



UNIVERSIDADE ESTADUAL DO TOCANTINS  
CÂMPUS DE PALMAS  
CURSO DE SISTEMAS DE INFORMAÇÃO

**ANÁLISE COMPARATIVA DE MODELOS DE REDES NEURAIIS  
PARA ANÁLISE DE SENTIMENTOS EM PORTUGUÊS  
COLOQUIAL**

JULIANO CORREIA PASSOS VIEIRA

Palmas - TO

2022



UNIVERSIDADE ESTADUAL DO TOCANTINS  
CÂMPUS DE PALMAS  
CURSO DE SISTEMAS DE INFORMAÇÃO

**ANÁLISE COMPARATIVA DE MODELOS DE REDES NEURAIS  
PARA ANÁLISE DE SENTIMENTOS EM PORTUGUÊS  
COLOQUIAL**

JULIANO CORREIA PASSOS VIEIRA

Projeto apresentado ao Curso de Sistemas de Informação da Universidade Estadual do Tocantins - UNITINS como parte dos requisitos para a obtenção do grau de Bacharel em Sistemas de Informação.

Palmas - TO

2022



## CURSO DE SISTEMAS DE INFORMAÇÃO

# ANÁLISE COMPARATIVA DE MODELOS DE REDES NEURAIS PARA ANÁLISE DE SENTIMENTOS EM PORTUGUÊS COLOQUIAL

JULIANO CORREIA PASSOS VIEIRA

Projeto apresentado ao Curso de Sistemas de Informação da Universidade Estadual do Tocantins - UNITINS como parte dos requisitos para a obtenção do grau de Bacharel em Sistemas de Informação.

---

**Me. Marco Antonio Firmino de Sousa**  
Orientador

---

**Douglas Chagas da Silva**  
Convidado 1

Documento assinado digitalmente  
**gov.br**  
JEAN NUNES RIBEIRO ARAÚJO  
Data: 31/07/2022 12:25:45-0300  
Verifique em <https://verificador.iti.br>

---

**Jean Nunes Ribeiro Araújo**  
Convidado 2

Palmas - TO  
2022



**Dados Internacionais de Catalogação na Publicação  
(CIP) Sistema de Bibliotecas da Universidade Estadual  
do Tocantins**

---

V658a

VIEIRA, Juliano Correia Passos

Análise comparativa de modelos de redes neurais  
para análise de sentimentos em português coloquial.  
Juliano Correia Passos Vieira. - Palmas, TO, 2022

Monografia Graduação - Universidade Estadual do  
Tocantins – Câmpus Universitário de Palmas - Curso de  
Sistemas de Informação, 2022.

Orientador: Marco Antonio Firmino de Sousa

1. Análise de sentimentos. 2. Processamento de  
linguagem natural. 3. Redes neurais.

**CDD 610.7**

---

TODOS OS DIREITOS RESERVADOS – A reprodução total ou parcial, de qualquer forma ou por  
qualquer meio deste documento é autorizado desde que citada a fonte. A violação dos direitos do  
autor (Lei nº 9.610/98) é crime estabelecido pelo artigo 184 do Código Penal.

Elaborado pelo sistema de geração automática de ficha catalográfica da UNITINS com os  
dados fornecidos pelo(a) autor(a).



GOVERNO DO  
**TOCANTINS**

## ATA DE DEFESA DE TRABALHO DE CONCLUSÃO DE CURSO DO CURSO DE SISTEMAS DE INFORMAÇÃO DA FUNDAÇÃO UNIVERSIDADE ESTADUAL DO TOCANTINS - UNITINS

Aos **27** dias do mês de **Junho** de **2022**, reuniu-se na Fundação Universidade Estadual do Tocantins, Câmpus Palmas, Bloco B, às **20:30 horas**, sob a Coordenação do Professor **Marco Antônio Firmino de Sousa**, a banca examinadora de Trabalho de Conclusão de Curso em Sistemas de Informação, composta pelos examinadores Professor **Marco Antônio Firmino de Sousa** (Orientador), Professor **Douglas Chagas da Silva** e Professor **Jean Nunes Ribeiro Araújo**, para avaliação da defesa do trabalho intitulado **“Análise Comparativa de Modelos de Redes Neurais para Análise de Sentimentos em Português Coloquial”** do acadêmico **Juliano Correia Passos Vieira** como requisito para aprovação na disciplina Trabalho de Conclusão de Curso (TCC). Após exposição do trabalho realizado pelo acadêmico e arguição pelos Examinadores da banca, em conformidade com o disposto no Regulamento de Trabalho de Conclusão de Curso em Sistemas de Informação, a banca atribuiu a pontuação: 9,0.


Sendo, portanto, o Acadêmico: ( X ) Aprovado ( ) Reprovado

Assinam esta Ata: 27/06/2022

Professor Orientador: Marco Antonio Firmino de Sousa

Examinador: Douglas Chagas da Silva

Examinador: Jean Nunes Ribeiro Araújo

Documento assinado digitalmente  
 JEAN NUNES RIBEIRO ARAUJO  
Data: 27/06/2022 22:33:09-0300  
Verifique em <https://verificador.iti.br>

**Prof. Marco Antonio Firmino de Sousa**  
**Presidente da Banca Examinadora**



Documento foi assinado digitalmente por MARCO ANTONIO FIRMINO DE SOUSA em 28/06/2022 15:04:41.

A autenticidade deste documento pode ser verificada no site <https://sgd-ati.to.gov.br/verificador>, informando o código verificador: 598ABDB2010D3EF4.

*Dedico este trabalho em especial, ao meu pai Francisco Vieira Campos (in memoriam) e a minha mãe, e , a toda minha família, que foram pilares da minha força para eu conseguir alcançar meus objetivos.*

# Agradecimentos

Agradeço primeiramente a Deus, pela minha vida, e por me dar forças para conseguir superar todos os obstáculos encontrados ao longo da jornada que é essa graduação.

Agradeço a minha mãe, em especial, por sempre ter batalhado muito para me ajudar a conquistar meus objetivos.

Agradeço ao meu amado pai Francisco Vieira Campos (in memoriam), que não está mais entre nós, mas continua sendo uma das minhas maiores forças na vida. Sua lembrança me inspira e me faz persistir.

Agradeço a minha família, por sempre ter me apoiado.

Agradeço ao grupo de colegas de turma, que se tornou uma grande amizade, criado nos primeiros períodos de graduação apelidado de "GdP", pelo companheirismo e pela ajuda.

Agradeço também ao meu orientador Me. Marco Antonio Firmino de Sousa por me orientar no desenvolvimento deste trabalho.

*“Que os nossos esforços desafiem as impossibilidades.  
Lembrai-vos de que as grandes proezas da história  
foram conquistadas do que parecia impossível.  
(Charles Chaplin)*



# Resumo

A análise de sentimentos é uma das principais tarefas no Processamento de Linguagem Natural - NLP, que tem como objetivo extrair as opiniões e sentimentos das pessoas à respeito de tópicos específicos, por exemplo, produtos, serviços, política etc. e pode ser aplicada e trazer grandes benefícios em diversos contextos como sistemas de apoio à decisão, planejamentos, marketing direcionado e campanhas políticas. Este trabalho tem como objetivo realizar uma análise comparativa entre modelos de redes neurais na tarefa de predição de sentimentos focada em português coloquial. Foram utilizados os modelos *Long Short Term Memory* (LSTM), LSTM Bidirecional e BERTimbau (*Bidirectional Encoder Representation from Transformers*). O dataset do IMDB, contendo 50.000 (cinquenta mil) resenhas de filmes, em português, sendo 25.000 avaliações positivas e negativas cada, foi utilizado para treinar os modelos. Após a realização da análise comparativa foi desenvolvido um serviço de análise de sentimentos em nuvem, para o português coloquial.

**Palavras-chaves:** PLN, Análise de Sentimentos, LSTM, BERTimbau, Aprendizado de Máquina.

# Abstract

Sentiment analysis is one of the main tasks in Data Processing. Natural Language - NLP, which aims to extract the opinions and feelings of people about information, e.g. products, services, policy, etc. and be applied and bring benefits in different contexts such as support systems planning, marketing planning and political campaigns. This work has how to perform a comparative analysis between neural network models and prediction of feelings with a focus on colloquial Portuguese. Long models were used Short Term Memory (LSTM), LSTM Bidirectional and BERTimbau (Bidirectional Encoder representation of Transformers). The IMDB dataset, containing 50,000 (fifty thousand) film reviews, in Portuguese, with 25,000 positive and negative estimates each, was used to learn the models. After performing the comparative analysis, it was developed a sentiment analysis service in the cloud, for colloquial Portuguese.

**Key-words:** PNL, Sentiment Analysis, LSTM, BERTimbau, Machine Learning.

# Lista de ilustrações

Figura 1 – Esquema de célula LSTM . . . . .	25
Figura 2 – Arquitetura <i>Transformer</i> . . . . .	27
Figura 3 – Distribuição de classes . . . . .	31
Figura 4 – Distribuição de sentenças tokenizadas . . . . .	31
Figura 5 – Distribuição de classes (Twitter) . . . . .	32
Figura 6 – LSTM Função de perda . . . . .	39
Figura 7 – LSTM Acurácia . . . . .	40
Figura 8 – LSTM Bidirecional Função de perda . . . . .	40
Figura 9 – LSTM Bidirecional Acurácia . . . . .	41
Figura 10 – LSTM Função de perda . . . . .	42
Figura 11 – LSTM Acurácia . . . . .	42
Figura 12 – LSTM Bidirecional Função de perda . . . . .	43
Figura 13 – LSTM Bidirecional Acurácia . . . . .	43
Figura 14 – LSTM Matriz de confusão . . . . .	44
Figura 15 – LSTM Bidirecional Matriz de confusão . . . . .	45
Figura 16 – LSTM Matriz de confusão . . . . .	46
Figura 17 – LSTM Bidirecional Matriz de confusão . . . . .	46
Figura 18 – BERT Matriz de confusão . . . . .	47
Figura 19 – LSTM Matriz de confusão . . . . .	48
Figura 20 – LSTM Bidirecional Matriz de confusão . . . . .	49
Figura 21 – LSTM Matriz de confusão . . . . .	50
Figura 22 – LSTM Bidirecional Matriz de confusão . . . . .	50
Figura 23 – BERT Matriz de confusão . . . . .	51
Figura 24 – Arquitetura do serviço . . . . .	53

# Lista de tabelas

Tabela 1 – Parâmetros do experimento 1 . . . . .	37
Tabela 2 – Parâmetros do experimento 2 . . . . .	37
Tabela 3 – Parâmetros do experimento 3 . . . . .	38
Tabela 4 – LSTM . . . . .	44
Tabela 5 – LSTM Bidirecional . . . . .	44
Tabela 6 – LSTM . . . . .	45
Tabela 7 – LSTM Bidirecional . . . . .	46
Tabela 8 – BERT . . . . .	47
Tabela 9 – LSTM . . . . .	48
Tabela 10 – LSTM Bidirecional . . . . .	48
Tabela 11 – LSTM . . . . .	49
Tabela 12 – LSTM Bidirecional . . . . .	50
Tabela 13 – BERT . . . . .	51

# Lista de Equações

1	<i>Forget Gate</i>	24
2	<i>Input Gate 1</i>	25
3	<i>Input Gate 2</i>	25
4	<i>Input Gate 3</i>	25
5	<i>Output Gate 1</i>	25
6	<i>Output Gate 2</i>	26

# Lista de abreviaturas e siglas

**API** - Application Programming Interface.

**BERT** - Bidirectional Encoder Representation from Transformers

**BiLSTM** - Bidirectional Long Short-Term Memory

**CNN** - Convolutional Neural Network

**CRF** - Conditional Random Field

**DNN** - Deep Neural Network

**GB** - Gigabyte

**GPU** - Graphics Processing Unit

**HTTPS** - Hyper Text Transfer Protocol Secure

**IMDB** - Internet Movie Database

**LSTM** - Long Short-Term Memory

**MLM** - Masked Language Model

**NLTK** - Natural Language Toolkit

**PLN/PNL** - Processamento de Linguagem Natural.

**RNN** - Recurrent Neural Network

**TPU** - Tensor Processing Unit

**URL** - Uniform Resource Locator

# Sumário

<b>1</b>	<b>INTRODUÇÃO</b>	<b>16</b>
<b>1.1</b>	<b>Justificativa</b>	<b>17</b>
<b>1.2</b>	<b>Objetivos</b>	<b>19</b>
1.2.1	Objetivo Geral	19
1.2.2	Objetivos Específicos	19
<b>2</b>	<b>REFERENCIAL TEÓRICO</b>	<b>20</b>
<b>2.1</b>	<b>Trabalhos Relacionados</b>	<b>20</b>
<b>2.2</b>	<b>Análise de sentimentos</b>	<b>20</b>
<b>2.3</b>	<b>Processamento de Linguagem Natural - PLN</b>	<b>21</b>
<b>2.4</b>	<b>Aprendizado de Máquina</b>	<b>22</b>
<b>2.5</b>	<b>Algoritmos para análise de sentimentos</b>	<b>23</b>
2.5.1	Explosão de Gradiente e Gradiente de Fuga	23
2.5.2	<i>Long Short Term Memory - LSTM</i>	24
2.5.3	LSTM Bidirecional	26
2.5.4	<i>Transformers</i>	26
2.5.5	<i>Bidirectional Encoder Representation from Transformers - BERT</i>	28
2.5.5.1	BERTimbau	29
<b>3</b>	<b>METODOLOGIA</b>	<b>30</b>
<b>3.1</b>	<b>Dataset</b>	<b>30</b>
<b>3.2</b>	<b><i>Dataset</i> de validação</b>	<b>31</b>
<b>3.3</b>	<b>Tecnologias utilizadas</b>	<b>32</b>
<b>3.4</b>	<b>Implementação dos modelos de <i>Machine Learning</i></b>	<b>34</b>
3.4.1	LSTM e LSTM Bidirecional	34
3.4.1.1	Hiperparâmetros	35
3.4.2	BERT (BERTimbau)	35
3.4.2.1	Hiperparâmetros	36
<b>3.5</b>	<b>Descrição dos experimentos</b>	<b>36</b>
3.5.1	Experimento 1	36
3.5.1.1	Treinamento de modelos LSTM e LSTM Bidirecional, com etapa de limpeza dos dados	36
3.5.2	Experimento 2	37
3.5.2.1	Treinamento BERT sem etapa de limpeza dos dados	37
3.5.3	Experimento 3	38

3.5.3.1	Treinamento de modelos LSTM e LSTM Bidirecional, sem etapa de limpeza dos dados . . . . .	38
3.5.4	Experimento 4 . . . . .	38
3.5.4.1	Comparativo dos modelos em predição do <i>dataset</i> de teste . . . . .	38
3.5.5	Experimento 5 . . . . .	38
3.5.5.1	Comparativo dos modelos em predição do <i>dataset</i> de validação, que contém comentários do twitter . . . . .	38
<b>4</b>	<b>RESULTADOS . . . . .</b>	<b>39</b>
<b>4.1</b>	<b>Métricas de avaliação . . . . .</b>	<b>39</b>
<b>4.2</b>	<b>Resultados do experimento 1 . . . . .</b>	<b>39</b>
4.2.1	LSTM e LSTM Bidirecional treinados com etapa de limpeza dos dados . . .	39
<b>4.3</b>	<b>Resultados do experimento 2 . . . . .</b>	<b>41</b>
4.3.1	BERT treinado sem etapa de limpeza dos dados . . . . .	41
<b>4.4</b>	<b>Resultados do experimento 3 . . . . .</b>	<b>41</b>
4.4.1	LSTM e LSTM Bidirecional treinados sem etapa de limpeza dos dados . . .	41
4.4.1.1	LSTM . . . . .	42
4.4.1.2	LSTM Bidirecional . . . . .	43
<b>4.5</b>	<b>Resultados do experimento 4 . . . . .</b>	<b>44</b>
4.5.1	Modelos Treinados com etapa de limpeza dos dados . . . . .	44
4.5.1.1	LSTM . . . . .	44
4.5.1.2	LSTM Bidirecional . . . . .	44
4.5.2	Modelos Treinados sem etapa de limpeza dos dados . . . . .	45
4.5.2.1	LSTM . . . . .	45
4.5.2.2	LSTM Bidirecional . . . . .	45
4.5.2.3	BERT . . . . .	47
<b>4.6</b>	<b>Resultados do experimento 5 . . . . .</b>	<b>47</b>
4.6.1	Modelos Treinados com etapa de limpeza dos dados . . . . .	47
4.6.1.1	LSTM . . . . .	47
4.6.1.2	LSTM Bidirecional . . . . .	48
4.6.2	Modelos Treinados sem etapa de limpeza dos dados . . . . .	49
4.6.2.1	LSTM . . . . .	49
4.6.2.2	LSTM Bidirecional . . . . .	50
4.6.2.3	BERT . . . . .	51
<b>4.7</b>	<b>Implementação do Serviço de Análise de Sentimentos em Nuvem .</b>	<b>52</b>
<b>5</b>	<b>CONCLUSÃO . . . . .</b>	<b>54</b>
	<b>REFERÊNCIAS . . . . .</b>	<b>55</b>



# 1 Introdução

A disseminação de informações nas redes sociais, como *Facebook*, *Twitter*, *Instagram*, fórum de notícias etc. ocorre de forma simultânea e em grande escala. As redes sociais tornaram-se um rico recurso de informação para empresas e pesquisadores que podem ser analisadas para obter informações valiosas usando a PLN (Processamento de Linguagem Natural) e técnicas de inteligência artificial. O vasto repositório de informações fornecidas nas plataformas de mídia social é não processado e bruto, e ao longo do tempo as tecnologias estão evoluindo para processar os dados e extrair informações valiosas disso. Essas informações podem ser analisadas e utilizadas no apoio à decisão e no desenvolvimento de políticas eficazes em diferentes áreas relacionadas a negócios, política, entretenimento, edificação médica e social (RANI; GILL; GULIA, 2021).

Seguindo essa ideia de explorar a opinião das pessoas, com o objetivo de extrair informações que surgiu a área de pesquisa chamada Análise de Sentimentos, que segundo (PANG; LEE, 2008) consiste num tipo de classificação de texto que trabalha com afirmações subjetivas. É conhecida também como mineração de opinião visto que processa a mesma para extrair informações sobre a percepção do público. É utilizado o Processamento de Linguagem Natural, para coletar e examinar palavras e frases de opinião de sentimento. De acordo com (MORENCY; MIHALCEA; DOSHI, 2011) a análise de sentimentos é definida como uma área de estudos com ênfase na identificação automática de estados privados, como opiniões, sentimentos, emoções, crenças, avaliações e especulações na linguagem natural. Esta área também atua na classificação de dados subjetivos como negativo, positivo ou neutro.

Para (LIU, 2012) a análise de sentimentos (conhecida também como mineração de opinião) avalia a opinião das pessoas, atitudes, apreciações e emoções diante de entidades, tópicos específicos, indivíduos, eventos e vossos atributos que indicam sentimentos negativos ou positivos. A análise de sentimentos possui utilização em várias áreas do mercado consumidor, por exemplo, avaliação de produtos, descoberta de tendências dos consumidores para melhorar campanhas de marketing, encontrar opiniões a partir de tópicos em alta, por exemplo, através do twitter.

Neste sentido, este trabalho visa implementar e comparar diferentes tipos de redes neurais para a tarefa de análise de sentimentos em português coloquial, visto que na maioria das redes sociais os usuários utilizam por meio de linguagem informal para se comunicar, expressar suas opiniões e comentar. Também será desenvolvido e disponibilizado um serviço de análise de sentimentos em nuvem a partir do modelo que obtiver melhor desempenho nos comparativos.

## 1.1 Justificativa

De acordo com (BEHERA et al., 2021), a análise de sentimento é considerada uma área de pesquisa em ascensão desde o início dos anos 2000. O autor (HUSSAIN; CAMBRIA, 2018) afirma que a análise de sentimentos tem como objetivo analisar e extrair conhecimento de informações subjetivas postadas na Internet. Devido à sua ampla gama de aplicações acadêmicas e organizacionais, e também ao crescimento exponencial da Web 2.0, a análise de sentimentos tem sido um campo de pesquisa em alta nas áreas de mineração de dados e processamento de linguagem natural (PNL) recentemente.

Para (ONAN, 2019) a análise de sentimentos é uma das principais tarefas no processamento de linguagem natural, onde são derivadas percepções, observações, sentimentos, opiniões ou julgamentos sobre um assunto específico. Para organizações empresariais, governos e tomadores de decisão individuais, o reconhecimento de sentimentos pode ser fundamental. Pode trazer grandes benefícios para governos, sistemas de apoio à decisão e indivíduos reconhecer opiniões públicas sobre políticas, produtos e organizações (ONAN; TOÇOĞLU, 2021).

(DIAS, 2012) A análise de sentimentos está sendo bastante estudada nas áreas de mineração de texto, mineração de dados e mineração da Web. O interesse por esta área de pesquisa se deve ao crescimento dos meios de comunicação na internet, como discussões, comentários em redes sociais, blogs, como o *Facebook* e *Twitter*, algo que proporcionou um grande volume de dados contendo opiniões de usuários, e que pode ser acessado de forma fácil. Então, a análise de sentimentos aplicada nesses dados forma uma grande e rica fonte de informações onde é possível entender expectativas e opiniões das pessoas a respeito de um serviço, produto, etc.

Visando a possibilidade de obter opiniões das pessoas sobre diversos temas, a análise de sentimentos tem despertado um grande interesse, tanto no âmbito científico, onde existem diversos desafios, mas também na área de negócios, por causa de tantos benefícios que são possíveis de se obter através da compreensão do sentimento das pessoas a respeito de produtos, serviços, aplicada a tomada de decisões, planejamentos, campanhas de marketing, entre outros. A análise de sentimentos está sendo usada em diversas áreas e em diversos propósitos:

- Análise de produtos, onde uma empresa ou usuários possuem interesse na opinião de consumidores sobre um determinado produto. Esta análise pode ser realizada com base nos comentários ou através da extração e sumarização das características do produto (ZHANG et al., 2009);
- Análise de lugares, onde um turista que planeja viajar pode usar as opiniões de terceiros para melhor definir sua viagem, assim indo a locais que receberam melhores

avaliações ([APPELQUIST et al., 2010](#));

- Análise de empresas na bolsa de valores, onde o objetivo é identificar o humor do mercado financeiro em relação às empresas na bolsa de valores com base nas opiniões de analistas, visando identificar a tendência dos preços das mesmas ([O'REILLY, 2007](#)).

## 1.2 Objetivos

### 1.2.1 Objetivo Geral

O objetivo geral deste projeto é implementar e comparar diferentes modelos de redes neurais para a tarefa de análise de sentimentos focada em português coloquial e desenvolver um serviço de análise de sentimentos em nuvem a partir do modelo que obtiver melhor desempenho nos comparativos.

### 1.2.2 Objetivos Específicos

- Pesquisar e utilizar diferentes tipos de modelos de deep learning;
- Comparar desempenhos de diferentes modelos de redes neurais, em específico de modelos de redes neurais (*Recurrent Neural Networks* (RNN) e arquitetura *Transformer*);
- Desenvolver e disponibilizar um serviço de análise de sentimentos em nuvem, para o português coloquial.

## 2 Referencial Teórico

### 2.1 Trabalhos Relacionados

A análise de sentimentos possui aplicação em diversas áreas do mercado consumidor, por exemplo, avaliação de produtos, descoberta de tendências dos consumidores, também nos âmbitos político, governamental, etc. de forma que pode trazer grandes benefícios. Devido à isso tem sido um campo de pesquisa em alta nas áreas de mineração de dados e processamento de linguagem natural.

Mesmo sendo um modelo recente, o BERTimbau já vem sendo aplicado em diversas tarefas de Processamento de Linguagem Natural. (LOPES; CORREA; FREITAS, 2021) Ajustaram m-BERT e BERTimbau para uma tarefa de extração de aspecto, já (LEITE et al., 2020) para classificação de sentenças tóxicas, onde superou outras soluções de *bag-of-words*. (JIANG et al., 2021) e (NETO et al., 2021) avaliaram o ajuste fino do BERTimbau para uma tarefa de detecção de ironia. (CARRIÇO; QUARESMA, 2021) Explorou formas diferentes de retirar características de sua camada de saída (*token* CLS, vetor máximo e vetor médio), considerando uma tarefa de similaridade semântica. Os autores (SILVA; SERAPIAO, 2018) treinaram uma rede neural convolucional (CNN) para a tarefa de detecção de discursos de ódio na língua portuguesa. O desempenho da rede desenvolvida foi testada em duas configurações diferentes: utilizando vetores pré-treinados e sem utilização de vetores pré-treinados. O autor (NETO et al., 2019) utilizou redes neurais para a tarefa de reconhecimento de entidades nomeadas no idioma português, ele utilizou vários modelos, com diferentes *embeddings* em combinação com a arquitetura BiLSTM-CRF em vários domínios específicos de textos-alvo, onde utilizou dados policiais, clínicos e geológicos.

### 2.2 Análise de sentimentos

Para (LIU, 2012), a análise de sentimento é definida como uma área de estudo que busca analisar as opiniões, sentimentos, avaliações e emoções das pessoas à respeito de tópicos específicos como, produtos, serviços, organizações, indivíduos, eventos, temas e seus respectivos atributos. O interesse por esta área de pesquisa se deve pelo desenvolvimento dos meios de comunicação na internet, como discussões, comentários em blogs, fóruns, redes sociais, como o *Facebook* e *Twitter*, que gerou volumosas quantidades de dados contendo opiniões de usuários. Então, a análise de sentimentos aplicada nesses dados forma uma grande e rica fonte de informações onde é possível entender expectativas e opiniões das pessoas a respeito de um serviço, produto, etc.

Segundo (BEHERA et al., 2021) uma enorme quantidade de dados é gerada continuamente quando os usuários publicam suas opiniões enquanto se comunicam entre si através de várias redes sociais como Twitter, *Facebook*, etc. As principais fontes de dados para a análise de sentimentos são as opiniões públicas associadas a um produto, organização, filme ou alguma outra entidade social. Estas revisões são muito importantes para a análise de negócios, pois podem desempenhar um papel crucial na tomada de decisões sobre seus produtos. A análise de sentimentos não é aplicada apenas para análises de serviços ou produtos, pode ser aplicada também para a previsão de ações, artigos de notícias e debates políticos. Por exemplo, no âmbito político, pode-se desejar saber as opiniões dos eleitores sobre alguns candidatos eleitorais ou partidos políticos. Os resultados das eleições podem sofrer influência pela previsão de sentimento das postagens políticas dos usuários. Várias plataformas como redes sociais, blogs e site são classificadas como ricas fontes de informação, pois os usuários publicam livremente suas opiniões e pensamentos a respeito de determinado tópico, algo que pode ser usado como recurso valioso na análise de sentimentos.

De acordo com (ONAN; TOÇOĞLU, 2021), a análise de sentimentos como uma das principais tarefas no processamento de linguagem natural, onde são derivadas observações, opiniões, percepções, sentimentos, ou julgamentos sobre um assunto específico. Reconhecer opiniões públicas à respeito de política, produtos, organizações e tendências pode agregar muito valor para governos, indivíduos e também sistemas de apoio à decisão.

## 2.3 Processamento de Linguagem Natural - PLN

Segundo (BIRD; KLEIN; LOPER, 2009) a linguagem natural consiste naquela utilizada em comunicações no dia-a-dia pelos seres humanos. A linguagem humana é altamente ambígua e variável e está sempre em constante metamorfose e evolução. As pessoas possuem facilidade em entender a linguagem, e possuem capacidade de expressar, perceber e interpretar significados bem matizados e elaborados, mas possuem grande dificuldade em entender e descrever formalmente as regras que regem a linguagem (GOLDBERG, 2017). Desta forma, utilizar por meio de processamento computacional para poder compreender e produzir linguagem humana se torna uma tarefa demasiadamente desafiadora.

O autor (GOLDBERG, 2017) define o Processamento de Linguagem Natural como um termo coletivo referente ao processamento computacional automático de linguagens humanas. Onde inclui tanto algoritmos que usam texto produzido por humanos como entrada, quanto algoritmos que produzem texto de aparência natural como saída. A necessidade destes algoritmos se torna cada vez maior em razão dos humanos produzirem quantidade de dados em forma de texto cada vez maior com o passar dos anos e esperam que as interfaces computacionais se comuniquem com eles utilizando linguagem natural.

O processamento de linguagem natural também é bastante desafiador, pois a linguagem humana possui muita ambiguidade, não é bem definida e está constantemente sofrendo mudanças.

Para ([CHOWDHURY, 2003](#)), o Processamento de Linguagem Natural é uma área de pesquisa e aplicação que tem como objetivo explorar a forma que os computadores podem ser utilizados para processar textos ou discursos em linguagem natural para poder realizar coisas úteis. Ainda de acordo com ele, os pilares da área de NLP se encontram em várias disciplinas, como a ciência da computação da informação, matemática, inteligência artificial e robótica, engenharia elétrica e eletrônica, psicologia, etc.

Este campo de pesquisa é muito amplo e possui muita aplicabilidade, por exemplo: tradução entre idiomas, análise de sentimentos, sumarização de textos, sistemas de pergunta e resposta, etc.

Abaixo estão algumas das técnicas de processamento de linguagem natural:

- **Tokenização:** esta etapa é muito importante no pré-processamento dos dados para extrair as palavras do texto. É responsável por dividir o texto em *tokens* com base em espaços em branco e pontuação ([GOLDBERG, 2017](#));
- **Remoção de *Stopwords*:** *stopwords* são palavras consideradas irrelevantes no texto. Geralmente são classificadas como preposições, pronomes, artigos, advérbios, e outras palavras auxiliares que são utilizadas com muita frequência;
- **Normalização morfológica:** ocorre a padronização das palavras, por exemplo, através da remoção acentos, convertendo letras maiúsculas para minúsculas, etc.

## 2.4 Aprendizado de Máquina

Com base em ([GÉRON, 2019](#)), o aprendizado de máquina como a ciência e a arte da programação de computadores que possibilita a eles aprenderem com os dados. De forma mais abrangente, o Aprendizado de Máquina é o campo de estudo que atribui aos computadores a habilidade de aprender sem ser evidentemente programados (Arthur Samuel, 1959).

Segundo ([ALVES et al., 2014](#)), aprendizado de Máquina consiste numa área da Inteligência Artificial que visa construir sistemas capazes de adquirir conhecimento de forma automática. O conhecimento é assimilado através de experiências acumuladas baseado na solução bem sucedida de problemas anteriores. As técnicas de aprendizado supervisionado destacam-se entre as técnicas de aprendizado, as quais utilizam-se dados previamente rotulados.

Os três tipos principais de Aprendizado de Máquina são ([LUDERMIR, 2021](#)):

- **Aprendizado Supervisionado:** para cada exemplo apresentado ao algoritmo de aprendizado é necessário apresentar o rótulo informando a classe que o exemplo pertence, por exemplo, na classificação de imagens, onde deve-se diferenciar imagens de cachorros e de gatos. Cada exemplo é apresentado por um vetor de atributos e pelo rótulo de sua respectiva classe associada. O algoritmo tem como objetivo desenvolver um classificador capaz de determinar de forma correta a classe de novos exemplos que ainda não foram rotulados. Este método é o mais utilizado.
- **Aprendizado Não Supervisionado:** os exemplos apresentados ao algoritmo não possuem rótulos. O algoritmo junta os exemplos pelas semelhanças encontradas em seus atributos. O algoritmo analisa os exemplos e busca determinar se alguns podem ser agrupados de alguma forma, formando conjuntos ou *clusters*. Após a determinação dos agrupamentos, é necessário realizar uma análise para definir o significado de cada agrupamento no contexto problema a ser analisado.
- **Aprendizado por Reforço:** o algoritmo não recebe a resposta certa, ao contrário disso o algoritmo recebe um sinal de reforço, de punição ou recompensa. O algoritmo realiza uma hipótese baseada nos exemplos apresentados e determinar se tal hipótese foi boa ou não. Esse tipo de aprendizado é bastante utilizado em jogos e robótica.

Neste projeto, o método utilizado foi aprendizado supervisionado.

## 2.5 Algoritmos para análise de sentimentos

De acordo com (ZHANG; WANG; LIU, 2018), as DNNs (*Deep Neural Network*) têm sido propostas para a análise de dados textuais, de forma concentrada principalmente no aprendizado da incorporação de palavras ou na execução de tarefas de aprendizado de máquina, como classificação e agrupamento nos vetores de recursos aprendidos. Entre os diversos tipos de redes profundas, as redes neurais convolucionais (CNNs) e as redes neurais recorrentes (RNNs) são as mais comuns em estudos referentes ao processamento de texto (ZHANG; WANG; LIU, 2018). O motivo para essa popularidade é a capacidade das CNNs em aprender padrões locais e o poder das RNNs com a modelagem sequencial. Apesar das RNNs serem adequadas para diversas aplicações de processamento de texto, elas sofrem de explosão de gradientes e gradientes de fuga quando há dependências de longo prazo nos dados de entrada (SONG; PARK; SHIN, 2019). Estas dependências são muito presentes na maioria das aplicações de PNL e principalmente na análise de sentimentos.

### 2.5.1 Explosão de Gradiente e Gradiente de Fuga

Segundo (HU; ZHANG; GE, 2021), o problema do gradiente de fuga no treinamento baseado em gradiente significa que os gradientes dos pesos da rede se aproximam de zero.



De acordo com (BROWNLEE, 2017) os gradientes de explosão consistem em um problema em que volumosos gradientes de erro se concentram e resultam em atualizações muito altos para os pesos do modelo da rede neural durante a etapa de treinamento. Gradiente de erro consiste na direção e na magnitude calculadas durante a etapa de treinamento de uma rede neural que é utilizada para atualizar os pesos na direção certa e na quantidade certa. Em redes neurais profundas, pode ocorrer o acúmulo dos gradientes de erro durante uma atualização e resultar em gradientes muito altos, que consequentemente resultam em grandes atualizações dos pesos da rede, e em uma rede instável. Os valores dos pesos podem tornar-se tão grandes quanto ao transbordamento e advir em valores de NaN.

Com o objetivo de resolver este problema, foram introduzidas *Long Short Term Memory* - LSTM, em português Redes de Memória de Curto Prazo. Em razão do seu potencial para resolver os problemas das RNNs padrão, o LSTM atraiu a atenção de pesquisadores da área de PLN (CAMBRIA; WANG; WHITE, 2014).

## 2.5.2 *Long Short Term Memory* - LSTM

LSTM é um tipo especial de RNN que foi projetado para lidar com o problema de desaparecimento e explosão de gradiente enfrentado por RNNs. LSTMs, assim como outros tipos de RNNs, geram sua saída com base na entrada da etapa de tempo atual e na saída da etapa de tempo anterior e enviam a saída atual para a próxima etapa de tempo. Cada unidade LSTM consiste em uma célula de memória  $c_t$ , que preserva seu estado sobre intervalos de tempo arbitrários e três portas não lineares, incluindo uma porta de entrada (*input gate*)  $i_t$ , uma porta de esquecimento (*forget gate*)  $f_t$  e uma porta de saída (*output gate*)  $o_t$ . Essas portas são projetadas para regular o fluxo de entrada e saída de informações na célula de memória (LIU; GUO, 2019).

Suponha que  $\sigma(\cdot)$ ,  $\tanh(\cdot)$ , e  $\odot$  são a função sigmoide elementar, função tangente hiperbólica e o produto, respectivamente.  $x_t$  e  $h_t$  são o vetor de entrada e o vetor de estado oculto no tempo  $t$ .  $\mathbf{U}$  e  $\mathbf{W}$  mostram as matrizes de peso de portas ou célula para a entrada  $x_t$  e estado oculto  $h_t$  e  $\mathbf{b}$  denotam os vetores de vies. A porta de esquecimento decide quais informações devem ser esquecidas emitindo um número entre  $[0,1]$ , como mostra a seguinte equação (LIU; GUO, 2019):

$$f_t = \sigma(\mathbf{W}_f \mathbf{h}_{t-1} + \mathbf{U}_f \mathbf{x}_t + \mathbf{b}_f) \quad (2.1)$$

Equacao 1 – *Forget Gate*

A porta de entrada decide quais novas informações precisam ser armazenadas, computando  $i_t$  e  $\tilde{c}_t$  e combinando-as de acordo com as equações a seguir:

$$\mathbf{i}_t = \sigma(\mathbf{W}_i \mathbf{h}_{t-1} + \mathbf{U}_i \mathbf{x}_t + \mathbf{b}_i) \quad (2.2)$$

Equacao 2 – *Input Gate 1*

$$\tilde{\mathbf{c}}_t = \tanh(\mathbf{W}_c \mathbf{h}_{t-1} + \mathbf{U}_c \mathbf{x}_t + \mathbf{b}_c) \quad (2.3)$$

Equacao 3 – *Input Gate 2*

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{c}}_t \quad (2.4)$$

Equacao 4 – *Input Gate 3*

A porta de saída decide quais partes do estado da célula devem ser emitidas de acordo com as equações a seguir:

$$\mathbf{o}_t = \sigma(\mathbf{W}_o \mathbf{h}_{t-1} + \mathbf{U}_o \mathbf{x}_t + \mathbf{b}_o) \quad (2.5)$$

Equacao 5 – *Output Gate 1*

Para capturar o contexto posterior além do contexto antecedente, o BiLSTM combina camadas para frente  $\vec{h}_t$  e para trás  $\overleftarrow{h}_t$

Abaixo está representado o esquema de uma célula LSTM:

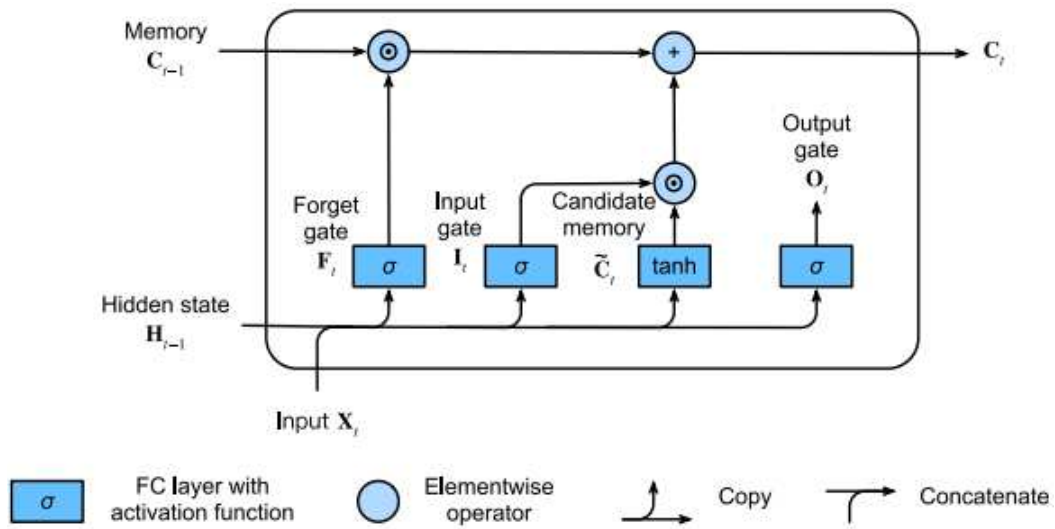


Figura 1 – Esquema de célula LSTM

Fonte: (ZHANG et al., 2021)

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t) \quad (2.6)$$

Equacao 6 – *Output Gate 2*

### 2.5.3 LSTM Bidirecional

Os LSTMs bidirecionais são uma extensão dos modelos LSTM, nos quais dois LSTMs são aplicados aos dados de entrada (SIAMI-NAMINI; TAVAKOLI; NAMIN, 2019). Na rede LSTM, a informação se propaga plenamente no sentido da esquerda para a direita, o que indica que o estado de tempo  $t$  depende apenas da informação anterior a  $t$ . Todavia, para caracterizar toda a semântica de uma revisão de entrada, as informações posteriores são tão eficazes quanto as anteriores. Assim, para uma melhor representação das informações contextuais, foi empregado o modelo LSTM Bidirecional - BiLSTM. O modelo BiLSTM é composto por duas redes LSTM e é capaz de ler revisões de entrada em ambas as direções, para frente e para trás (HAMEED; GARCIA-ZAPIRAIN, 2020).

Segundo (HAMEED; GARCIA-ZAPIRAIN, 2020), o LSTM normal processa as informações da esquerda para a direita e seu estado oculto pode ser representado como  $\vec{h}_t = LSTM(x_t, \vec{h}_{t-1})$  enquanto o LSTM inverso processa as informações da direita para a esquerda e seu estado oculto pode ser expresso como  $\overleftarrow{h}_t = LSTM(x_t, \overleftarrow{h}_{t+1})$ . E por fim, a saída do BiLSTM pode ser resumida concatenando os estados para frente e para trás como  $h_t = [\vec{h}_t, \overleftarrow{h}_t]$ .

### 2.5.4 Transformers

Segundo (VASWANI et al., 2017), arquiteturas recorrentes regularmente fatoram a computação ao longo das posições dos símbolos das sequências de entrada e saída. Ao alinhar as posições aos passos no tempo de computação, é gerada uma sequência de estados ocultos  $h_t$ , em função do estado oculto anterior  $h_{t-1}$  e da entrada para a posição  $t$ . Tal natureza fundamentalmente sequencial impede a paralelização nos exemplos de treinamento, o que se torna crítico em comprimentos de sequência mais longos, devido as restrições de memória limitarem o lote entre os exemplos.

*Transformer* consiste em uma arquitetura de modelo que ao invés de recorrência, utiliza inteiramente mecanismo de atenção para desenhar dependências globais entre entrada e saída. Surgiu como uma alternativa às arquiteturas RNNs, ele permite uma paralelização significativamente maior e inicialmente foi proposta para a tarefa de tradução (VASWANI et al., 2017).

De acordo com (VASWANI et al., 2017), esta arquitetura utiliza uma estrutura chamada *encoder-decoder*, em português codificador-decodificador. O codificador mapeia uma sequência de entrada de representações de símbolos ( $x_1, \dots, x_n$ ) para uma sequência

de representações contínuas  $z = (z_1, \dots, z_n)$ . Dado  $z$ , o decodificador gera uma sequência de saída  $(y_1, \dots, y_m)$  de símbolos um elemento por vez. A cada passo o modelo é auto-regressivo, consumindo os símbolos gerados anteriormente como entrada adicional ao gerar o próximo. O *Transformer* segue essa arquitetura geral utilizando auto-atenção empilhada e camadas totalmente conectadas ponto a ponto para o codificador e o decodificador, apresentadas nas metades esquerda e direita respectivamente, na figura abaixo:

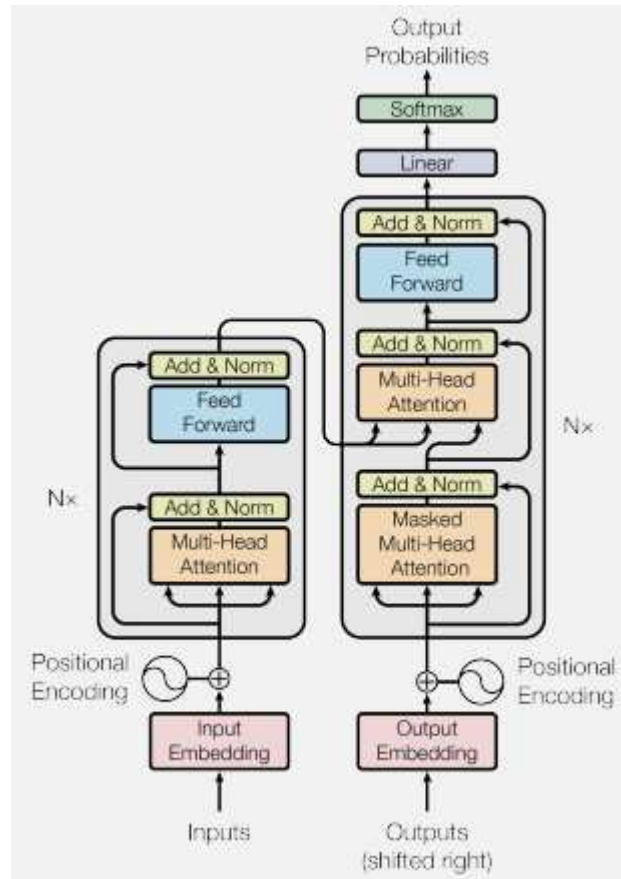


Figura 2 – Arquitetura *Transformer*

Fonte: ((VASWANI et al., 2017))

### 2.5.5 *Bidirectional Encoder Representation from Transformers* - BERT

O *Bidirectional Encoder Representation from Transformers* - BERT, em português, Representações de codificador bidirecional de Transformadores, é um algoritmo de *machine learning*, do *Google*, focado em processamento de linguagem natural que é projetado para considerar o contexto de uma palavra do lado esquerdo e direito simultaneamente. Apesar do conceito ser simples, ele melhora os resultados em várias tarefas de PNL, por exemplo sistemas de perguntas e respostas e análise de sentimentos (DEVLIN et al., 2019).

O pré-treinamento esquerdo e direito do BERT é realizado utilizando máscaras de modelo de linguagem modificadas, chamadas de modelagem de linguagem mascarada (MLM). Esse recurso MLM tem como objetivo mascarar uma palavra aleatória em uma frase com uma pequena probabilidade. Ao mascarar uma palavra, o modelo substitui a palavra por um *token* [MASK]. Posteriormente o modelo tenta prever a palavra mascarada usando o contexto da esquerda e da direita da palavra mascarada por meio de transformadores. Além da extração de contexto à esquerda e à direita através de MLM, o modelo BERT tem um objetivo chave adicional que o difere dos trabalhos anteriores, ou seja, a previsão da próxima sentença (HOANG; BIHORAC; ROUCES, 2019).

Segundo (DEVLIN et al., 2018), o BERT consiste num novo modelo de representação de linguagem, que usa utiliza rede *Transformer* bidirecional para pré-treinar um modelo de linguagem em um grande corpus e ajustar o modelo pré-treinado para outras tarefas. De acordo com (GAO et al., 2019), o modelo BERT para tarefas específicas é capaz de representar uma única sentença ou um par de sentenças como uma matriz consecutiva de *tokens*. Para determinado *token*, sua representação de entrada é estruturada somando seu *token*, segmento e incorporação de posição correspondentes. Para tarefa de classificação, a primeira palavra da sequência é identificada com um *token* único chamado [CLS], e uma camada totalmente conectada é conectada na posição [CLS] da última camada do codificador, por fim uma camada softmax completa a classificação da sentença ou par de sentenças.

O BERT possui duas configurações intensivas de parâmetros:

- **BERTbase:** O número de blocos *Transformer* é 12, o tamanho da camada oculta é 768, o número de cabeças de auto-atenção é 12 e o número total de parâmetros para o modelo pré-treinado é 110 milhões.
- **BERTlarge:** O número de blocos *Transformer* é 24, o tamanho da camada oculta é 1024, o número de cabeças de autoatenção é 16 e o número total de parâmetros para o modelo pré-treinado é 340 milhões.

O aprendizado de transferência, onde um modelo é treinado inicialmente em uma tarefa de origem e posteriormente ajustado em tarefas específicas, mudou o cenário de

aplicações de processamento de linguagem natural nos últimos anos. A estratégia de ajustar um grande modelo de linguagem pré-treinado foi vastamente adotada e alcançou desempenhos de ponta em diversas tarefas de PNL. Além de trazer melhorias no desempenho, o aprendizado de transferência reduz a quantidade de dados rotulados necessários para o aprendizado supervisionado em tarefas *downstream*. Porém o pré-treinamento desses grandes modelos de linguagem, requer enormes quantidades de dados não rotulados e recursos computacionais, há relatos de modelos sendo treinados utilizando milhares de GPUs ou TPUs e centenas de GBs de dados textuais brutos. Essa barreira de recursos limitou a disponibilidade desses modelos, desde o início, a modelos em inglês, chinês e multilíngue (SOUZA; NOGUEIRA; LOTUFO, 2020).

#### 2.5.5.1 BERTimbau

Grandes Modelos de linguagem pré-treinados podem ser recursos valiosos, principalmente para idiomas que possuem poucos recursos anotados, mas grande quantidade de dados não rotulados, como o português. Com isso em mente, (SOUZA; NOGUEIRA; LOTUFO, 2020) propôs o modelo BERT para o português brasileiro que foi apelidado como BERTimbau. Este modelo foi pré-treinado utilizando dados do brWac (FILHO et al., 2018), um corpus grande e diversificado que contém 2,68 bilhões de *tokens* de 3,53 milhões de documentos. Além do tamanho, o brWac é composto por documentos inteiro e sua metologia garante alta diversidade de domínio e qualidade de conteúdo, características que são importantíssimas para o pré-treinamento do BERT. O modelo BERTimbau foi treinado em dois tamanhos: Base(12 camadas, 768 dimensão oculta, 12 cabeças de atenção e 110 milhões de parâmetros) e Largo(24 camadas, 1024 dimensões ocultas, 16 cabeças de atenção e 330M parâmetros). O comprimento máximo da sentença foi definido como  $S = 512$  *tokens*.

(SOUZA; NOGUEIRA; LOTUFO, 2020) Avaliou o modelo em três tarefas de PNL: similaridade textual de sentença, reconhecimento de vinculação textual e reconhecimento de entidade nomeada e afirmou que o BERTimbau aprimora o estado da arte nessas tarefas em relação aos modelos multilíngues e abordagens monolíngues anteriores, confirmando a eficácia de grandes modelos de linguagem pré-treinados para o português. O modelo foi disponibilizado para a comunidade em bibliotecas de código aberto para fornecer bases sólidas para pesquisas futuras em PNL.

## 3 Metodologia

### 3.1 Dataset

É utilizado no presente trabalho um conjunto de dados do IMDB(*Internet Movie Database*), que possui 50.000(cinquenta mil) resenhas de filmes, em português, sendo 25.000 avaliações positivas e negativas cada.

O IMDB é o site mais usado para obter informações sobre filmes em todo o mundo. Devido à sua popularidade e devido à presença de um grande número de críticas relacionadas a filmes. É um dos conjuntos de dados de referência usados para análise de sentimentos em resenhas de filmes (KUMAR; HARISH; DARSHAN, 2019). Segundo (MESNIL et al., 2014) é um dos maiores conjuntos de dados de análise de sentimento disponíveis publicamente.

(ALVES et al., 2014) No idioma inglês existem diversos dados disponíveis já rotulados referentes a comentários de filmes, produtos e hotéis, que podem ser utilizados para treinamento e testes nestes domínios específicos. Contudo, outros idiomas e domínios carecem de dados rotulados para treinamento dos modelos de classificação.

O conjunto de dados original está no idioma inglês, mas foi traduzido de forma automática por (Gonçalves, Luís, 2018), através de ferramenta computacional e disponibilizado publicamente. É importante ressaltar que a tradução feita de forma automática pode não carregar todas as características linguísticas contidas na frase original, visto que cada idioma possui suas particularidades morfológicas, sintáticas e semânticas.

O *dataset* consiste num arquivo csv, que após ser convertido em um objeto *Data-Frame* da biblioteca Pandas ficou com 49459 registros. Abaixo está retratada a distribuição de registros de acordo com cada classe de sentimento, onde 0 significa Negativo e 1 Positivo. É possível notar que há um equilíbrio na distribuição, sendo 24694 registros com sentimento positivo e 24765 com sentimento negativo:

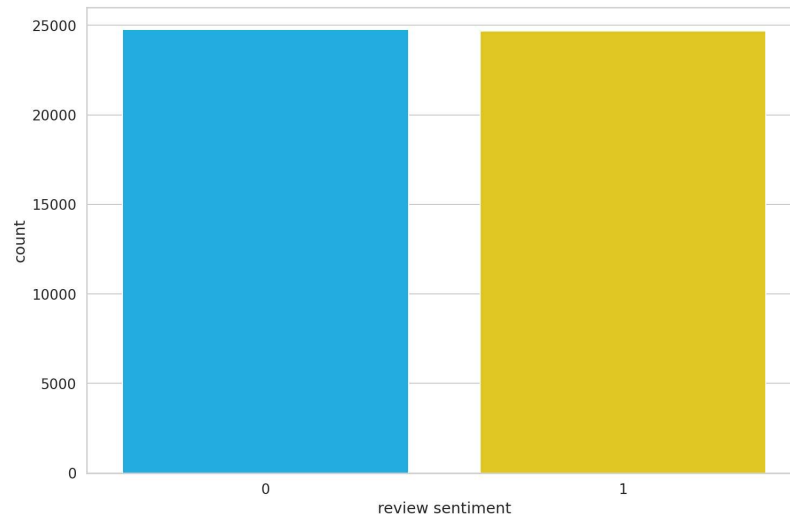


Figura 3 – Distribuição de classes

Fonte:(Autor)

O *dataset* passa por um processo de tokenização, de tal forma que as sentenças são convertidas em representações numéricas, posteriormente é gerado um gráfico para poder ser visualizada a distribuição das sentenças de acordo com a quantidade de palavras, onde nota-se que as sentenças de maior tamanho possuem 500 tokens:

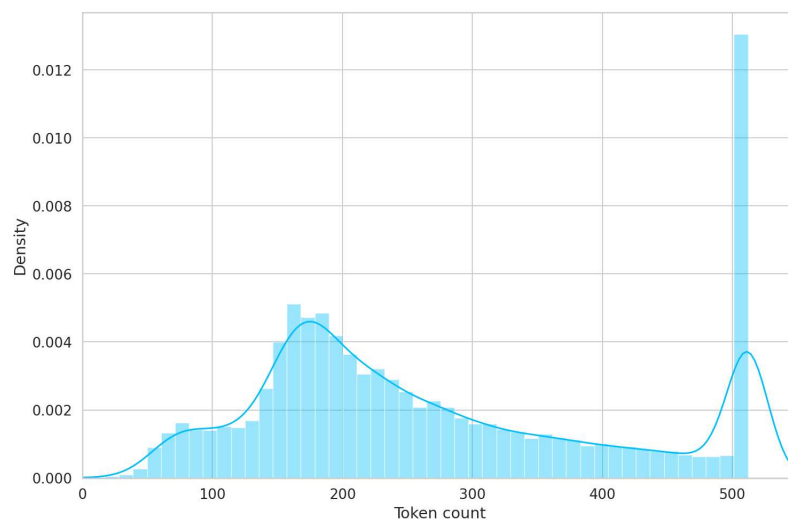


Figura 4 – Distribuição de sentenças tokenizadas

Fonte:(Autor)

### 3.2 *Dataset* de validação

O twitter é uma plataforma de mídia social que possui mais de 211 milhões de usuários ativos diários monetizáveis, postando cerca de 500 milhões de *tweets* todos os dias (Ahlgren, Matt, 2022). Originado em 2006, o twitter surgiu com a ideia de compartilhar



textos curtos, inicialmente eram limitados a 140 caracteres, posteriormente aumentou para 280, esses textos são chamados de *tweets*. Através desta plataforma os usuários podem compartilhar notícias, ideias, opiniões e comentar a respeito de diversos assuntos de forma fácil e rápida e para isso geralmente utilizam linguagem informal ou coloquial, onde pode conter gírias, siglas e ironia, ou seja, não há uma preocupação com as normas gramaticais.

Então é realizado um processo de extração de comentários da plataforma Twitter e posteriormente são transformados em um *dataset* para testar o desempenho dos modelos na tarefa de predição da polaridade de sentimentos de sentenças escritas em linguagem informal. Foram extraídos 40 comentários (20 de sentimento positivo e 20 de sentimento negativo) sobre os temas entretenimento, futebol e diversão, por serem temas que geralmente possuem mais opiniões expressas com linguagem coloquial. Estes comentários são de tamanhos diversos, têm comentários com frases longas e também com frases curtas. Após serem extraídos, os comentários foram salvos numa planilha CSV, e categorizados de acordo com seu respectivo sentimento, tal processo foi feito de forma manual. Abaixo está apresentado a distribuição dos comentários de acordo com sua classe de sentimento, sendo 0 para sentimento Negativo e 1 para Positivo:

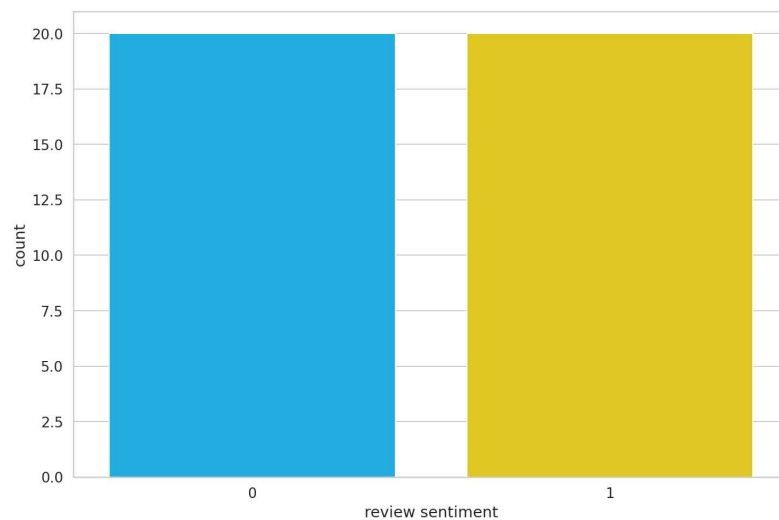


Figura 5 – Distribuição de classes (Twitter)

**Fonte:**(Autor)

### 3.3 Tecnologias utilizadas

- *Python*<sup>1</sup>: É a linguagem de programação mais popular para uso em inteligência artificial e *machine learning* (LUASHCHUK, Andrew., 2019), e uma das principais razões para isso é a extensa gama de bibliotecas (módulos com funções em código

---

<sup>1</sup> <https://www.python.org>

pré-escrito que permitem que o usuário obtenha determinadas funcionalidades) disponíveis para ela;

- *Docker*<sup>2</sup>: É uma plataforma de código aberto que utiliza virtualização de nível de sistema operacional para entregar software em pacotes intitulados como contêineres. Os contêineres ficam isolados uns dos outros e gerenciam seus próprios softwares, arquivos de configuração e bibliotecas;
- *FastAPI*<sup>3</sup>: É um *framework* para desenvolvimento de APIs com *Python*, é bastante robusto, intuitivo e de alto desempenho;
- *Heroku*<sup>4</sup>: É uma plataforma de nuvem como serviço baseada em contêiner focada em alta produtividade, que oferece suporte a várias linguagens de programação. É utilizada para implantar, gerenciar e dimensionar aplicações de diversos segmentos de forma flexível e fácil;
- VSCoDe: *Visual Studio Code*<sup>5</sup> é um editor de código-fonte desenvolvido pela *Microsoft*, que oferece suporte a diversas linguagens de programação, ferramentas de versionamento, complementação inteligente de código;
- *Colaboratory PRO*<sup>6</sup>: Também conhecida como *Colab* uma plataforma do *Google* especialmente adequada para aprendizado de máquina, análise de dados e educação, onde permite que os usuários desenvolvam e executem código *Python* arbitrário pelo direto no navegador. O *Colab* se baseia no projeto *Jupyter*, mas diferente do *Jupyter*, o *Colab* permite o uso de notebooks sem a necessidade de instalação ou download de arquivos. O *Colab* também oferece recursos de computação como GPUs, onde se faz necessário a utilização em diversas tarefas de aprendizado de máquina e análise de dados. Esta ferramenta oferece alguns planos, grátis e PRO (Pago), o plano PRO oferecem mais recursos computacionais e no presente trabalho foi utilizado este plano, pois o grátis não oferecia recursos suficientes para realizar os experimentos;
- *NumPy*<sup>7</sup>: Consiste numa biblioteca numérica em *Python*. Segundo (MALIK, Farhad., 2019), é uma das mais otimizadas para cálculos que envolvem *arrays* multidimensionais;
- *Pandas*<sup>8</sup>: É uma biblioteca da a linguagem *Python*, que fornece estruturas de dados de alto nível e funções projetadas para tornar o trabalho com dados estruturados ou tabulares veloz;

---

<sup>2</sup> <https://www.docker.com>

<sup>3</sup> <https://fastapi.tiangolo.com>

<sup>4</sup> <https://www.heroku.com>

<sup>5</sup> <https://code.visualstudio.com>

<sup>6</sup> <https://colab.research.google.com>

<sup>7</sup> <https://numpy.org>

<sup>8</sup> <https://pandas.pydata.org>

- *PyTorch*<sup>9</sup>: É baseado em *Torch*, um *framework* implementado em C e dedicado a computação científica. Segundo (JOHNS, Ray., 2020) e (O'CONNOR, Ryan., 2021), existem algumas vantagens em utilizá-lo em detrimento de outros, como: Melhor uso de memória e otimização, maior compatibilidade com *NumPy*, mais controle sobre as estruturas de modelos etc;
- *Keras*<sup>10</sup>: Consiste em uma biblioteca de *Deep Learning* que é implementada com o *TensorFlow* para várias linguagens e plataformas. Através dela é possível modelar e treinar modelos de redes neurais em poucas linhas de código;
- *Sklearn*<sup>11</sup>: É uma biblioteca da linguagem *Python* desenvolvida para aplicação prática de *machine learning*. Disponibiliza ferramentas eficientes e simples para análise preditiva de dados;
- *Seaborn*<sup>12</sup>: É uma biblioteca da linguagem *Python*, baseada em *Matplotlib*, que serve para a visualização de dados. Ele fornece uma interface de alto nível para desenhar gráficos estatísticos informativos e atraentes;
- *Matplotlib*<sup>13</sup>: É uma biblioteca abrangente para criar visualizações estáticas, e interativas em Python;
- *NLTK*<sup>14</sup>: É uma biblioteca da linguagem *Python* que oferece recursos utilizados para processamento de texto, classificação, marcação e tokenização;
- *Transformers*<sup>15</sup>: É uma biblioteca da linguagem *Python* que fornece modelos pré-treinados para baixar e treinar de forma fácil.

## 3.4 Implementação dos modelos de *Machine Learning*

### 3.4.1 LSTM e LSTM Bidirecional

A implementação dos modelos se dá pela linguagem de programação Python. São utilizadas as bibliotecas *Numpy*, *Pandas*, *Keras*, *Sklearn*, *Seaborn*, *NLTK*, *Matplotlib*. É definida uma camada *Embedding* onde as sentenças do *dataset* são convertidas em vetores numéricos, os valores contidos nesses vetores representam características sintáticas e semânticas das palavras. A entrada dessa camada consiste nos dados após terem passado pela etapa de Tokenização e pré-processamento. A saída dessa camada é dada como

---

<sup>9</sup> <https://pytorch.org>

<sup>10</sup> <https://keras.io>

<sup>11</sup> <https://scikit-learn.org>

<sup>12</sup> <https://seaborn.pydata.org>

<sup>13</sup> <https://matplotlib.org>

<sup>14</sup> <https://www.nltk.org>

<sup>15</sup> <https://huggingface.co/docs/transformers/main/en/index>

entrada na camada do modelo LSTM. A saída do modelo LSTM é dada como entrada numa camada densamente conectada que possui uma função de ativação *softmax*, essa camada recebe como entrada um vetor de valores reais e o converte em outro vetor onde os valores estarão no intervalo entre 0 e 1, esses valores convertidos são uma distribuição de probabilidades. O modelo foi compilado com uma função de perda *binary\_crossentropy* e otimizador Adam, a função de perda calcula e avalia quão bem o algoritmo está ao predizer os dados fornecidos. O otimizador é utilizado para fazer ajustes em alguns parâmetros como pesos e taxas de aprendizado durante o treinamento para melhorar o desempenho do modelo.

O LSTM Bidirecional é implementado da mesma maneira, a única diferença é que para implementar o LSTM Bidirecional, um modelo LSTM é passado como parâmetro para a camada do modelo LSTM Bidirecional.

#### 3.4.1.1 Hiperparâmetros

Abaixo estão detalhados os hiperparâmetros utilizados para a implementação dos modelos LSTM e LSTM Bidirecional:

- **epochs:** Quantidade de iterações em que o modelo foi exposto ao conjunto de dados de treinamento;
- **batch\_size:** Número de registros utilizados a cada atualização do gradiente;
- **max\_features:** Número máximo de palavras que serão armazenadas no vocabulário;
- **embed\_dim:** Dimensão de saída da camada *embedding*;
- **max\_sequence\_length:** Tamanho máximo das sentenças utilizadas durante o treinamento;
- **dropout:** Valor definido para a técnica *dropout*.

#### 3.4.2 BERT (BERTimbau)

A implementação do modelo se dá pela linguagem de programação *Python*. São utilizadas as bibliotecas *Numpy*, *Pandas*, *Keras*, *Sklearn*, *Seaborn*, *Matplotlib*, *PyTorch*, *Transformers*. Foi carregado um *tokenizer* com o modelo BERT pré-treinado (BERTimbau) através da biblioteca *Transformers*, o *tokenizer* será responsável por converter as palavras das sentenças em *tokens*. O modelo foi definido com uma camada BERT que recebe como entrada as sentenças tokenizadas, conta ainda com uma camada de *dropout* que foi importada da biblioteca *PyTorch*, essa camada serve para fazer regularização com o objetivo de prevenir sobre-ajustamento, e também uma camada totalmente conectada

para a saída. O otimizador utilizado foi o *AdamW*, que tem como função realizar ajustes em alguns parâmetros durante o treinamento para melhorar o desempenho do modelo, a função de perda *CrossEntropy* que tem como objetivo calcular e avaliar o quão bom o modelo está na tarefa de prever os exemplos fornecidos e a função de ativação *softmax* para obter as probabilidades previstas pelo modelo.

#### 3.4.2.1 Hiperparâmetros

Abaixo estão detalhados os hiperparâmetros utilizados para a implementação do modelo BERT ((BERTimbau):

- **EPOCHS:** Quantidade de iterações em que o modelo foi exposto ao conjunto de dados de treinamento;
- **BATCH\_SIZE:** Número de registros utilizados a cada atualização do gradiente;
- **MAX\_LEN :** Tamanho máximo das sentenças utilizadas durante o treinamento;
- **lr:** lr é uma abreviação de *learning rate*, em português significa taxa de aprendizado;
- **dropout:** Valor definido para a técnica dropout.

### 3.5 Descrição dos experimentos

#### 3.5.1 Experimento 1

##### 3.5.1.1 Treinamento de modelos LSTM e LSTM Bidirecional, com etapa de limpeza dos dados

Neste 1º experimento, o *dataset* utilizado para treinamento e teste dos modelos passa por uma etapa de pré-processamento, onde há uma limpeza nos dados, nesta etapa todas as palavras das sentenças foram transformadas em minúsculas, são removidos alguns caracteres especiais como "\$/:%&~" através de uma expressão regular. Letras que contém acentuações não são removidas. Nesta etapa também são removidas as *stopwords*, por meio da biblioteca NLTK. Após isto, ocorre o processo de tokenização das sentenças, através do método *Tokenizer* da biblioteca *Keras*, onde as sentenças são convertidas em representações numéricas e salvas num vocabulário, é definido um tamanho máximo de 5000 palavras para serem salvas no vocabulário. Por fim o *dataset* é dividido em duas partes, 80% para treino e 20% para teste.

Conforme mostrado na Figura 4, existem muitas sentenças que possuem tamanho de 500 *tokens*, por isso foi o hiperparâmetro `max_sequence_length` abaixo é definido com o valor 500, os demais hiperparâmetros são definidos com valores padrão:

Parâmetro	Valor
epochs	5
batch_size	32
max_features	5000
embed_dim	128
max_sequence_length	500
dropout	0.2

Tabela 1 – Parâmetros do experimento 1

Posteriormente, iniciou-se a etapa de treinamento dos dois modelos, na qual o modelo LSTM tem duração de aproximadamente 2 horas e 40 minutos e o LSTM Bidirecional de 3 horas e 15 minutos. Ao final de cada epoch os modelo são testados em alguns dados de validação.

### 3.5.2 Experimento 2

#### 3.5.2.1 Treinamento BERT sem etapa de limpeza dos dados

O dataset utilizado para treinamento e teste dos modelos passa por uma etapa de pré-processamento, onde há a tokenização das sentenças, posteriormente é dividido em duas partes, 80% para treino e 20% para teste. Conforme exposto na Figura 4, existem muitas sentenças que possuem tamanho de 500 tokens, por isso foi o hiperparâmetro MAX\_LEN abaixo é definido com o valor 500, é definido BATCH\_SIZE com valor 8 por limitação de recursos computacionais, mas normalmente é definido com valor 16 ou 32, os demais foram definidos valores padrão:

Parâmetro	Valor
EPOCHS	4
BATCH_SIZE	8
MAX_LEN	500
lr	2e-5
dropout	0.3

Tabela 2 – Parâmetros do experimento 2

Então, iniciou-se a etapa de treinamento, que tem duração de aproximadamente 6 horas e 30 minutos. Ao final de cada epoch o modelo foi testado em alguns dados de validação.

### 3.5.3 Experimento 3

#### 3.5.3.1 Treinamento de modelos LSTM e LSTM Bidirecional, sem etapa de limpeza dos dados

No presente experimento, o *dataset* passa por uma etapa de pré-processamento, mas desta vez não ocorre a etapa de limpeza dos dados, ocorrem apenas as etapas de tokenização, posteriormente a de divisão do *dataset* em treino e teste em 80% e 20% respectivamente.

É alterado o valor do `batch_size` para verificar se o desempenho dos modelos sofre algum impacto positivo, os demais hiperparâmetros são definidos com valores padrão:

Parâmetro	Valor
epochs	5
batch_size	16
max_features	5000
embed_dim	128
max_sequence_length	500
dropout	0.3

Tabela 3 – Parâmetros do experimento 3

Posteriormente, iniciou-se a etapa de treinamento dos modelos, onde o LSTM tem duração de aproximadamente 3 horas e 20 minutos e o LSTM Bidirecional de 4 horas e 10 minutos. Ao final de cada epoch os modelos são testados em alguns dados de validação.

### 3.5.4 Experimento 4

#### 3.5.4.1 Comparativo dos modelos em predição do *dataset* de teste

Após os modelos serem treinados, são submetidos à tarefa de predição no *dataset* de teste, que consiste em 20% (9892 registros) do *dataset* original, para comparar o desempenho dos modelos predizendo dados diferentes dos presentes no *dataset* de treinamento.

### 3.5.5 Experimento 5

#### 3.5.5.1 Comparativo dos modelos em predição do *dataset* de validação, que contém comentários do twitter

Após os modelos serem treinados, e testados no *dataset* de teste são submetidos à tarefa de predição no *dataset* de validação, para verificar o desempenho dos mesmos predizendo comentários em português informal.

## 4 Resultados

### 4.1 Métricas de avaliação

Abaixo estão explicadas as métricas utilizadas para avaliar o desempenho dos modelos:

- **Acurácia:** representa a performance geral do modelo. Dentre todas as classificações realizadas, quantas o modelo classificou corretamente;
- **Precisão:** dentre todas as classificações de classe X que o modelo fez, quantas estão corretas;
- **Recall:** dentre todas as situações de classe X como valor esperado, quantas estão corretas;
- **F1-score:** consiste numa média harmônica entre precisão e *recall*.

### 4.2 Resultados do experimento 1

#### 4.2.1 LSTM e LSTM Bidirecional treinados com etapa de limpeza dos dados

Ao final do treinamento são gerados os seguintes gráficos que mostram o comportamento da função de perda e acurácia durante a etapa de treinamento e validação.

#### LSTM

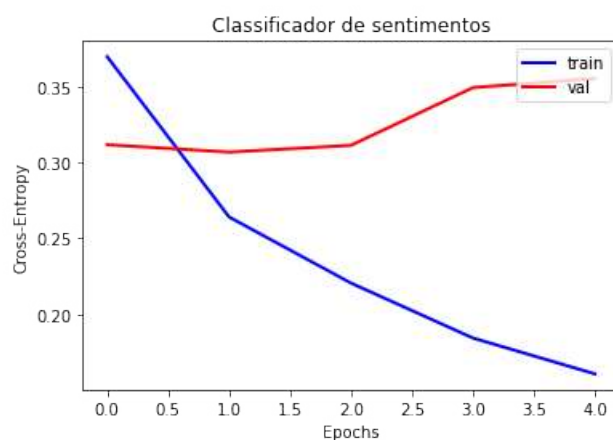


Figura 6 – LSTM Função de perda

Fonte:(Autor)



Conforme representado no gráfico acima, a função de perda nos dados de treinamento inicia em torno de 0.35 e ao longo das 5 épocas vai se aproximando de 0, já nos dados de validação inicia em torno de 0.30 e ao longo das 5 épocas eleva um pouco e se manteve em 0.35, indicando que o modelo teve mais dificuldade de prever o sentimento de dados que não estavam no *dataset* de treino.

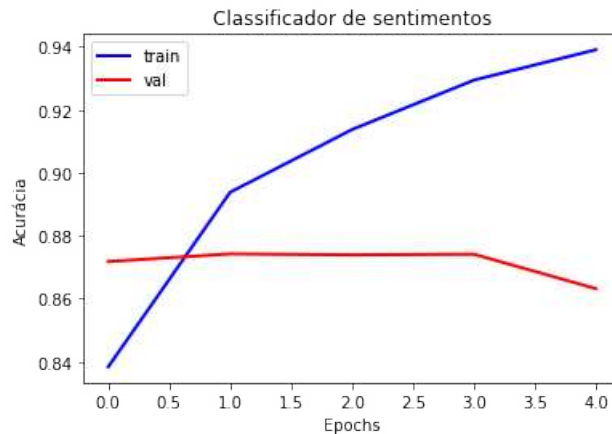


Figura 7 – LSTM Acurácia

Fonte:(Autor)

A acurácia do modelo, nos dados de treino inicia em torno de 0.84, ao final da 1ª época estava em 0.90, se manteve em constante movimento de subida, e ao final das 5 épocas ficou em torno de 0.94, já nos dados de validação atingiu aproximadamente 0.88, então se manteve constante, após isso houve um pequeno declínio e ao final da 5ª época encerrou em aproximadamente 0.86, mostrando que o modelo sofreu uma pequena perda no desempenho ao prever dados não contidos no *dataset* de treino.

### LSTM Bidirecional

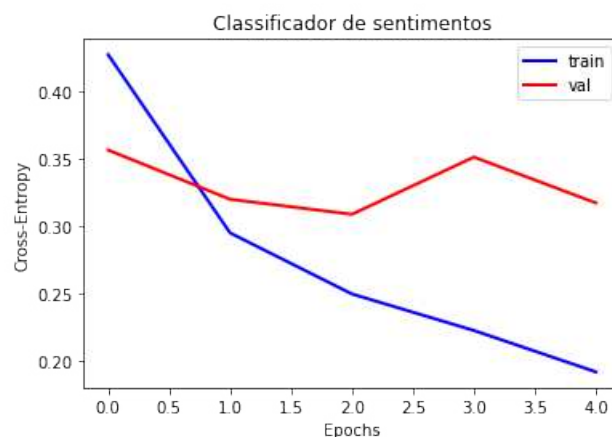


Figura 8 – LSTM Bidirecional Função de perda

Fonte:(Autor)

De acordo com o gráfico acima, pode-se observar que a função de perda nos dados

de treino, ao final da 1ª época, é superior a 0.40 e no decorrer das seguintes épocas mantém um movimento de descida até chegar em aproximadamente 0.20 ao final da 5ª época, e nos dados de validação, ao finalizar a 1ª época, a função de perda tem um valor de aproximadamente 0.35, posteriormente acontece um movimento de descida seguido de uma elevação, e depois finaliza em torno de 0.30, também sinalizando que o modelo tem maior dificuldade de predição nos dados de validação.

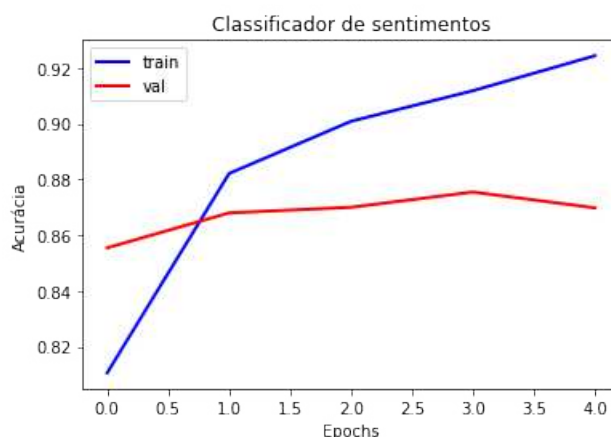


Figura 9 – LSTM Bidirecional Acurácia

Fonte:(Autor)

O modelo LSTM atingiu uma acurácia de 93.91% nos dados de treino e 86.32% nos dados de validação. Já o modelo LSTM Bidirecional atingiu uma acurácia de 92.44% nos dados de treino e 86.98% nos dados de validação. Obtém nos dados de treino uma acurácia inferior à alcançada pelo modelo LSTM, mas nos dados de validação é ligeiramente superior.

## 4.3 Resultados do experimento 2

### 4.3.1 BERT treinado sem etapa de limpeza dos dados

Ao final do experimento 2 o modelo atingiu uma acurácia de 96.28% e perda de 4% nos dados de treino e nos dados de validação acurácia de 94.27% e perda de 36.01%. Ao comparar estes resultados com os resultados do experimento 1 percebe-se que o BERT teve um desempenho superior.

## 4.4 Resultados do experimento 3

### 4.4.1 LSTM e LSTM Bidirecional treinados sem etapa de limpeza dos dados

Ao final do experimento 3 são gerados os seguintes gráficos que mostram o comportamento da função de perda e acurácia durante a etapa de treinamento e validação.

#### 4.4.1.1 LSTM

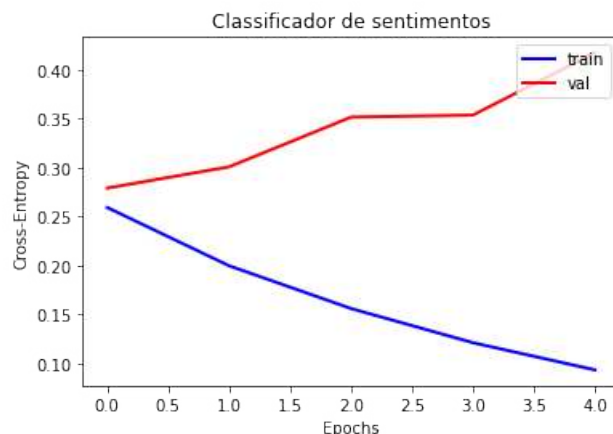


Figura 10 – LSTM Função de perda

**Fonte:**(Autor)

Durante a etapa de treinamento o modelo obtém uma função de perda um resultado de aproximadamente 0.30 ao final da 1ª época, nos dados de validação, nas épocas posteriores houve um movimento de elevação, finalizando em torno de 0.40, e nos dados de treino teve um valor de aproximadamente 0.25 ao final da 1ª época, subsequentemente teve um movimento constante de declínio, ficando em aproximadamente 0.10 ao final da 5ª época.

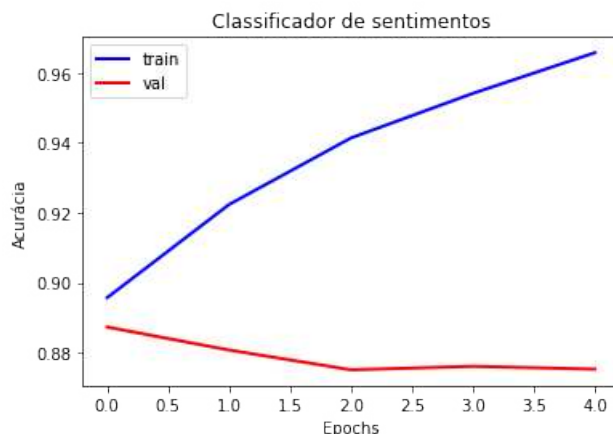


Figura 11 – LSTM Acurácia

**Fonte:**(Autor)

Conforme representa o gráfico acima, é possível observar que ao concluir a 1ª época, o modelo tem uma acurácia de 0.90, nos dados de treino, o valor da acurácia vai aumentando no decorrer das épocas, e ao final da 5ª o valor é de aproximadamente 0.96, já nos dados de validação, ao finalizar a 1ª época a acurácia é de aproximadamente 0.88, não há uma oscilação muito grande, e ao final das 5 épocas fica em torno de 0.88.

#### 4.4.1.2 LSTM Bidirecional

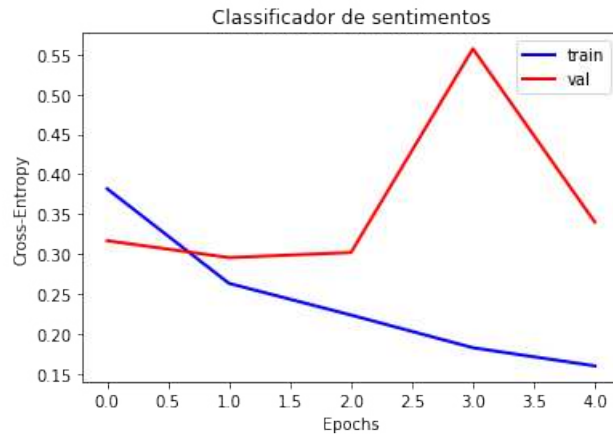


Figura 12 – LSTM Bidirecional Função de perda

**Fonte:**(Autor)

De acordo com o gráfico acima apresentado, ao finalizar a 1ª época, a função de perda do modelo é de aproximadamente 0.40 nos dados de treino, e ao decorrer das épocas vai declinando, até atingir algo em torno de 0.15 ao final da 5ª época, e nos dados de validação, o a função de perda obtém um resultado de aproximadamente 0.30, onde se manteve nessa faixa até o final da 3ª época, após isso ocorre um movimento de elevação seguido de um movimento de declínio, finalizando em torno de 0.35 ao concluir a 5ª época.

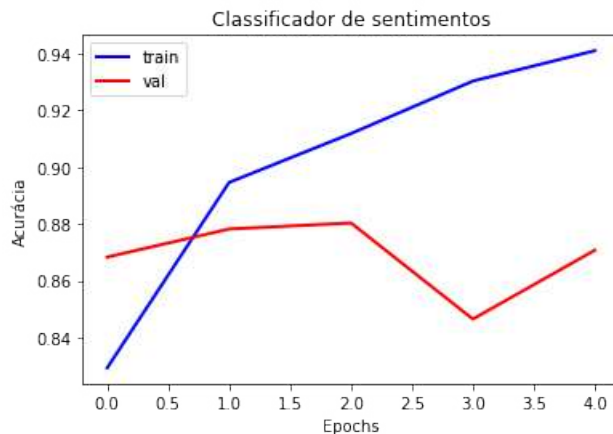


Figura 13 – LSTM Bidirecional Acurácia

**Fonte:**(Autor)

O modelo LSTM atingiu uma acurácia de 96.57% nos dados de treino e 87.53% nos dados de validação. Já o modelo LSTM Bidirecional atingiu uma acurácia de 94.10% nos dados de treino e 87.07% nos dados de validação.

## 4.5 Resultados do experimento 4

### 4.5.1 Modelos Treinados com etapa de limpeza dos dados

#### 4.5.1.1 LSTM

Após a realização do experimento 4 o modelo obtém os seguintes resultados:

	<b>precisão</b>	<b>recall</b>	<b>f1-score</b>	<b>support</b>
<b>Negativo</b>	0.85	0.90	0.87	5038
<b>Positivo</b>	0.88	0.83	0.86	4854

Tabela 4 – LSTM

Conforme apresentado na tabela acima, é possível observar que o modelo obtém uma precisão superior na predição de valores com sentimento positivo, já o *recall* é superior na predição de valores com sentimento negativo, e o f1-score que é uma média harmônica calculada com base na precisão e *recall*, o modelo obtém valores bem próximos nas duas classes Positivo e Negativo. A coluna *support* representa a quantidade de registros.

Então é gerada a seguinte matriz de confusão:

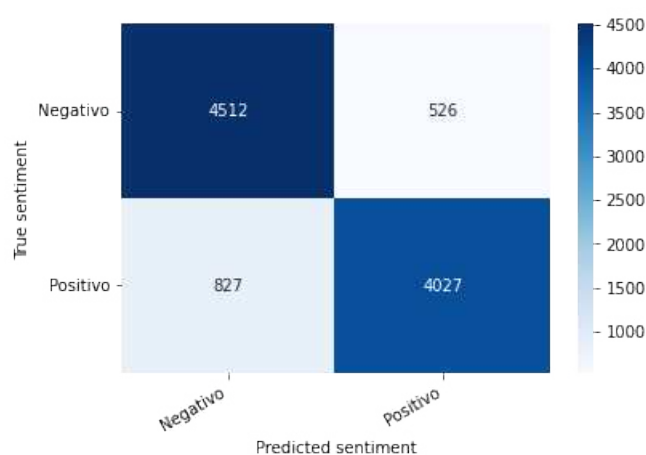


Figura 14 – LSTM Matriz de confusão

Fonte:(Autor)

#### 4.5.1.2 LSTM Bidirecional

O modelo obtém os seguintes resultados após o experimento 4:

	<b>precisão</b>	<b>recall</b>	<b>f1-score</b>	<b>support</b>
<b>Negativo</b>	0.88	0.86	0.87	5038
<b>Positivo</b>	0.86	0.88	0.87	4854

Tabela 5 – LSTM Bidirecional

De acordo com os dados descritos acima, percebe-se que o modelo obtém precisão superior na predição do sentimento negativo, o recall é superior na classe Positivo, já o f1-score o modelo teve um valor de 0.87 em ambas as classes de sentimento. Percebe-se que ao comparar estes resultados com os do LSTM apresentado anteriormente, é possível notar que o LSTM teve uma precisão superior na classe Positivo e inferior na classe Negativo.

Após isto é gerada a matriz de confusão a seguir:

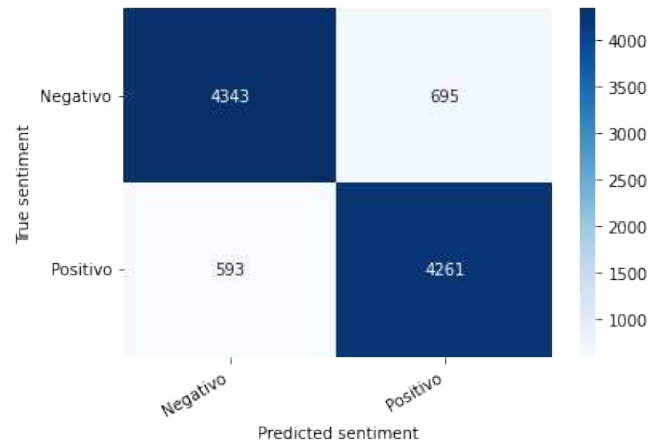


Figura 15 – LSTM Bidirecional Matriz de confusão

Fonte:(Autor)

## 4.5.2 Modelos Treinados sem etapa de limpeza dos dados

### 4.5.2.1 LSTM

O modelo obtém os seguintes resultados:

	precisão	recall	f1-score	support
<b>Negativo</b>	0.89	0.86	0.88	5038
<b>Positivo</b>	0.86	0.89	0.88	4854

Tabela 6 – LSTM

Conforme evidenciado acima, nota-se que o modelo obtém uma precisão de 0.89 na classe Negativo, já na classe Positivo é 0.03 pontos inferior, no *recall* é exatamente o inverso, já na métrica f1-score o modelo manteve uma média de 0.88 para ambas as classes.

A matriz de confusão abaixo representa as predições do modelo:

### 4.5.2.2 LSTM Bidirecional

A tabela abaixo retrata os resultados obtidos pelo modelo:

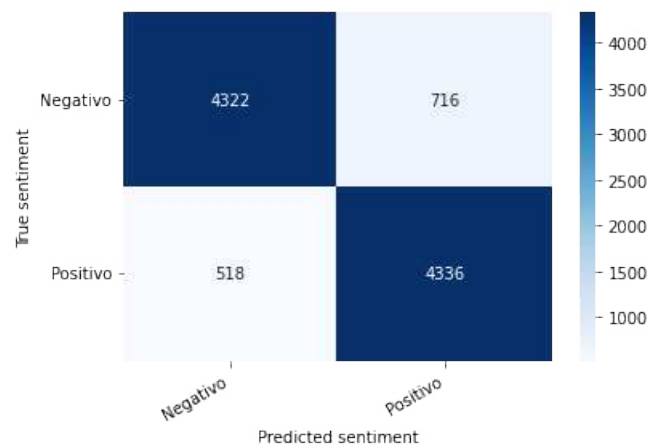


Figura 16 – LSTM Matriz de confusão

Fonte:(Autor)

	precisão	recall	f1-score	support
<b>Negativo</b>	0.90	0.84	0.87	5038
<b>Positivo</b>	0.84	0.91	0.87	4854

Tabela 7 – LSTM Bidirecional

Conforme apresentado acima, é possível notar que a precisão do modelo na classe Negativo é consideravelmente superior à classe Positivo, já o *recall* é superior na classe positivo, e o f1-score é o mesmo valor para ambas as classes.

Posteriormente é gerada a matriz de confusão que retrata as predições realizadas pelo modelo:

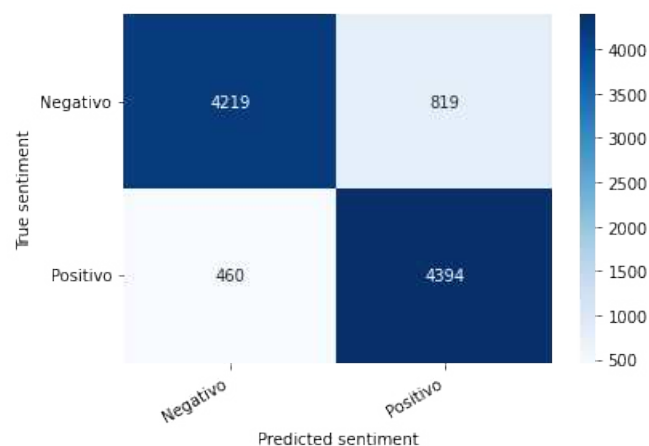


Figura 17 – LSTM Bidirecional Matriz de confusão

Fonte:(Autor)

### 4.5.2.3 BERT

O modelo obtém os seguintes resultados durante o experimento 4:

	<b>precisão</b>	<b>recall</b>	<b>f1-score</b>	<b>support</b>
<b>Negativo</b>	0.94	0.93	0.94	5038
<b>Positivo</b>	0.93	0.94	0.93	4854

Tabela 8 – BERT

Ao analisar os resultados apresentados acima, é possível observar que o modelo obtém valores mais equilibrados, para ambas as classes, a precisão é ligeiramente superior na classe Negativo.

Então é gerada a matriz de confusão durante o experimento 4:

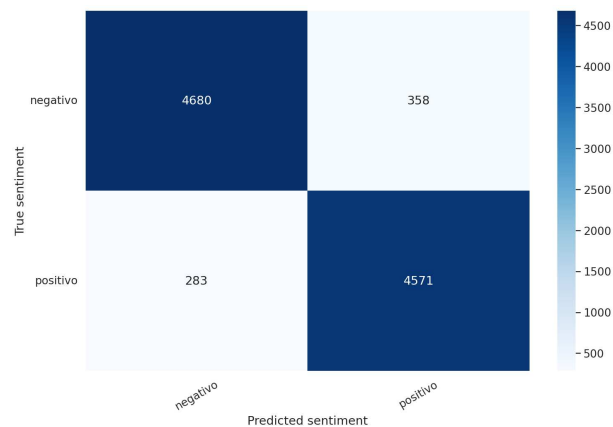


Figura 18 – BERT Matriz de confusão

**Fonte:**(Autor)

Observa-se que o modelo BERT obtém um desempenho superior aos demais modelos apresentados no presente experimento, predizendo as sentenças do *dataset* de teste com mais precisão. O BERT atinge resultados superiores em todas as métricas.

## 4.6 Resultados do experimento 5

### 4.6.1 Modelos Treinados com etapa de limpeza dos dados

#### 4.6.1.1 LSTM

O modelo obtém os seguintes resultados no experimento 5:

De acordo com os dados apresentados na tabela acima é possível observar que o modelo não obtém um desempenho satisfatório, onde atingiu uma precisão de 0.53 na classe Negativo e 0.52 na classe Positivo, indicando que conseguiu predizer de forma correta aproximadamente a metade dos registros.



	<b>precisão</b>	<i>recall</i>	<b>f1-score</b>	<i>support</i>
<b>Negativo</b>	0.53	0.40	0.46	20
<b>Positivo</b>	0.52	0.65	0.58	20

Tabela 9 – LSTM

A matriz de confusão abaixo representa as predições realizadas pelo modelo:

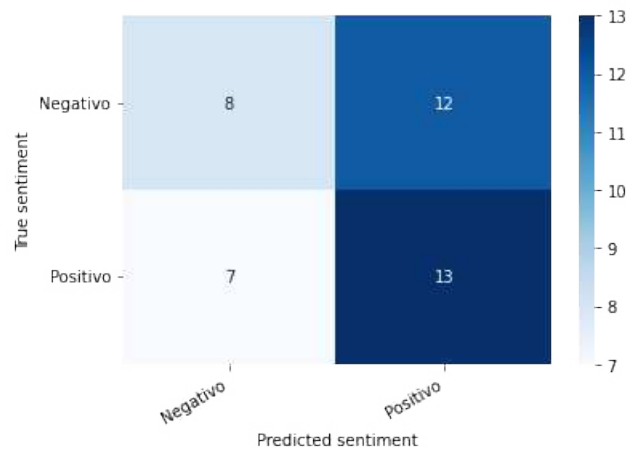


Figura 19 – LSTM Matriz de confusão

**Fonte:**(Autor)

#### 4.6.1.2 LSTM Bidirecional

O modelo obtém os seguintes resultados no experimento 5:

	<b>precisão</b>	<i>recall</i>	<b>f1-score</b>	<i>support</i>
<b>Negativo</b>	0.50	0.50	0.50	20
<b>Positivo</b>	0.50	0.50	0.50	20

Tabela 10 – LSTM Bidirecional

Conforme representado acima, nota-se que o modelo obtém o valor 0.50 em todas as métricas, indicando que conseguiu prever de forma correta apenas a metade dos registros contidos no *dataset*.

A matriz de confusão abaixo representa as predições realizadas pelo modelo:

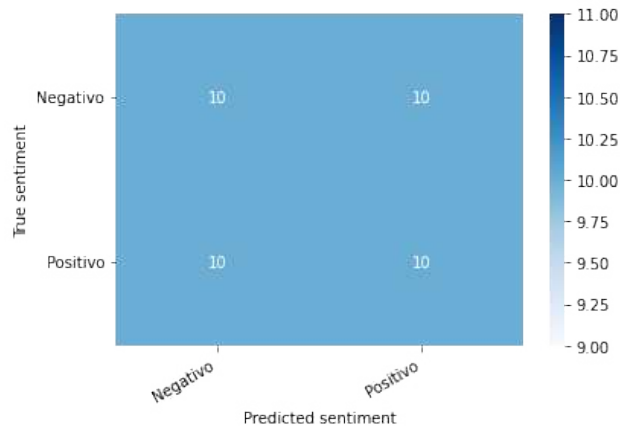


Figura 20 – LSTM Bidirecional Matriz de confusão

Fonte:(Autor)

## 4.6.2 Modelos Treinados sem etapa de limpeza dos dados

### 4.6.2.1 LSTM

O modelo obtém os seguintes resultados durante a realização do experimento 5:

	precisão	<i>recall</i>	f1-score	<i>support</i>
<b>Negativo</b>	0.43	0.30	0.35	20
<b>Positivo</b>	0.46	0.60	0.52	20

Tabela 11 – LSTM

Os dados representados acima mostram que o modelo obtém uma precisão superior na classe Positivo, o *recall* também é superior na classe Positivo, ele acertou mais predições positivas, mas também errou mais predições positivas. O f1-score também é superior na classe Positivo.

A matriz de confusão abaixo representa as predições realizadas pelo modelo:

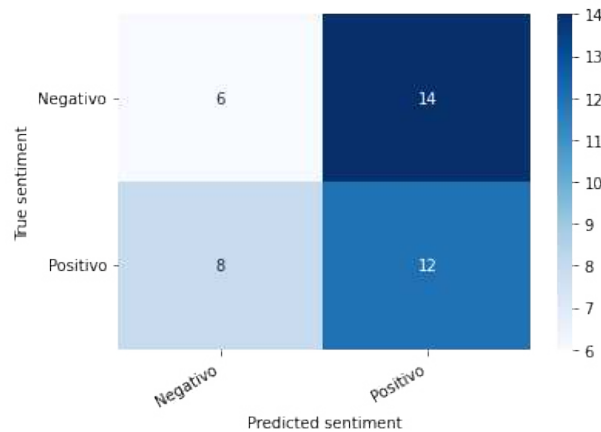


Figura 21 – LSTM Matriz de confusão

Fonte:(Autor)

#### 4.6.2.2 LSTM Bidirecional

Durante a realização do experimento 5 o modelo obtém os seguintes resultados:

	precisão	<i>recall</i>	f1-score	<i>support</i>
<b>Negativo</b>	0.53	0.50	0.51	20
<b>Positivo</b>	0.52	0.55	0.54	20

Tabela 12 – LSTM Bidirecional

Os dados acima mostram que o modelo obtém uma precisão maior na classe Negativo, o *recall* nesta classe é de 0.50, mostrando que o modelo conseguiu prever corretamente metade destes registros da classe Negativo. Ao observar a matriz de confusão abaixo, é possível notar que houveram mais previsões corretas na classe Positivo, porém a quantidade de falsos positivos é maior, por isso a precisão do modelo nesta classe é inferior.

A matriz de confusão abaixo representa as previsões realizadas pelo modelo:

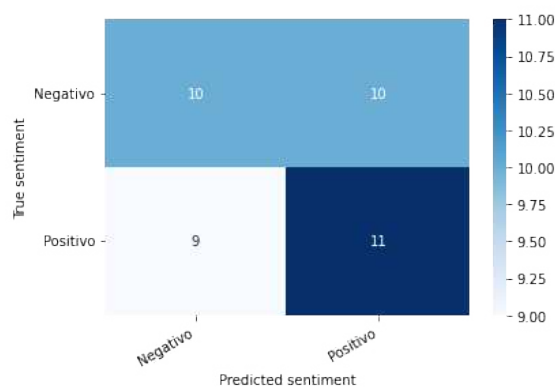


Figura 22 – LSTM Bidirecional Matriz de confusão

Fonte:(Autor)

Com base nos resultados acima apresentados nota-se que os modelos LSTM e LSTM Bidirecional sofrem de um problema chamado *overfitting*, pois o desempenho deles ao serem expostos à dados reais foram muito inferiores aos resultados obtidos com os dados de treino, constatou-se que só conseguiram prever de forma correta metade das sentenças, é praticamente um "chute" na predição.

#### 4.6.2.3 BERT

O modelo obtém os seguintes resultados durante o experimento 5:

	<b>precisão</b>	<b>recall</b>	<b>f1-score</b>	<b>support</b>
<b>Negativo</b>	0.95	1.00	0.98	20
<b>Positivo</b>	1.00	0.95	0.97	20

Tabela 13 – BERT

Conforme exibido acima, nota-se que o modelo obtém resultados superiores aos outros na realização deste experimento. O BERT conseguiu prever de forma correta quase todos os registros. Ao observar a matriz de confusão abaixo, percebe-se que todos os registros genuinamente negativos são preditos de forma correta, mas houve um falso negativo, por isso a precisão na classe Negativo é inferior a precisão na classe Positivo.

A matriz de confusão abaixo representa as predições realizadas pelo modelo:

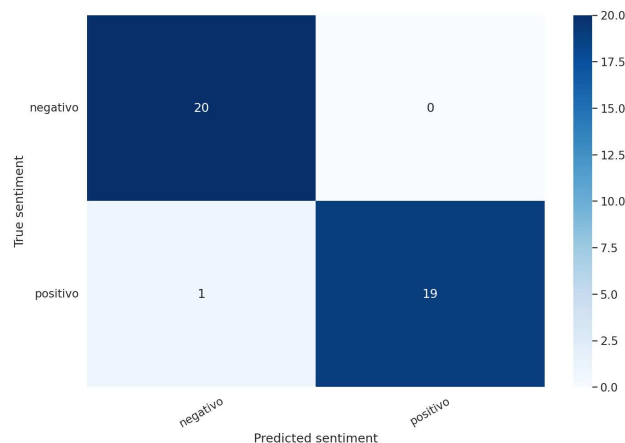


Figura 23 – BERT Matriz de confusão

**Fonte:**(Autor)

Após o modelo BERT atingir bons resultados, sendo treinado com dados brutos, ou seja, sem etapa de limpeza, os modelos LSTM e LSTM Bidirecional também são treinados com os dados brutos, para verificar se algum dos dois obtém resultados tão bons ou melhores que o BERT, mas após os experimentos verificou-se que ambos obtiveram resultados inferiores ao modelo BERT. Com base nos resultados apresentados evidenciou-se que o modelo BERT atingiu melhor desempenho do que os modelos LSTM e LSTM

Bidirecional, desde o treinamento até os experimentos de predição no *dataset* de teste e também no *dataset* de validação que contém os comentários retirados do twitter.

## 4.7 Implementação do Serviço de Análise de Sentimentos em Nuvem

Como relatado anteriormente, a análise de sentimentos pode ser aplicada em diversos contextos, por exemplo, para saber a opinião das pessoas a respeito de produtos, serviços, pode ser aplicada também a tomada de decisões, planejamentos, campanhas de marketing, entre outros. Visto isso, é desenvolvido um serviço de análise de sentimentos em nuvem, desacoplado e facilmente integrável em outras aplicações através de APIs. É desenvolvido utilizando as tecnologias *Python*, *FastAPI*, *Docker* e disponibilizado na plataforma *Heroku*.

O modelo BERT treinado no experimento 2 é o selecionado para compor este serviço, pois como é exposto acima ele obtém melhor desempenho nos experimentos. Após o modelo ser treinado é gerado um arquivo, pode-se dizer que este arquivo é a inteligência adquirida pelo modelo, arquivo este que é acoplado ao serviço. É utilizado o *framework* FastAPI para desenvolver a codificação do serviço, definir URLs da API e carregar o arquivo do modelo treinado. Após isto é desenvolvido um *container* para "empacotar" a aplicação, e posteriormente é feito *deploy* da mesma na plataforma *Heroku*, ao realizar o *deploy* da aplicação, o modelo treinado já é carregado e fica disponível para realizar a tarefa de predição da polaridade de sentimento.

Abaixo está ilustrada a arquitetura da aplicação desenvolvida:

A aplicação funciona da seguinte maneira: o cliente envia uma requisição HTTPS do tipo POST para a URL 'https://pt-sentiment-analyzer.herokuapp.com/predict', no corpo da requisição deve conter um atributo denominado como '*text*' contendo a frase que se deseja realizar a predição do sentimento, a aplicação ao receber a requisição, converte em representação numérica a frase recebida, então o modelo é exposto a essa frase para predizer o sentimento dela, e após realizada a predição o serviço retorna a requisição com a respectiva probabilidade do sentimento da frase.

Este serviço é desenvolvido com estas tecnologias citadas acima de modo que facilite sua escalabilidade e também a integração com outras aplicações. Se for preciso retreinar o modelo para substituir o que está rodando em produção não é necessário parar a aplicação, basta apenas treiná-lo em outra plataforma, como o *Google Colab*, e o arquivo gerado após o treinamento é inserido no lugar do arquivo que está no container da aplicação localmente, e posteriormente realizar o *deploy* do *container* através de alguns comandos, de forma fácil e rápida.

Atualmente o serviço é capaz de predizer o sentimento de uma sentença por

Arquitetura do serviço de análise de sentimentos em nuvem

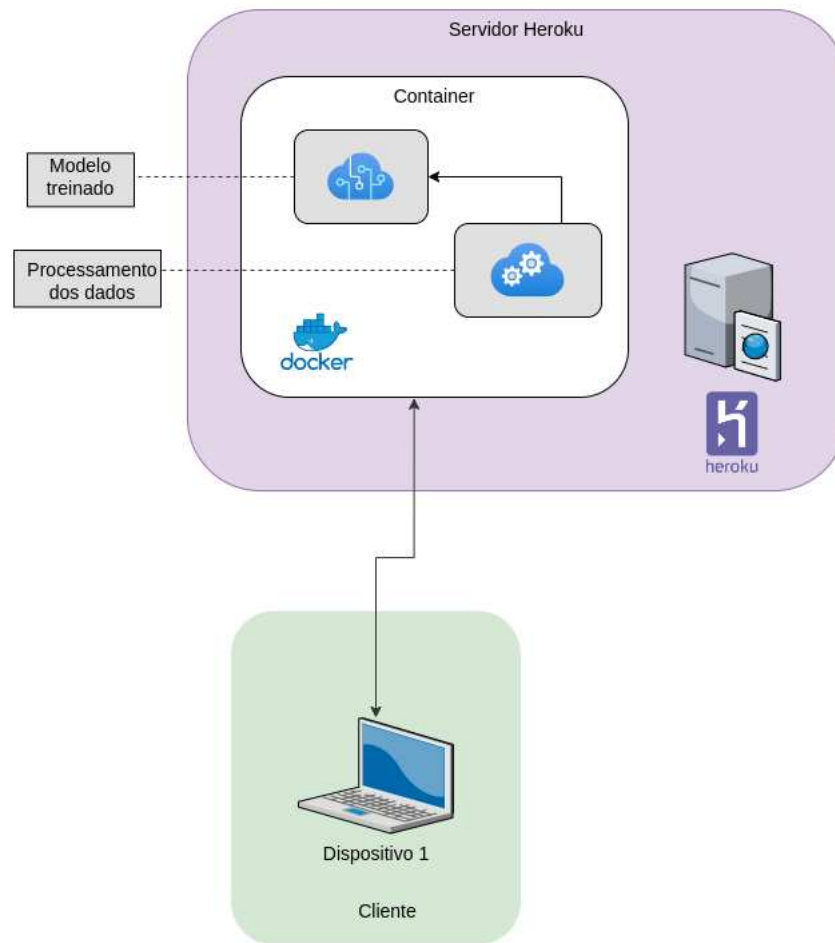


Figura 24 – Arquitetura do serviço

**Fonte:**(Autor)

requisição, mas futuramente será implementado outro recurso para prever várias sentenças por vez.

## 5 Conclusão

Neste trabalho foram realizados experimentos para comparar o desempenho de alguns modelos de redes neurais na tarefa de predição da polaridade de sentimentos focada em português coloquial. Após a realização dos experimentos foi desenvolvido um serviço de análise de sentimentos em nuvem.

Os modelos de redes neurais utilizados foram o *Long Short-Term Memory* - LSTM, LSTM Bidirecional e BERTimbau, este último é um *Bidirectional Encoder Representation from Transformers* - BERT. Estes modelos foram treinados com o *dataset* do IMDB, este conjunto de dados contém 50.000 resenhas de filmes rotuladas com o respectivo sentimento, sendo 25.000 avaliações positivas e negativas cada, e foi traduzido para o português por (Gonçalves, Luís, 2018).

Após treinados, os modelos foram submetidos a experimentos onde tiveram que prever o sentimento de sentenças contidas em um *dataset* de teste que consiste em 20% do *dataset* do IMDB, e um *dataset* de validação que contém sentenças retiradas do Twitter. Evidenciou-se que o modelo BERTimbau obteve resultados superiores aos dos outros modelos, então ele foi o modelo selecionado para compor o serviço de análise de sentimentos em nuvem focado em português coloquial. Este serviço foi desenvolvido utilizando tecnologias que facilitam a escalabilidade e integração com outras aplicações. Visto que análise de sentimentos pode ser aplicada em diversos contextos, por exemplo: tomada de decisões, planejamentos, campanhas de marketing, descobrir a opinião das pessoas a respeito de produtos, serviços etc. o projeto aqui desenvolvido poderá agregar valor para as mesmas, direcionadas ao português coloquial.

Trabalhos Futuros:

- Implementar outro recurso no serviço aqui desenvolvido, para prever várias sentenças por vez.

# Referências

- Ahlgren, Matt. *50+ Twitter Statistics Facts For 2022*. 2022. Disponível em: <<https://www.websiterating.com/research/twitter-statistics/>>. Acesso em: 20 de maio de 2022.
- ALVES, A. L. F. et al. Uma abordagem de análise de sentimentos espaço-temporal em microtextos. Universidade Federal de Campina Grande, 2014.
- APPELQUIST, D. et al. A standards-based. *Open and Privacy-aware Social Web*, 2010.
- BEHERA, R. K. et al. Co-lstm: Convolutional lstm model for sentiment analysis in social big data. *Information Processing & Management*, Elsevier, v. 58, n. 1, p. 102435, 2021.
- BIRD, S. et al. *Natural language processing with Python: analyzing text with the natural language toolkit*. [S.l.]: "O'Reilly Media, Inc.", 2009.
- BROWNLEE, J. Uma introdução delicada aos gradientes de explosão em redes neurais. *iCrowdNewswire*, 2017. Disponível em: <<https://icrowdnewswire.com/2017/12/18/uma-introducao-delicada-aos-gradientes-de-explosao-em-redes-neurais/>>.
- CAMBRIA, E. et al. Guest editorial: Big social data analysis. *Knowledge-Based Systems*, v. 69, p. 1–2, 2014. ISSN 0950-7051. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S0950705114002500>>.
- CARRIÇO, N.; QUARESMA, P. Sentence embeddings and sentence similarity for portuguese faqs. *Proc. IberSPEECH 2021*, p. 200–204, 2021.
- CHOWDHURY, G. G. Natural language processing. *Annual review of information science and technology*, Wiley Online Library, v. 37, n. 1, p. 51–89, 2003.
- DEVLIN, J. et al. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- DEVLIN, J. et al. BERT: Pre-training of deep bidirectional transformers for language understanding. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, 2019. p. 4171–4186. Disponível em: <<https://aclanthology.org/N19-1423>>.
- DIAS, D. C. *Text mining methods for mapping opinions from georeferenced documents*. Tese (Doutorado) — Master's thesis, Universidade Técnica de Lisboa, 2012.
- FILHO, J. A. W. et al. The brwac corpus: A new open resource for brazilian portuguese. In: *Proceedings of the eleventh international conference on language resources and evaluation (LREC 2018)*. [S.l.: s.n.], 2018.
- GAO, Z. et al. Target-dependent sentiment classification with bert. *IEEE Access*, v. 7, p. 154290–154299, 2019.



- GÉRON, A. *Mãos à Obra: Aprendizado de Máquina com Scikit-Learn & TensorFlow*. [S.l.]: Alta Books, 2019.
- GOLDBERG, Y. Neural network methods for natural language processing. *Synthesis lectures on human language technologies*, Morgan & Claypool Publishers, v. 10, n. 1, p. 1–309, 2017.
- Gonçalves, Luís. *Tradução do dataset IMdb para o português*. 2018. Disponível em: <https://www.kaggle.com/datasets/luisfredgs/imdb-ptbr>. Acesso em: 28 de setembro de 2021.
- HAMEED, Z.; GARCIA-ZAPIRAIN, B. Sentiment classification using a single-layered bilstm model. *Ieee Access*, IEEE, v. 8, p. 73992–74001, 2020.
- HOANG, M. et al. Aspect-based sentiment analysis using BERT. In: *Proceedings of the 22nd Nordic Conference on Computational Linguistics*. Turku, Finland: Linköping University Electronic Press, 2019. p. 187–196. Disponível em: <https://aclanthology.org/W19-6120>.
- HU, Z. et al. Handling vanishing gradient problem using artificial derivative. *IEEE Access*, v. 9, p. 22371–22377, 2021.
- HUSSAIN, A.; CAMBRIA, E. Semi-supervised learning for big social data analysis. *Neurocomputing*, v. 275, p. 1662–1673, 2018. ISSN 0925-2312. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0925231217316363>.
- JIANG, S. et al. Irony detection in the portuguese language using bert. *Proceedings http://ceur-ws.org ISSN*, v. 1613, p. 0073, 2021.
- JOHNS, Ray. *PyTorch vs TensorFlow for Your Python Deep Learning Project*. 2020. Disponível em: <https://realpython.com/pytorch-vs-tensorflow/#who-uses-pytorch>. Acesso em: 26 de maio de 2022.
- KUMAR, H. et al. Sentiment analysis on imdb movie reviews using hybrid feature extraction method. *International Journal of Interactive Multimedia & Artificial Intelligence*, v. 5, n. 5, 2019.
- LEITE, J. A. et al. *Toxic Language Detection in Social Media for Brazilian Portuguese: New Dataset and Multilingual Analysis*. arXiv, 2020. Disponível em: <https://arxiv.org/abs/2010.04543>.
- LIU, B. Sentiment analysis and opinion mining. In: . [S.l.: s.n.], 2012. v. 5. ISBN 978-3-642-19459-7.
- LIU, G.; GUO, J. Bidirectional lstm with attention mechanism and convolutional layer for text classification. *Neurocomputing*, v. 337, p. 325–338, 2019. ISSN 0925-2312. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0925231219301067>.
- LOPES, et al. Exploring bert for aspect extraction in portuguese language. *The International FLAIRS Conference Proceedings*, v. 34, Apr. 2021. Disponível em: <https://journals.flvc.org/FLAIRS/article/view/128357>.
- LUASHCHUK, Andrew. *Why I Think Python is Perfect for Machine Learning and Artificial Intelligence*. 2019. Disponível em: <https://towardsdatascience.com/8-reasons-why-python-is-good-for-artificial-intelligence-and-machine-learning-4a23f6bed2e6>. Acesso em: 26 de maio de 2022.

LUDERMIR, T. B. Inteligência artificial e aprendizado de máquina: estado atual e tendências. *Estudos Avançados*, SciELO Brasil, v. 35, p. 85–94, 2021.

MALIK, Farhad. *Why Should We Use NumPy?* 2019. Disponível em: <<https://medium.com/fintechexplained/why-should-we-use-numpy-c14a4fb03ee9>>. Acesso em: 26 de maio de 2022.

MESNIL, G. et al. Ensemble of generative and discriminative techniques for sentiment analysis of movie reviews. *arXiv preprint arXiv:1412.5335*, 2014.

MORENCY, L.-P. et al. Towards multimodal sentiment analysis: Harvesting opinions from the web. In: *Proceedings of the 13th International Conference on Multimodal Interfaces*. New York, NY, USA: Association for Computing Machinery, 2011. (ICMI '11), p. 169–176. ISBN 9781450306416. Disponível em: <<https://doi.org/10.1145/2070481.2070509>>.

NETO, A. M. dos S. A. et al. Sidi-nlp-team at idpt2021: Irony detection in portuguese 2021. In: *IberLEF@SEPLN*. [S.l.: s.n.], 2021.

NETO, J. F. d. S. et al. Reconhecimento de entidades nomeadas para o português usando redes neurais. Pontifícia Universidade Católica do Rio Grande do Sul, 2019.

ONAN, A. Two-stage topic extraction model for bibliometric data analysis based on word embeddings and clustering. *IEEE Access*, v. 7, p. 145614–145633, 2019.

ONAN, A.; TOÇOĞLU, M. A. A term weighted neural language model and stacked bidirectional lstm based framework for sarcasm identification. *IEEE Access*, IEEE, v. 9, p. 7701–7722, 2021.

O'CONNOR, Ryan. *PyTorch vs TensorFlow in 2022*. 2021. Disponível em: <<https://www.assemblyai.com/blog/pytorch-vs-tensorflow-in-2022/>>. Acesso em: 26 de maio de 2022.

O'REILLY, T. *What is web 2.0: Design patterns and business models for the next generations software*, September 2005. 2007.

PANG, B.; LEE, L. Opinion mining and sentiment analysis. *Foundations and Trends® in Information Retrieval*, v. 2, n. 1–2, p. 1–135, 2008. ISSN 1554-0669. Disponível em: <<http://dx.doi.org/10.1561/15000000011>>.

RANI, S. et al. Survey of tools and techniques for sentiment analysis of social networking data. *International journal of Advanced computer Science and applications*, v. 12, p. 222, 2021.

SIAMI-NAMINI, S. et al. The performance of lstm and bilstm in forecasting time series. In: *2019 IEEE International Conference on Big Data (Big Data)*. [S.l.: s.n.], 2019. p. 3285–3292.

SILVA, S.; SERAPIAO, A. Detecção de discurso de ódio em português usando cnn combinada a vetores de palavras. In: *Proceedings of KDMILE 2018, Symposium on Knowledge Discovery, Mining and Learning, São Paulo, SP, Brazil*. [S.l.: s.n.], 2018.

SONG, M. et al. Attention-based long short-term memory network using sentiment lexicon embedding for aspect-level sentiment analysis in korean. *Information Processing Management*, v. 56, n. 3, p. 637–653, 2019. ISSN 0306-4573. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S0306457318305612>>.

SOUZA, F. et al. Bertimbau: pretrained bert models for brazilian portuguese. In: SPRINGER. *Brazilian Conference on Intelligent Systems*. [S.l.], 2020. p. 403–417.

VASWANI, A. et al. Attention is all you need. *Advances in neural information processing systems*, v. 30, 2017.

ZHANG, A. et al. Dive into deep learning. *arXiv preprint arXiv:2106.11342*, 2021.

ZHANG, J. et al. A novel visualization method for distinction of web news sentiment. In: SPRINGER. *International Conference on Web Information Systems Engineering*. [S.l.], 2009. p. 181–194.

ZHANG, L. et al. 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.