráfico contendo um comparativo entre o treinamento dos comitês com os dois vetores. Assim, as barras na cor laranja representam o treinamento com o vetor original e a barra azul com a representação criada através dos codificadores automáticos.

Figura 21 – Tempo de Treinamento dos comitês classificadores, Vetor Original X Vetor de Representação produzido pelo codificador automático



Fonte: Elaboração própria.

Assim como para os classificadores, os comitês apresentam uma redução drástica do tempo do treinamento. Uma outra característica observada foi que os comitês com o algoritmo SVM entre os classificadores apresentavam um maior tempo de processamento necessário. Isso ocorreu devido ao modelo já apresentar essa particularidade, no qual

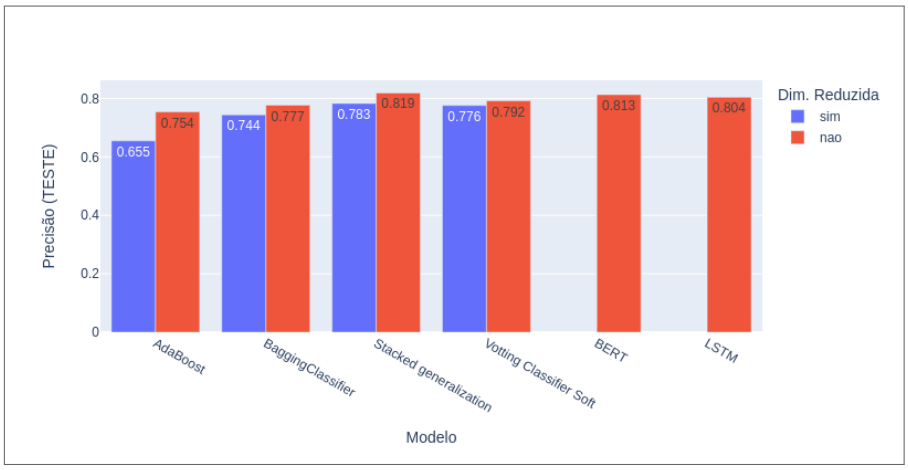
necessita de mais tempo para realizar todo o aprendizado. Porém, é visto também um aumento na qualidade da precisão obtida quando o utiliza.

Os comitês *AdaBoost* e *Baggin*, que utilizaram o classificador Árvore de decisões, alcançaram um custo de tempo bastante baixo, em ambos foram necessários somente 26 segundos com o vetor original. O treinamento com a representação desse vetor, reduziu o tempo a 4 e 5 segundos para o *bagging* e *AdaBoost*, respectivamente. Entretanto, mais adiante, será visto que, apesar do baixo custo, a qualidade das Árvores de decisão só melhoraram no *Bagging*. E, ainda assim, não superaram o comitê de árvores *RandomForest*.

O comitê *Stacked Generalization* treinado com o vetor original apresenta um tempo de processamento de mais de 1 hora, porém ao executá-lo com a redução da dimensionalidade o seu tempo diminui para menos de dois minutos. Desse modo, o comitê que demonstrou uma maior necessidade de tempo para o processamento consegue realizar todo o processamento com menos tempo necessário que o LSTM.

Essa qualidade alcançada pelos comitês é apresentada no gráfico da figura 22. Nele, as barras na cor azul indicam os modelos treinados com a representação do vetor de dimensionalidade reduzida e as barras na cor vermelha os que foram treinados com o vetor original. Desse modo, trazendo a comparação da qualidade dos treinos para validar os comitês classificadores.

Figura 22 – Precisão dos testes dos comitês classificadores, Vetor Original X Vetor de Representação produzido pelo codificador automático



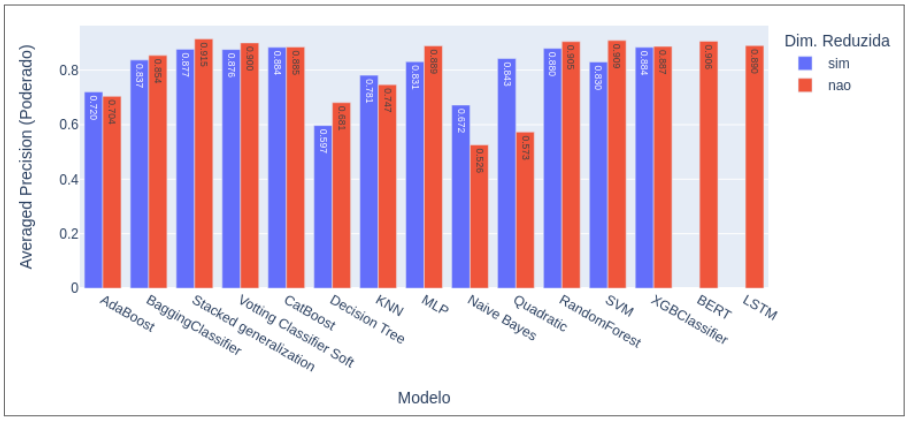
Fonte: Elaboração própria.

No gráfico 22 é apresentado que a qualidade entre os dois vetores (reduzido e o original) estão bastante próximas, não ultrapassando uma diferença de 0.1. Diferente do gráfico visto para os classificadores, não houve nenhum comitê que o resultado do treinamento com a representação do vetor produzisse uma qualidade superior aos treinados com o vetor original. Além disso, a melhoria na qualidade obtida com os comitês não ultrapassaram uma diferença de 0.1. Desse modo, para os comitês treinados com o vetor original e múltiplos classificadores, a maior qualidade obtida foi através do *stacked generalization*, que apresentou um aumento de 0.011 para o classificador QDA que teve a melhor precisão.

Ao comparar os resultados dos comitês treinados com o vetor reduzido aos modelos mais recentes (LSTM e BERT), é visto também uma aproximação de suas precisões. No qual, o comitê *Stacked Generalization* ultrapassa os modelos recentes com o vetor original, e com a redução se aproxima com uma diferença de aproximadamente 0.03. Além disso, houve uma necessidade de somente utilizar 1 época de treinamento, por não ser possível validar para mais de 5 épocas, pois ultrapassava o tempo de uso da plataforma.

Diante desses resultados, a partir desse experimento pode-se afirmar que os resultados dos comitês não demonstram uma grande vantagem em utilizá-los. Porém, antes de descartar essa técnica, foi avaliada a especificidade das classes em cada modelo através da precisão media, que atribui de 0 a 1 o quanto o modelo possui poder de predição com base nas métricas precisão e retorno para cada rótulo. Esses valores são apresentados no gráfico de barras da figura 23, contendo o resultado de todos os modelos e comitês classificadores.

Figura 23 – Índice de Predição, dos testes, dos classificadores e comitês classificadores (baseados na *Precisão Ponderada*), Vetor Original X Vetor de Representação produzido pelo codificador automático



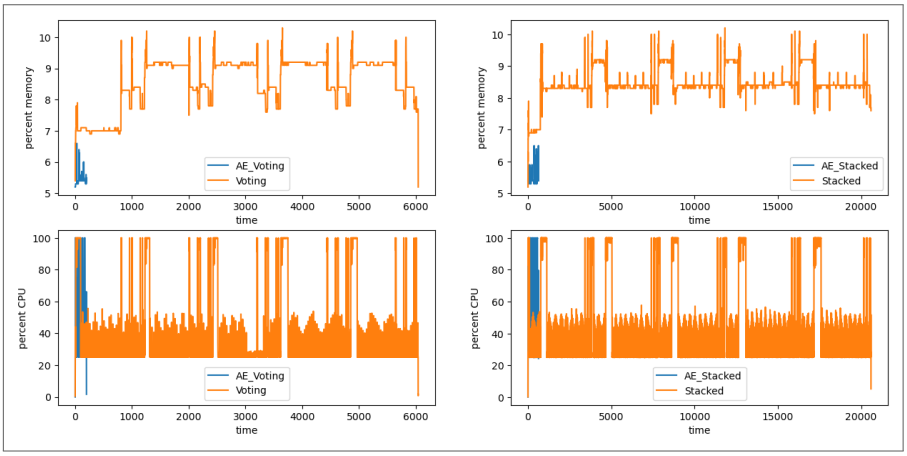
Fonte: Elaboração própria.

Por meio do gráfico 23, é visível que o algoritmo que alcançou maior índice de predição com o vetor original, foi o comitê *Stacked Generalization*, com 0.915. Entre os outros modelos, os que atingem melhor índice (abaixo dos maiores) são os comitês, com exceção do *AdaBoost* e os dois algoritmos *XGboost* e o *CatBoost*. Ao comparar os resultados obtidos entre os dois vetores, é visto que para alguns algoritmos, esse índice conseguiu uma melhora com o vetor produzido pelos codificadores automáticos.

As melhorias demonstram que o uso da técnica de codificadores automáticos podem dar mais poder de aprendizado e predição aos modelos. Outra observação a ser feita é, a importância da investigação do custo computacional de uso de memória e CPU pelos algoritmos aqui testados, assim como realizados nos experimentos de escolha da rede de codificadores. Foram escolhidos os comitês voto majoritário e empilhamento, por possuírem a melhor qualidade de precisão e índice de predição, para investigar o impacto no custo computacional ao utilizar cada vetor.

Além disso, a partir da métrica que determina a especificação (precisão media) dos classificadores, também é visto que com o uso dos codificadores essa medida alcançou resultados próximos, e iguais para alguns dos modelos, aos vistos nos algoritmos BERT e LSTM. Os gráficos exibidos na figura 52 apresentam os custos computacionais comparados entre os vetores para os modelos voto majoritário e empilhamento.

Figura 24 – Custos computacionais de comitês, Voto Majoritário e Empilhamento



Fonte: Elaboração própria.

A partir dos gráficos de consumo computacional, é visto que o uso de CPU não difere entre os vetores, possuem o mesmo grau de uso alcançado quase 100% do índice de processamento. Já nos custos de memória, é observada uma diminuição evidente, pois com a representação de vetor de menor dimensionalidade, o algoritmo produz um consumo abaixo de 8% em ambos os comitês. Desse modo, demonstra que o algoritmo consegue reduzir seu tempo computacional, por poder acessar mais variáveis do *corpus* com menos uso de memória.

O uso de memória é um fator que dita a possibilidade de um computador executar determinado processo, pois em muitos dos experimentos realizados neste trabalho foi um motivo de inviabilidade. Então, é considerado que com o uso dos vetores reduzidos pelas RCA, foi possível realizar o aprendizado dos modelos que não poderiam ser executados em máquinas de menor poder computacional. Porém, em alguns casos, somente seria verdade, caso a rede de codificadores fosse disponibilizada já pré-treinada. Pois a etapa de treinamento das redes codificadoras são bastante custosas.

A partir desses resultados, é visto que existe um impacto positivo no uso das redes de codificadores na etapa de pré-processamento. Uma rede de codificadores automáticos já treinada, possibilita a redução dos custos do treinamento dos modelos, ao aplicar a redução através dos codificadores automáticos. Além disso, também é visto que, mesmo sem uma exploração mais ampla dos parâmetros do modelo (visando a melhoria dos seus resultados), a qualidade dos algoritmos treinados com a representação do vetor produzida pelas RCA, não tem um grande decaimento. Assim, resulta em um processo de treinamento menos custoso e mais rápido.

Diante de todo o exposto, o próximo subcapítulo irá tratar da validação dos comitês de classificadores *Stacked Generalization*, escolhido por mostrar maior qualidade que os demais treinados. Desse modo, será comparado os parâmetros de qualidade obtida pelo comitê com o vetor de representação e o vetor original. Além desses critérios, também será visto o custo de tempo operacional dos modelos com cada vetor, para poder analisar as vantagens e desvantagens do seu uso com a técnica de rede de codificadores automáticos para redução de dimensionalidade.

5.2 VALIDAÇÃO DO COMITÊ CLASSIFICADOR *STACKED GENERALIZATION*

A partir da escolha do comitê de classificadores *Stacked Generalization*, foram realizados os experimentos para validação de sua qualidade com o vetor representação produzido pela rede de codificadores automáticos. Para essa avaliação, foram comparados os resultados do treinamento com diversas amostragens do *corpus* TBCOV e também com o intuito de comparar com os vetores TFIDF originais e o vetor de representação reduzido.

O comitê foi treinado com 147216 *tweets* rotulados do *corpus* TBCOV, não equivalendo a todos os *tweets* disponíveis na língua portuguesa deste corpus. Além disso, só foi possível realizar o treino para toda essa quantidade de *tweets* com os vetores reduzidos através das redes de codificadores automáticos. Pois, com os vetores originais, o treinamento ultrapassou o tempo de treinamento disponível através das máquinas virtuais disponibilizadas pela plataforma *kaggle*.

Para comparar a evolução do tempo computacional necessário para treinar o comitê com o vetor original e o reduzido, foram realizadas três amostragens do corpus. A primeira amostra foi com um vetor de 7500 *tweets* rotulados, a segunda com 15000 e a terceira com 30000. Esses valores foram escolhidos por conseguirem alcançar todo o processamento nas máquinas virtuais e comparar os resultados de custos entre os treinamentos com vetores originais e a representação do vetor reduzido.

Essa comparação entre as amostras é apresentada na tabela 24, que traz a quantidade de *tweets* no corpus, o tempo de processamento para a finalização do treinamento e a precisão obtida. Para assim, ser capaz de avaliar quais impactos o comitê teve com a utilização de uma rede de codificadores para reduzir a dimensionalidade dos vetores. Além de validar a qualidade obtida no comitê através da proposta neste trabalho.

Na tabela 24 é exibido que, à medida que a quantidade de *tweets* é aumentada para treinar o modelo, o tempo de processamento acresce muito juntamente ao comitê treinado com o vetor original. Assim, o distanciamento entre as precisões obtidas no teste entre o comitê treinado com o vetor original e a sua representação produzida pelo CA diminui, no primeiro experimento é visto uma diferença de 6% no resultado e no último experimento essa divergência é de 3,78%.

O gráfico na figura 25 apresenta a tendência da precisão obtida no teste dos experimentos com diferentes quantidades de *tweets*. A partir disso, fica mais evidente a aproximação

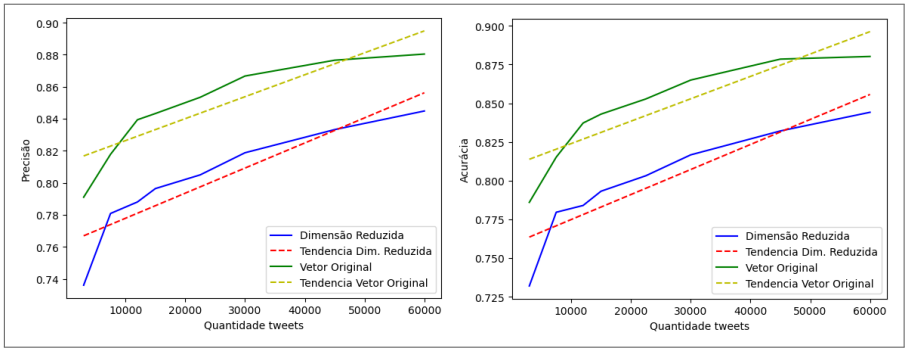
Tabela 24 – Métricas de comparação produzidas pelos testes do comitê *stacked generalization* com vetor original e vetor de representação para diferentes amostragens

Reduzido	Quantidade de *Tweets*	**Horas**	**Minutos**	**Segundos**	**Precisão Ponderada (Teste)**
Sim	7500	0	2	8	0.78
Não	7500	1	0	38	0.82
Sim	15000	0	5	14	0.80
Não	15000	1	26	51	0.84
Sim	30000	0	17	6	0.82
Não	30000	8	32	1	0.87

Fonte: Elaboração própria.

entre os resultados. Assim, demonstra que com uma maior quantidade de dados de treino nos CA e na técnica TFIDF, é possível alcançar resultados ainda melhores.

Figura 25 – Gráfico da tendência da precisão de teste dos modelos treinados com vetor original e a sua representação de dimensão reduzida.



Fonte: Elaboração própria.

A partir das linhas de tendência no gráfico da figura 25, é presumível que as retas possam se encontrar. Isso ocorre pois, mesmo que a reta da tendência do modelo treinado com o vetor original esteja inclinada para cima, essa inclinação para os modelos treinados com a representação reduzida, tem um ângulo um pouco maior. Dessa forma, isso pode apontar que os modelos treinados com o vetor reduzido tendem a melhorar mais com uma maior quantidade de dados.

Assim, o comitê foi treinado para todo o *corpus* somente com o vetor reduzido. Pois, desse modo, foi alcançado todo o processamento nos limites estipulados pela plataforma na qual foram realizados os experimentos. Diante dos seus resultados, foram realizados todos os mesmos testes de validação e resultado, para visualizar a qualidade do modelo final. A tabela 25 mostra o tempo de processamento, qualidade de precisão em teste e o tamanho total de *tweets* treinados.

Tabela 25 – Métricas resultantes de treino e teste do comitê final, treinado com todo o *corpus* TBCOV coletado.

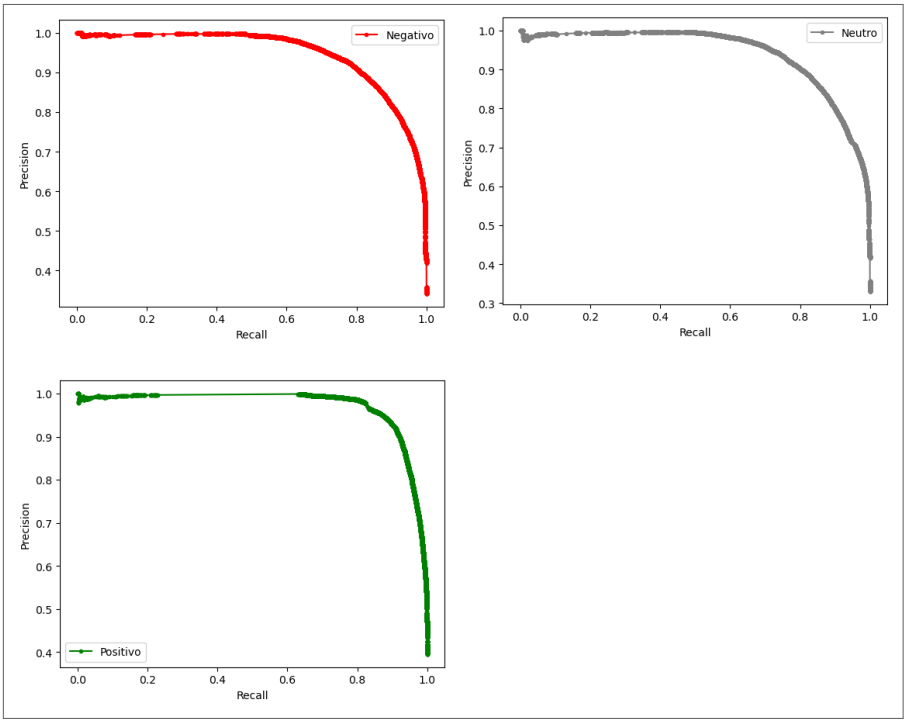
Quantidade de *Tweets*	**Horas**	**Minutos**	**Segundos**	**Acurácia Treino**	**Acurácia Teste**	**Precisão Treino**	**Precisão Teste**
147216	9	10	38	97.620	87.531	97.622	87.517

Fonte: Elaboração própria.

A qualidade obtida com a técnica de CA e o comitê utilizado para classificar o *corpus* coletado para este estudo, pode ser considerada adequada, ao observar os trabalhos da literatura. Esses estudos demonstram que os níveis de acurácia encontrados estão entre 55% e 96%, para técnicas de classificadores dinâmicos, sustentando a equivalência da utilização dos métodos aqui propostos (CALADO, 2019; SILVA, 2020; SANTOS, 2020). E através da tabela onde é apresentado a acurácia atingida pelos modelos aqui apresentados, é visto que sua qualidade esta nesta aproximação.

O comitê foi treinado com 98148 *tweets* e validado com 49068 e teve um aumento no tempo de processamento para mais de 9 horas.O modelo demonstra ter um bom nível de predição de texto para o contexto no qual foi treinado, pois apresenta uma acurácia de teste próxima a 90%. Por meio dos gráficos na figura 26, é observado o grau de sensibilidade de cada rótulo no comitê classificador.

Figura 26 – Gráficos da Precision-Recall Curve dos rótulos de sentimento treinados para o comitê classificador *stacked generalization*



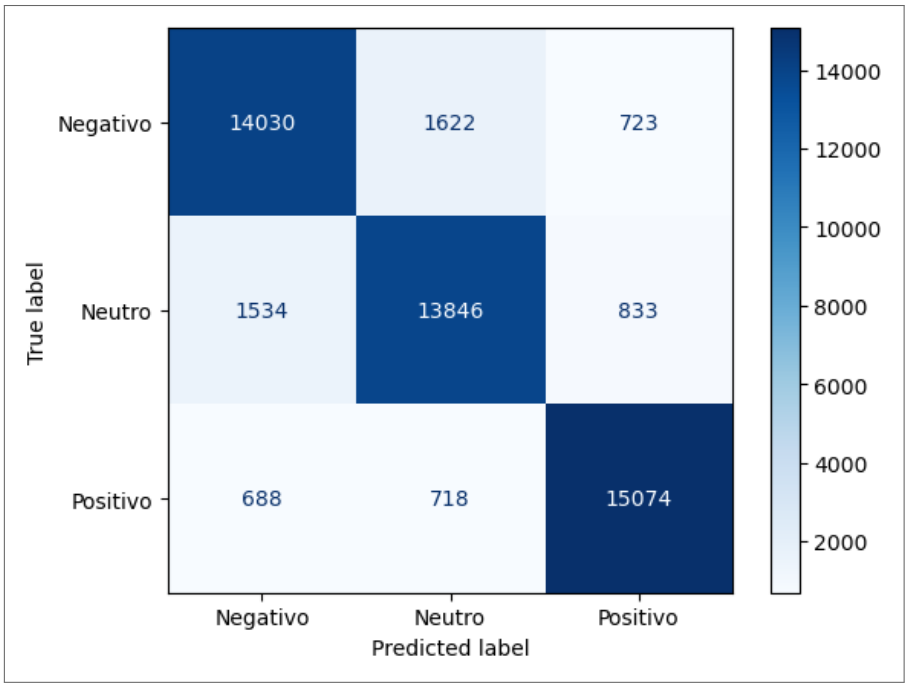
Fonte: Elaboração própria.

Ao analisar os gráficos da figura 26, é visível que a sensibilidade dos rótulos Negativo e Neutro tiveram uma aproximação muito grande, obtendo um valor igual. Já o rótulo positivo apresenta um nível melhor que os outros, pois possui um valor superior aos demais. Porém, a curva Roc é calculada para classificadores binários, o que não é o caso desse estudo. Então, para conseguir estimar a sensibilidade desse modelo, o cálculo foi realizado com base em um rótulo em comparação com os outros.

Essa equivalência, vista através do cálculo da precisão ponderada entre os rótulos negativo e neutro, também pode-se notar através da matriz de confusão com o resultado

dos testes do comitê. Essa matriz é apresentada na figura 27 para os 49068 *tweets* do *corpus* de teste. Na matriz é visto que os rótulos neutro e negativo possuem o mesmo comportamento de classificação errado entre si. Assim, o rótulo positivo possui uma taxa de maior qualidade, pois a predição errada neste rótulo não ultrapassa 1500 *tweets* de 16356.

Figura 27 – Matriz de confusão dos resultados dos testes com 49068 *tweets*, com o comitê *stacked generalization*



Fonte: Elaboração própria.

Já os demais rótulos predizem de forma errada mais de 2300 *tweets* de aproximadamente 16300 rotulados para cada um deles. Desse modo, entende-se que há uma tendência de textos neutros, no contexto da pandemia, serem preditos como negativos com uma maior frequência que o positivo e o mesmo para o negativo. Na tabela 26 são apresentadas outras métricas de avaliação de resultados, que demonstram essa mesma característica, vista através da PR curva e da matriz de confusão.

Diante dos resultados vistos na tabela 26, fica claro que o comitê possui uma maior qualidade na predição com o rótulo positivo. Mas, observando a média ponderada das métricas observadas, seus resultados ficam próximo de 90%, demonstrando uma qualidade boa o suficiente para realizar estudos contextualizados na pandemia de COVID-19.

Ao observar a acurácia nos resultados de teste dos experimentos com o comitê classificador, técnicas selecionadas e redução da dimensionalidade com a rede neural RCA, é visto uma qualidade de 82,09%. No estudo de Farias (2022), foi possível, através da utilização do dicionário léxico SentiLex-PT, alcançar uma acurácia de 68,5%. Assim, nota-se que os resultados obtidos através do estudo aqui realizado foram superiores.

Além disso, no trabalho de Gomes (2022), é realizado uma exploração dos modelos

Tabela 26 – Métricas do teste do comitê *Stacked generalization*, treinado com todo o *corpus* TBCOV coletado.

	Precisão	**Revocação**	**F1-Score**	**Suporte**
Negativo	0.86	0.86	0.86	16375
Neutro	0.86	0.85	0.85	16213
Positivo	0.91	0.91	0.91	16480
Acurácia			0.88	49068
Macro avg	0.88	0.88	0.88	49068
Média Ponderada	0.88	0.88	0.88	49068

Fonte: Elaboração própria.

para a aplicação da classificação, sem uma exploração, também, das técnicas da PLN na etapa de pré-processamento. Através desse estudo, ele conseguiu alcançar através do seu melhor modelo (OpLexicon), uma acurácia de 39% em seus testes. O pesquisador também explorou dados do *Twitter* relacionados à COVID-19, porém, concentrado nas reações a partir das politicas de vacinação entre os políticos com maiores menções ou engajamento.

Assim como visto no trabalho anterior, a classificação do sentimento foi baseada em um dicionário léxico e não alcançou uma acurácia acima de 70%. Entretanto, como é enfatizado por Farias (2022), essas qualidades obtidas são consideradas adequadas por se tratarem de dicionários léxicos. Por outro lado, no estudo de Peixoto (2021), é realizado um comparativo de três modelos preditivos, que realizam a redução da dimensionalidade em diferentes cenários de experimento para a classificação de resolutividade de reclamações de clientes.

Os resultados desse estudo foram equiparáveis aos vistos neste trabalho, com uma acurácia próxima a 75% no modelo que demonstrou maior qualidade. Assim, se perceber que, utilizando os algoritmos preditores que usam a rede neural RCA para reduzir a dimensionalidade, a qualidade dos resultados dos testes ultrapassaram aos estudos com dicionários léxicos.

Para a melhor comparação entre os resultados obtidos neste trabalho, foram treinados modelos BERT e LSTM com o *corpus* utilizado no comitê aqui criado. Desse modo, foi visado realizar uma comparação entre os resultados de teste entre os modelos de estado da arte com maior utilização entre pesquisadores da área de PLN. O modelo BERT está entre os mais utilizados, pois utiliza redes neurais que simulam uma IA para realizar as predições dos textos.

Os resultados desses dois modelos são comparados ao comitê elaborado neste subcapítulo através de amostragens de 7500, 12000 e 30000 *tweets*. A tabela 34 apresenta os resultados com a melhor acurácia vista nas redes para a validação do LSTM, BERT e o comitê.

Tabela 27 – Métricas resultantes de treino e teste do comitê final, treinado com todo o *corpus* TBCOV coletado.

Quantidade de *Tweets*	**Acurácia LSTM (% TESTE)**	**Acurácia BERT (% TESTE)**	**Acurácia COMITÊ (% TESTE)**
7500	78,32	54,51	76,32
12000	81,45	53,58	79,26
30000	84,67	56,22	82,09

Fonte: Elaboração própria.

Na tabela 34 é mostrado que o modelo BERT apresentou uma acurácia muito abaixo das que foram vistas para os outros modelos, porém a rede neural utilizada para seu treino foi o BERTimbau[6], que se trata da técnica BERT pré-treinada para a língua portuguesa. Por outro lado, o modelo LSTM produziu resultados acima do que foi visto no comitê desenvolvido neste subcapítulo, porém essa qualidade foi próxima ao do *Stacked Generalization* treinada com o vetor de dimensionalidade reduzida.

Além disso, o tempo de treinamento observado entre o comitê e a técnica de rede neural LSTM com 30000 *tweets* foram equiparáveis, o tempo necessário para realizar todo o processamento dos algoritmos foi reduzido nas duas técnicas. Porém, o modelo LSTM foi executado com 5 épocas e no gráfico da figura 120 no apêndice F é visto que o algoritmo não alcança sua curva de tendência total, mostrando que ainda se pode executar por mais épocas, até alcançar o melhor resultado.

Em virtude dos aspectos abordados, o comitê de classificadores treinado através das técnicas propostas neste trabalho apresenta resultados análogos aos métodos de estado da arte vistos neste subcapítulo. Desse modo, demonstra o poder das redes em manter a qualidade de predição, mesmo reduzindo a dimensionalidade. Assim, o modelo treinado foi salvo através da biblioteca *pickle*, disponibilizada através da linguagem de programação *Python*.

Então, todo o *corpus* coletado para os estudos através das ferramentas expostas foi rotulado e salvo em CSV. Por meio desse arquivo foi utilizado o *pandas* para criar *data-frame's* de cada respectivo *corpus* rotulado e, assim, foram produzidos os gráficos a serem analisados no próximo subcapítulo.

5.3 PREDIÇÃO E ANÁLISE DO *CORPUS* BRASIL-COVID-PANDEMIA 2020-2021

Para as análises realizadas neste subcapítulo, serão utilizados dados provenientes da Fundação Oswaldo Cruz (Fiocruz), que durante toda a pandemia armazenou e disponibilizou[7] informações acerca das mortes, casos e vacinação dentro do Brasil. Assim, será comparado como a evolução da doença afetou as discussões sobre o assunto dentro do país, avaliando a polaridade demonstrada no corpo do texto, na rede social *Twitter*.

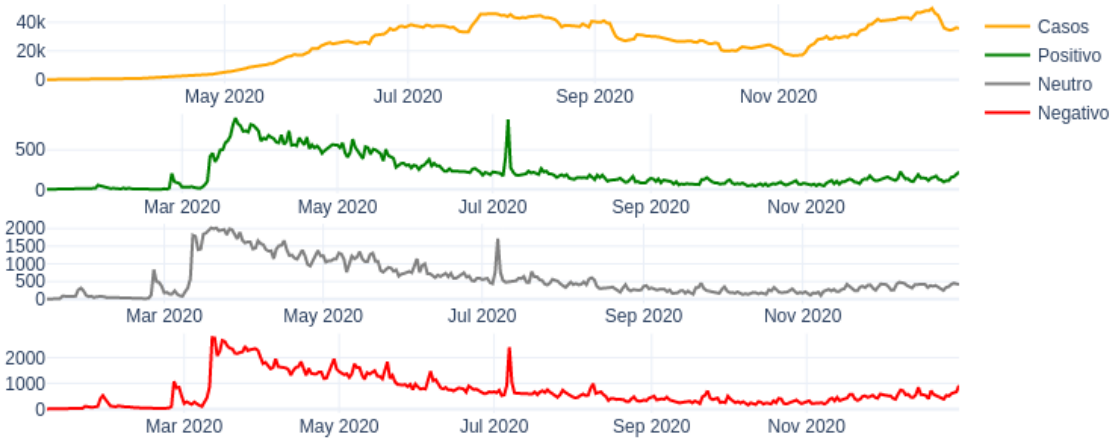
[6] Disponível no repositório: https://github.com/neuralmind-ai/portuguese-bert. Acesso em: 16 de Maio de 2023

[7] Dados disponibilizados em: https://bigdata-covid19.icict.fiocruz.br/. Acesso em: 6 de outubro de 2022.

Toda a investigação realizada neste subcapítulo é pautada sobre o recorte pandêmico de janeiro de 2020 até dezembro de 2021, como ressaltado nos capítulos anteriores. No qual, está situado a maior quantidade de *tweets* produzidos devido às particularidades impostas sobre a pandemia. A partir dos gráficos da figura 123, será analisada a oscilação da polaridade dos sentimentos apresentados pelos usuários durante o ano de 2021, comparados aos números médios dos casos de COVID-19 no ano de 2020. Nele, é visto quatro gráficos, o primeiro é a evolução dos casos, o segundo representa o sentimento positivo, o terceiro apresenta o sentimento neutro e o último mostra o sentimento negativo.

Figura 28 – Oscilação de sentimento X Evolução dos casos de Covid-19

2020



Fonte: Elaboração própria, dados provenientes da ICICT.

Por meio do gráfico apresentado acima, é exposto que, mesmo com um baixo índice de casos registrados, a quantidade de *tweets* publicados são maiores que os vistos ao longo do resto do ano de 2020. Inicialmente, os textos nas três polaridades apresentam a mesma inclinação, porém com a polaridade negativa se sobressaindo em quantidade de textos publicados. O número de casos por dia continua aumentando após o mês de pico da doença, já a quantidade de *tweets* por dia rotulados em suas polaridades continuam em queda.

Essa queda vista nos sentimentos pode ser atribuída a uma diminuição no engajamento das hashtags exploradas. Porém, ainda assim, são percebidos aumentos para cada polaridade, no qual os *tweets* rotulados como negativos e neutros mostram crescimentos nos meses iniciais, até próximo ao mês de julho, ultrapassando mais de 500 *tweets* por dia. Nos meses subsequentes a junho de 2020, a quantidade de *posts* rotulados como positivo caem drasticamente, sendo observados valores abaixo de 100 por dia.

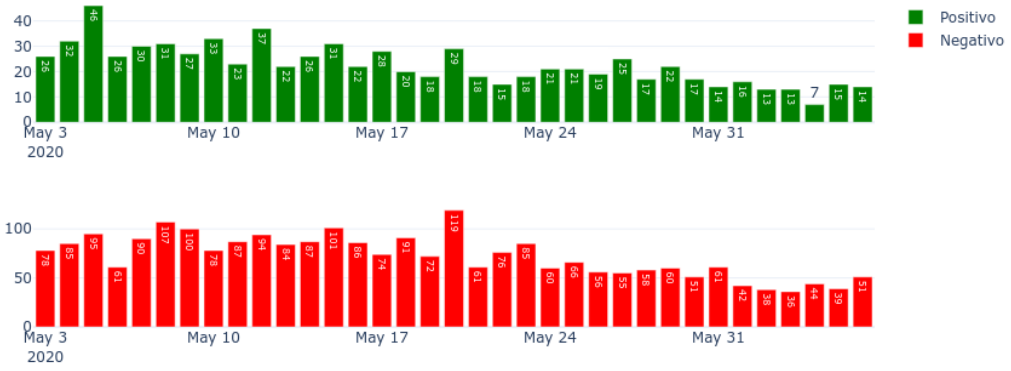
Já para os sentimentos Neutros e Negativos, esse número se mantém acima de 100, na polaridade negativa há picos que podem passar de 200 *tweets* no dia. Assim, demonstra que, durante o período de 2020 com o aumento dos casos e também com a falta de um medicamento próprio para tratar o vírus, os usuários do *Twitter* fizeram posts predominantemente negativos. O início do mês de novembro foi um período no qual o número de casos por dia demonstrou uma grande redução.

Essa redução vista nos casos também foi comentada pelos usuários do *Twitter*. E se observa um pico na polaridade positiva e uma redução para menos de 100 *tweets* negativos por dia, nesse mesmo período. Além disso, o ano de 2020 foi marcado por diversas medidas não farmacêuticas que buscavam controlar a propagação. Porém, essas ações foram aplicadas em diversos níveis de isolamento em cada cidade/estado do país.

Para analisar como essas medidas restritivas foram sentidas pelos usuários do *Twitter*, foram escolhidas três cidades brasileiras em quatro diferentes regiões, para estudar a viração de polaridade. As escolhidas foram Fortaleza, Niterói e Curitiba em seus períodos de *lockdown* definidos pelos governadores do estado e prefeitos. Todas essas cidades tiveram um período de isolamento social mais restrito no mês de maio, diferindo somente entre suas datas iniciais e finais.

A cidade de Fortaleza, no Ceará, teve o período de *lockdown* decretado para 8 de maio até dia 20 do mesmo mês, porém prorrogado até 1 de junho. Para analisar, foram coletados 4892 *tweets* com localização dentro ou próximo à cidade, para serem rotulados e analisada a polaridade manifestada durante esse período, com 5 dias antes do início e 5 dias depois do final. Desse modo, a figura 29 apresenta um gráfico de linhas, mostrando a convergência das polaridades observadas em *tweets* com localização próximas à região da cidade de Fortaleza.

Figura 29 – Oscilação de sentimento, Fortaleza - CE (3 de maio a 6 de junho de 2020)



Fonte: Elaboração própria.

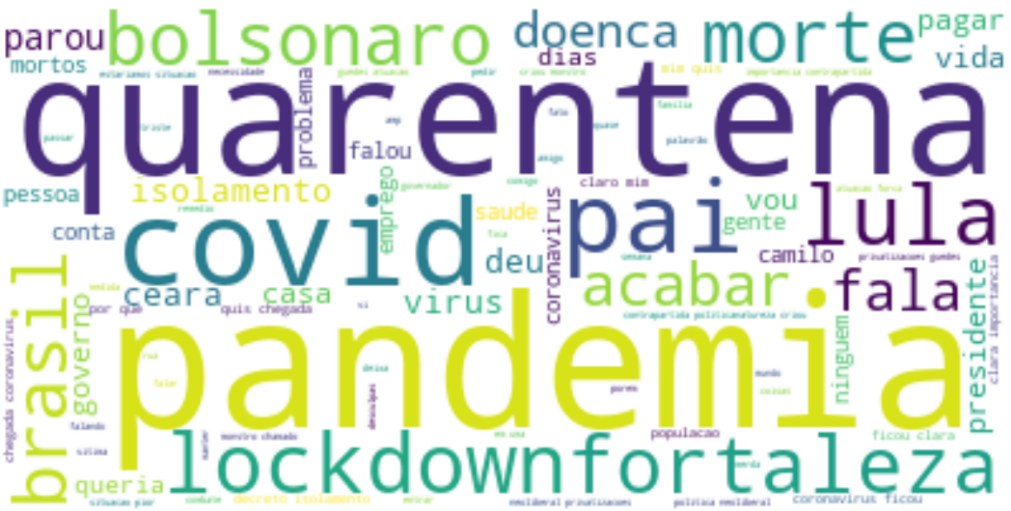
Por meio dos gráficos na figura 29, é exposto que, no primeiro dia e no último decretado para o *lockdown*, a polaridade negativa apresenta dois picos. Assim, demonstram ser maiores que as quantidades vistas para as outras duas polaridades, expressando que parte

dos usuários que utilizaram as *hashtags* para expor alguma opinião sobre a COVID-19 e o *lockdown* demonstraram um posicionamento negativo.

Porém, durante o período do *lockdown*, a polaridade positiva apresenta diversos picos de *tweets*, mas com quantidades inferiores aos *tweets* negativos para o mesmo dia. Isso mostra que, predominantemente, a polaridade negativa foi apresentada durante esse período de *lockdown* aplicado em Fortaleza, conforme os textos rotulados por nosso modelo. Após o período determinado pelo *lockdown*, essa polaridade positiva é reduzida.

Ao buscar a investigação do motivo da queda de *tweets* rotulados como positivos após o dia 20 de maio, foi criada uma nuvem de palavras que mostra a ocorrência das palavras por tamanho. Essa nuvem é apresentada na figura 30, no qual as palavras maiores são as que apresentam maior ocorrência e são as menores as menos citadas. Essa nuvem é gerada com base nos textos rotulados como negativos, pois, assim, será possível ver prováveis motivos da predominância do sentimento negativo após esse período.

Figura 30 – Nuvem de palavras para polaridade Negativa 20 de maio de 2020 - Fortaleza



Fonte: Elaboração própria.

Algumas palavras em destaque na nuvem gerada para o dia 20 de maio são: covid, pandemia, *Lockdown*, quarentena, doença, morte e Fortaleza. Esses termos têm correlação com a medida decretada na cidade de Fortaleza e a situação imposta pelo vírus. O nome Camilo aparece no lado direito da imagem e se trata do governador do estado do Ceará no ano de 2020. No qual, ele demonstra estar relacionado com os termos isolamento e dias, pois mostram que o estado prorrogou o regime de *lockdown* até o dia 1 de junho de 2020.

Além disso, o nome de duas figuras políticas influentes são também vistas, sendo eles Lula e Bolsonaro. O primeiro foi presidente do país até 2010 e o segundo o presidente em exercício no ano de 2020 e 2021. Possivelmente, a aparição dos seus nomes tem a ver com o contexto politico enfrentado no período, no qual era muito comum a comparação ou citação deles em assuntos políticos. Então, esse pico de *tweets* rotulados como negativos,

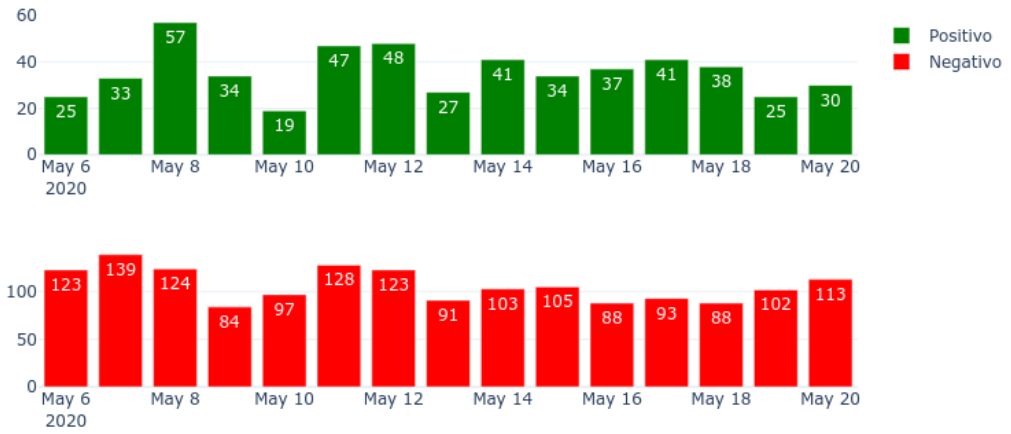
podem ser deduzidos como uma reação à continuação da medida restritiva na cidade e também ao contexto político vivenciado.

Desse modo, se conclui que a queda dos *tweets* positivos ocorre por uma insatisfação com a prorrogação da medida restritiva em Fortaleza. Por outro lado, pode também indicar um abandono das *hashtags*, utilizadas para a criação do *corpus* pelos usuários na cidade. Já quando é observada uma nuvem de palavras com os *tweets* rotulados como positivos no período do *lockdown* com a prorrogação, é evidenciado que os usuários demonstravam usar palavras de cuidados com a vida, combate à doença, além de apoio a projetos e ONGs que se popularizaram no período por ajudarem pessoas em situação de risco alimentar.

Essa nuvem de palavras é apresentada na figura 134, no apêndice I, no qual também são expostas outras nuvens de palavras para datas de picos com as polaridades negativa e positiva. Por fim, é visto que o sentimento negativo já se mantém durante o período de isolamento mais restrito e reduz após o fim do isolamento. Na nuvem de palavra para o sentimento negativo no período do isolamento na figura 130, é visível o quanto o termo morte, casa e *lockdown* é citado. Isso demonstra o impacto das restrições e dos óbitos em decorrência do coronavírus sobre os usuários.

Ao observar os gráficos da figura 31, os resultados obtidos para a cidade de Niterói, é visto que durante o período de *lockdown* (11 a 15 de maio), a polaridade negativa inicia em um pico de 73 *tweets* diarios. Então, sofre uma redução de *tweets* até o dia 13, e a partir disso, voltar a subir sem ultrapassar o pico inicial. A polaridade positiva apresenta um comportamento semelhante ao visto na negativa, o qual inicia em um pico, tem uma descida até o dia 13 e volta a subir até o dia 15.

Figura 31 – Oscilação de sentimento, Niterói - RJ (6 a 20 de maio de 2020)



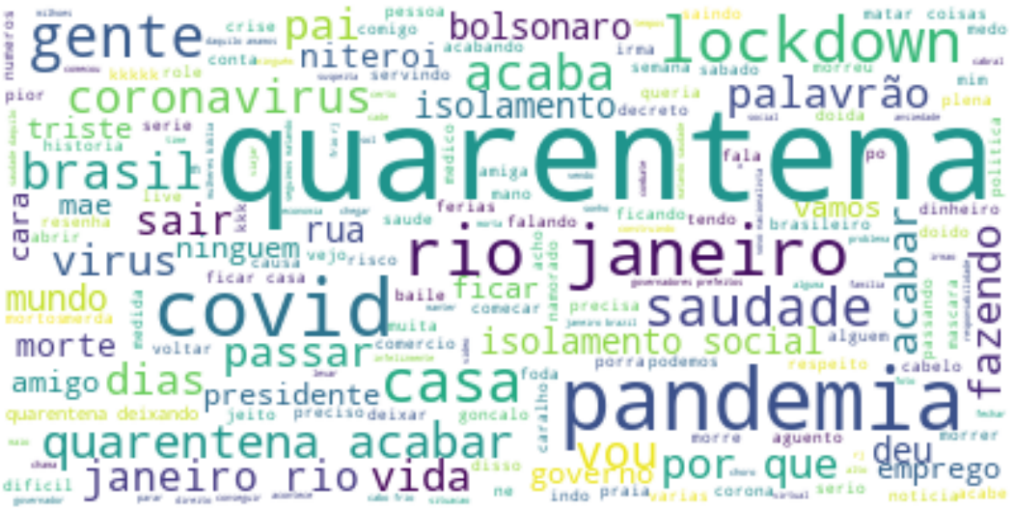
Fonte: Elaboração própria.

Após o último dia de *lockdown* (dia 15), a polaridade negativa tem uma leve inclinação de subida e a positiva demonstra uma pequena oscilação. Contudo, essa diferença entre as polaridades negativas e positivas ocorrem somente na perspectiva da quantidade entre os *tweets* de cada polaridade. Ao serem comparadas, a polaridade negativa se sobressai

em quantidade de *tweets* diários. As postagens classificadas como negativas ultrapassam o valor de 30, enquanto as positivas estão limitadas a menos que este valor.

Para avaliar os tópicos de relevância observados pelos usuários de Niterói através da nossa pesquisa em palavras-chave sobre o *lockdown* implementado na cidade, também foi feita uma nuvem de palavras. Esse gráfico pode ser visto na figura 32, que apresenta as palavras mais vistas entre os textos coletados com a localização dentro da cidade. Os textos utilizados para criar esse gráfico de palavras foram tirados de um recorte de tempo dos 5 dias de lockdown, para, desse modo, ver quais palavras foram mais apresentadas como negativas neste período.

Figura 32 – Nuvem de palavras para polaridade Negativa 11 a 15 de maio de 2020 - Niterói



Fonte: Elaboração própria.

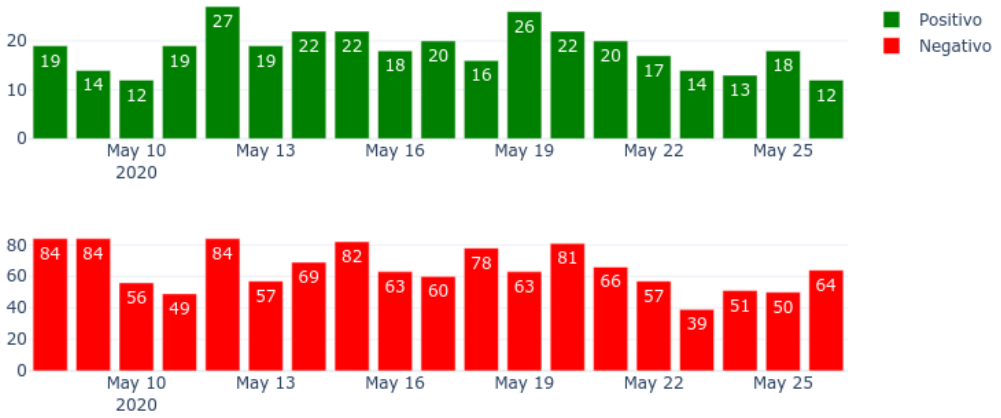
Os termos com maior destaque nesta nuvem de palavras foram: quarentena, pandemia e Rio de Janeiro, que mostram o período no qual foi feito o recorte, além dos dois primeiros termos estarem nas palavras-chave de busca. Já o termo "acaba", que aparece com destaque, também tem forte relação com quarentena. Na nuvem de palavras da parte inferior esquerda, "quarentena acabar"aparece com destaque, mostrando o desejo do término dos isolamentos naquele período.

Portanto, é visto que nessa cidade existe uma ampla quantidade de termos equiparáveis. Essas palavras apresentam um nível de destaque semelhante, e em sua grande maioria são acerca da medida de isolamento e a vontade de que termine, para que possa retornar à normalidade. Somente uma figura política aparece nessa nuvem de palavras que é o presidente em exercício do ano. A palavra morte, que em Fortaleza teve grande destaque, nessa cidade aparece com um menor peso.

Para finalizar as análises dos decretos de lockdown, são apresentados os gráficos na figura 33, que mostram a oscilação da polaridade dos *tweets* diários, no período de 8 a 26 de maio de 2020, na cidade de Curitiba. Essa cidade implementou um período de

lockdown de 8 dias, começando no dia 13 e terminando no dia 21 de maio. No total, foram coletados 2390 *tweets* de usuários com localização na cidade de Curitiba.

Figura 33 – Oscilação de sentimento, Curitiba - PR (8 a 26 de maio de 2020)



Fonte: Elaboração própria.

É observado que nos 5 dias anteriores ao início do *lockdown*, o sentimento negativo era predominante em quantidade e teve um declínio de dois dias, voltando a subir no dia anterior ao início do decreto. Durante o período de quarentena rígida na cidade, os *tweets* rotulados como de polaridade negativa oscilaram com vários picos sempre maiores ou próximos a 80 postagens diárias. Já os *tweets* com polaridade positiva em nenhum momento do recorte de tempo ultrapassou 30 postagens diárias.

A partir do fim do decreto, após o dia 21 de maio, é visto no gráfico uma queda na polaridade positiva, enquanto na polaridade negativa há um aumento. Porém, até o último dia do recorte de tempo, na polaridade negativa, a quantidade de *tweets* não chega aos picos vistos durante o *lockdown*. Ao buscar entender os possíveis motivos da predominância negativa, foi plotado o gráfico de nuvem de palavras na figura 34.

A partir dessa nuvem de palavras é mostrado que os termos com maior destaque são relacionados ao *lockdown*: quarentena, covid, pandemia e coronavírus. Alguns outros termos com menor ênfase aos relatados acima, porém, com destaque são vistos também: presidente, isolamento, fazendo, morte, Bolsonaro, ficar e cloroquina. Muitos desses termos citados são também relacionados ao momento da pandemia, mas um se trata de um remédio (Cloroquina), anunciado pelo presidente como uma possível maneira de conter o vírus.

Esse mesmo remédio não é visto como termo entre as palavras na nuvem plotada para o mesmo período, com *tweets* rotulados na polaridade positiva na figura 136. Isso mostra que o remédio está relacionado a textos predominantemente negativos, é visto que, uma parte dos usuários no *Twitter* o citaram em contexto negativo. Os outros termos relacionados aos *tweets* de polaridade positiva são: família, vida, pós-quarentena, combate, isolamento e quarentena. Esses termos estão relacionados ao *lockdown* e ao que ocorrerá

Figura 34 – Nuvem de palavras para polaridade Negativa 13 a 21 de maio de 2020 - Curitiba

Fonte: Elaboração própria.

posteriormente. Desse modo, demonstra uma preocupação com as mortes causadas pela doença, e, principalmente, em combater a doença como exemplo os termos de maior destaque: combate e família.

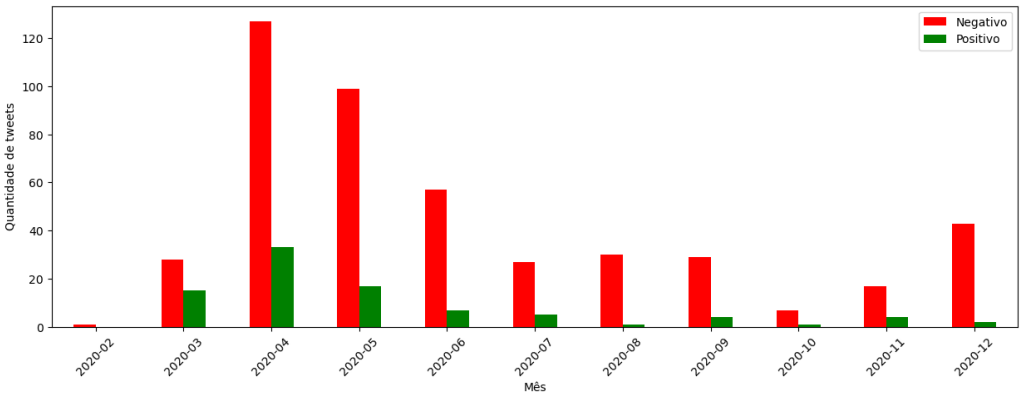
Além do remédio, visto nos *tweets* de polaridade negativa, o termo presidente e Bolsonaro também são muito comuns nas nuvens plotadas para *tweets* rotulados como negativos. Por fim, é visto que os *tweets* coletados para os períodos de quarentena foram majoritariamente negativos, mostrando que os usuários das cidades analisadas expressaram opiniões negativas durante estes momentos de quarentena.

Uma outra medida adotada no país, dessa vez em âmbito nacional, foi o pagamento do chamado auxílio emergencial. Essa foi uma das medidas para combater a vulnerabilidade alimentar que o país poderia cair, devido às outras ações e também aos efeitos da pandemia de COVID-19. Esse auxílio teve seu primeiro pagamento no dia 9 de abril de 2020 e foi uma das medidas importantes aplicada no país. Então, foi realizada uma investigação de como os usuários do *Twitter* expressaram suas opiniões acerca desse auxílio no ano de 2020. Para isso, foi filtrado a partir do *corpus* rotulado, os textos que continham o termo dentro do seu escopo.

O gráfico de barras apresentado na figura 35, mostra a quantidade de *tweets* por mês no ano de 2020 com referência ao auxílio emergencial. Assim como foi observada a polaridade negativa superior nas cidades que aplicaram o lockdown, também houve uma superioridade de textos rotulados como negativo neste tema, no ano de 2020. O termo "auxílio emergencial"é visto, inicialmente, a partir de março, mês que o governo também anunciou a criação do projeto de lei que daria esse auxílio às famílias em vulnerabilidade social.

Em abril é visto um crescimento na quantidade de opiniões postadas no *Twitter*, demonstrando uma grande superioridade da polaridade negativa. Os meses seguintes apre-

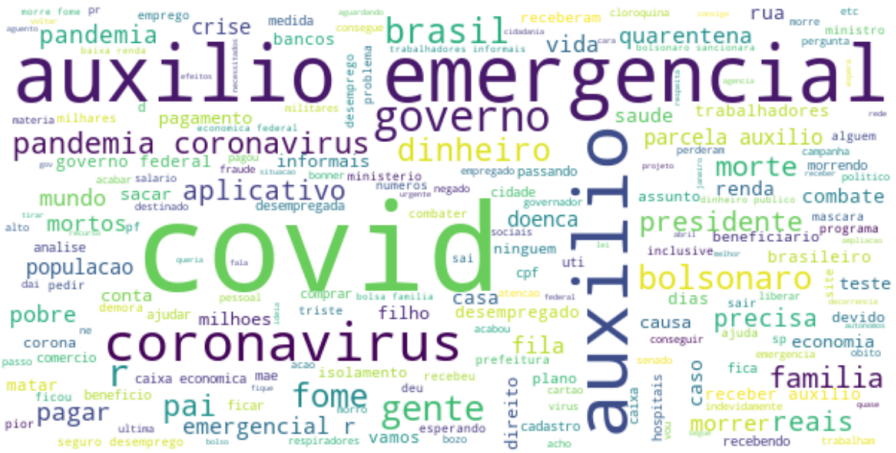
Figura 35 – Comparação de polaridade de*tweets* sobre auxílio emergencial - 2020



Fonte: Elaboração própria.

sentam uma redução, porém com a mesma tendência entre as polaridades, nas quais os textos são sempre rotulados como negativos em maior quantidade. Na busca de avaliar quais termos tiveram maior destaque nos meses de abril, maio e junho, foi feito um gráfico de nuvem de palavras com os textos negativos filtrados com o termo "auxílio emergencial". Esse gráfico é apresentado na figura 36.

Figura 36 – Nuvem de palavras para polaridade Negativa 1 de abril a 30 de junho de 2020, textos sobre auxílio emergencial



Fonte: Elaboração própria.

A partir da nuvem de palavras na figura 36, é visto que auxilio emergencial é o termo com maior destaque, porém isso era esperado, pois os textos foram filtrados a partir deste termo. As outras palavras em evidência são: "parcela auxílio", dinheiro, cadastro, covid, fila, governo, conta, aplicativo. O termo dinheiro faz referência ao que seria recebido através do auxílio emergência, já o termo parcela se refere às parcelas divulgadas pelo governo que seriam pagas. Os termos cadastro, conta e aplicativo têm relação por se tratarem da plataforma criada pela Caixa para o pagamento das famílias cadastradas no auxílio emergencial. Além disso, nos meses iniciais houve muita procura para tirar dúvidas

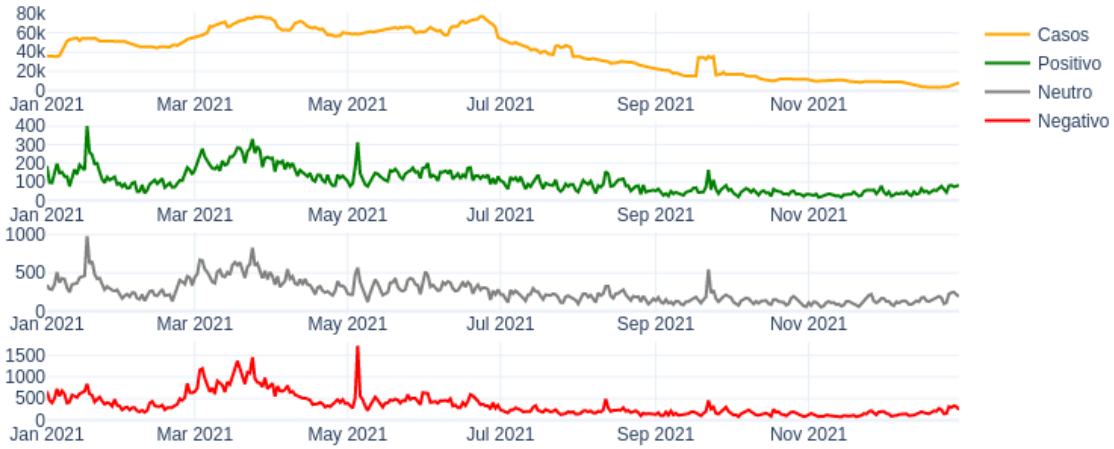
sobre o cadastramento e quem deveria recebê-los.

Porém, apesar dessa plataforma disponibilizada pelo governo, muitas famílias não dispunham de acesso a uma ferramenta ou entendimento de como fazê-lo. Desse modo, muitas pessoas necessitaram se dirigir aos bancos durante a pandemia e o *lockdown* para poder realizar o saque desse dinheiro. Assim, são vistos os termos fila, covid, *lockdown*, sair e pandemia que apresentam relação com as enormes filas formadas em frente aos bancos. Isso demonstra que, os usuários que postaram no *Twitter* opinaram de forma negativa com relação aos efeitos causados pela necessidade de obter esse auxílio. Pois muitas pessoas tiveram que sair em meio à pandemia e também durante os decretos de *lockdown* para formar filas e, assim, conseguir obter o valor disposto pelo auxílio emergencial.

Por outro lado, o ano de 2021 não foi marcado por muitos decretos com medidas não farmacêuticas, como os que foram vistos no ano anterior. Isso ocorreu, pois houve o surgimento das vacinas para tratar a COVID-19, porém algumas liberações ocorridas no final do ano anterior, influenciaram no aumento de casos e óbitos naquele ano.

Os gráficos apresentados na figura 124 apresentam a oscilação dos sentimentos rotulados no *corpus* criado para esse estudo e também a evolução diária das infecções por COVID-19, registradas naquele ano. Para conseguir entender a inclinação dos casos registrados, esse número diário de ocorrências da doença são fornecidos por uma média móvel dos últimos 7 dias, fazendo com que a linha apresente as curvas de maneira mais suave.

Figura 37 – Oscilação de sentimento X Evolução dos casos de Covid-19 2021



Fonte: Elaboração própria, dados provenientes da ICICT.

Em janeiro de 2021, como apresentado na figura 124, é visto um pico de *tweets* rotulados nas polaridades positiva e neutra. Enquanto isso, para esse mesmo mês, a polaridade

negativa não apresenta um grande aumento ou pico de *tweets*. Após o mês de fevereiro, os *tweets* rotulados como negativos exibem um grande aumento, enquanto nas outras duas polaridades, esse aumento não tem uma grande inclinação. Esse crescimento entre o final do mês de fevereiro e começo de maio também é visto no número médio de casos diários.

Após setembro é visto um pequeno pico nos casos, mas não é observado nos *tweets* rotulados algum impacto. No resto do ano, tanto os casos, quanto os *tweets* têm uma queda e se mantêm em uma tendência linear. Ao observar os gráficos da oscilação das polaridades e a evolução dos óbitos no apêndice I para o ano de 2021, mostra-se que apesar dos picos nos casos retratado no mês de setembro, os óbitos permaneceram em uma tendência de queda após o mês de julho de 2021, que também segue a queda de *tweets* negativos após esse mesmo mês.

A partir do mês de julho, também é visto a quantidade de *tweets* rotulados como positivo em determinados dias, passarem a quantidade de *tweets* negativos. Isso expõe que, a partir de julho, com a queda na média movel de óbitos e casos no país, os usuários do *Twitter* no Brasil passaram a fazer postagens mais positivas. Além desse fator, em julho de 2021 o índice de vacinação com a primeira dose contra o coronavírus ultrapassava 40% da população.

Para buscar entender o aumento da polaridade negativa entre o final de fevereiro e começo de maio, foi criada uma nuvem de palavras para observar os tópicos em maior destaque na rede social. Essa nuvem de palavra é apresentada na figura 38, no qual os termos com tamanho superior determinam a sua maior incidência entres os textos utilizados para gerar a imagem.

Figura 38 – Nuvem de palavras para polaridade Negativa 20 de fevereiro a 12 de maio de 2021 - Brasil.

Fonte: Elaboração própria.

O termo com maior destaque se trata do acrônimo covid, que divide a visibilidade com coronavírus, Brasil e mundo, ressaltando a volta do aumento de casos e mortes pela doença no país e no mundo. Muitas cidades tiveram a necessidade de voltar a implementar

medidas de isolamento social mais restritas. Um outro termo que aparece e denota a um acontecimento de bastante notoriedade é o "cpi covid". Pois diz respeito a investigação de casos de desvio de dinheiro fornecido pelo governo às cidades. Além disso, os termos presidente e Bolsonaro aparecem novamente, por se tratarem do líder político naquele ano e também por cobranças realizadas pelos usuários aos posicionamentos do então presidente.

Os outros termos que que não foram vistos nas nuvens de palavras anteriores foram: negacionista e "tratamento precoce"são agora vistos. Essas palavras podem estar relacionadas com a insatisfação pelas atitudes dos políticos e suas falas. Assim, é visto que houve uma reação negativa ao modo como foi conduzido o período pandêmico no país, e às restrições impostas em determinadas regiões do Brasil.

Além disso, foram investigados os dias com maior quantidade de *tweets* positivos e negativos, para tentar entender quais foram as notícias ou momentos que tiveram maior impacto nos sentimentos dos usuários do *Twitter*. As datas com as maiores quantidades são apresentadas na tabela 28, que também mostra a quantidade de *tweets* que esse dia teve em cada polaridade.

Tabela 28 – Tabela de datas com maior quantidade de*tweets* rotulados nas polaridades: Positiva e Negativa.

Data	**Polaridade**	**Quantidade**
2021-01-17	Positivo	334
2021-05-05	Negativo	1033

Fonte: Elaboração própria.

A data com maior quantidade de *tweets* positivos foi o dia 17 de janeiro com 334*tweets*. Nesse dia, ocorreu a primeira aplicação da vacina contra a COVID-19, CoronaVac, em uma enfermeira no estado de São Paulo, capital. Ao observar a nuvem de palavras na figura 39, que foi gerada com base nesses 334 *tweets*, é visto que os termos com maior destaque fazem referência a esse fato.

Diante dos termos que podem ser observados na figura 39, é visto que os que detém maior destaque, apresentam relação com a vacina, como: vacina, parabéns, covid, Butantan, aprovada, via, vida, viva ciência. Desse modo, fica claro que o sentimento positivo para aquela data tem total correlação com a vacina aplicada na enfermeira em São Paulo, capital. O nome da própria vacina e o órgão que a liberou em critério de urgência, também é citado e tem destaque, mostrando que os usuários brasileiros do *Twitter* que realizaram publicações naquela data, demonstravam uma reação positiva à ocasião.

Ao analisar o dia com maior quantidade de opiniões rotuladas como negativas por nosso comitê de classificadores na figura 40, é visto que o termo com maior destaque é o nome do humorista Paulo Gustavo. Essa marcante ênfase a esse nome ocorre por seu falecimento no dia anterior, em consequência de complicações da COVID-19. Os usuá-

Figura 39 – Nuvem de palavras para polaridade Positiva 17 de janeiro de 2021 - Brasil

Fonte: Elaboração própria.

rios nessa data expressaram grande tristeza em relação a esse ocorrido, assim, pode-se relacionar as palavras tristeza, triste, "complicações covid", "morte Paulo"entre outras em destaque, ao falecimento do humorista.

Figura 40 – Nuvem de palavras para polaridade Negativa 5 de maio de 2021 - Brasil

Fonte: Elaboração própria.

Além dessas palavras, se mantêm em destaque as expressões: "cpi covid", vacina, presidente, Bolsonaro e morte, que estão ligadas ao desconforto demonstrado pelos usuários em relação às políticas aplicadas no país. A manutenção desses termos se mantêm em todo o período de pandemia, para os meses iniciais do ano de 2021, o que remete ao crescimento dos casos no país. De acordo com estudos (MORAES, 2021; MOURA et al., 2021; OLIVEIRA et al., 2021), esse crescimento ocorreu pela chegada da segunda onda da pandemia no país, em decorrência das liberações e abrandamento das medidas restritivas nas cidades

brasileiras com a chegada do fim do ano de 2020.

Essa segunda onda ocorreu após o mês de janeiro de 2021 e teve seu pico no final do mês de março. Iisso foi visto através dos usuários, que postaram muitas opiniões negativas nos meses de março e abril. Apesar de ter ocorrido a vacinação em janeiro, o país só bateu mais de 50% da população vacinada, com a 1ª dose, após agosto. A partir da segunda dose depois de outubro, é quando se pode observar uma queda de fato, tanto no uso dos termos relacionados à COVID-19 nas postagens, como também, na polaridade negativa. Os gráficos mostrando essa relação estão apresentados no apêndice I.

Após o pico de casos e óbitos ocorridos na segunda onda da pandemia de COVID-19, no ano de 2021, o número de *tweets* relacionado ao tema começou a sofrer uma queda. Desse modo, houve uma equiparação nas quantidades rotuladas na polaridade negativa com a positiva, como visto na figura 124, mostrando que os usuários expressaram opiniões mais positivas com o decorrer da redução de fatalidades pela doença. De acordo com nosso comitê classificador, pode-se afirmar que existe uma correlação entre as ocorrências da pandemia de COVID-19 e as opiniões dos usuários.

Por fim, é visto que com o modelo aqui treinado, o *corpus* montado para este trabalho teve majoritariamente rótulos na polaridade negativa durante o período da pandemia. Assim como visto na etapa de validação do modelo, o rótulo positivo tem melhores índices de classificação que as outras duas polaridades. Pois, existe uma classificação errada entre a polaridade negativa e neutra, por terem sempre termos próximos com relação à COVID-19.

Essa ligação que há entre essa duas polaridades, faz com que muitas notícias, que deveriam ser vistas como neutras, sejam também classificadas como negativas. Desse modo, faz com que esse rótulo tenha uma maior quantidade que os demais. Além disso, durante a pandemia a quantidade de notícias acerca do vírus, utilizando os termos usados para coletar os textos, foram crescendo à medida que passava o período de pandemia. Por esse motivo, o número de *tweets* neutros e negativos sempre são mostrados superiores aos positivos.

5.4 CONCLUSÃO

Portanto, foi visto neste capítulo diversos experimentos e validações de seus resultados, para construir o modelo que classificou o *corpus* proposto para este estudo. Entre as técnicas aplicadas, o estudo das metodologias na etapa de pré-processamento foi demonstrado como importante. Ao ser analisado o uso das técnicas *Stemming* e *Lemmatization* em conjunto com o dicionário léxico *NetSpeak-BR*, foi notada uma melhoria nos resultados para o *corpus* disponibilizado no *Kaggle*.

Além disso, foram avaliadas as metodologias *word embbending* em conjunto com as listas de palavras produzidas pelas processos anteriores. Assim, através dos vetores produzidos pelas técnicas *word embbending*, foi realizada uma classificação para validar os

resultados e escolher qual vetor seria usado no treinamento das redes de codificadores automáticos. Desse modo, foi visto que os vetores TFIDF com o parâmetro *n-gram*= (1,1), produziram os classificadores de maior qualidade.

Nessa etapa, foi realizado um estudo de parâmetros das redes de CA, de acordo com o que foi visto no estudo de Ameur, Jamoussi e Hamadou (2018). Foi percebido que, que diferente dos resultados visto pelos autores, não foram atingidos bons resultados, reduzindo o vetor a 20% de sua dimensionalidade. Pois, quanto maior a quantidade de nós na camada de saída do codificador, ainda demonstrava uma melhoria na qualidade, independente da quantidade de camadas na rede.

Desse modo, foi utilizada a rede com menor quantidade de nós, resultantes na codificação com a melhor qualidade. Nos testes com classificadores multi-classe, foi notado que o resultado dos algoritmos KNN e QDA foram superiores, quando treinados com o vetor representação, produzido pela rede de codificadores automáticos. Para os demais classificadores, foram vistos resultados muito próximos no teste. Mostrando que, a utilização da técnica produziu medidas equiparáveis aos que foram vistos com o vetor original.

Por fim, foram experimentados os algoritmos com maior qualidade em diversos comitês de classificadores. Isso foi feito, para encontrar um melhor resultado na acurácia e demais índices de qualidades dos modelos. Então através desse comitê, foram rotulados os *corpus* a serem estudados para avaliar a polaridade evidenciada pelos usuários do *Twitter*.

Em suma, pode-se, por meio de toda a experimentação e avaliação do agrupamento de diversas técnicas, alcançar resultados equiparáveis aos que foram vistos nos trabalhos da literatura. É mostrado que como alguns algoritmos produziram resultados superiores através da redução da dimensionalidade por CA, poderiam alcançar maior qualidade, se fossem explorados mais parâmetros da rede de codificadores. Além disso, com a redução da dimensionalidade não vinculada a um modelo, foi realizado um estudo mais amplo através das metodologias da AM.

6 CONCLUSÕES E TRABALHOS FUTUROS

Nesta dissertação foi abordada a tarefa de análise de sentimentos automático através de vetores reduzidos por meio da rede de aprendizado profundo *autoencoder*. A pesquisa visa reduzir o custo computacional envolvido na tarefa de classificação dos sentimentos e ainda assim manter sua qualidade próxima as que foram vistas nos trabalhos de estado da arte.

Além disso, o presente trabalho possui como objetivo a melhoria dos resultados da rede de codificadores automáticos foram utilizadas diversas técnicas de pré-processamento de dados para elaborar um conjunto que produzisse uma qualidade superior as demais. Também, para a classificação foi estudado diversos classificadores únicos para escolher um conjunto de classificador e aplica-los a um comitê para produzir uma qualidade melhor. Assim, rotular o *corpus* coletado durante os dois anos de pandemia (2020-2021) contextualizados sobre as *hashtag's* mais utilizadas sobre o tema da COVID-19.

O estudo buscou realizar analises comparativas entre diversos métodos de pré-processamento, processamento e pós para escolher o conjunto de técnicas mais adequados para o estudo. De modo a desenvolver um modelo treinado com uma exploração e validação que fundamente o seu uso na tarefa de predição do *corpus* coletado. Visando alcançar esse resultado foram utilizados dois *corpus* rotulados para o treino dos modelos, um *corpus* contextualizado sobre a COVID-19 e um sem tema especifico, sendo eles, o *corpus* TBCOV e o *kaggle*, respectivamente.

A partir dessa exploração dos *corpus* rotulados foi possível responder como é a diferença entre um *corpus* contextualizado e um sem tema especifico. Foi visto que, existe de fato uma diferença entre eles, o *corpus* generalista demonstrou ter uma menor qualidade em seus resultados, mesmo com a exploração de técnicas e parâmetros dos algoritmos. Isso ocorre por não se ter um vocabulário muito especifico que denote cada polaridade fazendo com que os algoritmos não consigam criar uma grande distinção entre os rótulos negativo, positivo e neutro.

Por outro lado, com o *corpus* contextualizado sobre o tema da COVID-19, demonstrou um grau maior de qualidade em seus testes. Onde foi observado um índice superior a todos os resultados visto com o *corpus* anterior. Sabendo que, com esse *corpus* disponibilizado pelo trabalho de Imran, Qazi e Ofli (2022a), o vocabulário demonstrado apresenta uma maior distinção entre o positivo e o negativo e neutro. Onde para esses dois últimos (negativo e neutro) se vê uma maior proximidade por existirem muitos termos relacionados a óbitos e casos.

Desse modo, o *corpus* TBCOV produziu qualidades superiores as que foram vistas no outro, obtendo 80% de acurácia e precisão nos testes de nós e camadas com o vetor produzido pelo *autoencoder*. Enquanto no *kaggle* foi visto, em seu melhor resultado de

teste, uma acurácia de 66%. Por tais resultados observados, foi escolhido o uso do *corpus* TBCOV, produzido por um algoritmo multi-linguístico, para realizar a classificação e predição do *corpus* que foi criado para esta dissertação.

Portanto, através da utilização da rede neural redutora da dimensionalidade desassociado do algoritmo de classificação, possibilita a compreensão dos resultados de diversos classificadores da literatura. Através disso, foi possível avaliar quais, destes algoritmos, apresentaram um grau maior de qualidade em seus resultados. Os modelos tradicionais da AM, o que mostrou maior qualidade foi o SVM, obtendo acurácias com até 15% a mais que os outros.

Além desse algoritmo, foi visto que algoritmos como *XGboost*, *RandomForest*, MLP e o *CatBoost* alcançaram resultados próximo ou acima de 80% de acurácia no teste. Assim, os classificadores únicos mostraram-se com capacidade de produzir a predição equiparável ao comitê de arvores aleatórias (*RandomForest*). Portanto, foi possível ver uma melhora do resultado dos testes dos algoritmos QDA e KNN com o uso dos vetores de dimensionalidade reduzida através das redes de codificadores automáticos. Assim foi observado que algoritmos tradicionais reproduziram uma qualidade inferior aos mais utilizados na literatura, e com o uso do RCA alguns dos algoritmos obtiveram uma melhoria.

Desse modo, foram escolhidos cinco algoritmos para compor comitês de classificadores com o intuito de avaliar se através dessa técnica era possível produzir qualidades ainda melhores que as demais já investigadas. As técnicas tradicionais foram utilizadas, como SVM e MLP juntamente com os dois algoritmos que apresentaram um aumento de sua qualidade com a redução da dimensionalidade, QDA e KNN. Pois, foi visto como uma forma de generalizar o comitê e produzir resultados que englobassem diferentes predições durante a classificação.

E realmente, foi percebida uma melhoria na qualidade obtida com o comitê, no qual diferente dos classificadores únicos, esse agrupamento conseguiu chegar a um resultado próximo aos 90% de acurácia no teste. Isso mostrou sua importância para o trabalho, e assim foram experimentadas diversas técnicas de comitês classificadores que possibilitavam o treino com multi-classes. Alguns algoritmos foram estudados e entre eles se destacou o *Stacked Generalization* que foi utilizado para produzir a polaridade dos textos do *corpus* criado.

A utilização da RCA para reduzir a dimensionalidade do vetor produzido pelo *corpus* rotulado, de forma independente, mostrou alcançar resultados equiparáveis aos obtidos com o vetor original. A partir dos resultados obtidos no testes com algoritmos tradicionais da AM e os que demonstraram maior qualidade na literatura, é visto que a utilização dessa redução da dimensionalidade pode manter até 97% da qualidade obtida sem seu uso. Assim, demonstra que o uso das RCA produzem resultado equiparáveis aos que são obtidos sem a interferência desta técnica.

Então, com a qualidade de 0,88 de acurácia no teste, o *corpus* Brasil-Covid-Pandemia

(2020-2021) foi rotulado pelo comitê classificador *stacked generalization* com os algoritmos SVM, MLP, QDA, KNN e árvore de decisões. A partir desses textos categorizados dentro das polaridades negativas, positivas e neutras, então, foi possível analisar cada período da pandemia e como influenciaram na emoção dos usuários do *Twitter*. Os períodos analisados nesta dissertação foram momentos importantes da pandemia, como a sua chegada no Brasil, o meses iniciais e de picos de casos e óbitos. Por fim, também foram analisados momentos com maior incidência das polaridades positiva e negativa.

A partir dos resultados da predição, foi possível observar que durante todo o período pandêmico do ano de 2020, a polaridade negativa se destacava em relação às demais. No início da pandemia, nos meses de março e abril, junto com os primeiro picos dos índices da COVID-19, a predominância da emoção negativa era maior que as demais polaridades. Além disso, outro fator observado, foi que a curva de *tweets* rotulados como positivo e negativo seguiam o índice de óbitos e casos disponibilizados pelo ICICT. Demonstrando que durante o aumento dos casos e óbitos da doença no país, assim como no período de aplicação de medidas rígidas o sentimento predominante era o negativo, e o positivo oscilava de acordo com os números observados na base de dados.

Buscando entender os sentimentos restritos aos cidades-estados (Fortaleza-CE, Niterói-RJ e Curitiba-PR) que aplicaram o *lockdown* foi observado o alto número de *tweets* no mês de abril e maio de 2020. No qual foi visto que a predominância do sentimento negativo estaria relacionado à aplicação dessa medida não farmacológica, que de acordo com os resultados obtidos nesta dissertação. Isso expõe que, inicialmente, tais medidas com alta rigidez provocaram desconforto nos usuários por terem de se habituar a um novo contexto. Além disso também foi possível correlacionar as mortes aos sentimentos negativos através dos gráficos de nuvens de palavras apresentados.

Diante das discussões apresentadas, foi possível compreender que, nos estados que foram aplicados o lockdown, a polaridade predominante foi a negativa. Pois, através da nuvens de palavras é possível relacionar o alto número com a vontade dos usuários de desejarem que tudo o que estava ocorrendo devido ao vírus da COVID-19 passe e que os dias voltem a ser como antes da pandemia. Já nos textos com a emoção positiva, as palavras que mais se destacavam eram relacionadas ao pedido de cuidado com as vidas das pessoas.

Em suma, o sentimento mais evidente em todo o período da pandemia analisado é o negativo. A cada período esse sentimento está relacionado a um momento especifico em alguns momentos, a polaridade negativa associada à quantidade de mortes, à necessidade de permanecer em suas residências, aos casos da doença e também a falta de políticas utilizadas no ano de 2020.

Portanto, a partir do modelo pré-treinado da rede de codificadores automático disponibilizado através deste trabalho, a etapa de treinamento custosa dessas redes é eliminada. Porém, esse modelo pré-treinado apenas serve para o estudo de refinamento dos parâ-

metros dos modelos classificadores em trabalhos futuros e não da rede de codificadores. A utilização dessa rede de codificadores sobre *corpus* em português busca validar seus resultados em outro contexto atrelado a saúde pública ou períodos pandêmicos.

Desse modo, além do seu uso interligado ao refinamento dos modelos classificadores, podem ser vistos também, como trabalhos futuros, a exploração dos parâmetros das redes codificadoras, buscando melhorar o desempenho dos modelos através dos vetores de representação gerados. Da mesma forma, um refinamento dos parâmetros dos comitês utilizados, visando obter um maior desempenho no modelo e utilizando um *corpus* da língua portuguesa com redução de sua dimensionalidade.

REFERÊNCIAS

ABEDIN, A. F.; MAMUN, A. I. A.; NOWRIN, R. J.; CHAKRABARTY, A.; MOSTAKIM, M.; NASKAR, S. K. A deep learning approach to integrate human-level understanding in a chatbot. *arXiv preprint arXiv:2201.02735*, 2021.

AKEN, B. van; HERRMANN, S.; LÖSER, A. What do you see in this patient? behavioral testing of clinical nlp models. *arXiv preprint arXiv:2111.15512*, 2021.

AMEUR, H.; JAMOUSSI, S.; HAMADOU, A. B. A new method for sentiment analysis using contextual auto-encoders. *Journal of Computer Science and Technology*, Springer, v. 33, n. 6, p. 1307–1319, 2018.

ANTÓN, M. Los eventos de deportes electrónicos (esports) como herramienta de promoción turística. In: *Actas XII Congreso Virtual Internacional sobre Turismo y Desarrollo*. [S.l.: s.n.], 2018. p. 77–89.

ARIA, M.; CUCCURULLO, C.; D'ANIELLO, L.; MISURACA, M.; SPANO, M. Thematic analysis as a new culturomic tool: The social media coverage on covid-19 pandemic in italy. *Sustainability*, MDPI, v. 14, n. 6, p. 3643, 2022.

BAR, Y.; BAR, K.; ITZHAK, I.; NISELBAUM, C. S.; DERSHOWITZ, N.; SHACHAR, E.; WEISS-MEILIK, A.; GOLAN, O.; WOLF, I.; MENES, T. et al. The impact of tumor detection method on genomic and clinical risk and chemotherapy recommendation in early hormone receptor positive breast cancer. *The Breast*, Elsevier, v. 60, p. 78–85, 2021.

BARBERIA, L. G.; CANTARELLI, L. G.; OLIVEIRA, M. L. C. d. F.; MOREIRA, N. d. P.; ROSA, I. S. C. The effect of state-level social distancing policy stringency on mobility in the states of brazil. *Revista de Administração Pública*, SciELO Brasil, v. 55, p. 27–49, 2021.

BARBOSA, A. F.; CAMPELO, C. Processamento de linguagem natural em artefatos textuais educacionais: Um mapeamento sistemático no contexto brasileiro. In: SBC. *Anais do XXXI Simpósio Brasileiro de Informática na Educação*. [S.l.], 2020. p. 1433–1442.

BARRETO, I. C. d. H. C.; FILHO, R. V. C.; RAMOS, R. F.; OLIVEIRA, L. G. d.; MARTINS, N. R. A. V.; CAVALCANTE, F. V.; ANDRADE, L. O. M. d.; SANTOS, L. M. P. Colapso na saúde em manaus: o fardo de não aderir às medidas não farmacológicas de redução da transmissão da covid-19. *Saúde em debate*, SciELO Public Health, v. 45, p. 1126–1139, 2021.

BEHL, S.; RAO, A.; AGGARWAL, S.; CHADHA, S.; PANNU, H. Twitter for disaster relief through sentiment analysis for covid-19 and natural hazard crises. *International journal of disaster risk reduction*, Elsevier, v. 55, p. 102101, 2021.

BELLODI, E.; BERTAGNON, A.; GAVANELLI, M. Comparing emotion and sentiment analysis tools on italian anti-vaccination for covid-19 posts. In: *Proceedings of the Sixth Workshop on Natural Language for Artificial Intelligence (NL4AI 2022) co-located with*

21th International Conference of the Italian Association for Artificial Intelligence (AI IA 2022)*. [S.l.: s.n.], 2022.

BENEVENUTO, F.; RIBEIRO, F.; ARAÚJO, M. Métodos para análise de sentimentos em mídias sociais. *Sociedade Brasileira de Computação*, 2015.

BHARTI, U.; BAJAJ, D.; BATRA, H.; LALIT, S.; LALIT, S.; GANGWANI, A. Medbot: Conversational artificial intelligence powered chatbot for delivering tele-health after covid-19. In: IEEE. *2020 5th international conference on communication and electronics systems (ICCES)*. [S.l.], 2020. p. 870–875.

BONTA, V.; JANARDHAN, N. K. N. A comprehensive study on lexicon based approaches for sentiment analysis. *Asian Journal of Computer Science and Technology*, v. 8, n. S2, p. 1–6, 2019.

CALADO, R. B. *Mineração de Dados Não Estruturados Utilizando uma Combinação de Redes Complexas e Ensemble Dinâmico*. Dissertação (Mestrado) — Universidade de Pernambuco, 2019.

CAMBRIA, E.; LI, Y.; XING, F. Z.; PORIA, S.; KWOK, K. Senticnet 6: Ensemble application of symbolic and subsymbolic ai for sentiment analysis. In: *Proceedings of the 29th ACM international conference on information & knowledge management*. [S.l.: s.n.], 2020. p. 105–114.

CANDIDO, D. D. S.; WATTS, A.; ABADE, L.; KRAEMER, M. U.; PYBUS, O. G.; CRODA, J.; OLIVEIRA, W. D.; KHAN, K.; SABINO, E. C.; FARIA, N. R. Routes for covid-19 importation in brazil. *Journal of Travel Medicine*, Oxford University Press, v. 27, n. 3, p. taaa042, 2020.

CHAKRABORTY, K.; BHATIA, S.; BHATTACHARYYA, S.; PLATOS, J.; BAG, R.; HASSANIEN, A. E. Sentiment analysis of covid-19 tweets by deep learning classifiers—a study to show how popularity is affecting accuracy in social media. *Applied Soft Computing*, Elsevier, v. 97, p. 106754, 2020.

CHEN, L. Curse of dimensionality. In: _____. *Encyclopedia of Database Systems*. Boston, MA: Springer US, 2009. p. 545–546. ISBN 978-0-387-39940-9. Disponível em: <https://doi.org/10.1007/978-0-387-39940-9_133>.

COVID, E. E. Principais variantes do sars-cov-2 notificadas no brasil. *RBAC*, v. 53, n. 2, p. 109–116, 2021.

DARWIS, S. A.; PHAM, D. N.; PHENG, A. J.; HOE, O. H. Evaluating the impact of removing less important terms on sentiment analysis. *researchgate*, p. 6–18, 2019.

DAS, N.; SADHUKHAN, B.; CHATTERJEE, T.; CHAKRABARTI, S. Effect of public sentiment on stock market movement prediction during the covid-19 outbreak. *Social network analysis and mining*, Springer, v. 12, n. 1, p. 1–22, 2022.

DEEPAK, S.; CHITTURI, B. Deep neural approach to fake-news identification. *Procedia Computer Science*, Elsevier, v. 167, p. 2236–2243, 2020.

DESHMUKH, P. R.; PHALNIKAR, R. Information extraction for prognostic stage prediction from breast cancer medical records using nlp and ml. *Medical & Biological Engineering & Computing*, Springer, v. 59, n. 9, p. 1751–1772, 2021.

EDARA, D. C.; VANUKURI, L. P.; SISTLA, V.; KOLLI, V. K. K. Sentiment analysis and text categorization of cancer medical records with lstm. *Journal of Ambient Intelligence and Humanized Computing*, Springer, p. 1–17, 2019.

ELAYAN, S.; SYKORA, M.; SHANKARDASS, K.; ROBERTSON, C.; FEICK, R.; SHAUGHNESSY, K.; HAYDN, L.; JACKSON, T. The stresscapes ontology system: Detecting and measuring stress on social media. In: *7th European Conference on Social Media ECSM 2020*. [S.l.: s.n.], 2020. p. 74–82.

ENGELMANN, D.; PANISSON, A.; MAGISTRALI, L.; ROSA, J.; CARLOTTO, E.; DAHLEM, J.; TELLI, D.; VIEIRA, R.; BORDINI, R.; CALIENDO, P. Inteligência artificial no apoio à tomada de decisões no direito tributário. *Revista de Direitos Fundamentais e Tributação*, v. 1, p. 45–64, 2020.

FARIAS, F. L. *Aplicação de mineração de textos e análise de sentimentos a postagens do Twitter acerca das vacinas contra a Covid-19*. Dissertação (Mestrado) — UFVJM, 2022.

FARIAS, F. L.; OLIVEIRA, L. S. C. de. Mineração de textos e análise de sentimentos aplicados a postagens do twitter acerca das vacinas contra a covid-19. *Research, Society and Development*, v. 11, n. 13, p. e364111335490–e364111335490, 2022.

FRONDELIUS, T.; ATKOVA, I.; MIETTUNEN, J.; RELLO, J.; JANSSON, M. M. Diagnostic and prognostic prediction models in ventilator-associated pneumonia: Systematic review and meta-analysis of prediction modelling studies. *Journal of Critical Care*, Elsevier, v. 67, p. 44–56, 2022.

GADINI, S. L. Uma outra 'fábrica de mentiras': Análise de notícias falsas no twitter pelas agência lupa, aos fatos e boatos. org sobre a pandemia do coronavírus no brasil. *Jornalismo em tempos da pandemia do novo coronavírus*, Ria Editorial, p. 399, 2020.

GASPAR, J. d. S.; REIS, Z. S. N.; OLIVEIRA, I. J. R. d.; SILVA, A. P. C. d.; DIAS, C. d. S. Introdução à análise de dados em saúde com python. In: *Introdução à análise de dados em saúde com Python*. [S.l.: s.n.], 2023. p. 130–130.

GENCOGLU, O. Large-scale, language-agnostic discourse classification of tweets during covid-19. *Machine Learning and Knowledge Extraction*, Multidisciplinary Digital Publishing Institute, v. 2, n. 4, p. 603–616, 2020.

GHOSH, R.; RAVI, K.; RAVI, V. A novel deep learning architecture for sentiment classification. In: IEEE. *2016 3rd international conference on recent advances in information technology (RAIT)*. [S.l.], 2016. p. 511–516.

GOMES, A. M. *Análise de sentimentos dos conteúdos gerados pelos usuários no Twitter a partir das comunicações dos políticos em relação a vacina da Covid-19*. Dissertação (Mestrado) — Universidade Federal do Rio Grande do Norte, 2022.

GUPTA, N.; AGRAWAL, R. Application and techniques of opinion mining. In: *Hybrid Computational Intelligence*. [S.l.]: Elsevier, 2020. p. 1–23.

HAJIBABAEE, P.; MALEKZADEH, M.; HEIDARI, M.; ZAD, S.; UZUNER, O.; JONES, J. H. An empirical study of the graphsage and word2vec algorithms for graph multiclass classification. In: IEEE. *2021 IEEE 12th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*. [S.l.], 2021. p. 0515–0522.

HAMID, A.; SHIEKH, N.; SAID, N.; AHMAD, K.; GUL, A.; HASSAN, L.; AL-FUQAHA, A. Fake news detection in social media using graph neural networks and nlp techniques: A covid-19 use-case. *arXiv preprint arXiv:2012.07517*, 2020.

HAN, J.; KAMBER, M.; PEI, J. Data mining concepts and techniques third edition. *University of Illinois at Urbana-Champaign Micheline Kamber Jian Pei Simon Fraser University*, 2012.

HUAI, S.; VOORDE, T. Van de. Which environmental features contribute to positive and negative perceptions of urban parks? a cross-cultural comparison using online reviews and natural language processing methods. *Landscape and Urban Planning*, Elsevier, v. 218, p. 104307, 2022.

IGLECIAS, W. T. Respostas à pandemia: a experiência brasileira em comparação com outros países da américa latina. *Políticas públicas e Covid-19: a experiência brasileira*, p. 17–42, 2022.

IMRAN, M.; QAZI, U.; OFLI, F. Tbcov: Two billion multilingual covid-19 tweets with sentiment, entity, geo, and gender labels. *Data*, Multidisciplinary Digital Publishing Institute, v. 7, n. 1, p. 8, 2022.

IMRAN, M.; QAZI, U.; OFLI, F. Tbcov: Two billion multilingual covid-19 tweets with sentiment, entity, geo, and gender labels. *Data*, v. 7, n. 1, 2022. ISSN 2306-5729. Disponível em: <https://www.mdpi.com/2306-5729/7/1/8>.

JACQUES, N.; SILVEIRA, M. F. d.; HALLAL, P. C.; MENEZES, A.; HORTA, B. L.; MESENBURG, M. A.; HARTWIG, F. P.; BARROS, A. J. Uso de máscara durante a pandemia de covid-19 no brasil: resultados do estudo epicovid19-br. *Cadernos de Saúde Pública*, SciELO Public Health, v. 38, p. e00271921, 2022.

KAUR, S.; KAUL, P.; ZADEH, P. M. Monitoring the dynamics of emotions during covid-19 using twitter data. *Procedia Computer Science*, Elsevier, v. 177, p. 423–430, 2020.

KHADIJA, A.; ZAHRA, F. F.; NACEUR, A. Ai-powered health chatbots: Toward a general architecture. *Procedia Computer Science*, Elsevier, v. 191, p. 355–360, 2021.

KHALIL, F.; PIPA, G. Is deep-learning and natural language processing transcending the financial forecasting? investigation through lens of news analytic process. *Computational Economics*, Springer, v. 60, n. 1, p. 147–171, 2022.

KIM, K. An improved semi-supervised dimensionality reduction using feature weighting: Application to sentiment analysis. *Expert Systems with Applications*, Elsevier, v. 109, p. 49–65, 2018.

KIM, T.; LEE, B. Multi-attention multimodal sentiment analysis. In: *Proceedings of the 2020 International Conference on Multimedia Retrieval*. [S.l.: s.n.], 2020. p. 436–441.

KOH, J. X.; LIEW, T. M. How loneliness is talked about in social media during covid-19 pandemic: Text mining of 4,492 twitter feeds. *Journal of psychiatric research*, Elsevier, 2020.

KOH, J. X.; LIEW, T. M. How loneliness is talked about in social media during covid-19 pandemic: Text mining of 4,492 twitter feeds. *Journal of psychiatric research*, Elsevier, v. 145, p. 317–324, 2022.

LEE, G.; NHO, K.; KANG, B.; SOHN, K.-A.; KIM, D. Predicting alzheimer's disease progression using multi-modal deep learning approach. *Scientific reports*, Nature Publishing Group, v. 9, n. 1, p. 1–12, 2019.

LEVIS, M.; WESTGATE, C. L.; GUI, J.; WATTS, B. V.; SHINER, B. Natural language processing of clinical mental health notes may add predictive value to existing suicide risk models. *Psychological medicine*, Cambridge University Press, v. 51, n. 8, p. 1382–1391, 2021.

LI, M.; XU, H.; TU, Z.; SU, T.; XU, X.; WANG, Z. A deep learning based personalized qoe/qos correlation model for composite services. In: IEEE. *2022 IEEE International Conference on Web Services (ICWS)*. [S.l.], 2022. p. 312–321.

MAIA, A. G.; MARTINEZ, J. D. M.; MARTELETO, L. J.; SERENO, L. G.; GUIMARAES, C. Big data de redes sociais pode prever focos de epidemias? 2021.

MAIA, N. N.; SALTON, G. D. Utilização de machine learning para classificação de sentimentos no idioma portguês-brasil using machine learning for sentiment classification in the brazilian-portuguese language. *Brazilian Journal of Development*, v. 8, n. 6, p. 43568–43580, 2022.

MANDELA, N. Autobiografia de nelson mandela, um longo caminho para a liberdade (trad. port., no original long walk to freedom). *Planeta, Lisboa*, 2016.

MANGURI, K. H.; RAMADHAN, R. N.; AMIN, P. R. M. Twitter sentiment analysis on worldwide covid-19 outbreaks. *Kurdistan Journal of Applied Research*, p. 54–65, 2020.

MARTINS, A. F. M. *A utilização das redes sociais para comunicar ciência: o caso do INL*. Tese (Doutorado), 2023.

MASSONI, G. Análise de textos por meio de processos estocásticos na representação word2vec. Universidade Federal de São Carlos, 2021.

MATSUMOTO, G. H. P. Fatores de sucesso para canais de live streaming de jogos online na percepção dos usuários brasileiros da twitch. tv. Universidade Federal do Rio de Janeiro, 2019.

MCSHANE, M.; NIRENBURG, S. *Linguistics for the Age of AI*. [S.l.]: Mit Press, 2021.

MELLO, M. R. G. de; CAMILLO, E. da S.; SANTOS, B. R. P. dos. Big data e inteligência artificial: aspectos éticos e legais mediante a teoria crítica. *Complexitas–Revista de Filosofia Temática*, v. 3, n. 1, p. 50–60, 2019.

MERRYTON, A. R.; AUGASTA, M. G. An empirical analysis of fake news detection with nlp and machine learning techniques. *Advanced Engineering Science*, ResearchGate, v. 54, p. 1351–1363, 2022.

MOHAN, S.; SOLANKI, A. K.; TALUJA, H. K.; SINGH, A. et al. Predicting the impact of the third wave of covid-19 in india using hybrid statistical machine learning models: A time series forecasting and sentiment analysis approach. *Computers in Biology and Medicine*, Elsevier, v. 144, p. 105354, 2022.

MORAES, R. F. d. A segunda onda da pandemia (mas não do distanciamento físico): covid-19 e políticas de distanciamento social dos governos estaduais no brasil. Instituto de Pesquisa Econômica Aplicada (Ipea), 2021.

MORSHED, S. A.; KHAN, S. S.; TANVIR, R. B.; NUR, S. Impact of covid-19 pandemic on ride-hailing services based on large-scale twitter data analysis. *Journal of Urban Management*, Elsevier, v. 10, n. 2, p. 155–165, 2021.

MOURA, E. C.; SILVA, E. N. d.; SANCHEZ, M. N.; CAVALCANTE, F. V.; OLIVEIRA, L. G. d.; OLIVEIRA, A.; FRIO, G. S.; SANTOS, L. M. P. Disponibilidade de dados públicos em tempo oportuno para a gestão: análise das ondas da covid-19. 2021.

NA, T.; CHENG, W.; LI, D.; LU, W.; LI, H. Insight from nlp analysis: Covid-19 vaccines sentiments on social media. *arXiv preprint arXiv:2106.04081*, 2021.

NANDINI, G. S.; KUMAR, A. S.; CHIDANANDA, K. Dropout technique for image classification based on extreme learning machine. *Global Transitions Proceedings*, Elsevier, v. 2, n. 1, p. 111–116, 2021.

NASCIMENTO, R. da S.; SANTOS, G. dos; CARVALHO, F.; GUEDES, G. Avaliando contribuições na substituição de termos informais em classificação de texto de redes sociais com netspeak-br. In: SBC. *Anais do X Brazilian Workshop on Social Network Analysis and Mining*. [S.l.], 2021. p. 181–186.

NASEEM, U.; RAZZAK, I.; KHUSHI, M.; EKLUND, P. W.; KIM, J. Covidsenti: A large-scale benchmark twitter data set for covid-19 sentiment analysis. *IEEE Transactions on Computational Social Systems*, IEEE, v. 8, n. 4, p. 1003–1015, 2021.

OLIVEIRA, L. R.; GOUVEIA, A. S. A.; MATIAS, D. A.; SILVA, W. S.; SANTOS, V. D. dos; TOAZZA, M. R. Análise epidemiológica da segunda onda de covid-19 no estado da bahia. *Revista Eletrônica Acervo Saúde*, v. 13, n. 4, p. e7006–e7006, 2021.

OMUYA, E. O.; OKEYO, G.; KIMWELE, M. Sentiment analysis on social media tweets using dimensionality reduction and natural language processing. *Engineering Reports*, Wiley Online Library, p. e12579, 2022.

OMUYA, E. O.; OKEYO, G.; KIMWELE, M. Sentiment analysis on social media tweets using dimensionality reduction and natural language processing. *Engineering Reports*, Wiley Online Library, v. 5, n. 3, p. e12579, 2023.

ORENGO, V. M.; HUYCK, C. R. A stemming algorithmm for the portuguese language. In: *spire*. [S.l.: s.n.], 2001. v. 8, p. 186–193.

OYEBODE, O.; ORJI, R. Deconstructing persuasive strategies in mental health apps based on user reviews using natural language processing. In: *BCSS@ PERSUASIVE*. [S.l.: s.n.], 2020.

OYEBODE, O.; ORJI, R. Persuasive strategies in mental health apps: A natural language processing approach. In: *PERSUASIVE (Adjunct)*. [S.l.: s.n.], 2020.

PALANI, S.; RAJAGOPAL, P.; PANCHOLI, S. T-bert–model for sentiment analysis of micro-blogs integrating topic model and bert. *arXiv preprint arXiv:2106.01097*, 2021.

PATEL, R.; CHOUDHARY, V.; SAXENA, D.; SINGH, A. K. Lstm and nlp based forecasting model for stock market analysis. In: IEEE. *2021 First International Conference on Advances in Computing and Future Communication Technologies (ICACFCT)*. [S.l.], 2021. p. 52–57.

PEIXOTO, L. H. R. *Aprendizado de Máquina Aplicado no Atendimento de Reclamações de Clientes*. Tese (Doutorado) — Universidade de São Paulo, 2021.

PESSANHA, G. R. G.; FIDELIS, T. O.; FREIRE, C. D.; SOARES, E. A. # fiqueemcasa: Análise de sentimento dos usuários do twitter em relação ao covid19. *HOLOS*, v. 5, p. 1–20, 2020.

PIMENTEL, P. L. B. et al. A psicologia das emergências e dos desastres frente à polidemia de covid-19: diálogos entre saúde mental, vulnerabilidades, envelhecimentos e sociabilidade online. Universidade Federal da Paraíba, 2021.

PINHEIRO, R. H. W. Combinação de classificadores em diferentes espaços de características para classificação de documentos. Universidade Federal de Pernambuco, 2017.

PLATTO, S.; XUE, T.; CARAFOLI, E. Covid19: an announced pandemic. *Cell Death & Disease*, Nature Publishing Group, v. 11, n. 9, p. 1–13, 2020.

POKHAREL, B. P. Twitter sentiment analysis during covid-19 outbreak in nepal. *Available at SSRN 3624719*, 2020.

POLIGNANO, M.; BASILE, P.; GEMMIS, M. D.; SEMERARO, G. Hate speech detection through alberto italian language understanding model. In: *NL4AI@ AI* IA*. [S.l.: s.n.], 2019. p. 1–13.

PONTES, G. M. *Configuração automática de dados no ANSA*. Dissertação (Mestrado), 2021.

QIAN, C.; MATHUR, N.; ZAKARIA, N. H.; ARORA, R.; GUPTA, V.; ALI, M. Understanding public opinions on social media for financial sentiment analysis using ai-based techniques. *Information Processing & Management*, Elsevier, v. 59, n. 6, p. 103098, 2022.

RAHMAN, M. M.; ALI, G. M. N.; LI, X. J.; SAMUEL, J.; PAUL, K. C.; CHONG, P. H.; YAKUBOV, M. Socioeconomic factors analysis for covid-19 us reopening sentiment with twitter and census data. *Heliyon*, Elsevier, v. 7, n. 2, p. e06200, 2021.

ROEHRS, A.; COSTA, C. A. da; RIGHI, R. da R.; RIGO, S. J.; WICHMAN, M. H. Toward a model for personal health record interoperability. *IEEE journal of biomedical and health informatics*, IEEE, v. 23, n. 2, p. 867–873, 2018.

ROMAN, R. L. D. Atributos valorizados por consumidores digitais. 2017.

RONCERO, M. A.; GARCÍA, F. G. Deportes electrónicos. una aproximación a las posibilidades comunicativas de un mercado emergente. *Questiones publicitarias*, n. 19, p. 98–115, 2014.

RONG, W.; NIE, Y.; OUYANG, Y.; PENG, B.; XIONG, Z. Auto-encoder based bagging architecture for sentiment analysis. *Journal of Visual Languages & Computing*, Elsevier, v. 25, n. 6, p. 840–849, 2014.

SANTOS, N. R. d. *Análise e classificação de rumores em redes sociais*. Dissertação (Mestrado) — Universidade de São Paulo, 2020.

SHARMA, N.; SONI, M.; KUMAR, S.; KUMAR, R.; DEB, N.; SHRIVASTAVA, A. Supervised machine learning method for ontology-based financial decisions in stock market: Ontology-based financial decisions in stock market. *ACM Transactions on Asian and Low-Resource Language Information Processing*, ACM New York, NY.

SHI, W.; LIU, D.; YANG, J.; ZHANG, J.; WEN, S.; SU, J. Social bots' sentiment engagement in health emergencies: A topic-based analysis of the covid-19 pandemic discussions on twitter. *International Journal of Environmental Research and Public Health*, MDPI, v. 17, n. 22, p. 8701, 2020.

SILVA, C. V. M. *Análise exploratória e experimental sobre detecção inteligente de fake news*. Dissertação (Mestrado) — Universidade Federal de Sergipe, 2020.

SOUFYANE, A.; ABDELHAKIM, B. A.; AHMED, M. B. An intelligent chatbot using nlp and tf-idf algorithm for text understanding applied to the medical field. In: *Emerging Trends in ICT for Sustainable Development*. [S.l.]: Springer, 2021. p. 3–10.

SOUSA, G.; PEDRONETTE, D. C. G.; BALDASSIN, A.; PRIVATTO, P. I. M.; GASETA, M.; GUILHERME, I. R.; COLOMBO, D.; AFONSO, L. C. S.; PAPA, J. P. Pattern analysis in drilling reports using optimum-path forest. In: IEEE. *2018 International Joint Conference on Neural Networks (IJCNN)*. [S.l.], 2018. p. 1–8.

SOUSA, W. G. de; FIDELIS, R. A.; BERMEJO, P. H. de S.; GONÇALO, A. G. da S.; MELO, B. de S. Artificial intelligence and speedy trial in the judiciary: Myth, reality or need? a case study in the brazilian supreme court (stf). *Government Information Quarterly*, Elsevier, v. 39, n. 1, p. 101660, 2022.

SOUZA, A.; MAIA, J. E. B. Agente inteligente para classificação de notícias por assunto. *Anais do Computer on the Beach*, p. 714–723, 2019.

STEVENSON, M.; MUES, C.; BRAVO, C. The value of text for small business default prediction: A deep learning approach. *European Journal of Operational Research*, Elsevier, v. 295, n. 2, p. 758–771, 2021.

SUZUKI, M.; YAMAGISHI, N.; TSAI, Y.-C.; ISHIDA, T.; GOTO, M. English and taiwanese text categorization using n-gram based on vector space model. In: *2010 International Symposium On Information Theory Its Applications*. [S.l.: s.n.], 2010. p. 106–111.

TISSIER, J.; GRAVIER, C.; HABRARD, A. Near-lossless binarization of word embeddings. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. [S.l.: s.n.], 2019. v. 33, n. 01, p. 7104–7111.

TRUICĂ, C.-O.; APOSTOL, E.-S.; ȘERBAN, M.-L.; PASCHKE, A. Topic-based document-level sentiment analysis using contextual cues. *Mathematics*, MDPI, v. 9, n. 21, p. 2722, 2021.

UMER, M.; IMTIAZ, Z.; ULLAH, S.; MEHMOOD, A.; CHOI, G. S.; ON, B.-W. Fake news stance detection using deep learning architecture (cnn-lstm). *IEEE Access*, IEEE, v. 8, p. 156695–156706, 2020.

VASILEIOU, M. V.; MAGLOGIANNIS, I. G. et al. The health chatbots in telemedicine: Intelligent dialog system for remote support. *Journal of Healthcare Engineering*, Hindawi, v. 2022, 2022.

VEIGA, H. M. F. C. d. *Text mining e twitter: o poder das redes sociais num mercado competitivo*. Tese (Doutorado), 2016.

YILDIRIM, S. Comparing deep neural networks to traditional models for sentiment analysis in turkish language. *Deep learning-based approaches for sentiment analysis*, Springer, p. 311–319, 2020.

ZHANG, J.; LIU, F.; XU, W.; YU, H. Feature fusion text classification model combining cnn and bigru with multi-attention mechanism. *Future Internet*, MDPI, v. 11, n. 11, p. 237, 2019.

ZHANG, L.; WANG, S.; LIU, B. Deep learning for sentiment analysis: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, Wiley Online Library, v. 8, n. 4, p. e1253, 2018.

ZHOU, Z.; GUAN, H.; BHAT, M. M.; HSU, J. Fake news detection via nlp is vulnerable to adversarial attacks. *arXiv preprint arXiv:1901.09657*, 2019.

APÊNDICE A – COVID - VARIANTES

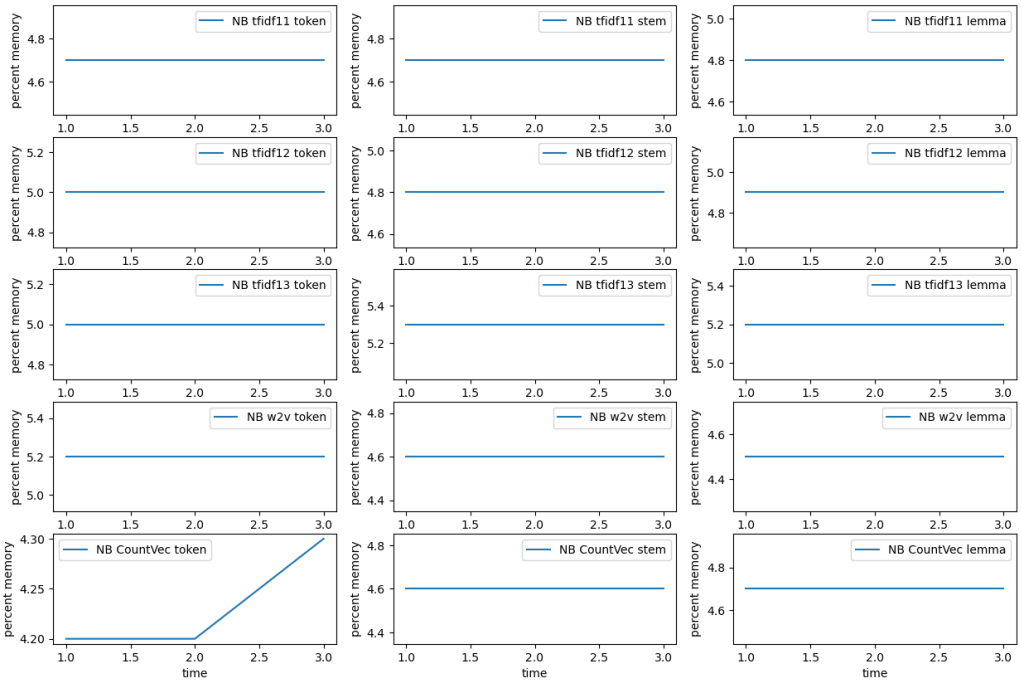
Quadro 1 – Variantes de interesse e variantes de preocupação do SARS-CoV-2.

OMS	Nextrain clado	Pango linhagem	GISAID clado	Primeira detecção	Primeiras amostras	Mutações características
Classificação Variantes de preocupação						
Alfa	20J/501Y.V1	B.1.1.7	GRY GR/501Y.V1	Reino Unido	Set 2020	H69/V70del, Y144del, N501Y, A570D, P681H, S106/G107/F108del
Beta	20H/501Y.V2	B.1.351	GH/501Y.V2	África do Sul	Ago 2020	L242/A243/L244del, K417N, E484K, N501Y, S106/G107/F108del
Gama	20J/501Y.V3	B.1.1.28.1 conhecida como P.1	GR/501Y.V3	Brasil e Japão	Dez 2020	K417T, E484K, N501Y, S106/G107/F108del
Delta	21A/S:478K	B.1.617.2	G/452R.V3	Índia	Out 2020	T19R, (G142D), 156del, 157del, R158G, L452R, T478K, D614G, P681R, D950N
Classificação Variantes de interesse						
Epsilon	20C/S.452R	B.1.427/ B.1.429	GH/452R.V1	Estados Unidos da América	Jun 2020	L452R, W152C, S13I, D614G
Zeta	20B/S.484K	B.1.1.28.2 conhecida como P.2	GR	Brasil	Abril 2020	L18F, T20N, P.26S, F157L, E484K, D614G, S929I, V1176F
Eta	20A/S484K	B.1.525	G/484K.V3	Múltiplos países	Dez 2020	H69-V70del, Y144del, Q52R, E484K, Q677H, D616G, S929I, V1176F
Teta	20B/S:265C	B.1.1.28.3 conhecida como P.3	GR	Filipinas e Japão	Fev 2021	141-143del, E484K, N501Y, P681H
Iota	20C/S:484K	B.1.526	GH	Estados Unidos da América	Nov 2020	LSF, T95I, D253G, D614G, V483A, H655Y, G669S, Q949R, N1187D
Kapa	21A/S:154K	B.1.617.1	G/452R.V3	Índia	Out 2020	(T95I), G142D, E154K, L452R, E484Q, D614G, P681R, Q1071H
Lambda	21G	C.37	GR/452Q.V1	Peru	Dez 2020	246-252del, G75V, T76I, L452Q, F490S, D614G e T859N

Quadro 2 – Fonte: Adaptado de (COVID, 2021)

APÊNDICE B – EXPERIMENTOS COM TECNICAS DE *WORD EMBEDDING* NO CORPUS PORTUGUESE TWEETS FOR SENTIMENT ANALYSIS (KAGGLE)

Figura 41 – Custos de memória com algoritmo *Naive Bayes* para experimentos com técnicas de *word embedding*



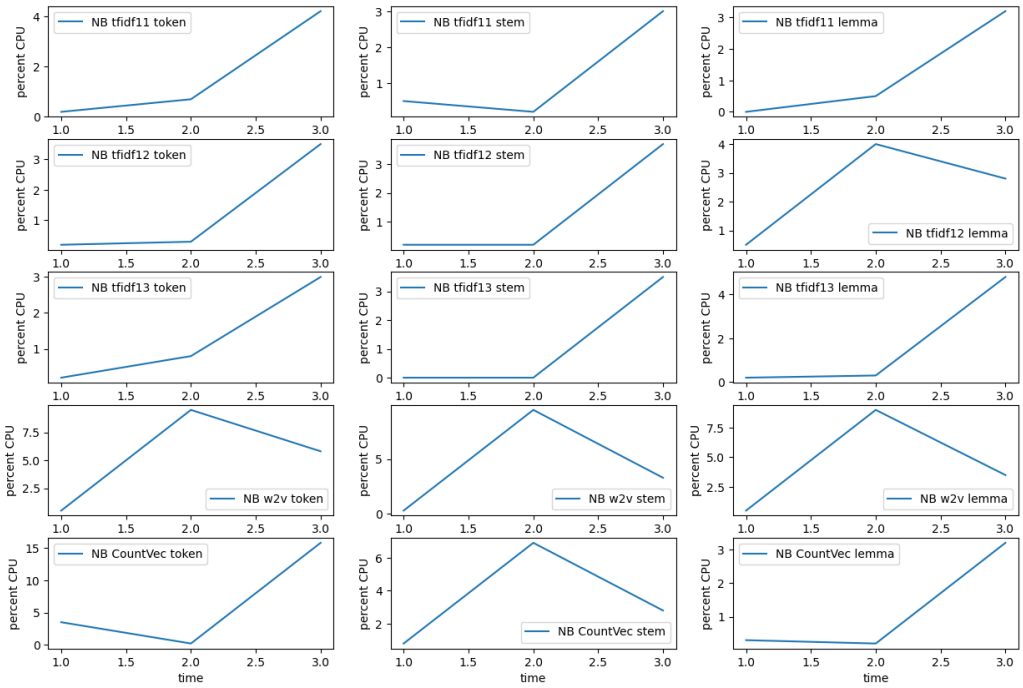
Fonte: Elaboração própria.

Tabela 29 – Tabela de métricas resultantes utilizando técnicas de *Word embedding* com a base *kaggle*.

Tec. Incorporação	**Tec. Vetorização**	**Modelo**	**Precisão**	**Recall**	**F1**	**Acurácia**	**Precisão Ponderada**
CountVector	Token	naive_bayes	0.682520	0.676133	0.673404	0.676133	0.989051
		svc	0.664099	0.647600	0.643265	0.647600	0.992537
		mlp	0.658319	0.653333	0.652895	0.653333	0.999547
	stem	naive_bayes	0.595366	0.562400	0.557145	0.562400	0.946073
		svc	0.583168	0.540133	0.533264	0.540133	0.965958
		mlp	0.580701	0.540800	0.535281	0.540800	0.985606
	lemma	naive_bayes	0.676102	0.673333	0.670281	0.673333	0.986142
		svc	0.653298	0.646267	0.643305	0.646267	0.989871
		mlp	0.634805	0.625467	0.625529	0.625467	0.999112
tfidf (1-1)	Token	naive_bayes	0.686530	0.681067	0.676898	0.681067	0.963662
		svc	0.708098	0.689333	0.691141	0.689333	0.989542
		mlp	0.661522	0.654800	0.657344	0.654800	0.999691
	stem	naive_bayes	0.681639	0.680133	0.677496	0.680133	0.969712
		svc	0.695789	0.687333	0.688219	0.687333	0.988892
		mlp	0.634812	0.630133	0.630423	0.630133	0.999772
	lemma	naive_bayes	0.686078	0.681867	0.679801	0.681867	0.966783
		svc	0.696896	0.688267	0.688832	0.688267	0.987450
		mlp	0.639799	0.634800	0.636089	0.634800	0.999463
tfidf (1-2)	Token	naive_bayes	0.682554	0.673200	0.668276	0.673200	0.955464
		svc	0.706095	0.681333	0.681477	0.681333	0.985084
		mlp	0.681718	0.668800	0.672211	0.668800	0.998601
	stem	naive_bayes	0.598542	0.565600	0.555993	0.565600	0.945483
		svc	0.573436	0.559733	0.550566	0.559733	0.959593
		mlp	0.582716	0.542933	0.535333	0.542933	0.984811
	lemma	naive_bayes	0.664950	0.664533	0.657461	0.664533	0.965228
		svc	0.674096	0.667333	0.667177	0.667333	0.982898
		mlp	0.626257	0.617733	0.616840	0.617733	0.999588
tfidf (1-3)	Token	naive_bayes	0.679879	0.666000	0.662046	0.666000	0.949685
		svc	0.703149	0.679200	0.676969	0.679200	0.983378
		mlp	0.677449	0.653067	0.658494	0.653067	0.996814
	stem	naive_bayes	0.598920	0.566133	0.556538	0.566133	0.945415
		svc	0.573445	0.559867	0.550686	0.559867	0.960345
		mlp	0.579966	0.543333	0.537050	0.543333	0.985500
	lemma	naive_bayes	0.665909	0.666133	0.658807	0.666133	0.965135
		svc	0.673096	0.666400	0.666241	0.666400	0.982570
		mlp	0.629930	0.625333	0.626941	0.625333	0.999320
word2vec	Token	naive_bayes	0.389325	0.373333	0.316534	0.373333	0.977227
		svc	0.562646	0.563467	0.555759	0.563467	0.984735
		mlp	0.577208	0.568133	0.567159	0.568133	0.992483
	stem	naive_bayes	0.361688	0.378400	0.324854	0.378400	0.956777
		svc	0.517996	0.513067	0.497664	0.513067	0.984232
		mlp	0.547362	0.542000	0.531094	0.542000	0.974179
	lemma	naive_bayes	0.369956	0.396667	0.349269	0.396667	0.945855
		svc	0.516049	0.520533	0.504103	0.520533	0.975175
		mlp	0.542180	0.537067	0.521348	0.537067	0.974230

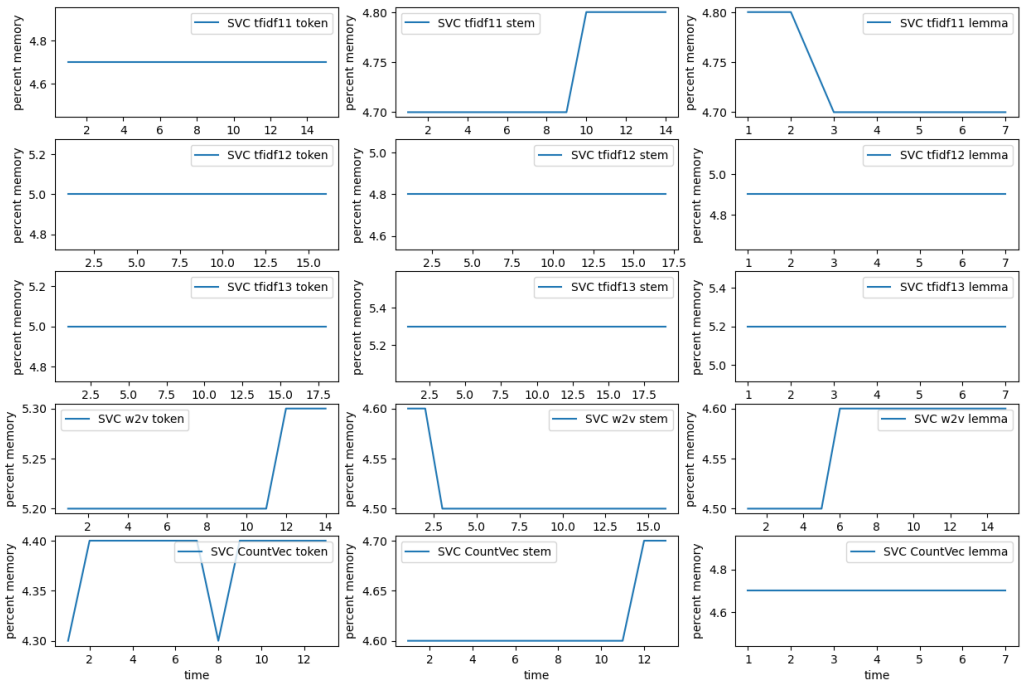
Fonte: Elaboração própria.

Figura 42 – Custos de CPU com algoritmo *Naive Bayes* para experimentos com técnicas de *word embedding*



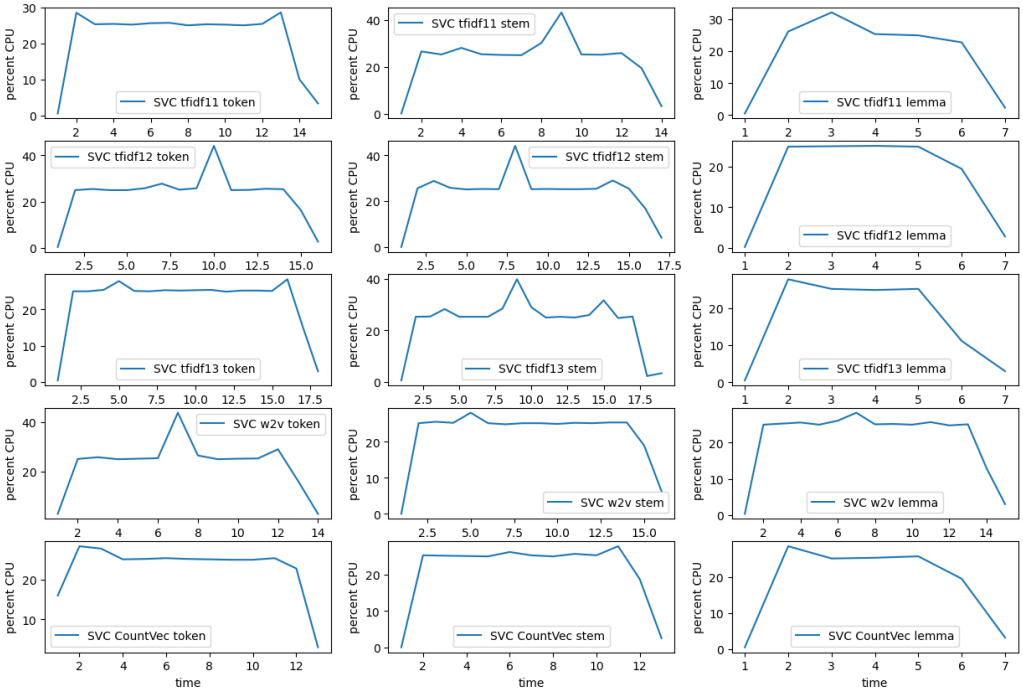
Fonte: Elaboração própria.

Figura 43 – Custos de memória com algoritmo SVC para experimentos com técnicas de *word embedding*



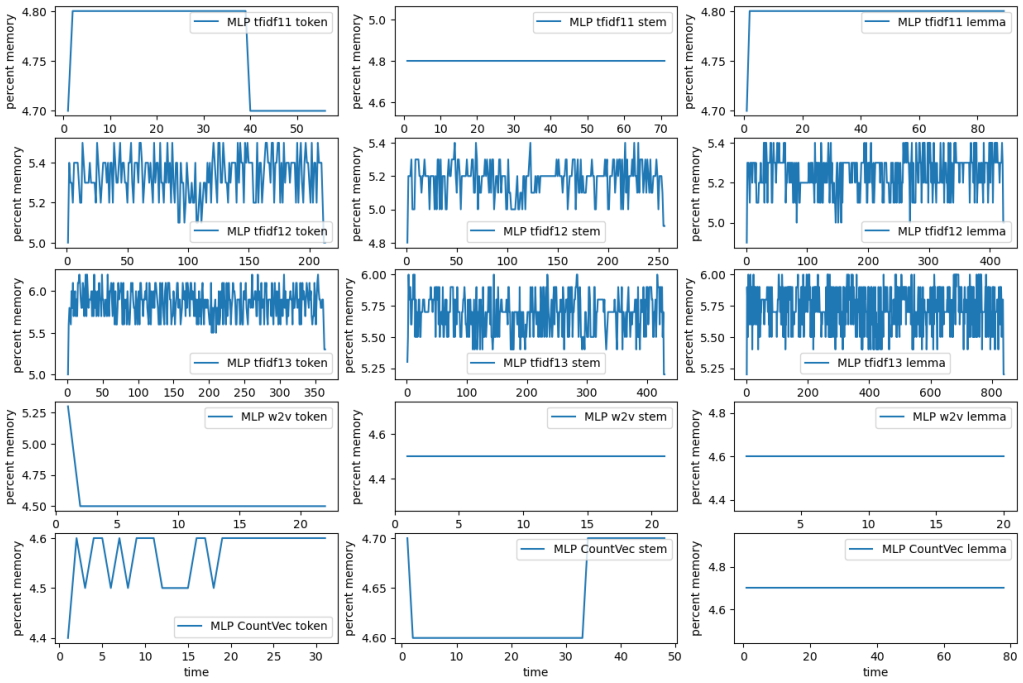
Fonte: Elaboração própria.

Figura 44 – Custos de CPU com algoritmo SVC para experimentos com técnicas de *word embedding*



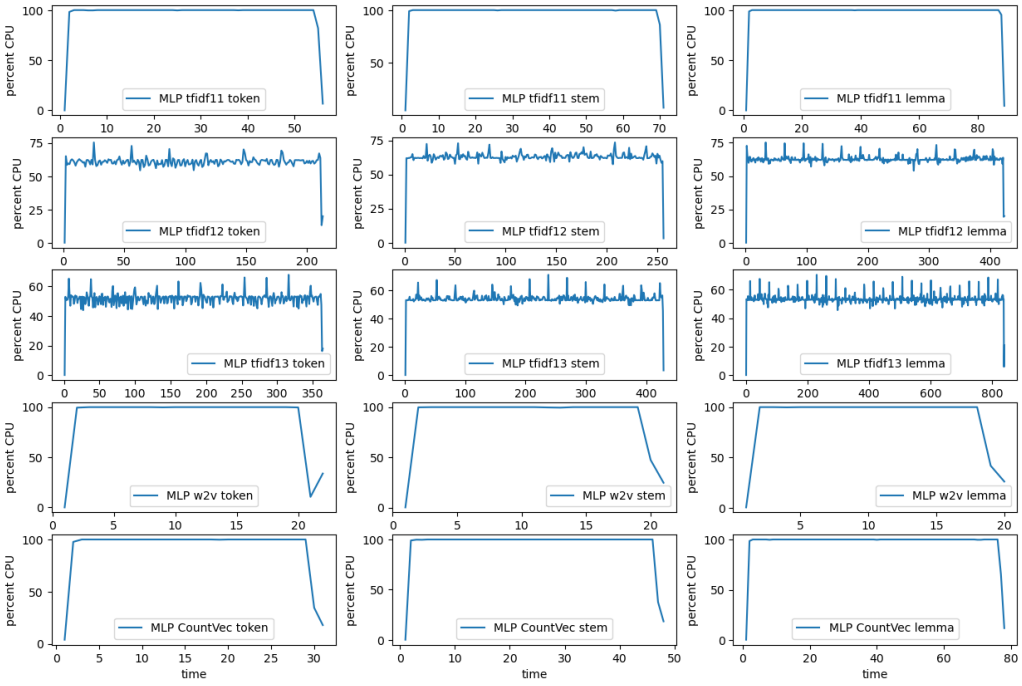
Fonte: Elaboração própria.

Figura 45 – Custos de memória com algoritmo MLP para experimentos com técnicas de *word embedding*



Fonte: Elaboração própria.

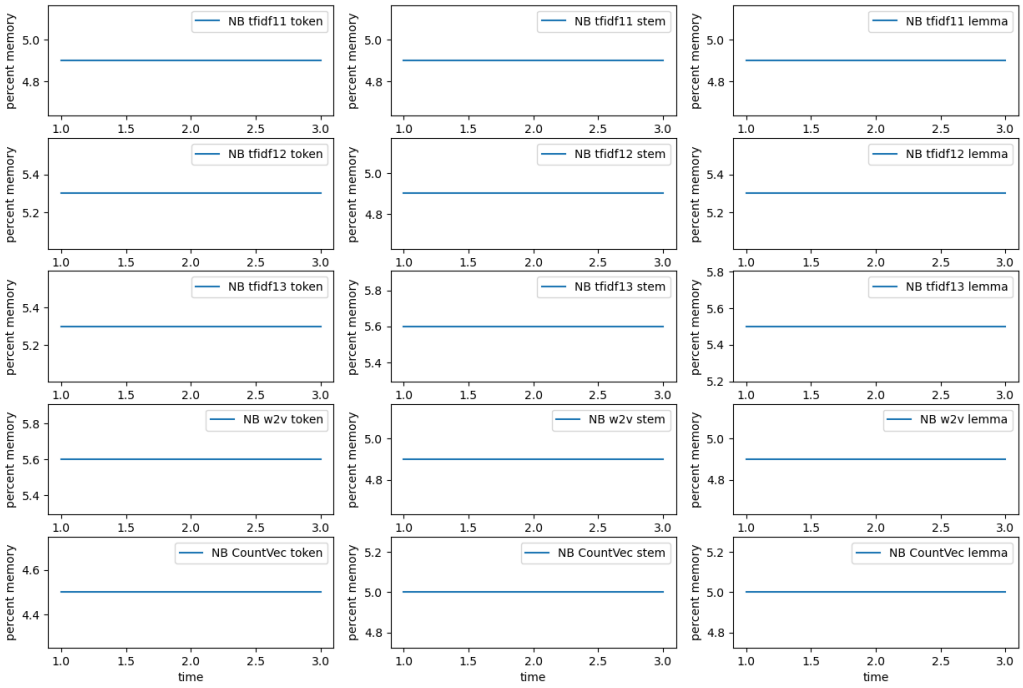
Figura 46 – Custos de CPU com algoritmo MLP para experimentos com técnicas de *word embedding*



Fonte: Elaboração própria.

APÊNDICE C – EXPERIMENTOS COM TECNICAS DE *WORD EMBEDDING* NO CORPUS TWO BILLION MULTILINGUAL COVID-19 (TBCOV)

Figura 47 – Custos de memória com algoritmo *Naive Bayes* para experimentos com técnicas de *word embedding*



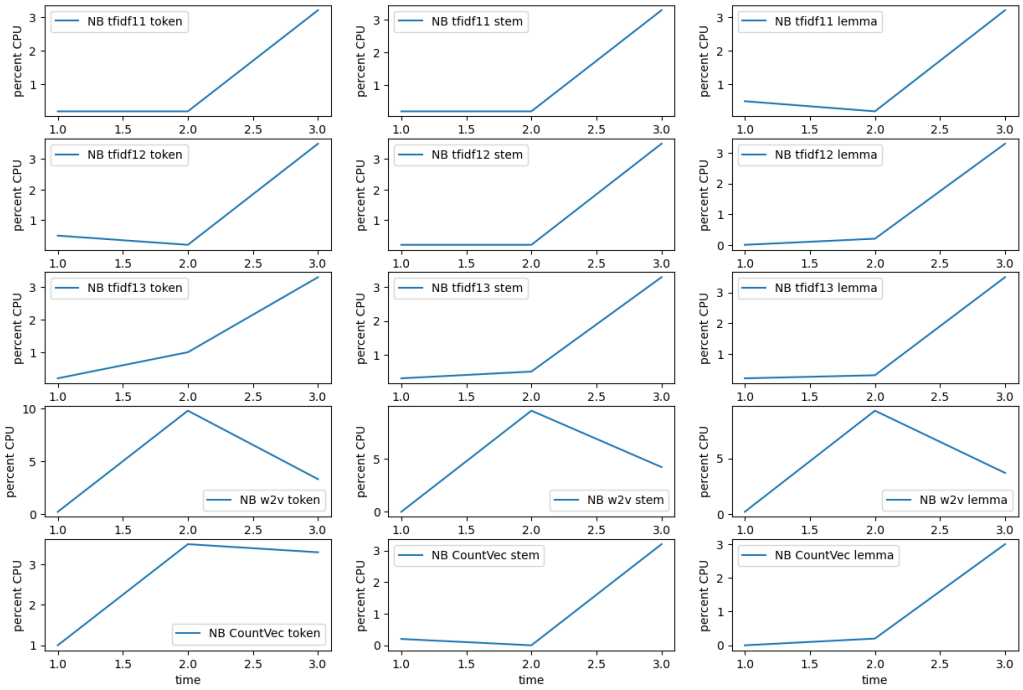
Fonte: Elaboração própria.

Tabela 30 – Tabela de métricas resultantes utilizando técnicas de *Word embedding* com a base TBCOV.

Tec. Incorporação	**Tec. Vetorização**	**Modelo**	**Precisão**	**Recall**	**F1**	**Acurácia**	**Precisão Ponderada**
CountVector	Token	naive_bayes	0.781339	0.772800	0.774606	0.772800	0.998356
		svc	0.790820	0.789867	0.790031	0.789867	0.996850
		mlp	0.799303	0.796667	0.797333	0.796667	0.999952
	Stemização	naive_bayes	0.580377	0.575600	0.573498	0.575600	0.952736
		svc	0.659378	0.628800	0.634712	0.628800	0.981166
		mlp	0.655944	0.625600	0.625609	0.625600	0.994718
	Lemmatização	naive_bayes	0.772229	0.760267	0.762166	0.760267	0.997555
		svc	0.798814	0.797600	0.797771	0.797600	0.997486
		mlp	0.795699	0.794800	0.794800	0.794800	0.999966
tfidf (1-1)	Token	naive_bayes	0.785384	0.776933	0.778757	0.776933	0.992303
		svc	0.818158	0.802800	0.805041	0.802800	0.992408
		mlp	0.799245	0.798133	0.797909	0.798133	0.999925
	Stemização	naive_bayes	0.784386	0.774000	0.775935	0.774000	0.994392
		svc	0.826695	0.816000	0.817931	0.816000	0.995317
		mlp	0.797587	0.797733	0.797318	0.797733	0.999955
	Lematização	naive_bayes	0.785100	0.775867	0.777666	0.775867	0.993406
		svc	0.825115	0.812933	0.814962	0.812933	0.994414
		mlp	0.825115	0.812933	0.814962	0.812933	0.994414
tfidf (1-2)	Token	naive_bayes	0.795500	0.788533	0.790020	0.788533	0.990009
		svc	0.814782	0.791333	0.793603	0.791333	0.985815
		mlp	0.809379	0.802000	0.802449	0.802000	0.999465
	Stemização	naive_bayes	0.578888	0.574133	0.572463	0.574133	0.936764
		svc	0.666714	0.637200	0.643285	0.637200	0.991010
		mlp	0.655810	0.625867	0.625406	0.625867	0.975620
	Lematização	naive_bayes	0.775808	0.767600	0.769111	0.767600	0.992113
		svc	0.818195	0.805333	0.807380	0.805333	0.995544
		mlp	0.791897	0.790400	0.790208	0.790400	0.999958
tfidf (1-3)	Token	naive_bayes	0.798153	0.791333	0.792737	0.791333	0.989810
		svc	0.814384	0.786267	0.788349	0.786267	0.982800
		mlp	0.813649	0.795867	0.797102	0.795867	0.999172
	Stemização	naive_bayes	0.578864	0.574133	0.572452	0.574133	0.936722
		svc	0.666714	0.637200	0.643285	0.637200	0.991919
		mlp	0.655341	0.624533	0.624169	0.624533	0.984176
	Lematização	naive_bayes	0.775626	0.767467	0.768940	0.767467	0.992100
		svc	0.818239	0.805067	0.807120	0.805067	0.995557
		mlp	0.784998	0.784400	0.784030	0.784400	0.999937
word2vec	Token	naive_bayes	0.471506	0.454800	0.446636	0.454800	0.903679
		svc	0.642946	0.605333	0.605154	0.605333	0.987638
		mlp	0.650083	0.643200	0.638404	0.643200	0.987599
	multirow3*Stemização	naive_bayes	0.469254	0.447200	0.439089	0.447200	0.890291
		svc	0.651759	0.611733	0.611833	0.611733	0.985096
		mlp	0.662439	0.645200	0.641724	0.645200	0.986702
	Lematização	naive_bayes	0.466063	0.445200	0.432665	0.445200	0.921738
		svc	0.651228	0.603200	0.603249	0.603200	0.983228
		mlp	0.657910	0.636667	0.633805	0.636667	0.984765

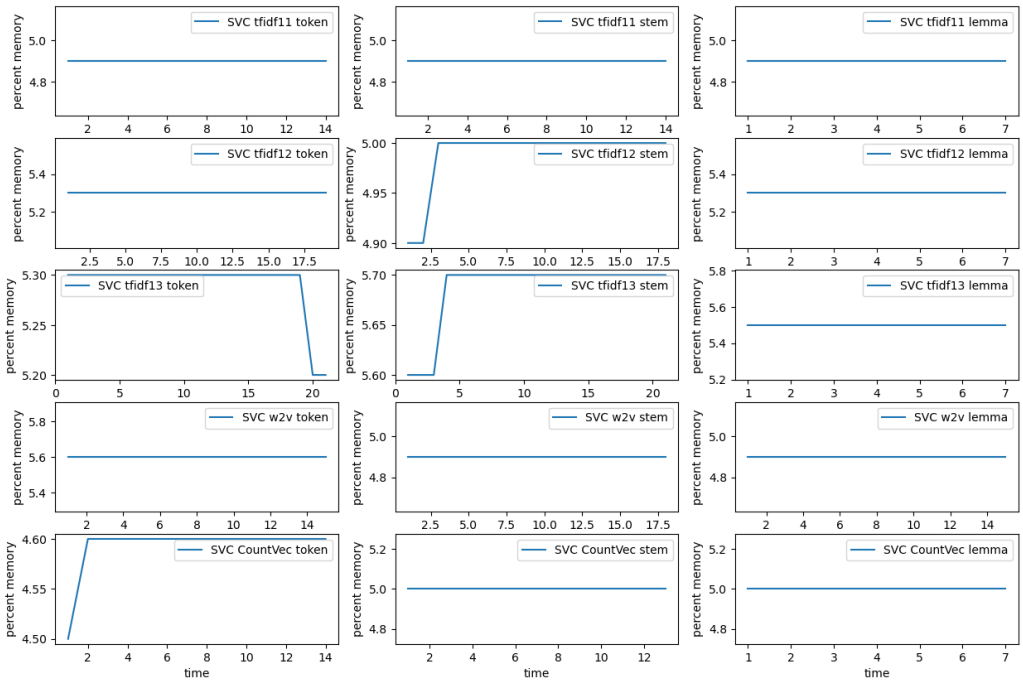
Fonte: Elaboração própria.

Figura 48 – Custos de CPU com algoritmo *Naive Bayes* para experimentos com técnicas de *word embedding*



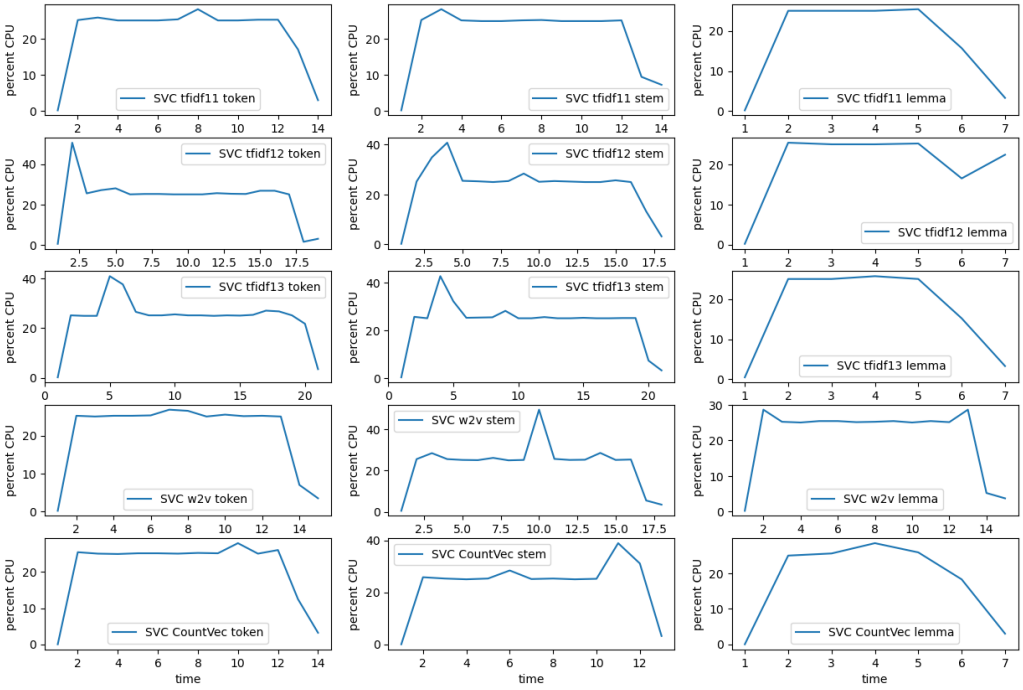
Fonte: Elaboração própria.

Figura 49 – Custos de memória com algoritmo SVC para experimentos com técnicas de *word embedding*



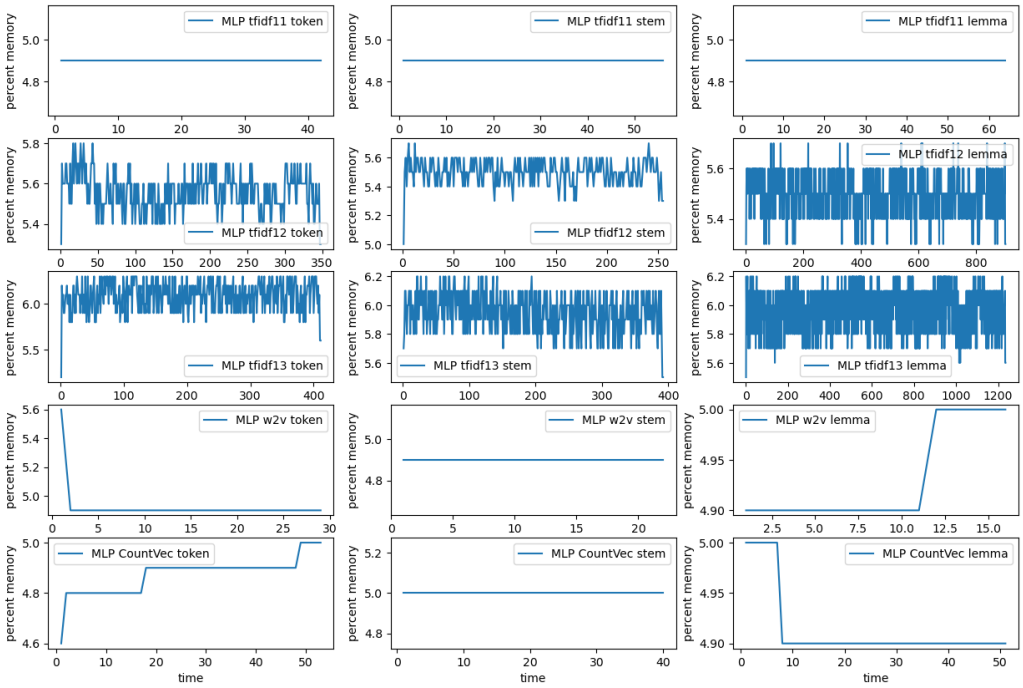
Fonte: Elaboração própria.

Figura 50 – Custos de CPU com algoritmo SVC para experimentos com técnicas de *word embedding*



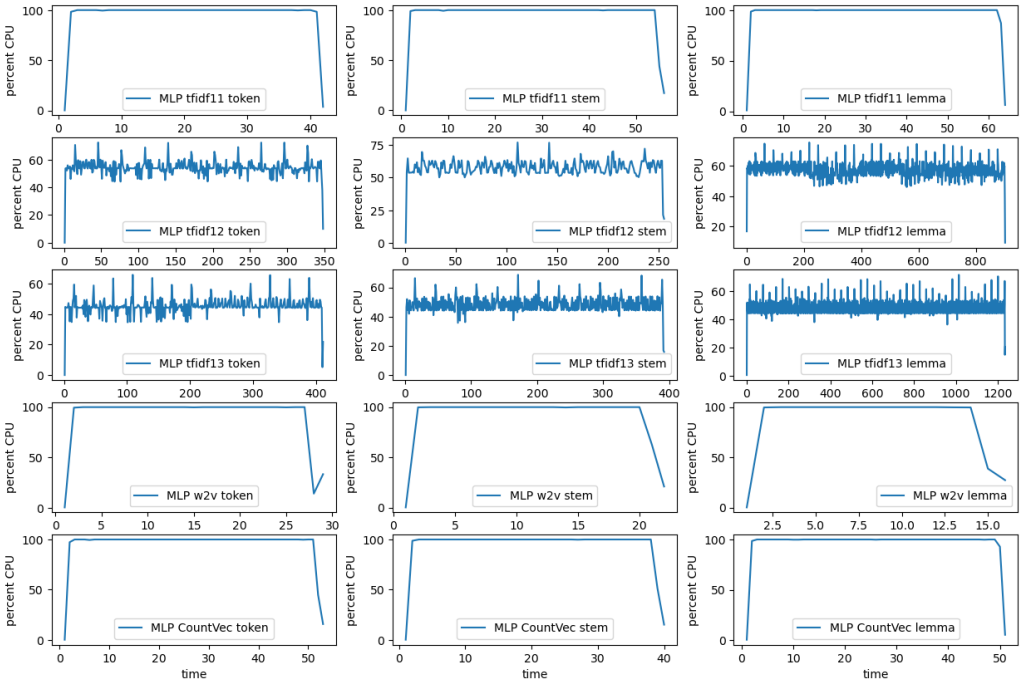
Fonte: Elaboração própria.

Figura 51 – Custos de memória com algoritmo MLP para experimentos com técnicas de *word embedding*



Fonte: Elaboração própria.

Figura 52 – Custos de CPU com algoritmo MLP para experimentos com técnicas de *word embedding*



Fonte: Elaboração própria.

APÊNDICE D – EXPERIMENTOS COM TECNICAS DE *WORD EMBEDDING* E MODELOS DE APRENDIZADO PROFUNDO NOS CORPUS TWO BILLION MULTILINGUAL COVID-19 (TBCOV) E PORTUGUESE TWEETS FOR SENTIMENT ANALYSIS (KAGGLE)

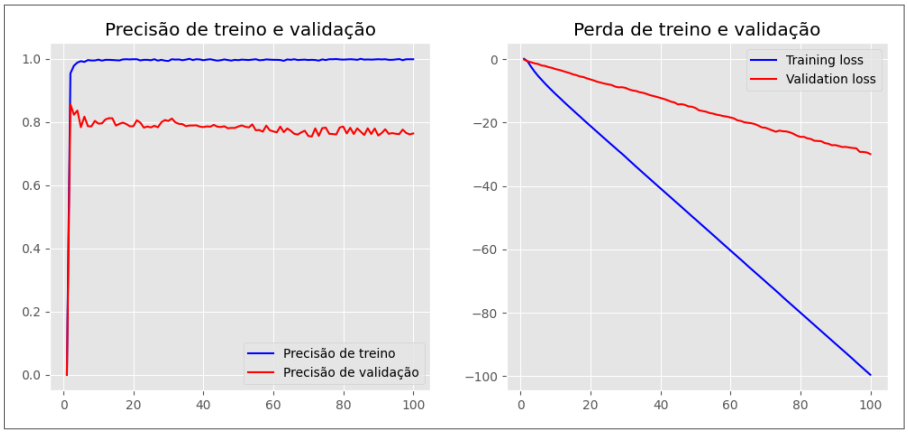
D.1 PORTUGUESE TWEETS FOR SENTIMENT ANALYSIS (KAGGLE)

D.1.1 *CountVector*

D.1.1.1 Rede Neural Recorrente com 4 camadas

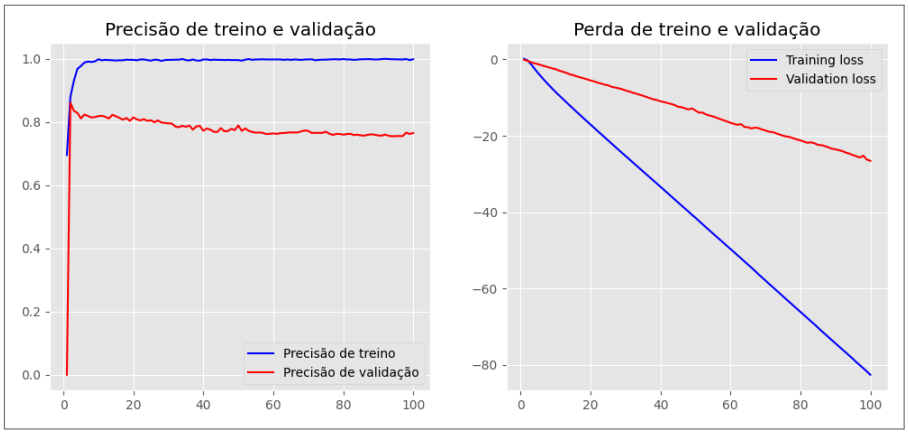
Ativação através da função ReLu na camada de entrada e sigmoide nas camadas restante. Produzidas através da técnica de incorporação de palavras *CountVector* e comparadas entre *tokenização*, *stemming* e *lemmatization*.

Figura 53 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor *CountVector* e técnica *tokenização*



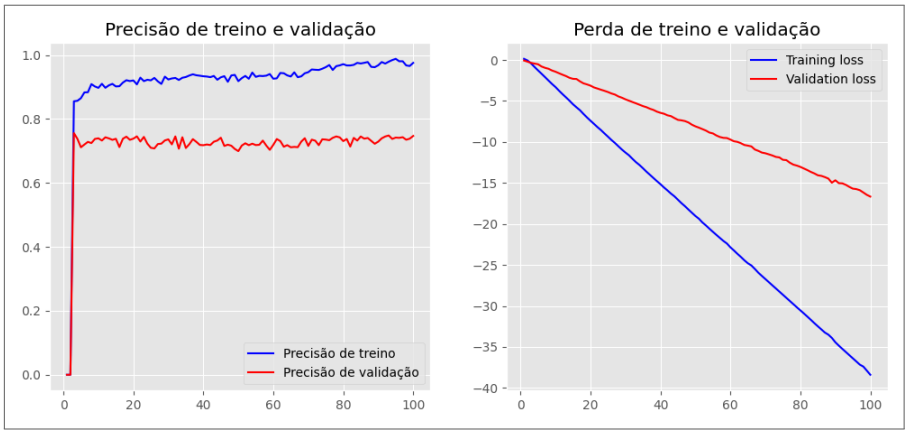
Fonte: Elaboração própria.

Figura 54 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor *CountVector* e técnica *stemming*



Fonte: Elaboração própria.

Figura 55 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor *CountVector* e técnica *lemmatization*

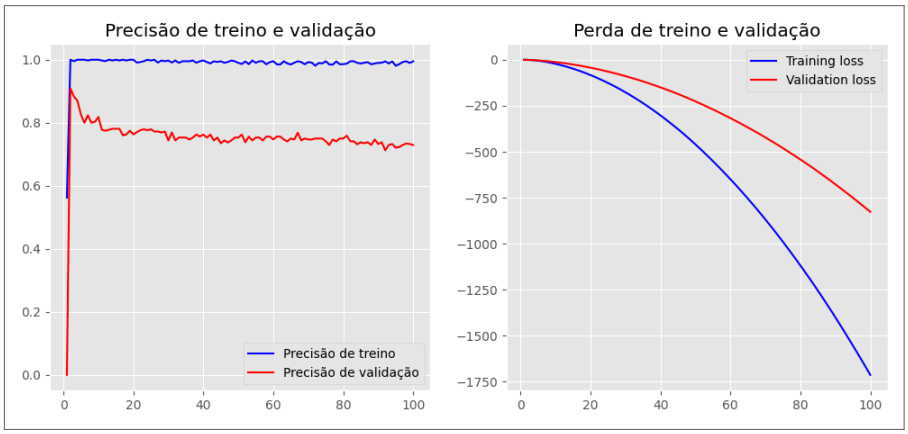


Fonte: Elaboração própria.

D.1.1.2 Rede neural recorrente com 2 camadas

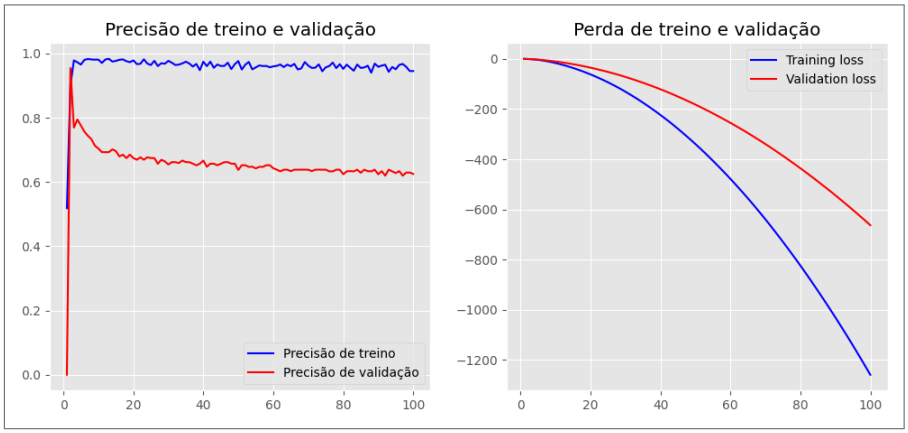
Ativação através da função ReLu na camada de entrada e sigmoide na camada restante. Produzidas através da técnica de incorporação de palavras *CountVector* e comparadas entre *tokenização*, *stemming* e *lemmatization*.

Figura 56 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor *CountVector* e técnica *tokenização*



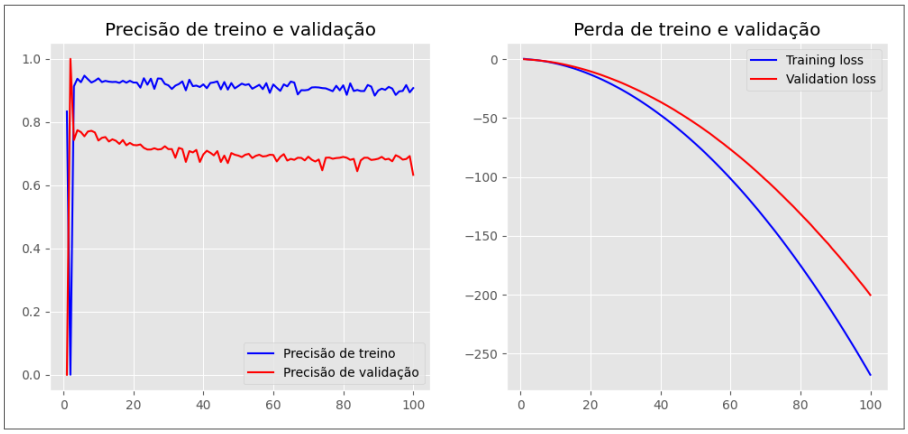
Fonte: Elaboração própria.

Figura 57 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor *CountVector* e técnica *stemming*



Fonte: Elaboração própria.

Figura 58 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor *CountVector* e técnica *lemmatization*

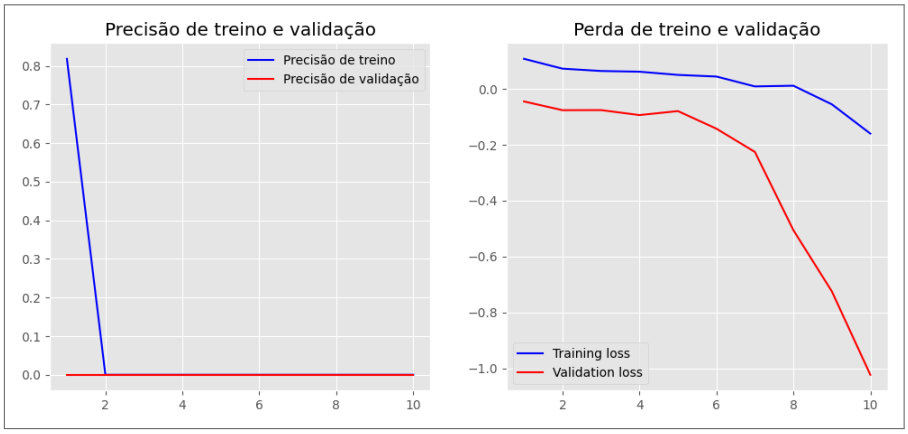


Fonte: Elaboração própria.

D.1.1.3 Rede Neural Convolucional com 5 camadas

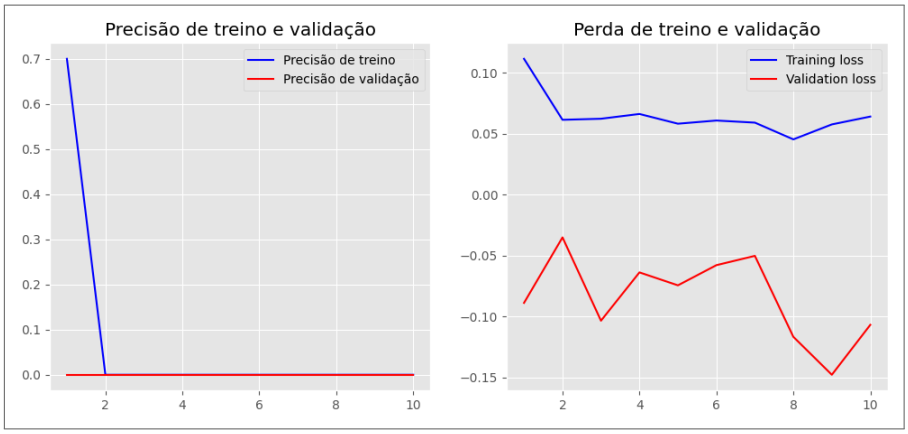
Produzidas através da técnica de incorporação de palavras *CountVector* e comparadas entre *tokenização*, *stemming* e *lemmatization*.

Figura 59 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Convolucional para treino e validação. Realizado com o vetor *CountVector* e técnica *tokenização*



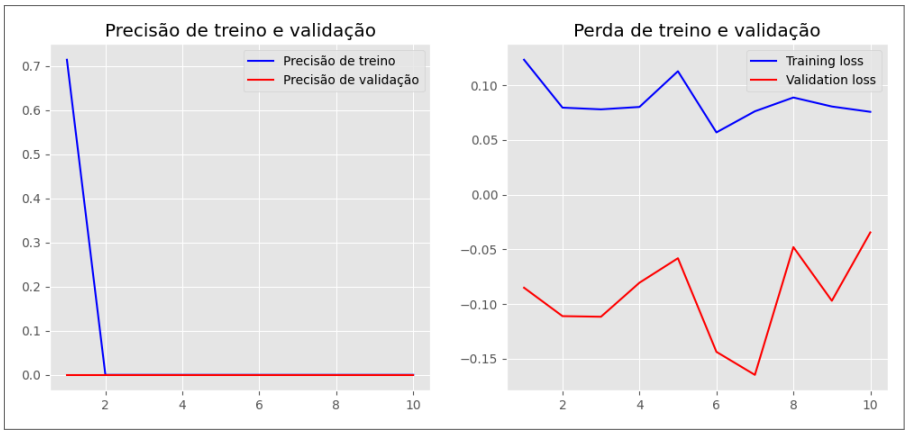
Fonte: Elaboração própria.

Figura 60 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Convolucional para treino e validação. Realizado com o vetor *CountVector* e técnica *stemming*



Fonte: Elaboração própria.

Figura 61 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Convolucional para treino e validação. Realizado com o vetor *CountVector* e técnica *lemmatization*



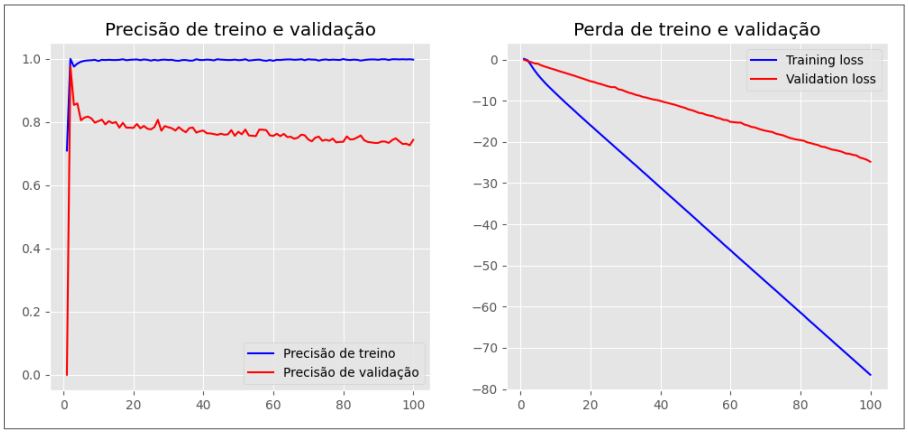
Fonte: Elaboração própria.

D.1.2 *Term frequency–inverse document frequency (TFIDF)*

D.1.2.1 Rede Neural Recorrente com 4 camadas

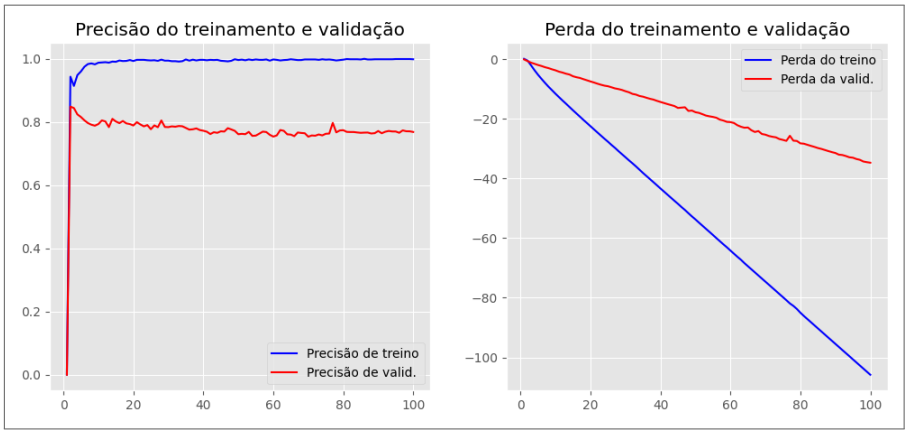
Ativação através da função ReLu na camada de entrada e sigmoide nas camadas restante. Produzidas através da técnica de incorporação de palavras TFIDF e comparadas entre *tokenização*, *stemming* e *lemmatization*.

Figura 62 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,1)$ e técnica *tokenização*



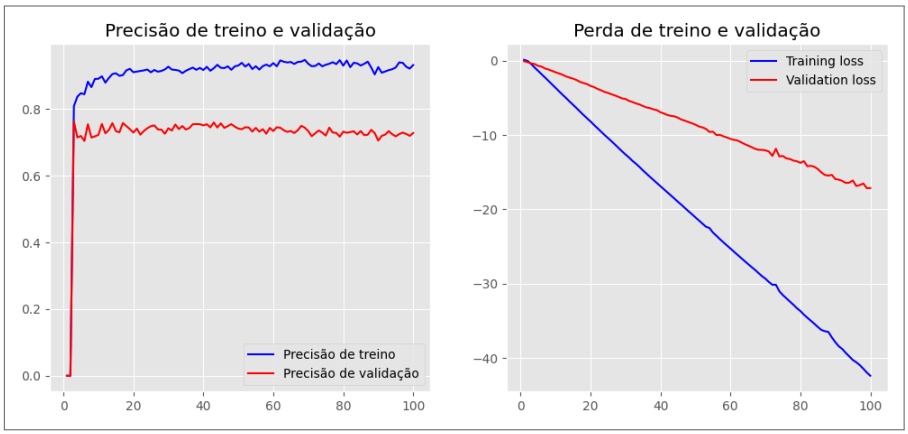
Fonte: Elaboração própria.

Figura 63 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,1)$ e técnica *stemming*



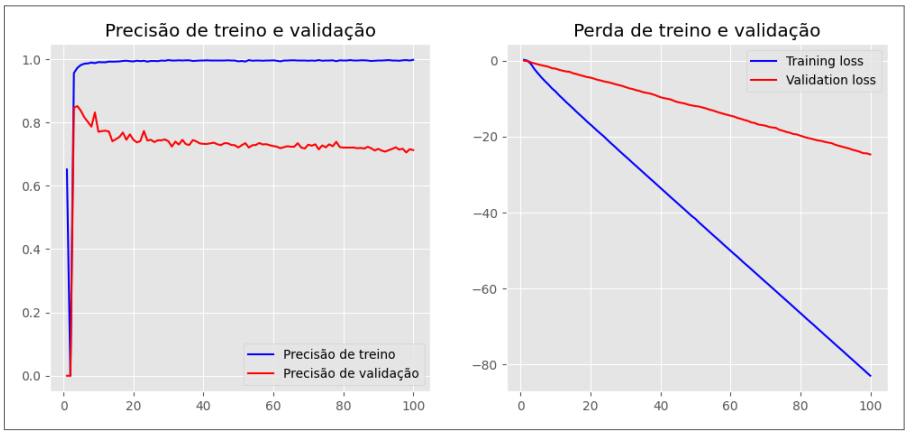
Fonte: Elaboração própria.

Figura 64 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor TFIDF $n - gram = (1, 1)$ e técnica *lemmatization*



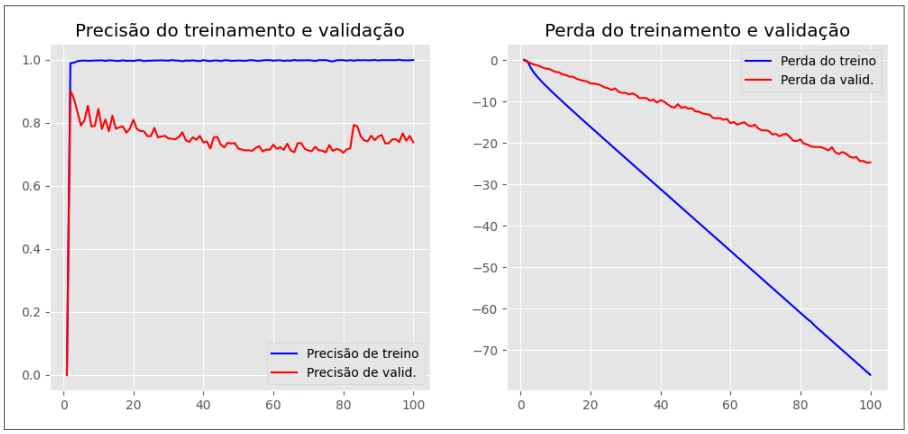
Fonte: Elaboração própria.

Figura 65 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor TFIDF $n - gram = (1, 2)$ e técnica *tokenização*



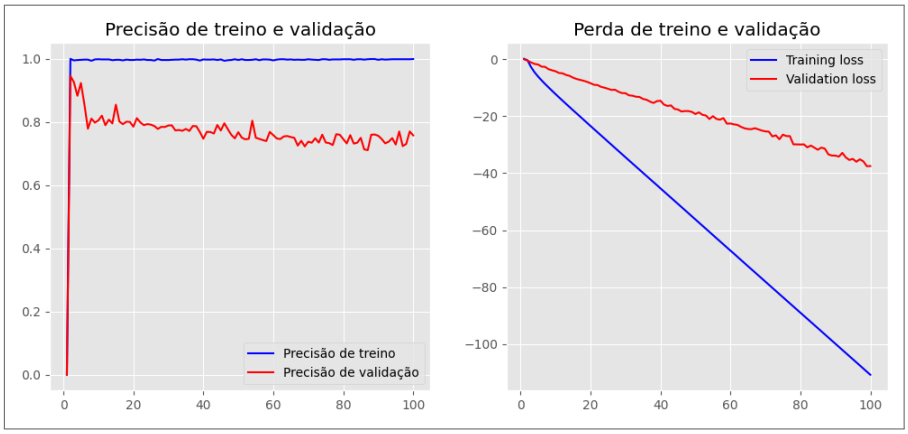
Fonte: Elaboração própria.

Figura 66 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor TFIDF $n - gram = (1, 2)$ e técnica *stemming*



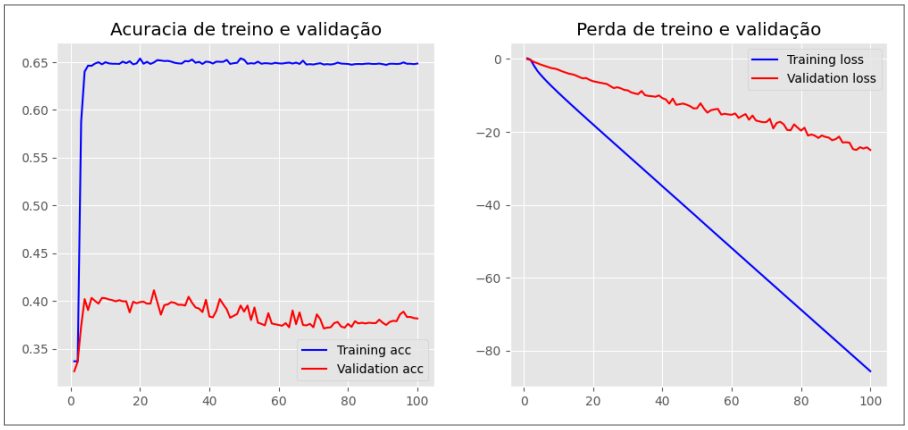
Fonte: Elaboração própria.

Figura 67 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,3)$ e técnica *tokenização*



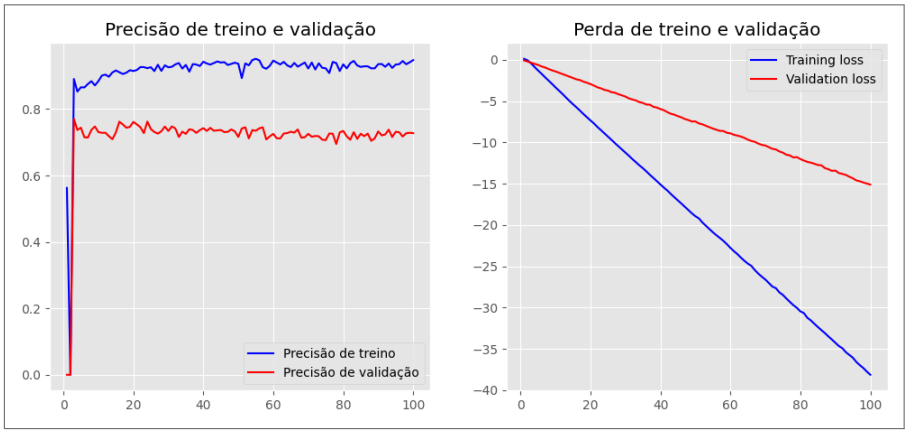
Fonte: Elaboração própria.

Figura 68 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,3)$ e técnica *stemming*



Fonte: Elaboração própria.

Figura 69 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,3)$ e técnica *lemmatization*

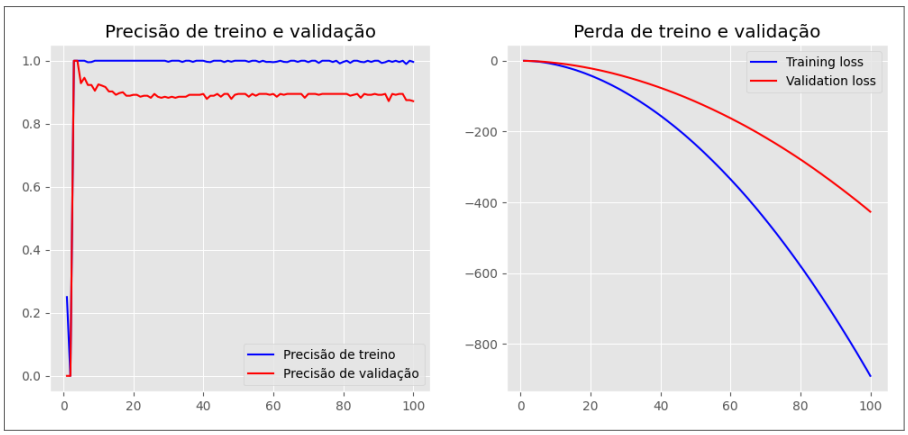


Fonte: Elaboração própria.

D.1.2.2 Rede neural recorrente com 2 camadas

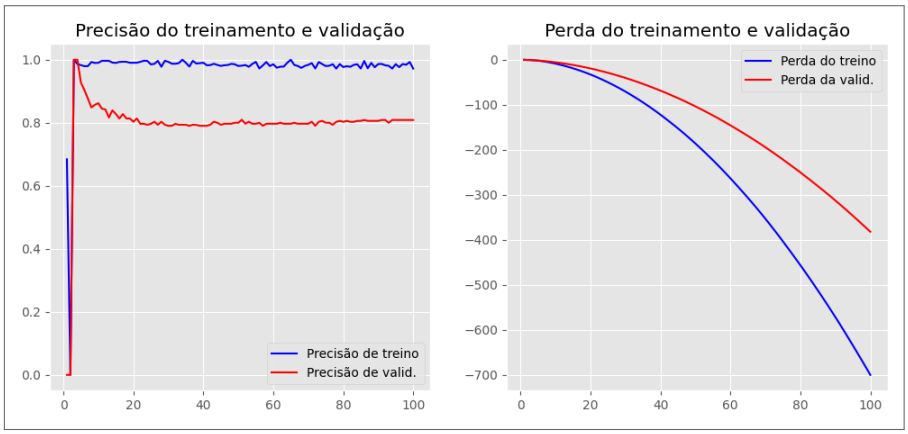
Ativação através da função ReLu na camada de entrada e sigmoide na camada restante. Produzidas através da técnica de incorporação de palavras TFIDF e comparadas entre *tokenização*, *stemming* e *lemmatization*.

Figura 70 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor TFIDF $n - gram = (1, 1)$ e técnica *tokenização*



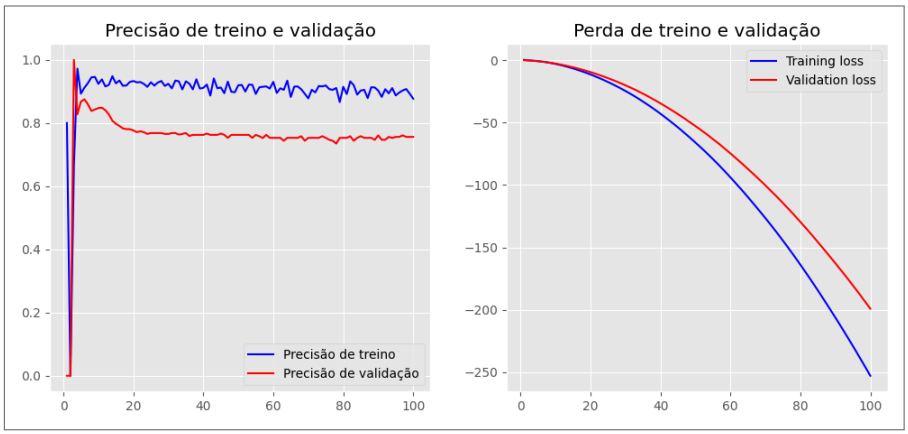
Fonte: Elaboração própria.

Figura 71 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor TFIDF $n - gram = (1, 1)$ e técnica *stemming*



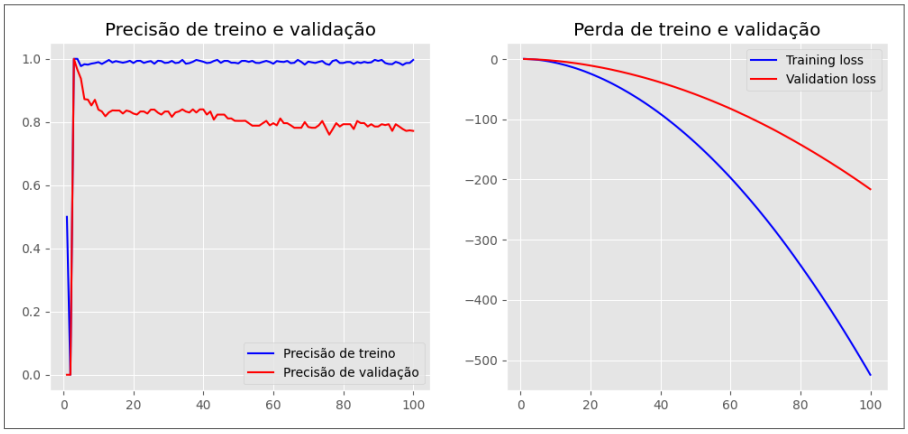
Fonte: Elaboração própria.

Figura 72 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,1)$ e técnica *lemmatization*



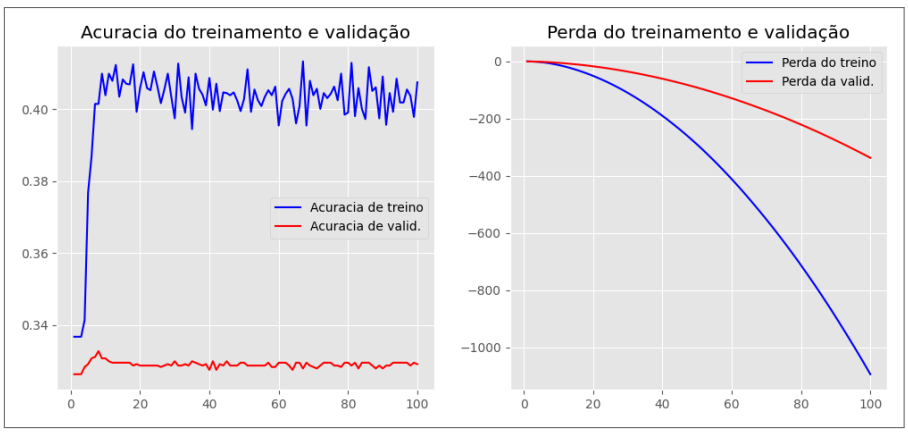
Fonte: Elaboração própria.

Figura 73 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,2)$ e técnica *tokenização*



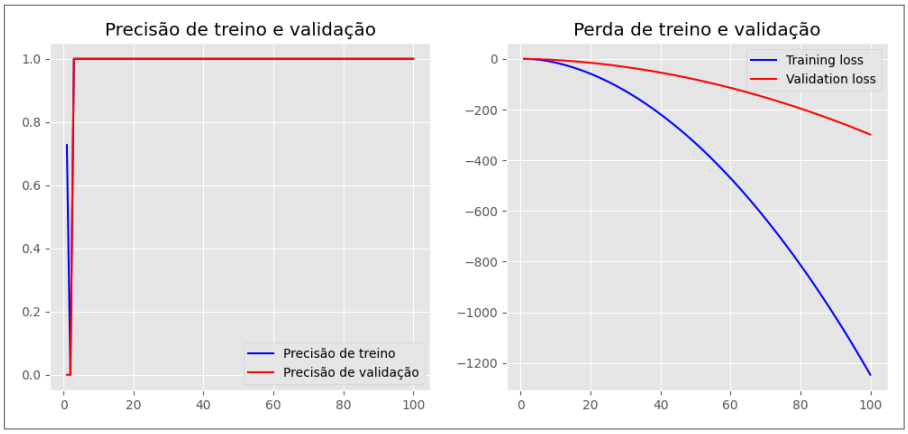
Fonte: Elaboração própria.

Figura 74 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,2)$ e técnica *stemming*



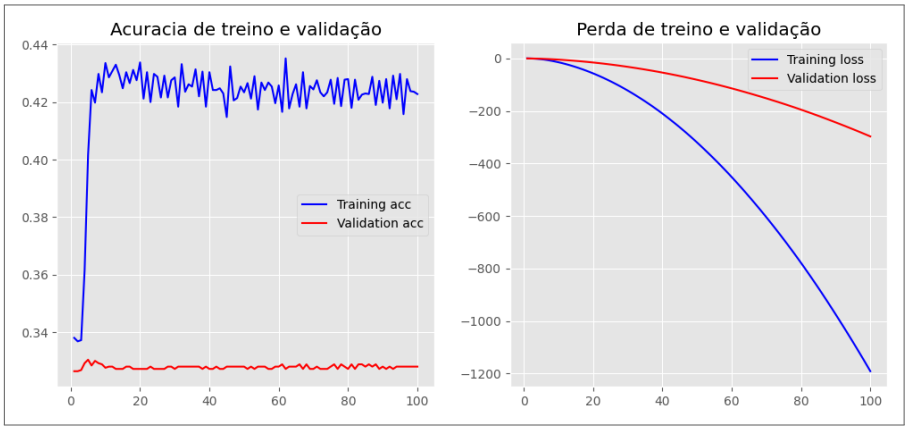
Fonte: Elaboração própria.

Figura 75 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,3)$ e técnica *tokenização*



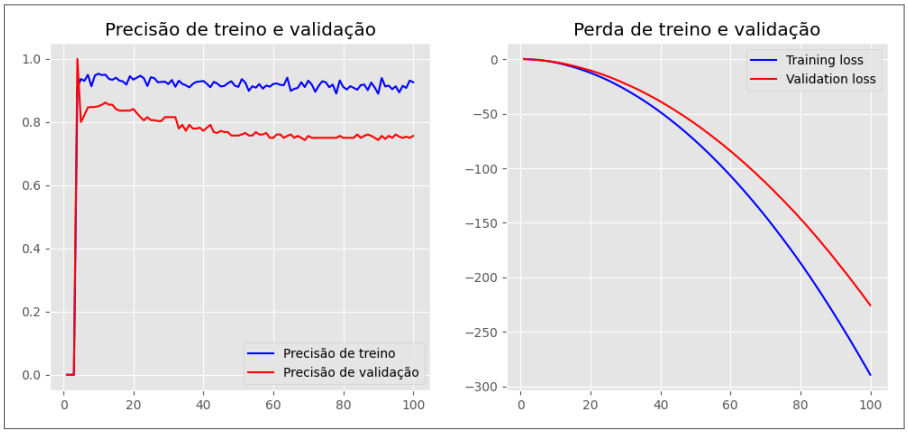
Fonte: Elaboração própria.

Figura 76 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,3)$ e técnica *stemming*



Fonte: Elaboração própria.

Figura 77 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,3)$ e técnica *lemmatization*

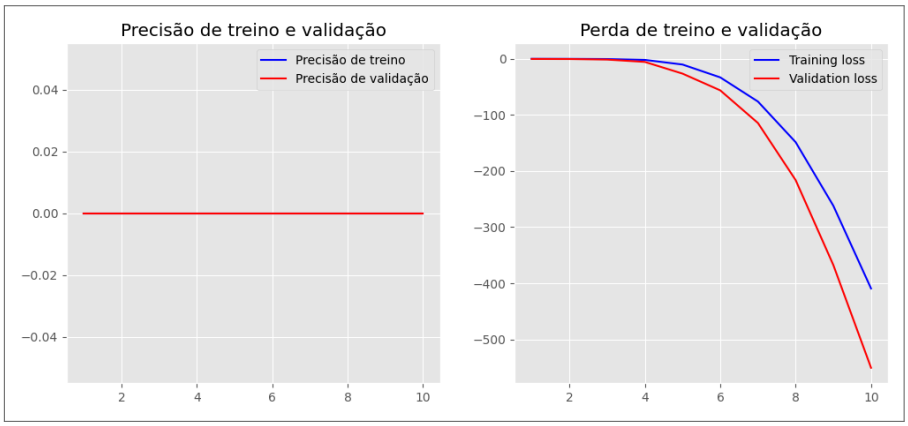


Fonte: Elaboração própria.

D.1.2.3 Rede Neural Convolucional com 5 camadas

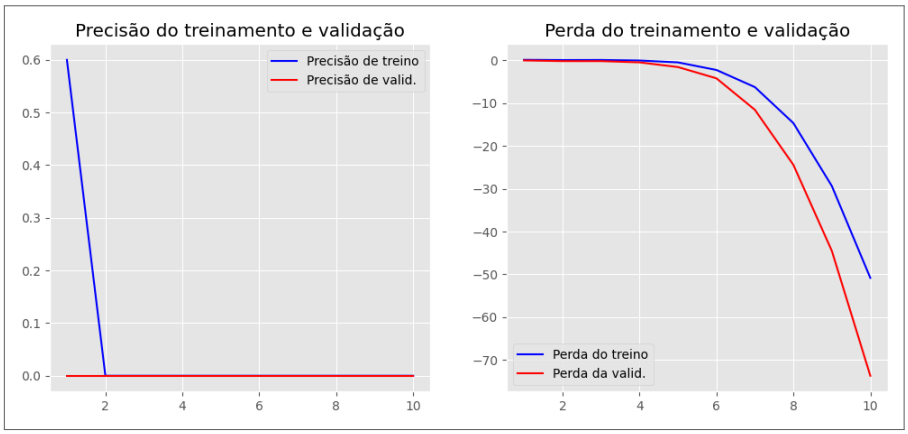
Produzidas através da técnica de incorporação de palavras *CountVector* e comparadas entre *tokenização*, *stemming* e *lemmatization*.

Figura 78 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Convolucional para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,1)$ e técnica *tokenização*



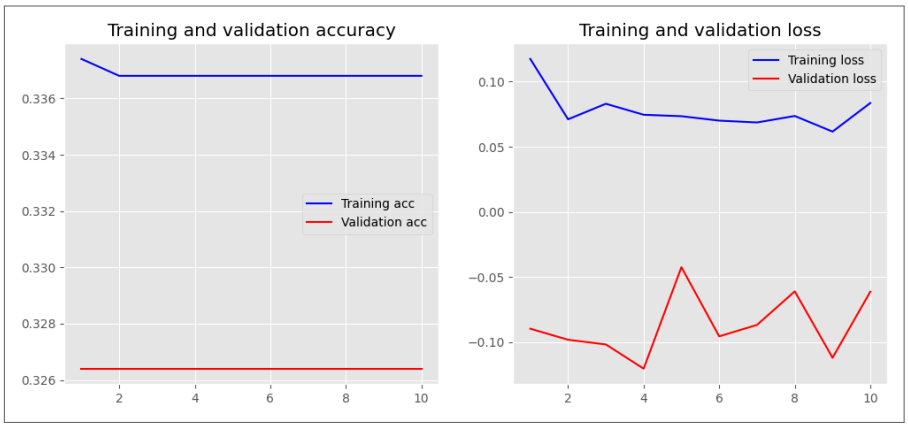
Fonte: Elaboração própria.

Figura 79 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Convolucional para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,1)$ e técnica *stemming*



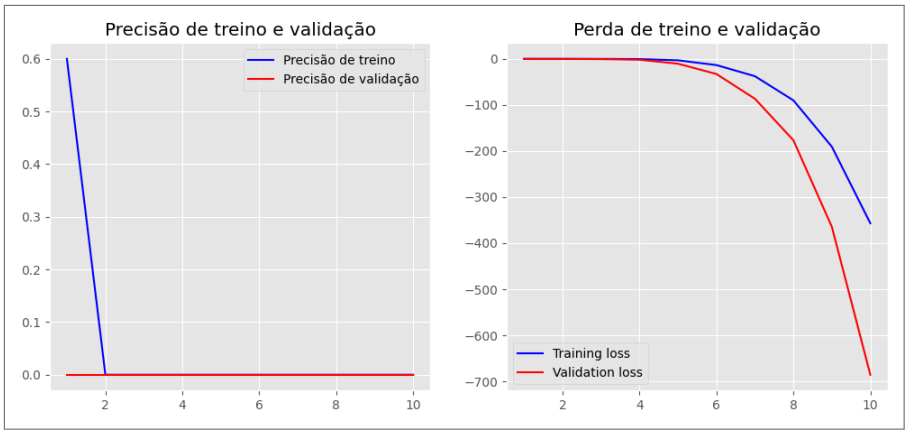
Fonte: Elaboração própria.

Figura 80 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Convolucional para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,1)$ e técnica *lemmatization*



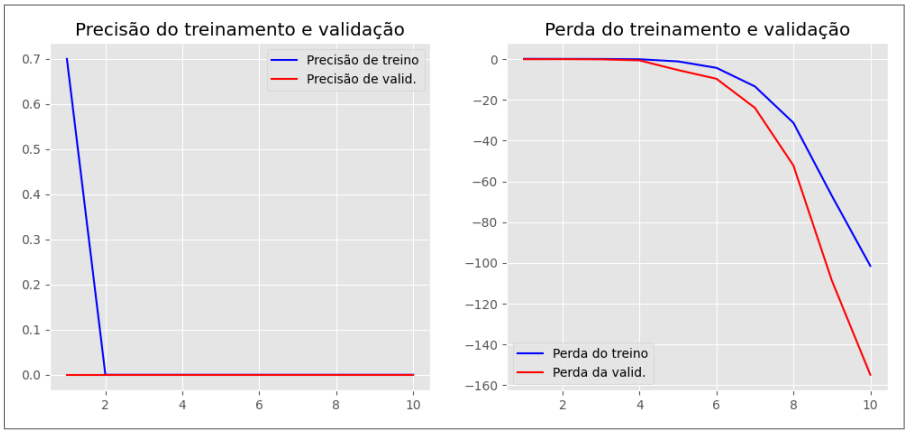
Fonte: Elaboração própria.

Figura 81 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Convolucional para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,2)$ e técnica *tokenização*



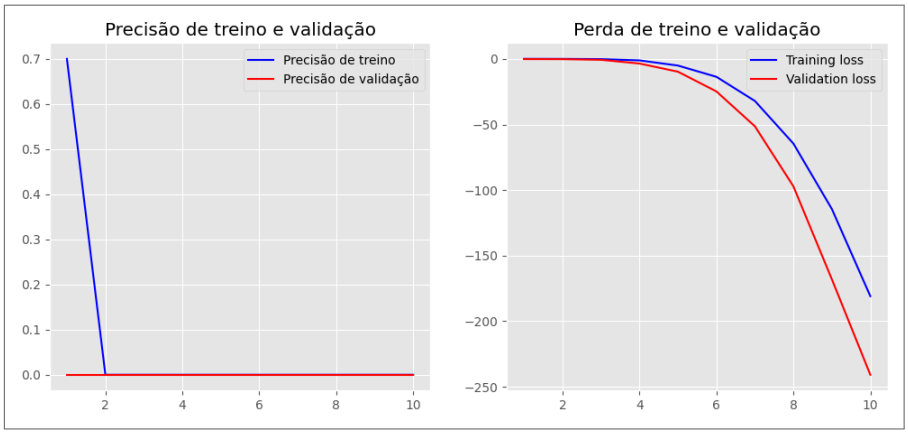
Fonte: Elaboração própria.

Figura 82 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Convolucional para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,2)$ e técnica *stemming*



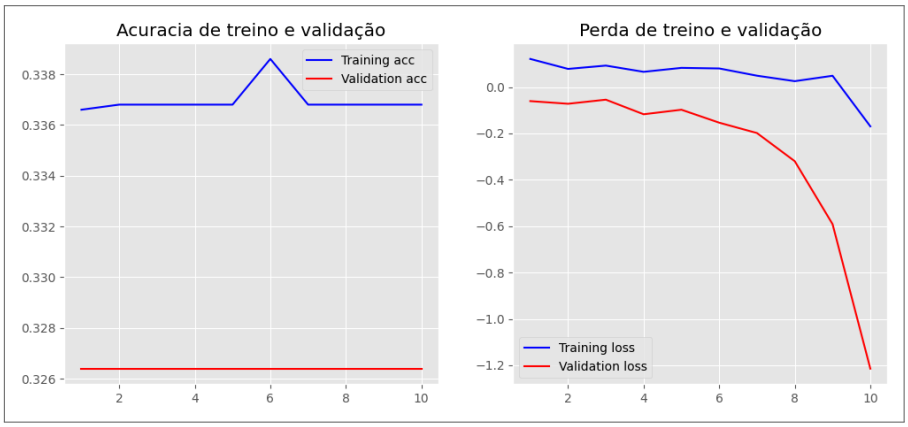
Fonte: Elaboração própria.

Figura 83 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Convolucional para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,3)$ e técnica *tokenização*



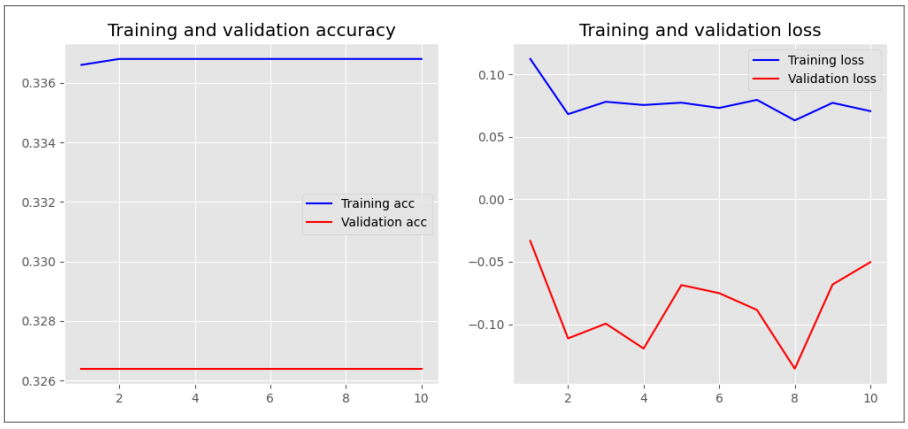
Fonte: Elaboração própria.

Figura 84 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Convolucional para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,3)$ e técnica *stemming*



Fonte: Elaboração própria.

Figura 85 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Convolucional para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,3)$ e técnica *lemmatization*



Fonte: Elaboração própria.

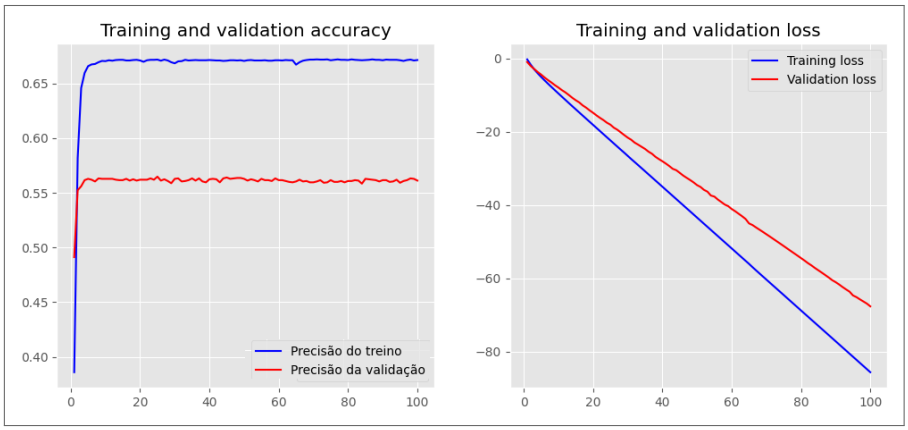
D.2 TWO BILLION MULTILINGUAL COVID-19 (TBCOV)

D.2.1 *CountVector*

D.2.1.1 Rede Neural Recorrente com 4 camadas

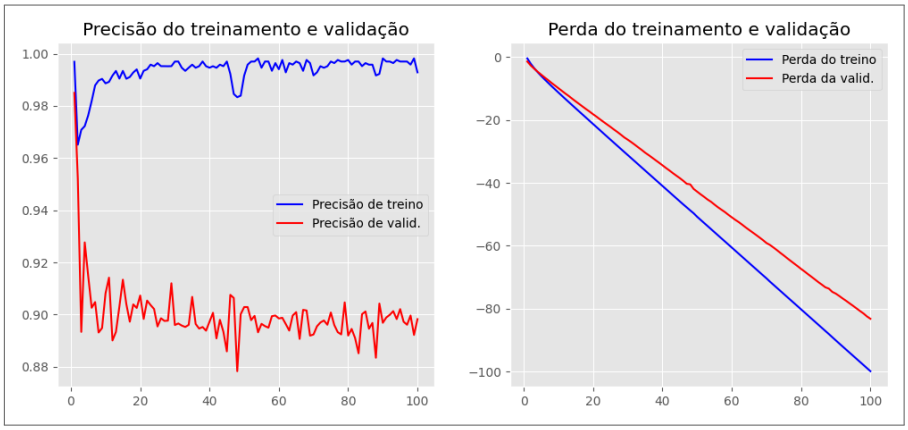
Ativação através da função ReLu na camada de entrada e sigmoide nas camadas restante. Produzidas através da técnica de incorporação de palavras *CountVector* e comparadas entre *tokenização*, *stemming* e *lemmatization*.

Figura 86 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor *CountVector* e técnica *tokenização*



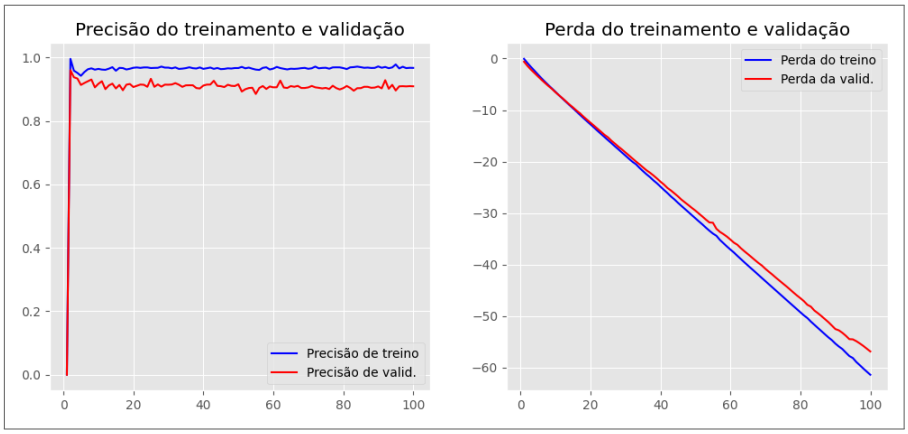
Fonte: Elaboração própria.

Figura 87 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor *CountVector* e técnica *stemming*



Fonte: Elaboração própria.

Figura 88 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor *CountVector* e técnica *lemmatization*

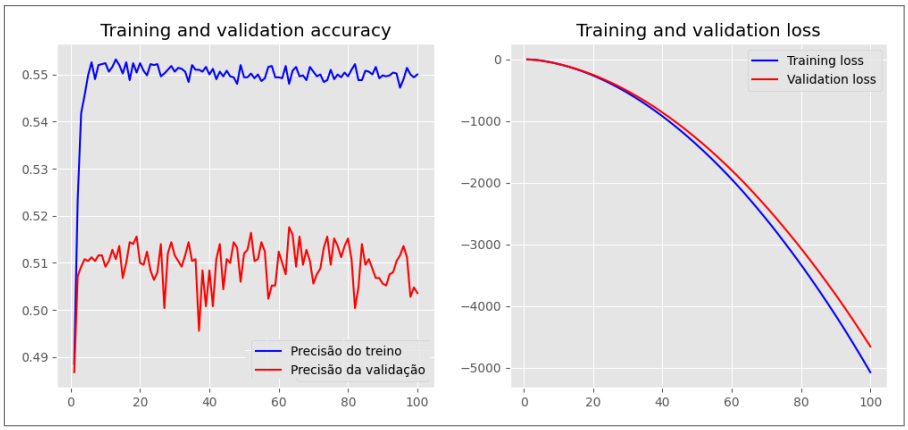


Fonte: Elaboração própria.

D.2.1.2 Rede neural recorrente com 2 camadas

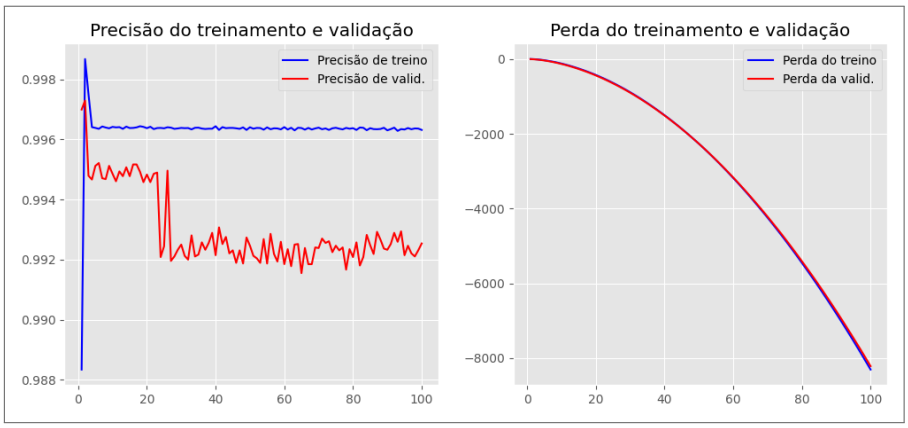
Ativação através da função ReLu na camada de entrada e sigmoide na camada restante. Produzidas através da técnica de incorporação de palavras *CountVector* e comparadas entre *tokenização*, *stemming* e *lemmatization*.

Figura 89 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor *CountVector* e técnica *tokenização*



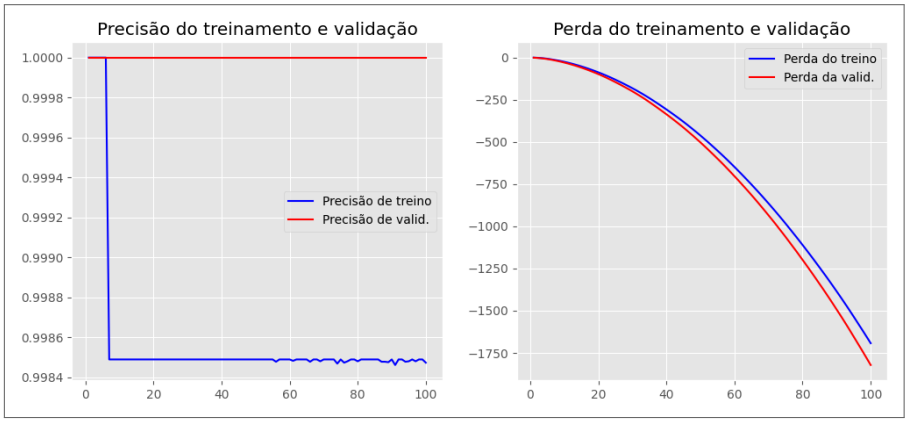
Fonte: Elaboração própria.

Figura 90 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor *CountVector* e técnica *stemming*



Fonte: Elaboração própria.

Figura 91 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor *CountVector* e técnica *lemmatization*



Fonte: Elaboração própria.

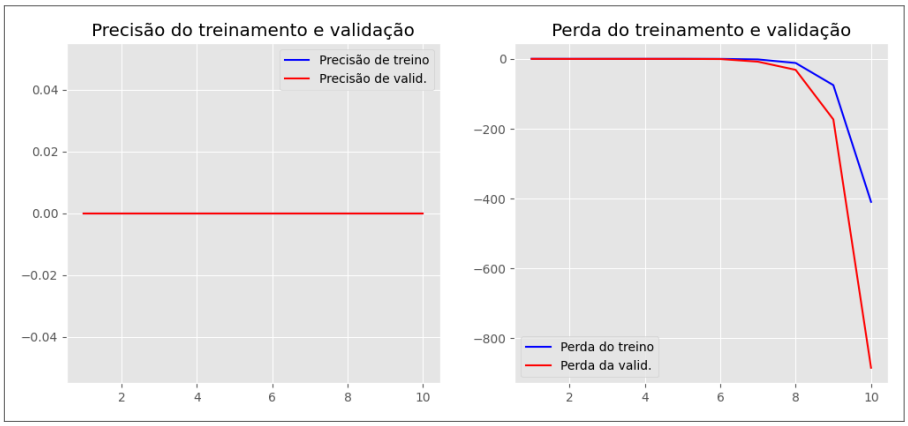
D.2.1.3 Rede Neural Convolucional com 5 camadas

Produzidas através da técnica de incorporação de palavras *CountVector* e comparadas entre *tokenização*, *stemming* e *lemmatization*.

Figura 92 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Convolucional para treino e validação. Realizado com o vetor *CountVector* e técnica *tokenização*

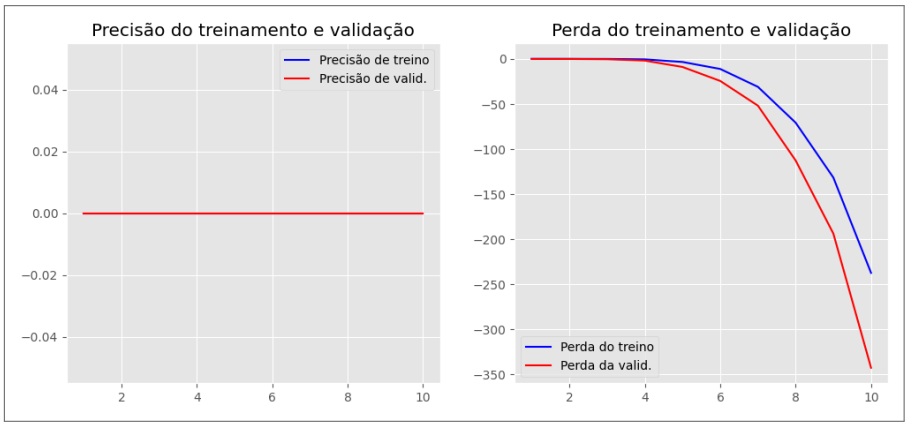
Fonte: Elaboração própria.

Figura 93 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Convolucional para treino e validação. Realizado com o vetor *CountVector* e técnica *stemming*



Fonte: Elaboração própria.

Figura 94 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Convolucional para treino e validação. Realizado com o vetor *CountVector* e técnica *lemmatization*



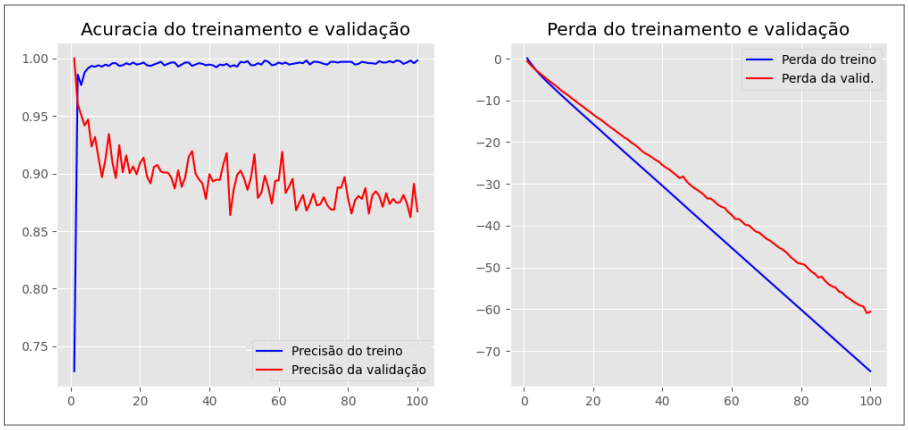
Fonte: Elaboração própria.

D.2.2 *Term frequency–inverse document frequency (TFIDF)*

D.2.2.1 Rede Neural Recorrente com 4 camadas

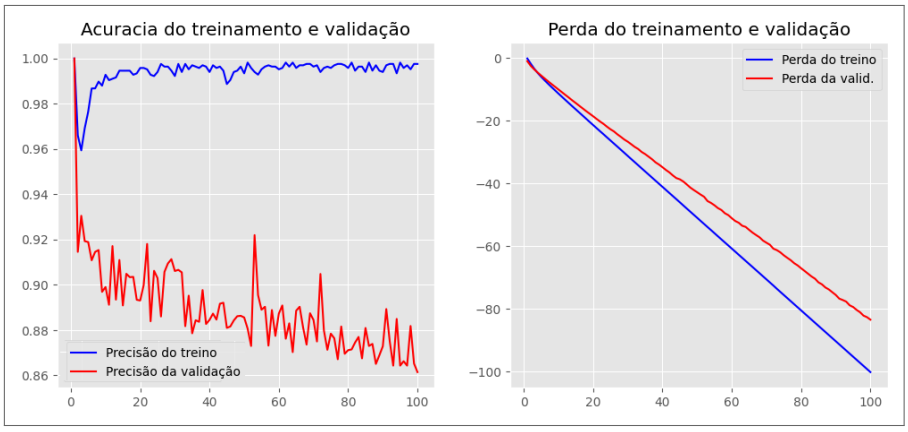
Ativação através da função ReLu na camada de entrada e sigmoide nas camadas restante. Produzidas através da técnica de incorporação de palavras TFIDF e comparadas entre *tokenização*, *stemming* e *lemmatization*.

Figura 95 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,1)$ e técnica *tokenização*



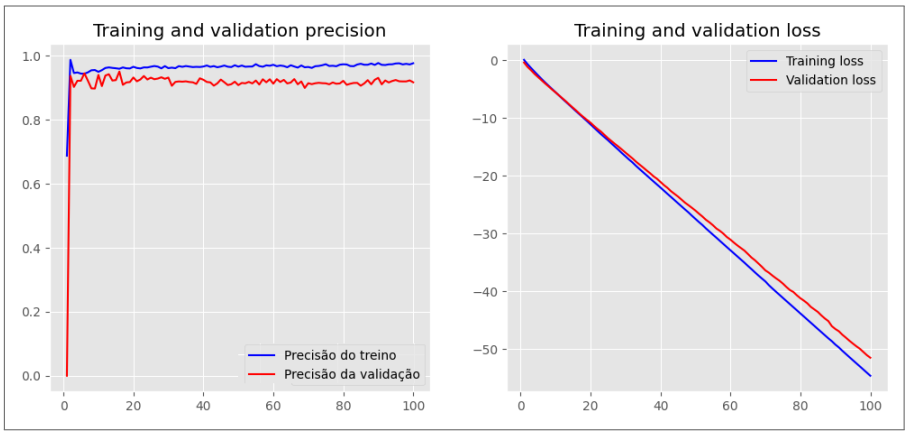
Fonte: Elaboração própria.

Figura 96 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,1)$ e técnica *stemming*



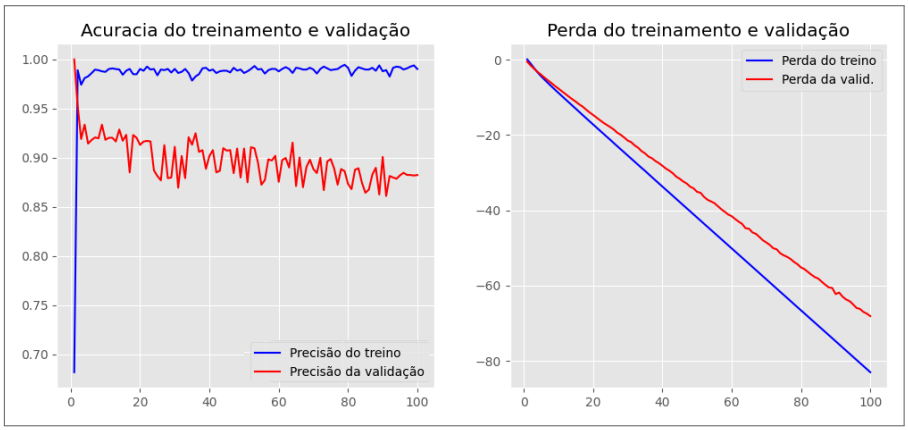
Fonte: Elaboração própria.

Figura 97 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,1)$ e técnica *lemmatization*



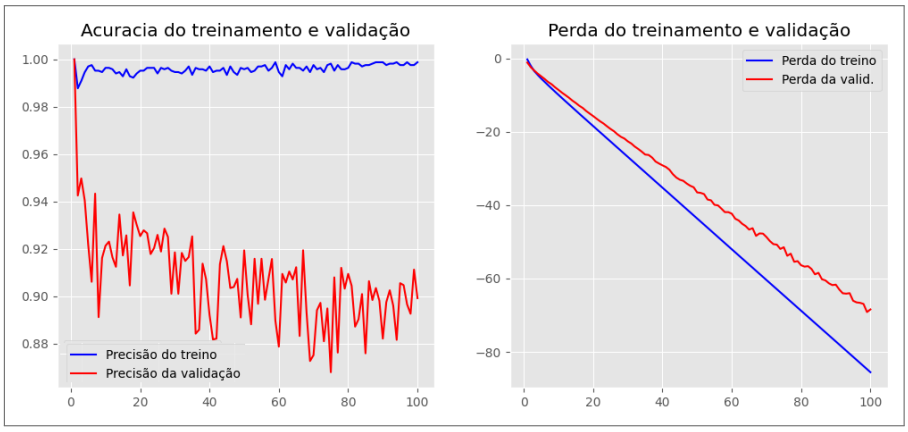
Fonte: Elaboração própria.

Figura 98 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,2)$ e técnica *tokenização*



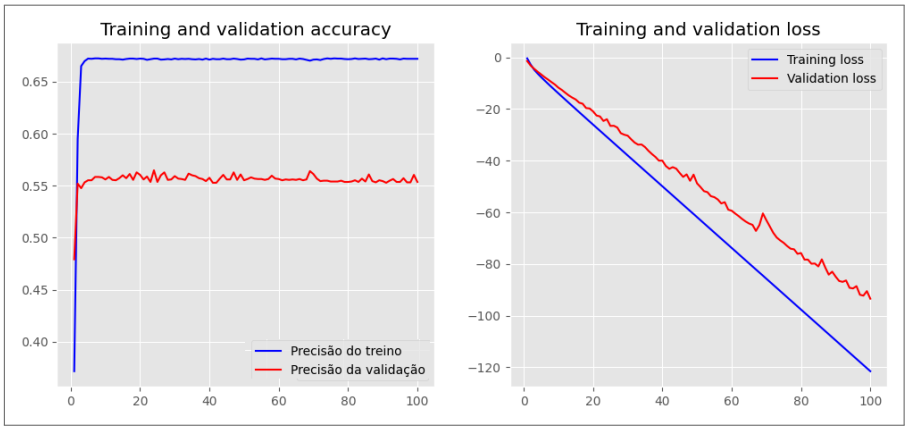
Fonte: Elaboração própria.

Figura 99 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,2)$ e técnica *stemming*



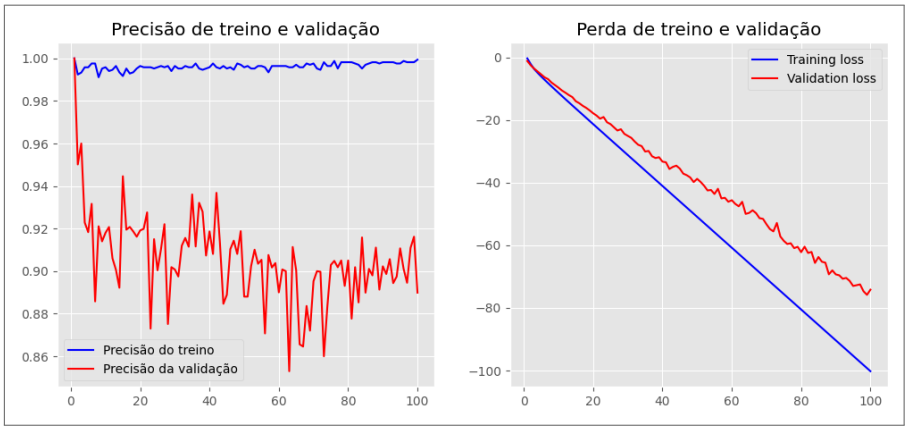
Fonte: Elaboração própria.

Figura 100 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,3)$ e técnica *tokenização*



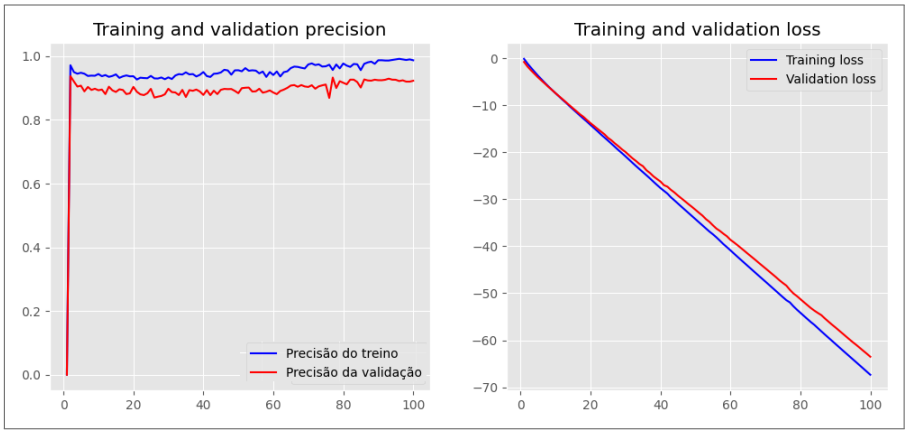
Fonte: Elaboração própria.

Figura 101 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,3)$ e técnica *stemming*



Fonte: Elaboração própria.

Figura 102 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,3)$ e técnica *lemmatization*

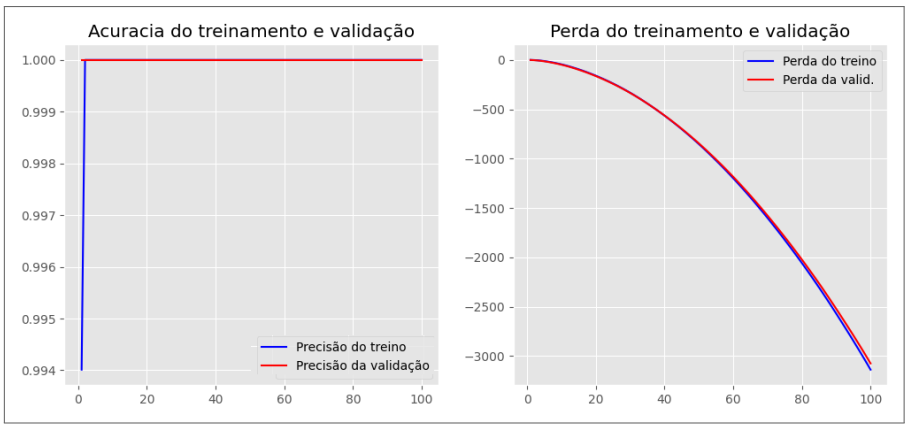


Fonte: Elaboração própria.

D.2.2.2 Rede neural recorrente com 2 camadas

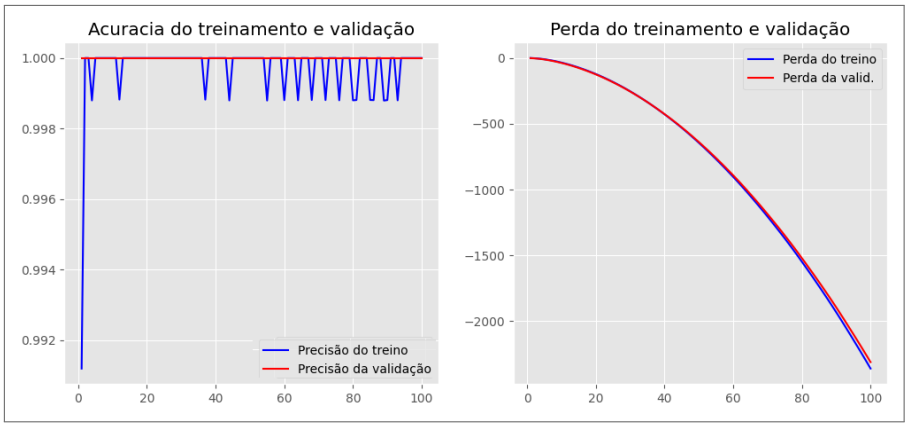
Ativação através da função ReLu na camada de entrada e sigmoide na camada restante. Produzidas através da técnica de incorporação de palavras TFIDF e comparadas entre *tokenização*, *stemming* e *lemmatization*.

Figura 103 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor TFIDF $n - gram = (1, 1)$ e técnica *tokenização*



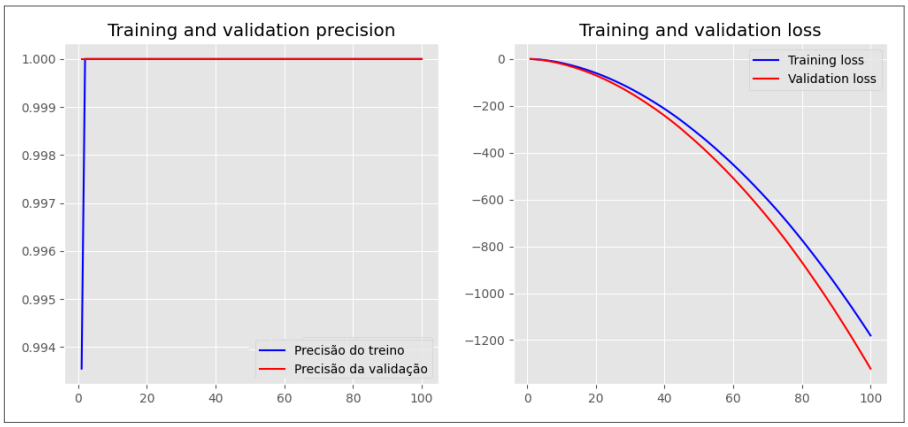
Fonte: Elaboração própria.

Figura 104 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor TFIDF $n - gram = (1, 1)$ e técnica *stemming*



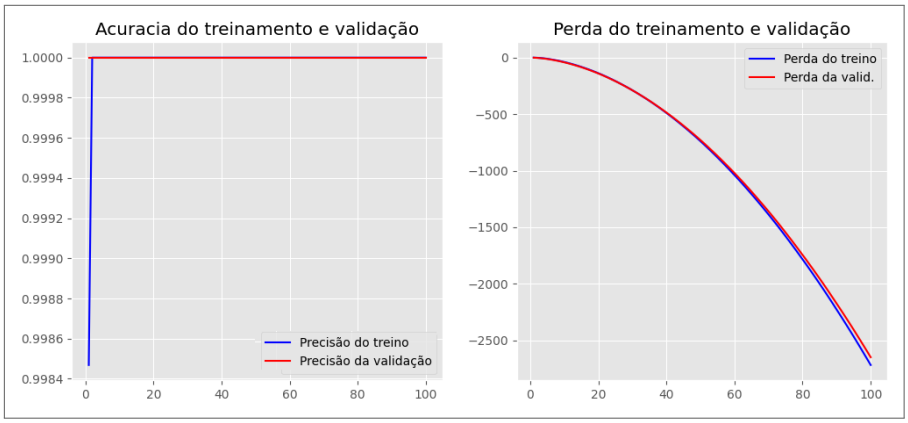
Fonte: Elaboração própria.

Figura 105 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,1)$ e técnica *lemmatization*



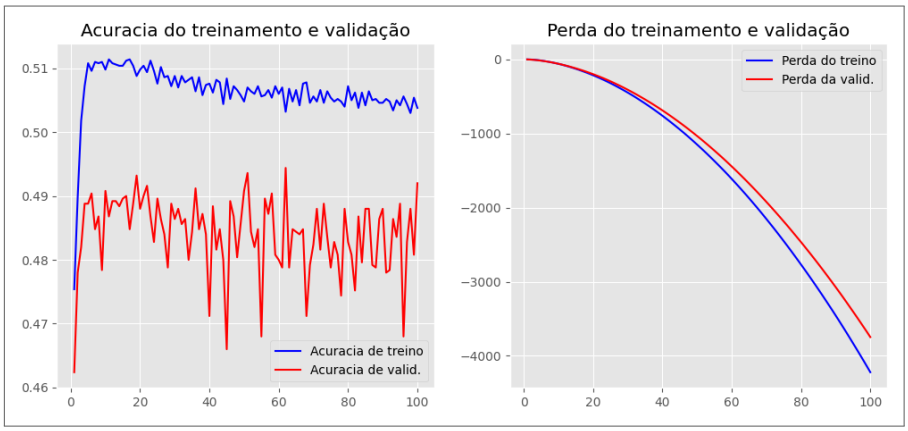
Fonte: Elaboração própria.

Figura 106 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,2)$ e técnica *tokenização*



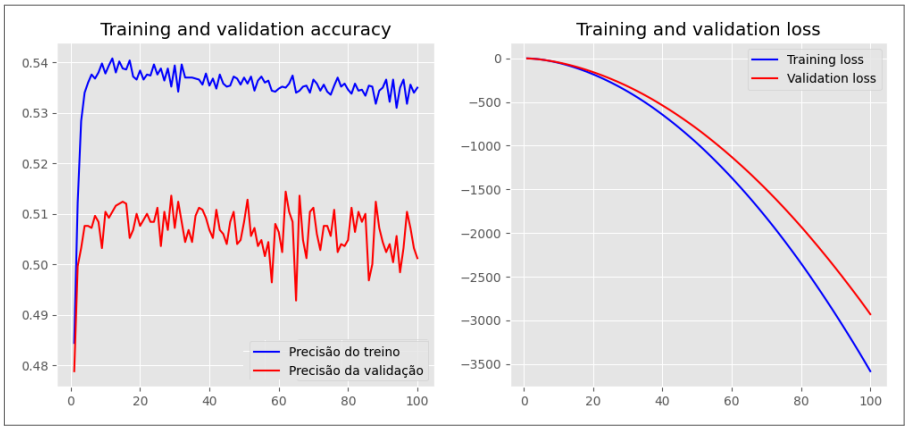
Fonte: Elaboração própria.

Figura 107 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,2)$ e técnica *stemming*



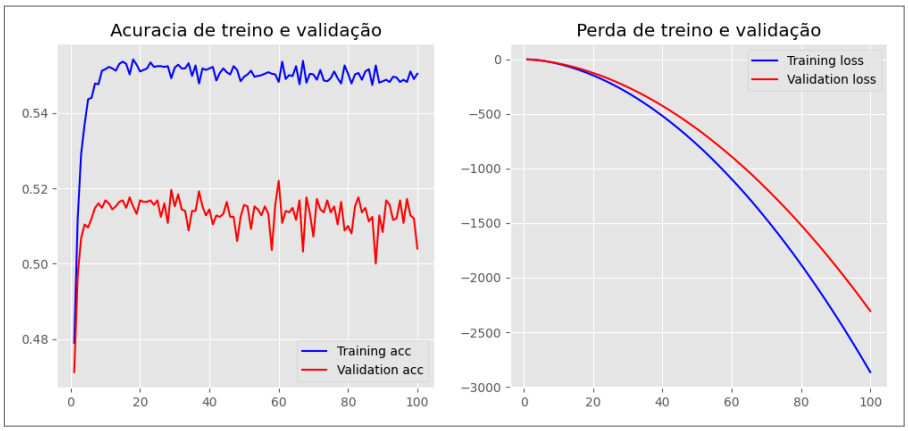
Fonte: Elaboração própria.

Figura 108 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,3)$ e técnica *tokenização*



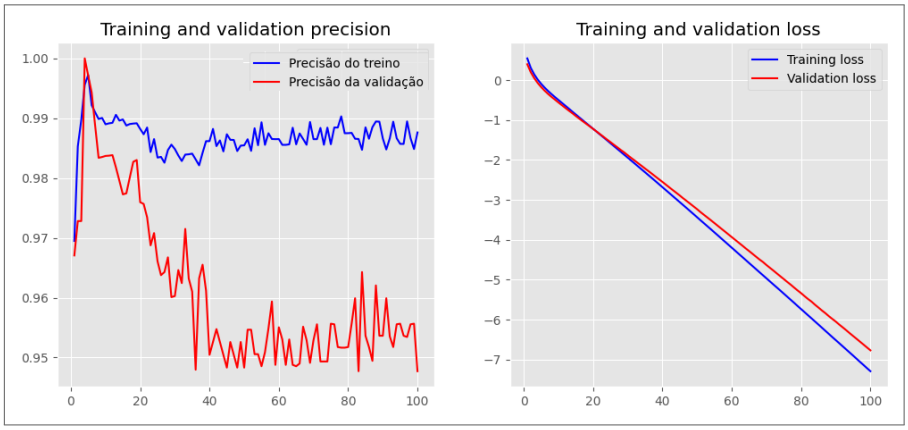
Fonte: Elaboração própria.

Figura 109 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,3)$ e técnica *stemming*



Fonte: Elaboração própria.

Figura 110 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Recorrente para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,3)$ e técnica *lemmatization*

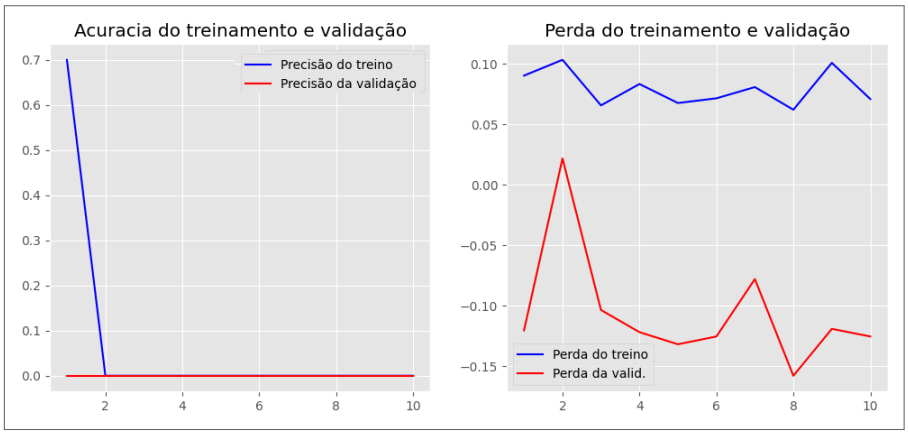


Fonte: Elaboração própria.

D.2.2.3 Rede Neural Convolucional com 5 camadas

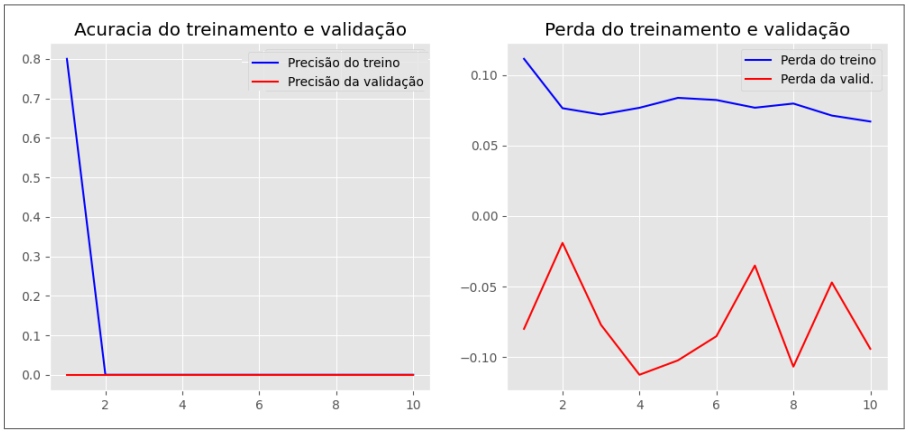
Produzidas através da técnica de incorporação de palavras *CountVector* e comparadas entre *tokenização*, *stemming* e *lemmatization*.

Figura 111 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Convolucional para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,1)$ e técnica *tokenização*



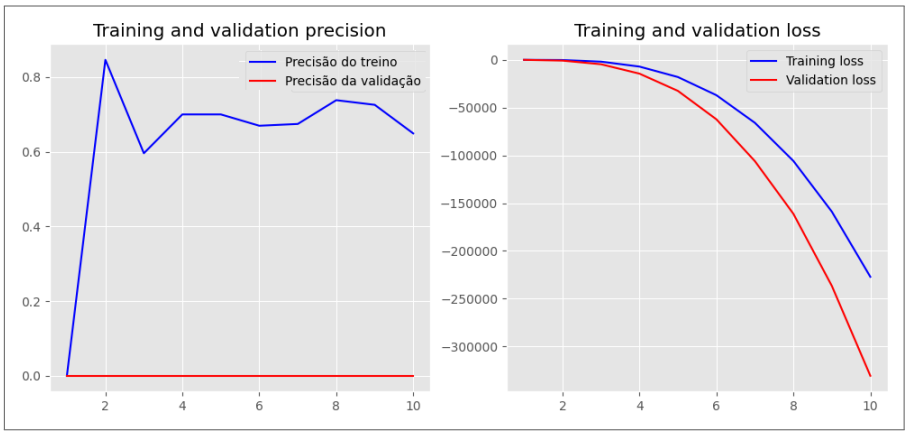
Fonte: Elaboração própria.

Figura 112 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Convolucional para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,1)$ e técnica *stemming*



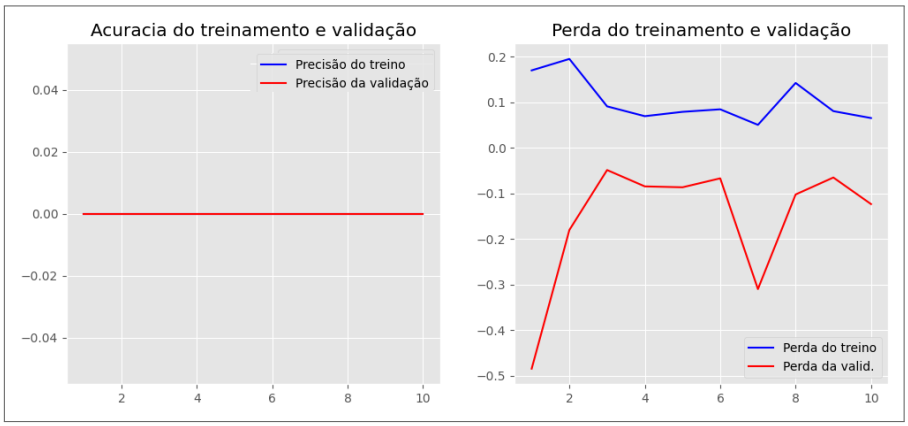
Fonte: Elaboração própria.

Figura 113 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Convolucional para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,1)$ e técnica *lemmatization*



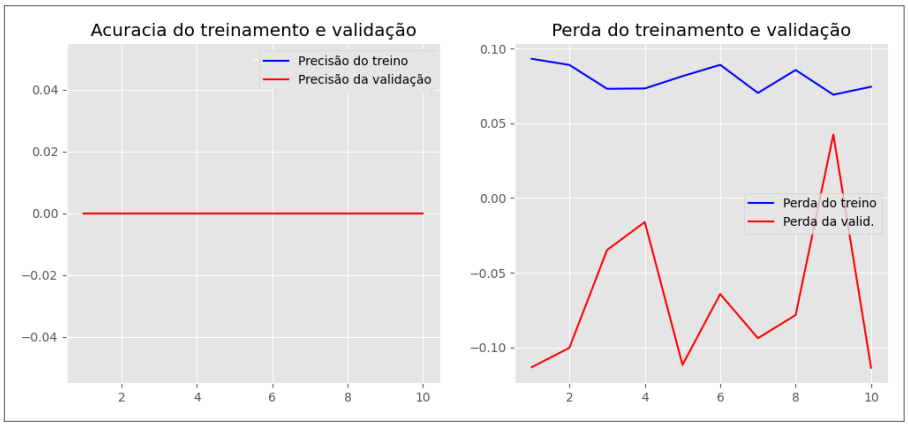
Fonte: Elaboração própria.

Figura 114 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Convolucional para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,2)$ e técnica *tokenização*



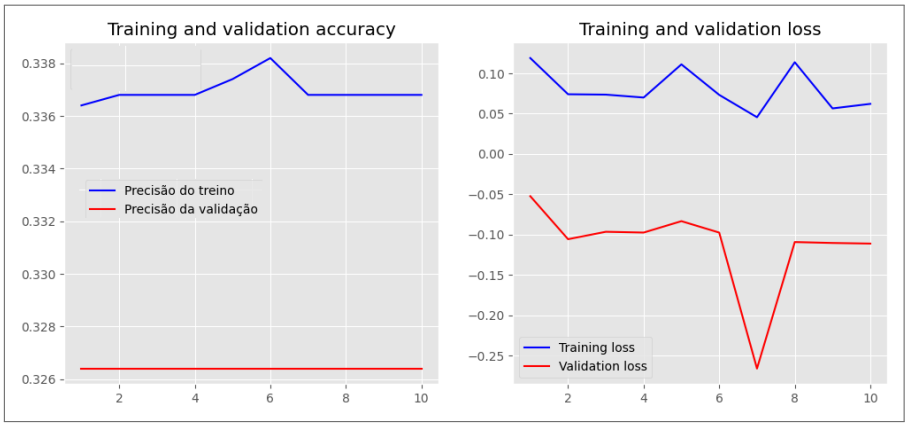
Fonte: Elaboração própria.

Figura 115 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Convolucional para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,2)$ e técnica *stemming*



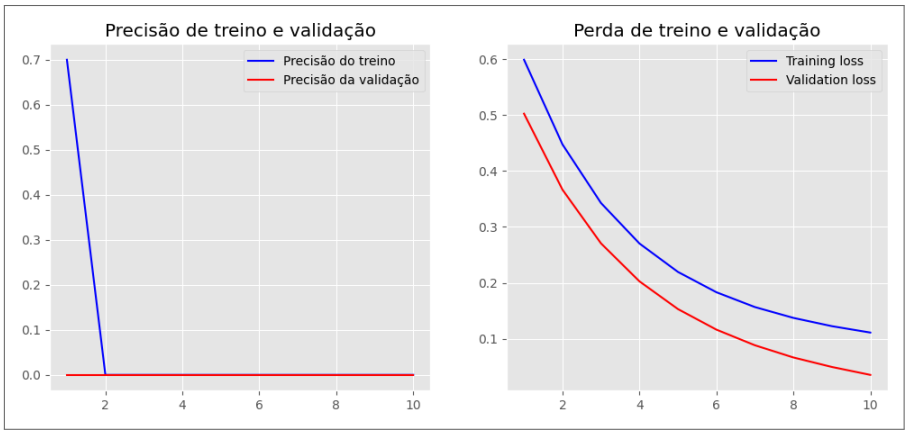
Fonte: Elaboração própria.

Figura 116 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Convolucional para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,3)$ e técnica *tokenização*



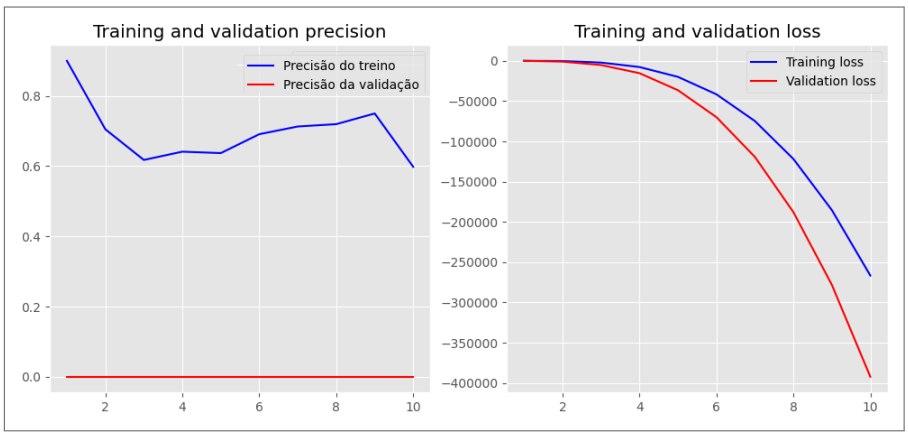
Fonte: Elaboração própria.

Figura 117 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Convolucional para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,3)$ e técnica *stemming*



Fonte: Elaboração própria.

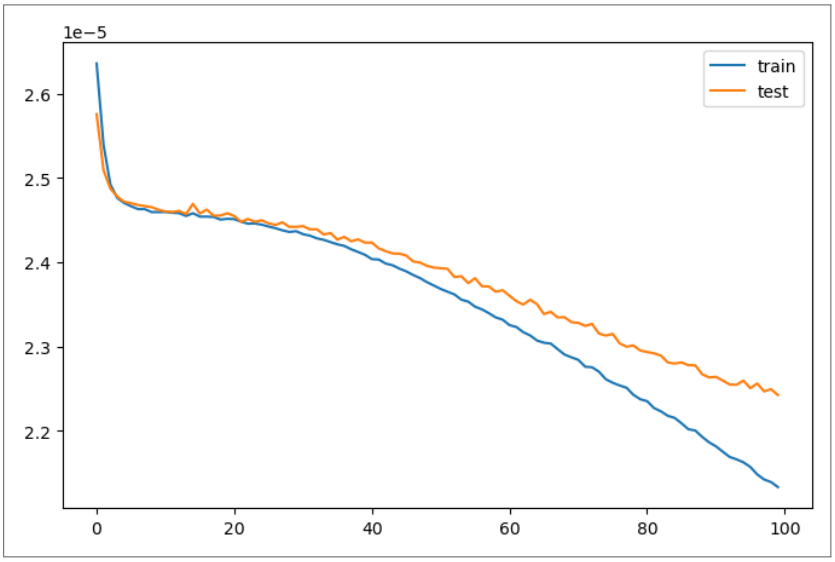
Figura 118 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da Rede Neural Convolucional para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,3)$ e técnica *lemmatization*



Fonte: Elaboração própria.

APÊNDICE E – REDE DE CODIFICADORES AUTOMÁTICOS COM 6 NÍVEIS APLICADOS AO CORPUS PORTUGUESE *TWEETS FOR SENTIMENT ANALYSIS* (KAGGLE)

Figura 119 – Treino e Validação - Perda, Dimensionalidade: 200, Linear/ReLu [**N-GRAM = (1, 2)**]



Fonte: Elaboração própria.

Tabela 31 – Resultados do codificador automático e técnica de incorporação de palavras TFIDF com stemização. (*Kaggle*) [**N-GRAM = (1, 2)**]

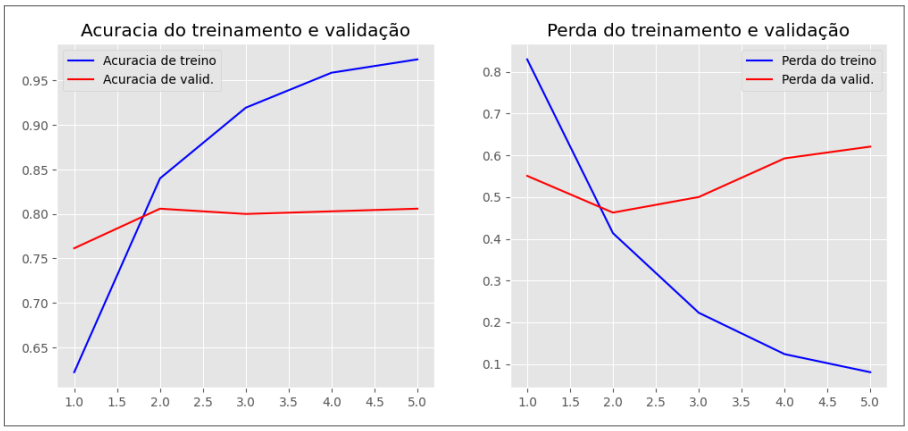
Reduzido	Classificador	Acurácia	Roc AUC	Treino Tempo (Segundos)
Sim	Naive Bayes	0.580	0.752	0.010
Não	Naive Bayes	0.674	0.850	0.003
Sim	SVC	0.595	0.783	13.214
Não	SVC	0.684	0.852	11.796
Sim	MLP	0.546	0.737	37.070
Não	MLP	0.624	0.804	68.870

Fonte: Elaboração própria.

APÊNDICE F – EXPERIMENTOS COM AS TECNICAS LSTM E BERT NOS CORPUS TWO BILLION MULTILINGUAL COVID-19 (TBCOV) E *KAGGLE*

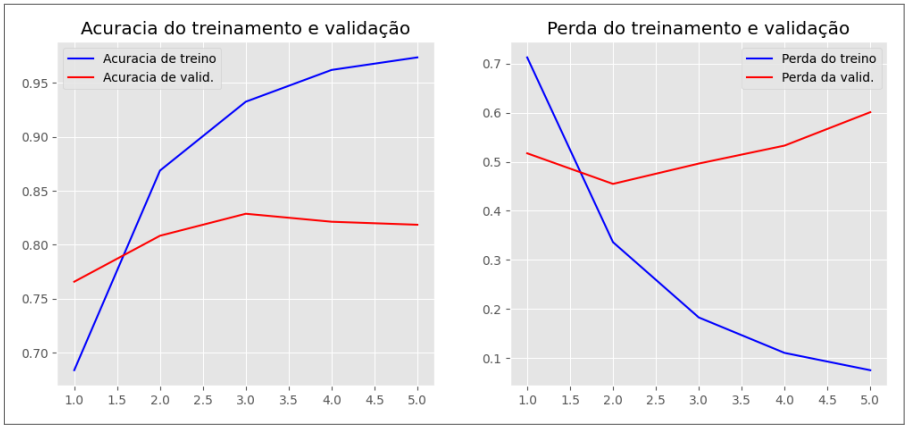
F.1 *LONG SHORT-TERM MEMORY (LSTM)*

Figura 120 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da rede neural LSTM para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,1)$ produzido a partir da amostragem de 7500 *tweets* do *corpus* TBCOV com técnica *stemming*



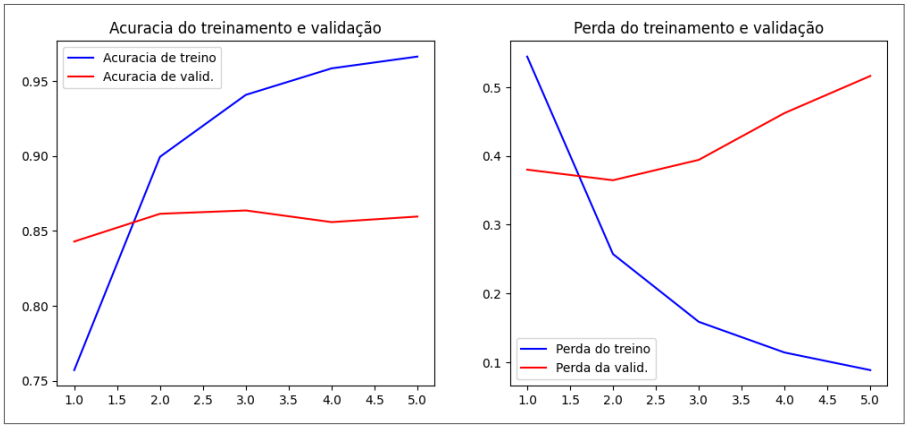
Fonte: Elaboração própria.

Figura 121 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da rede neural LSTM para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,1)$ produzido a partir da amostragem de 12000q *tweets* do *corpus* TBCOV com técnica *stemming*



Fonte: Elaboração própria.

Figura 122 – Gráfico de tendência de acurácia e perda obtida durante o aprendizado da rede neural LSTM para treino e validação. Realizado com o vetor TFIDF $n-gram = (1,1)$ produzido a partir da amostragem de 30000 *tweets* do *corpus* TBCOV com técnica *stemming*



Fonte: Elaboração própria.

F.2 *BIDIRECTIONAL ENCODER REPRESENTATIONS FOR TRANSFORMERS (BERT)*

Tabela 32 – Métricas resultantes da validação da rede neural BERT, treinado com uma amostra de 7500 *tweets* do *corpus* TBCOV.

Métrica	**Valor**
Loss	0.0960
Accuracy	0.5451
Precision	0.7935
Recall	0.8482
Specificity	0.7895

Fonte: Elaboração própria.

Tabela 33 – Métricas resultantes da validação da rede neural BERT, treinado com uma amostra de 12000 *tweets* do *corpus* TBCOV.

Métrica	**Valor**
Loss	0.1284
Accuracy	0.5358
Precision	0.8436
Recall	0.8166
Specificity	0.8563

Fonte: Elaboração própria.

Tabela 34 – Métricas resultantes da validação da rede neural BERT, treinado com uma amostra de 30000 *tweets* do *corpus* TBCOV.

Métrica	**Valor**
Loss	0.1203
Accuracy	0.5622
Precision	0.8709
Recall	0.7958
Specificity	0.8883

Fonte: Elaboração própria.

APÊNDICE G – EXPERIMENTOS COM CODIFICADORES AUTOMATICOS NO CORPUS PORTUGUESE TWEETS FOR SENTIMENT ANALYSIS (KAGGLE)

Tabela 35 – Experimentos com nós em codificadores automáticos com 5 camadas. (*Kaggle*)

NOS	MODEL	PRECISAO	RECALL	F1	ACURACY	ROC_AUC
50	NAIVE_BAYES	0.653	0.503	0.457	0.503	0.736
50	SVC	0.606	0.572	0.572	0.572	0.772
50	MLP	0.588	0.565	0.569	0.565	0.754
150	NAIVE_BAYES	0.646	0.562	0.551	0.562	0.769
150	SVC	0.65	0.61	0.613	0.61	0.802
150	MLP	0.593	0.58	0.583	0.58	0.767
300	NAIVE_BAYES	0.609	0.568	0.561	0.568	0.765
300	SVC	0.645	0.617	0.621	0.617	0.808
300	MLP	0.611	0.558	0.565	0.558	0.759
600	NAIVE_BAYES	0.635	0.591	0.585	0.591	0.777
600	MLP	0.666	0.637	0.64	0.637	0.816
600	SVC	0.606	0.552	0.533	0.552	0.78
900	NAIVE_BAYES	0.608	0.578	0.576	0.578	0.761
900	SVC	0.657	0.636	0.639	0.636	0.812
900	MLP	0.602	0.57	0.569	0.57	0.782
1200	NAIVE_BAYES	0.59	0.576	0.574	0.576	0.773
1200	SVC	0.651	0.627	0.631	0.627	0.81
1200	MLP	0.617	0.616	0.616	0.616	0.796
1500	NAIVE_BAYES	0.594	0.574	0.571	0.574	0.771
1500	SVC	0.653	0.628	0.632	0.628	0.816
1500	MLP	0.618	0.573	0.579	0.573	0.789
1700	NAIVE_BAYES	0.591	0.573	0.572	0.573	0.778
1700	SVC	0.618	0.618	0.617	0.618	0.804
1700	MLP	0.652	0.631	0.635	0.631	0.818
1900	NAIVE_BAYES	0.608	0.586	0.584	0.586	0.783
1900	SVC	0.668	0.643	0.646	0.643	0.822
1900	MLP	0.616	0.621	0.618	0.621	0.804
2200	NAIVE_BAYES	0.626	0.607	0.609	0.607	0.786
2200	SVC	0.67	0.652	0.655	0.652	0.826
2200	MLP	0.675	0.597	0.599	0.597	0.813

Fonte: Elaboração própria.

Tabela 36 – Experimentos com nós em codificadores automáticos com 4 camadas. (*Kaggle*)

NOS	MODEL	PRECISAO	RECALL	F1	ACURACY	ROC_AUC
50	NAIVE_BAYES	0.613	0.513	0.476	0.513	0.736
50	SVC	0.61	0.58	0.579	0.58	0.772
50	MLP	0.542	0.552	0.54	0.552	0.749
150	NAIVE_BAYES	0.621	0.568	0.558	0.568	0.765
150	SVC	0.636	0.606	0.609	0.606	0.797
150	MLP	0.589	0.588	0.586	0.588	0.771
300	NAIVE_BAYES	0.584	0.559	0.551	0.559	0.765
300	SVC	0.636	0.61	0.614	0.61	0.804
300	MLP	0.597	0.59	0.593	0.59	0.768
600	NAIVE_BAYES	0.614	0.582	0.575	0.582	0.773
600	MLP	0.652	0.628	0.631	0.628	0.811
600	SVC	0.599	0.608	0.601	0.608	0.799
900	NAIVE_BAYES	0.582	0.56	0.557	0.56	0.759
900	SVC	0.647	0.632	0.634	0.632	0.811
900	MLP	0.615	0.609	0.608	0.609	0.795
1200	NAIVE_BAYES	0.584	0.576	0.572	0.576	0.771
1200	SVC	0.644	0.624	0.627	0.624	0.806
1200	MLP	0.603	0.591	0.592	0.591	0.781
1500	NAIVE_BAYES	0.605	0.594	0.591	0.594	0.783
1500	SVC	0.652	0.63	0.634	0.63	0.817
1500	MLP	0.684	0.569	0.567	0.569	0.797
1700	NAIVE_BAYES	0.577	0.564	0.563	0.564	0.774
1700	SVC	0.607	0.59	0.588	0.59	0.792
1700	MLP	0.644	0.626	0.63	0.626	0.812
1900	NAIVE_BAYES	0.602	0.584	0.581	0.584	0.779
1900	SVC	0.663	0.643	0.646	0.643	0.816
1900	MLP	0.624	0.604	0.606	0.604	0.79
2200	NAIVE_BAYES	0.62	0.607	0.607	0.607	0.783
2200	SVC	0.664	0.646	0.65	0.646	0.819
2200	MLP	0.649	0.623	0.624	0.623	0.811

Fonte: Elaboração própria.

Tabela 37 – Experimentos com nós em codificadores automáticos com 3 camadas. (*Kaggle*)

NOS	MODEL	PRECISAO	RECALL	F1	ACURACY	ROC_AUC
50	NAIVE_BAYES	0.613	0.513	0.476	0.513	0.736
50	SVC	0.61	0.58	0.579	0.58	0.772
50	MLP	0.572	0.565	0.561	0.565	0.756
150	NAIVE_BAYES	0.621	0.568	0.558	0.568	0.765
150	SVC	0.636	0.606	0.609	0.606	0.797
150	MLP	0.592	0.58	0.579	0.58	0.763
300	NAIVE_BAYES	0.584	0.559	0.551	0.559	0.765
300	SVC	0.636	0.61	0.614	0.61	0.804
300	MLP	0.585	0.576	0.577	0.576	0.768
600	NAIVE_BAYES	0.614	0.582	0.575	0.582	0.773
600	MLP	0.652	0.628	0.631	0.628	0.811
600	SVC	0.609	0.584	0.59	0.584	0.79
900	NAIVE_BAYES	0.582	0.56	0.557	0.56	0.759
900	SVC	0.647	0.632	0.634	0.632	0.811
900	MLP	0.644	0.598	0.59	0.598	0.788
1200	NAIVE_BAYES	0.584	0.576	0.572	0.576	0.771
1200	SVC	0.644	0.624	0.627	0.624	0.807
1200	MLP	0.606	0.574	0.581	0.574	0.776
1500	NAIVE_BAYES	0.594	0.578	0.575	0.578	0.772
1500	SVC	0.646	0.625	0.628	0.625	0.814
1500	MLP	0.609	0.6	0.568	0.6	0.794
1700	NAIVE_BAYES	0.577	0.564	0.563	0.564	0.774
1700	SVC	0.584	0.593	0.577	0.593	0.798
1700	MLP	0.644	0.626	0.63	0.626	0.812
1900	NAIVE_BAYES	0.602	0.584	0.581	0.584	0.779
1900	SVC	0.663	0.643	0.646	0.643	0.816
1900	MLP	0.604	0.576	0.534	0.576	0.796
2200	NAIVE_BAYES	0.62	0.607	0.607	0.607	0.783
2200	SVC	0.664	0.646	0.65	0.646	0.819
2200	MLP	0.61	0.617	0.605	0.617	0.81

Fonte: Elaboração própria.

APÊNDICE H – EXPERIMENTOS COM CODIFICADORES AUTOMATICOS NO CORPUS TWO BILLION MULTILINGUAL COVID-19 (TBCOV)

Tabela 38 – Experimentos com nós em codificadores automáticos com 5 camadas. (*TBCOV*)

NOS	MODEL	PRECISAO	RECALL	F1	ACURACY	ROC_AUC
50	NAIVE_BAYES	0.683	0.562	0.552	0.562	0.727
50	SVC	0.731	0.727	0.727	0.727	0.885
50	MLP	0.717	0.709	0.708	0.709	0.891
100	NAIVE_BAYES	0.682	0.578	0.574	0.578	0.765
100	SVC	0.748	0.744	0.745	0.744	0.9
100	MLP	0.735	0.734	0.732	0.734	0.904
350	NAIVE_BAYES	0.698	0.61	0.602	0.61	0.775
350	SVC	0.766	0.761	0.762	0.761	0.914
350	MLP	0.749	0.748	0.747	0.748	0.909
600	NAIVE_BAYES	0.691	0.597	0.59	0.597	0.784
600	SVC	0.789	0.78	0.781	0.78	0.928
600	MLP	0.738	0.737	0.735	0.737	0.91
900	NAIVE_BAYES	0.68	0.584	0.579	0.584	0.779
900	SVC	0.782	0.778	0.778	0.778	0.927
900	MLP	0.763	0.758	0.759	0.758	0.91
1200	NAIVE_BAYES	0.693	0.606	0.598	0.606	0.797
1200	SVC	0.778	0.773	0.774	0.773	0.927
1200	MLP	0.753	0.721	0.721	0.721	0.904
1400	NAIVE_BAYES	0.69	0.598	0.591	0.598	0.792
1400	SVC	0.783	0.772	0.774	0.772	0.928
1400	MLP	0.739	0.731	0.73	0.731	0.908
1600	NAIVE_BAYES	0.7	0.612	0.603	0.612	0.808
1600	SVC	0.8	0.792	0.794	0.792	0.933
1600	MLP	0.782	0.714	0.718	0.714	0.904
1800	NAIVE_BAYES	0.689	0.599	0.591	0.599	0.798
1800	SVC	0.783	0.778	0.779	0.778	0.93
1800	MLP	0.765	0.747	0.747	0.747	0.908
2000	NAIVE_BAYES	0.685	0.592	0.586	0.592	0.793
2000	SVC	0.797	0.789	0.791	0.789	0.934
2000	MLP	0.77	0.764	0.765	0.764	0.915

Fonte: Elaboração própria.

Tabela 39 – Experimentos com nós em codificadores automáticos com 4 camadas. (*TBCOV*)

NOS	MODEL	PRECISAO	RECALL	F1	ACURACY	ROC_AUC
50	NAIVE_BAYES	0.639	0.563	0.554	0.563	0.697
50	SVC	0.735	0.73	0.731	0.73	0.894
50	MLP	0.72	0.719	0.719	0.719	0.894
100	NAIVE_BAYES	0.677	0.588	0.582	0.588	0.74
100	SVC	0.762	0.751	0.753	0.751	0.901
100	MLP	0.737	0.735	0.736	0.735	0.901
350	NAIVE_BAYES	0.678	0.587	0.581	0.587	0.784
350	SVC	0.771	0.77	0.77	0.77	0.918
350	MLP	0.743	0.744	0.743	0.744	0.911
600	NAIVE_BAYES	0.693	0.602	0.595	0.602	0.784
600	SVC	0.782	0.78	0.781	0.78	0.927
600	MLP	0.763	0.758	0.76	0.758	0.912
900	NAIVE_BAYES	0.691	0.601	0.593	0.601	0.793
900	SVC	0.79	0.784	0.786	0.784	0.928
900	MLP	0.758	0.747	0.748	0.747	0.913
1200	NAIVE_BAYES	0.687	0.594	0.588	0.594	0.802
1200	SVC	0.788	0.778	0.78	0.778	0.93
1200	MLP	0.752	0.714	0.712	0.714	0.906
1400	NAIVE_BAYES	0.679	0.586	0.58	0.586	0.786
1400	SVC	0.792	0.784	0.786	0.784	0.931
1400	MLP	0.758	0.757	0.755	0.757	0.906
1600	NAIVE_BAYES	0.689	0.599	0.592	0.599	0.797
1600	SVC	0.804	0.796	0.797	0.796	0.935
1600	MLP	0.754	0.734	0.731	0.734	0.907
1800	NAIVE_BAYES	0.688	0.596	0.589	0.596	0.798
1800	SVC	0.799	0.794	0.795	0.794	0.934
1800	MLP	0.778	0.758	0.76	0.758	0.909
2000	NAIVE_BAYES	0.697	0.608	0.6	0.608	0.787
2000	SVC	0.803	0.795	0.796	0.795	0.932
2000	MLP	0.755	0.724	0.72	0.724	0.903

Fonte: Elaboração própria.

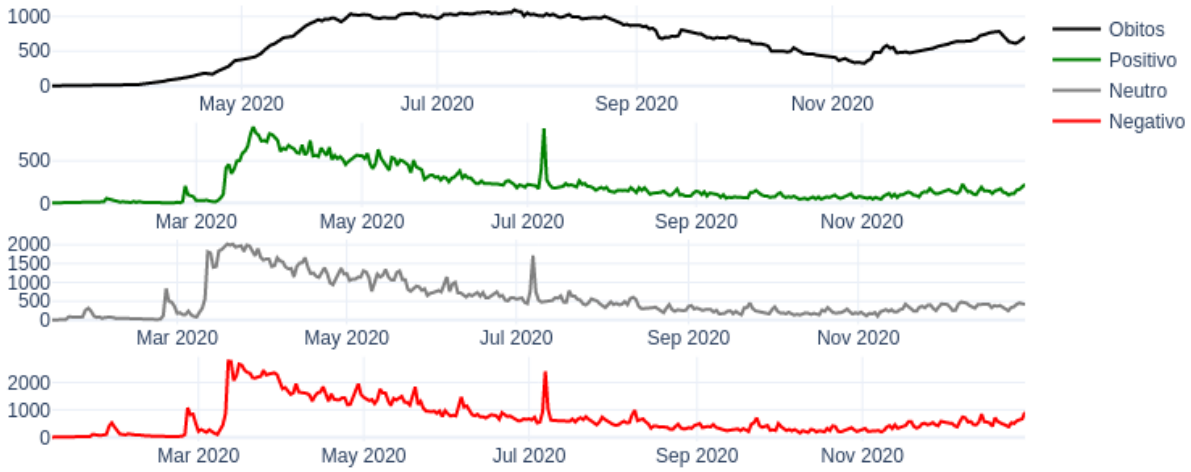
Tabela 40 – Experimentos com nós em codificadores automáticos com 3 camadas. (*TBCOV*)

NOS	MODEL	PRECISAO	RECALL	F1	ACURACY	ROC_AUC
50	NAIVE_BAYES	0.635	0.566	0.558	0.566	0.727
50	SVC	0.744	0.741	0.741	0.741	0.888
50	MLP	0.729	0.729	0.729	0.729	0.904
100	NAIVE_BAYES	0.647	0.551	0.548	0.551	0.744
100	SVC	0.745	0.744	0.744	0.744	0.899
100	MLP	0.737	0.734	0.734	0.734	0.9
350	NAIVE_BAYES	0.68	0.59	0.584	0.59	0.79
350	SVC	0.773	0.765	0.766	0.765	0.918
350	MLP	0.739	0.74	0.74	0.74	0.901
600	NAIVE_BAYES	0.685	0.597	0.59	0.597	0.797
600	SVC	0.779	0.772	0.773	0.772	0.923
600	MLP	0.769	0.767	0.767	0.767	0.919
900	NAIVE_BAYES	0.693	0.602	0.595	0.602	0.79
900	SVC	0.786	0.776	0.778	0.776	0.929
900	MLP	0.765	0.754	0.757	0.754	0.915
1200	NAIVE_BAYES	0.68	0.592	0.585	0.592	0.79
1200	SVC	0.784	0.78	0.781	0.78	0.931
1200	MLP	0.762	0.756	0.756	0.756	0.912
1400	NAIVE_BAYES	0.7	0.613	0.604	0.613	0.794
1400	SVC	0.789	0.783	0.785	0.783	0.932
1400	MLP	0.754	0.753	0.753	0.753	0.908
1600	NAIVE_BAYES	0.692	0.603	0.595	0.603	0.787
1600	SVC	0.795	0.788	0.789	0.788	0.93
1600	MLP	0.749	0.734	0.732	0.734	0.903
1800	NAIVE_BAYES	0.699	0.611	0.601	0.611	0.794
1800	SVC	0.802	0.794	0.795	0.794	0.934
1800	MLP	0.782	0.764	0.766	0.764	0.912
2000	NAIVE_BAYES	0.694	0.606	0.597	0.606	0.793
2000	SVC	0.794	0.786	0.787	0.786	0.934
2000	MLP	0.761	0.761	0.759	0.761	0.911

Fonte: Elaboração própria.

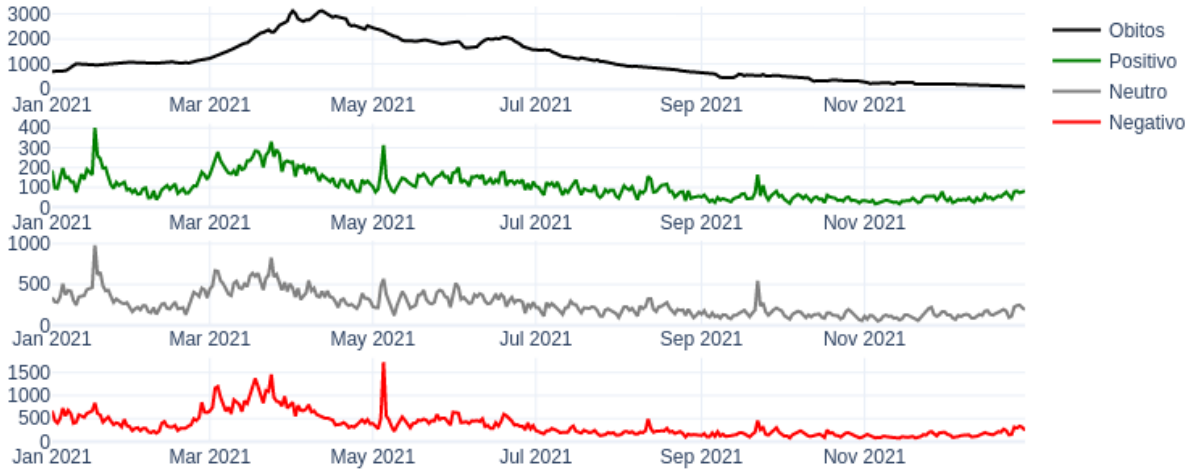
APÊNDICE I – CORPUS BRASIL. GRÁFICOS E ANALISES DOS RÓTULOS DO MODELO *STACKEDGENERALIZATION*

Figura 123 – Oscilação de sentimento X Evolução dos obitos de Covid-19 2020



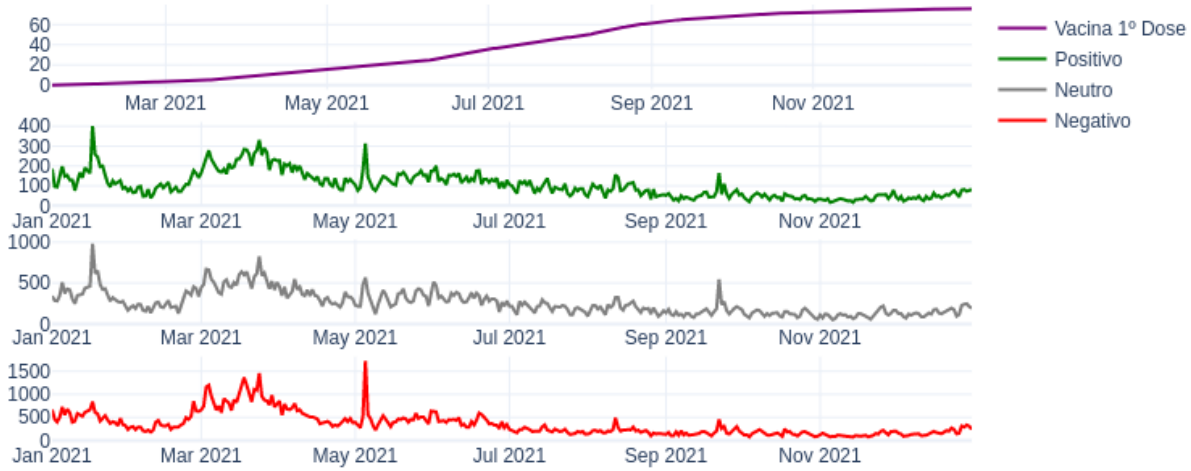
Fonte: Elaboração própria, dados provenientes da ICICT.

Figura 124 – Oscilação de sentimento X Evolução dos obitos de Covid-19 2021



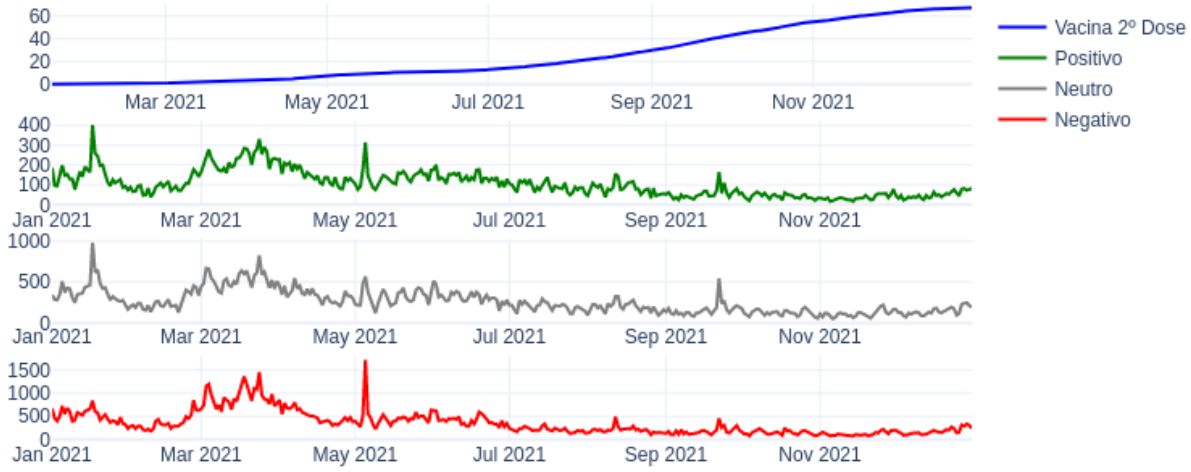
Fonte: Elaboração própria, dados provenientes da ICICT.

Figura 125 – Oscilação de sentimento X Evolução da vacina 1º dose de Covid-19 2021



Fonte: Elaboração própria, dados provenientes da ICICT.

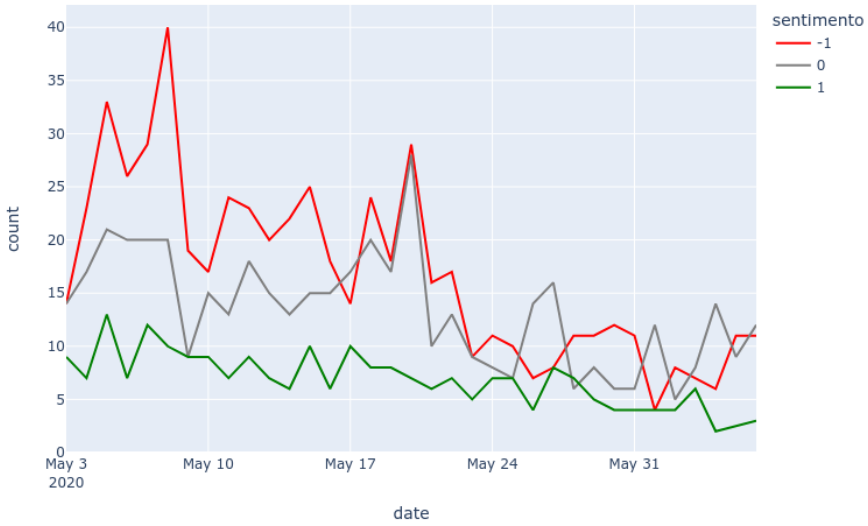
Figura 126 – Oscilação de sentimento X Evolução da vacina 2º dose de Covid-19 2021



Fonte: Elaboração própria, dados provenientes da ICICT.

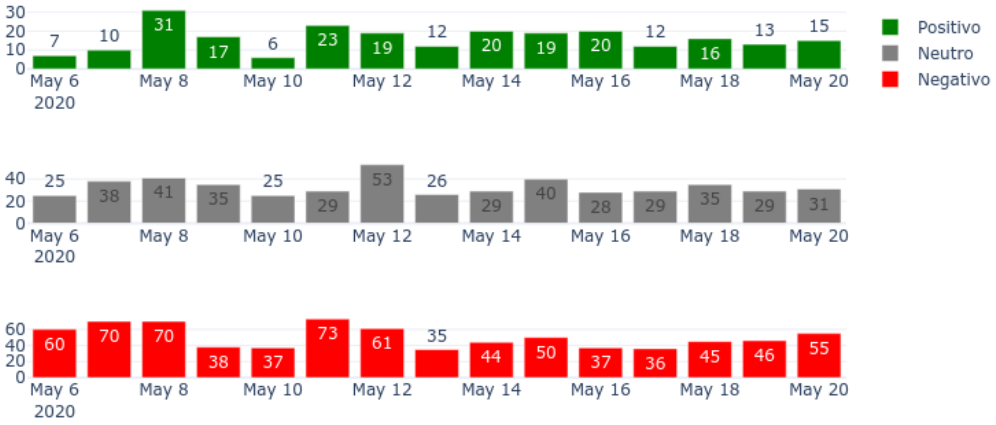
I.1 FORTALEZA *LOCKDOWN*, DE 3 DE MAIO A 6 DE JUNHO DE 2020

Figura 127 – Oscilação de sentimento, Fortaleza - CE (3 de maio a 6 de junho de 2020) - Linhas



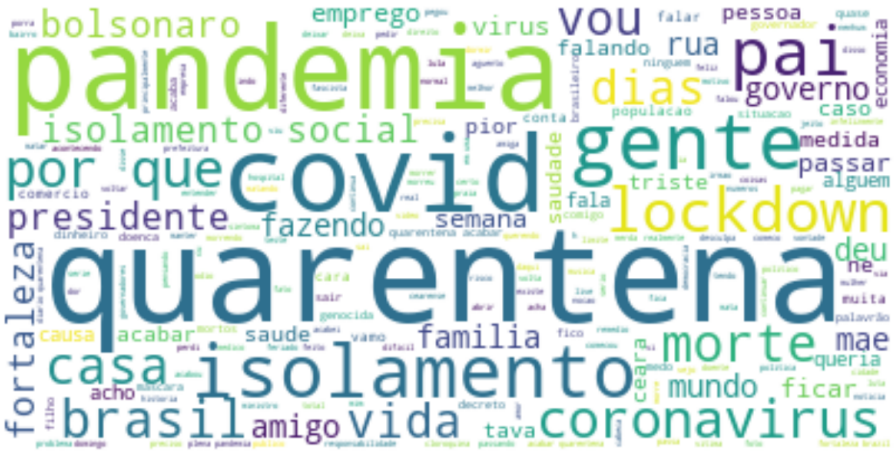
Fonte: Elaboração própria. Gráfico que mostra a oscilação por meio de linhas das polaridades classificadas pelo modelo *stacked generalization* sob o corpus de *tweets* filtrado para a região de Fortaleza.

Figura 128 – Oscilação de sentimento, Niterói - RJ (6 a 20 de maio de 2020)



Fonte: Elaboração própria.

Figura 129 – Nuvem de palavras para polaridade Positiva 8 de maio a 1 de junho de 2020



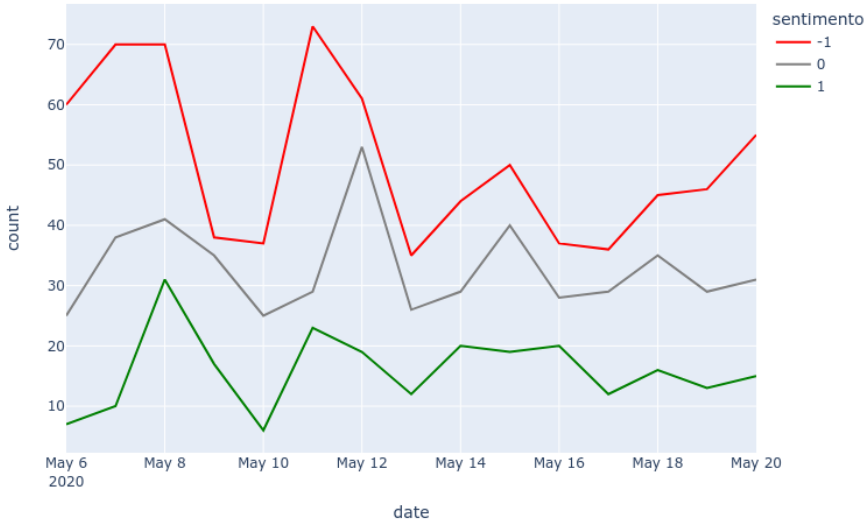
Fonte: Elaboração própria.

Figura 130 – Nuvem de palavras para polaridade Negativo 8 de maio a 1 de junho de 2020

Fonte: Elaboração própria.

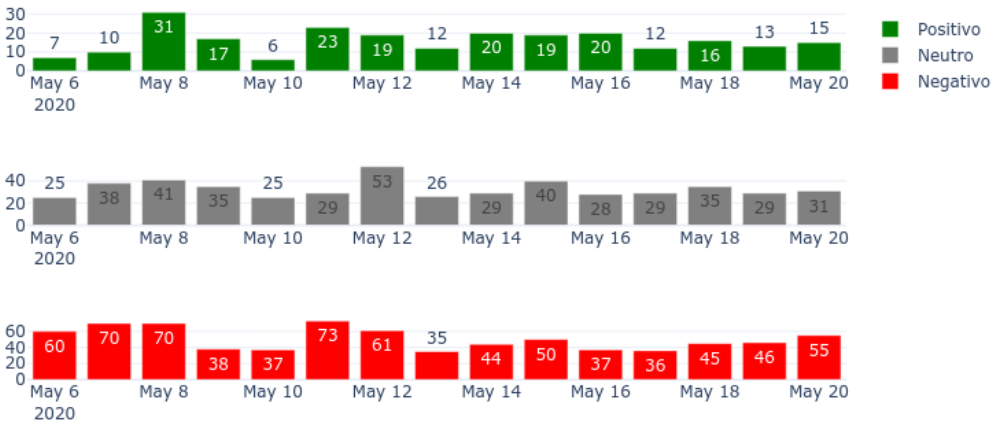
I.2 NITERÓI *LOCKDOWN*, DE 3 DE MAIO A 6 DE JUNHO DE 2020

Figura 131 – Oscilação de sentimento, Niterói - RJ (3 de maio a 6 de junho de 2020) - Linhas



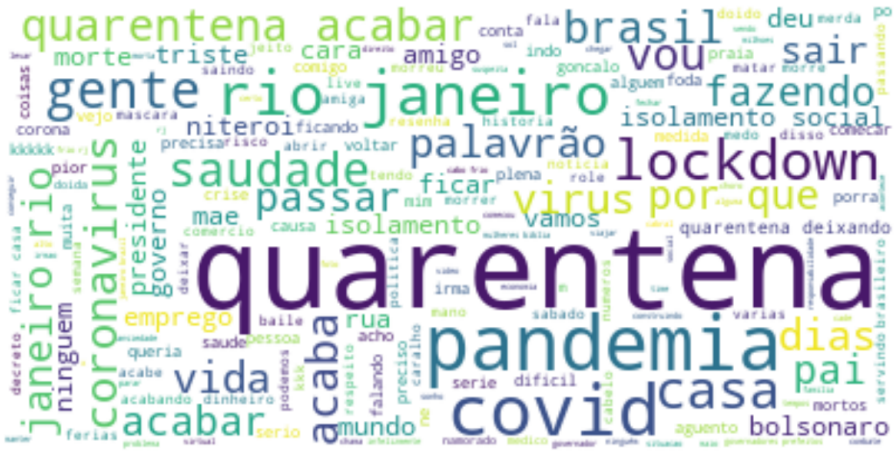
Fonte: Elaboração própria. Gráfico que apresenta a oscilação por meio de linhas das polaridades classificadas pelo modelo *stacked generalization* sob o corpus de *tweets* filtrado para a região de Niterói.

Figura 132 – Oscilação de sentimento, Niterói - RJ (6 a 20 de maio de 2020)



Fonte: Elaboração própria.

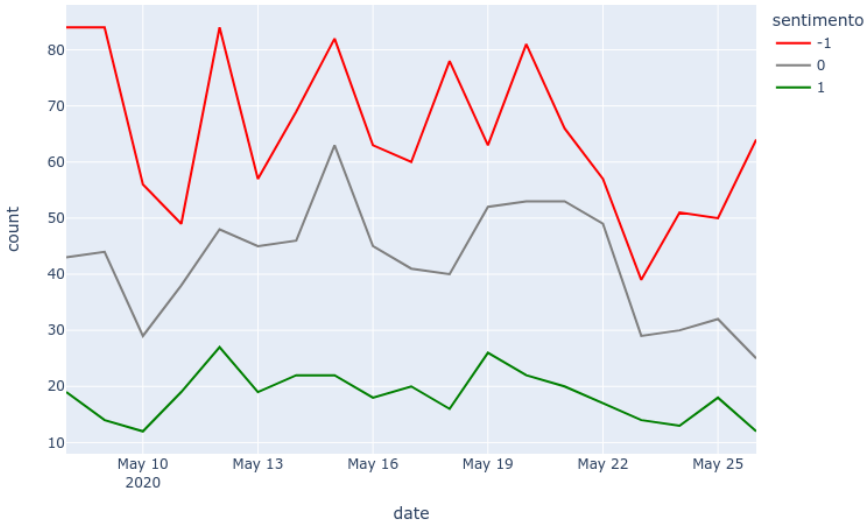
Figura 133 – Nuvem de palavras para polaridade Positiva 13 a 21 de maio de 2020



Fonte: Elaboração própria.

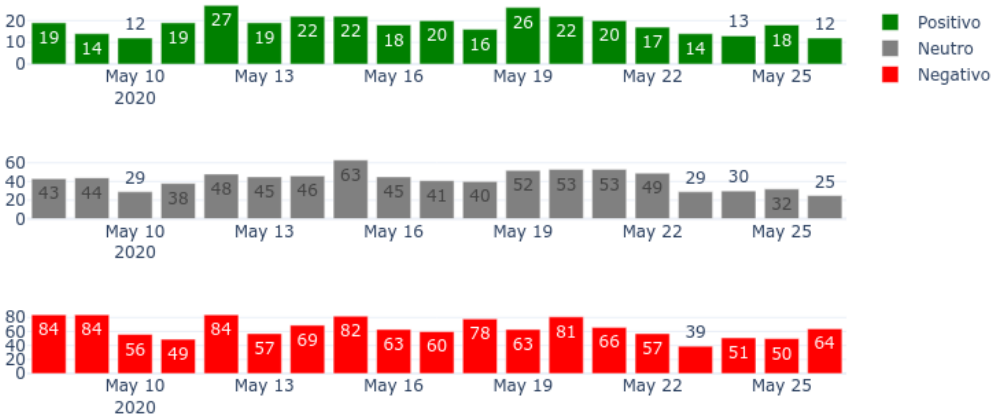
I.3 CURITIBA *LOCKDOWN*, DE 6 A 26 DE MAIO DE 2020

Figura 134 – Oscilação de sentimento, Curitiba - PR (3 de maio a 6 de junho de 2020) - Linhas



Fonte: Elaboração própria. Gráfico que exibe a oscilação por meio de linhas das polaridades classificadas pelo modelo *stacked generalization* sob o corpus de *tweets* filtrado para a região de Curitiba.

Figura 135 – Oscilação de sentimento, Curitiba - PR (8 a 26 de maio de 2020)



Fonte: Elaboração própria.

Figura 136 – Nuvem de palavras para polaridade Positiva 13 a 21 de maio de 2020

Fonte: Elaboração própria.