UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
Câmpus de São José do Rio Preto

MATEUS HIRAMATSU FIORI

UTILIZAÇÃO DE REPRESENTAÇÕES VISUAIS PARA A ANÁLISE SENTIMENTAL EM DOCUMENTOS DE TEXTO

SÃO JOSÉ DO RIO PRETO
2021

Mateus Hiramatsu Fiori

Utilização de Representações Visuais para a Análise Sentimental em Documentos de Texto

Dissertação apresentada como parte dos requisitos para obtenção do título de Mestre em Ciência da Computação, junto ao programa de Pós-Graduação em Ciência da Computação da Universidade Estadual Paulista "Júlio de Mesquita Filho", Instituto de Biociências, Letras e Ciências Exatas, Câmpus São José do Rio Preto.

Financiadora: CAPES

Orientador: Prof. Dr. Felipe Fernandes Fanchini

Coorientador: Prof. Dr. João Paulo Papa

SÃO JOSÉ DO RIO PRETO
2021

F519u

Fiori, Mateus Hiramatsu

Utilização de representações visuais para análise sentimental em documentos de texto / Mateus Hiramatsu Fiori. -- São José do Rio Preto, 2021

64 p. : il., tabs.

Dissertação (mestrado) - Universidade Estadual Paulista (Unesp), Instituto de Biociências Letras e Ciências Exatas, São José do Rio Preto

Orientador: Felipe Fernandes Fanchini

Coorientador: João Paulo Papa

1. NLP. 2. CNN. 3. Análise Sentimental. I. Título.

Sistema de geração automática de fichas catalográficas da Unesp. Biblioteca do Instituto de Biociências Letras e Ciências Exatas, São José do Rio Preto. Dados fornecidos pelo autor(a).

Essa ficha não pode ser modificada.

Mateus Hiramatsu Fiori

Utilização de Representações Visuais para a Análise Sentimental em Documentos de Texto

Dissertação apresentada como parte dos requisitos para obtenção do título de Mestre em Ciência da Computação, junto ao programa de Pós-Graduação em Ciência da Computação da Universidade Estadual Paulista "Júlio de Mesquita Filho", Instituto de Biociências, Letras e Ciências Exatas, Câmpus São José do Rio Preto.

Financiadora: CAPES

Banca Examinadora

Prof. Dr. Felipe Fernandes Fanchini
UNESP - Campus de Bauru
Orientador

Prof. Dr. Thiago A. S. Pardo
USP - Campus de São Carlos

Prof. Dr. Kelton Augusto Pontara da Costa
UNESP - Campus de Bauru

BAURU
1 de março de 2021

Dedico este trabalho aos meus iluminados pais, Claudio e Raulina. E à minha namorada e grande amiga Isis...

Agradecimentos

Agradeço aos meus pais por proporcionarem toda a estrutura e base necessária para a realização deste trabalho.

Agradeço à minha namorada, Isis, por sempre acreditar em mim, mesmo nas horas que nem eu mesmo acreditava.

Ao meu orientador por me guiar durante todo o período de desenvolvimento do projeto.

Ao IBILCE.

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

E agradeço a Deus por todos os momentos de reflexão e confortos psicológicos proporcionados.

Science is not only a disciple of reason but also one of romance and passion.
(HAWKINGS, 2010)

Resumo

É notável o crescimento e popularização da utilização de conceitos e métodos de inteligência artificial, mais especificamente do aprendizado de máquina, para resolução de problemas cotidianos. Através da análise de dados históricos em combinação com um alto poder computacional é possível se obter resultados que superam a capacidade humana. O presente trabalho utiliza-se deste preceito para analisar e classificar documentos de texto de acordo com os sentimentos expressos nestes, sejam eles positivos ou negativos. Em outras palavras, desenvolveu-se um modelo capaz de determinar se uma crítica de determinado filme é positiva ou negativa. No trabalho, foram utilizados diversos tipos de algoritmos de aprendizagem, desde os mais simples, como a Regressão Logística, até modelos mais complexos como as Redes Neurais Convolucionais. A abordagem inicial foi transformar os dados textuais em representações numéricas coerentes, transformá-las em imagens e utilizá-las em diversos métodos de aprendizagem de máquina, fazendo uso de camadas convolucionais. Observou-se que a abordagem proposta possui uma performance melhor quando comparada aos algoritmos de aprendizagem sem as camadas convolucionais, que são ferramentas exclusivas para o tratamento de imagem. Isso mostra que o uso de técnicas de tratamento de imagem se mostra promissor quando o intuito é a analise de sentimentos.

Palavras-chave: CNN, NLP, Redes Neurais, Aprendizado de Máquina, Aprendizado Profundo

Abstract

It is remarkable the growth and popularization of the use of artificial intelligence's concepts and methods, more specifically machine learning's methods, for solving daily life problems. Through the analysis of historical data in combination with a high computational processing power it is possible to obtain results that surpass human capacity. The present work utilizes this precept to analyze and to classify text documents according to the feelings expressed on it, where these feelings could be described as positive or negative. In other words, we have developed an algorithm capable of determining whether a review of a given film is positive or negative. In the work, it is used several types of learning algorithms, from the simplest, such as Logistic Regression, to more complex models such as Convolutional Neural Networks. The initial approach was to transform the textual data into coherent numerical representations, transform them into images and use it in several machine learning methods, using convolutional layers. It was observed that the proposed approach has a better performance when compared to learning algorithms without convolutional layers, which are exclusive tools for image treatment. This shows that the use of image treatment techniques is promising when the intention is to analyze the feelings.

Keywords: CNN, NLP, Neural Networks, Machine Learning, Deep Learning

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

NLP	Natural Language Processsing - Processamento de Linguagem Natural
CNN	Convolutional Neural Network - Rede Neural Convolucional
MLP	Multi-Layer Perceptron - Perceptron Multi Camadas
LGBM	Light Gradient Boosting Machine
BOW	Bag of Words - Saco de Palavras
CBOW	Continuous Bag of Words - Saco de Palavras Contínuo
DBOW	Distributed Bag of Words - Saco de Palavras Distribuído
DM	Distribute Memory - Memória Distribuída
IMDB	Internet Movie Database - Banco de dados de filmes da internet
SST2	Stanford Sentiment Tree Binary - Árvore Binária de Sentimentos de Stanford

Sumário

1 Introdução

Desde os primórdios da computação, a necessidade de automatizar e imitar o comportamento e as ações dos seres humanos está fortemente presente nos conceitos e ideias geradas para a resolução de problemas do cotidiano (TURING, 1950). Com isso, a computação que é conhecida hoje em dia já nasce com um ramo que possui o próprio ser humano como fonte de inspiração. Pode-se visualizar tal fato com a popularização das redes neurais e seus derivados, tais como modelos matemáticos/estatísticos, assim como o ramo da visão computacional e computação gráfica, que aproxima os padrões utilizados nos seres humanos para a compreensão de imagens, cores, profundidade e afins pela máquina.

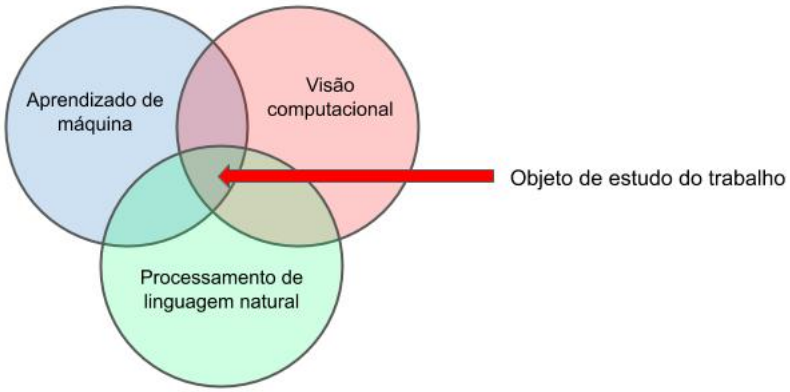
As redes neurais estão fortemente ligadas ao ramo do aprendizado de máquina, que por sua vez faz parte do ramo da inteligência artificial. Apesar deste último ser um assunto de longa data, a prova empírica de métodos mais robustos e complexos, como o aprendizado em profundidade, só pôde ser validada atualmente pela disponibilidade de recursos computacionais elevados (GOODFELLOW; BENGIO; COURVILLE, 2016).

Junto a isso, algumas outras áreas também se tornam relevantes em termos de utilizar o ser humano como base, sendo elas o processamento de linguagem natural (PLN), que tem como objetivo principal tornar a máquina capaz de entender as linguagens pelas quais os seres humanos se comunicam (MANNING; SCHüTZE, 1999), assim como o ramo de análise de sentimento, que através da composição de técnicas de PLN e aprendizado de máquina, visa encontrar a polaridade ou sentimento descrito por um documento (LIU, 2012).

Para ilustrar, a Figura 1 demonstra as áreas de atuação cobertas por esse trabalho. Aqui, propomos uma nova abordagem para a detecção de sentimentos em documentos de texto onde, através da utilização de camadas de convolução, e outros algoritmos de classificação, classificamos imagens geradas a partir de representações matriciais de textos a ela submetidas. Com isso, é notável a utilização do aprendizado de máquina aplicado às resoluções de problemas, pela classificação e/ou regressão feitas a partir da análise e extração de características de dados históricos, com o intuito de encontrar padrões para predições de eventos e que forneçam informações relevantes para a resolução do problema final (GOODFELLOW; BENGIO; COURVILLE, 2016).

Para que tal feito seja possível, primeiramente é necessário coletar dados, devidamente classificados, de textos disponíveis na internet que tratem do objeto de estudo do trabalho. Em seguida, técnicas de aprendizado em profundidade somado às técnicas de processamento de imagem e de linguagem natural (PLN) também serão necessárias para que o documento de texto seja processado, estruturado e então utilizado em camadas de convolução como entrada para um algoritmo de aprendizagem de máquina (LECUN et al., 1998) projetado para

Figura 1 – Diagrama de Venn ilustrando as áreas cobertas pelo trabalho.



Três grandes áreas da computação: aprendizado de máquina, visão computacional e processamento de linguagem natural utilizadas em conjunto para definir o escopo do trabalho, representação utilizando diagrama de Venn. Fonte: Feito pelo autor

a classificação do sentimento do documento. Uma das estruturas cruciais de nosso trabalho são as denominadas camadas de convoluções, responsável por extrair as características mais importantes das imagens a ela submetida. Esta camada será utilizada como etapa diferencial do trabalho e serão abordados mais detalhadamente nos capítulos seguintes.

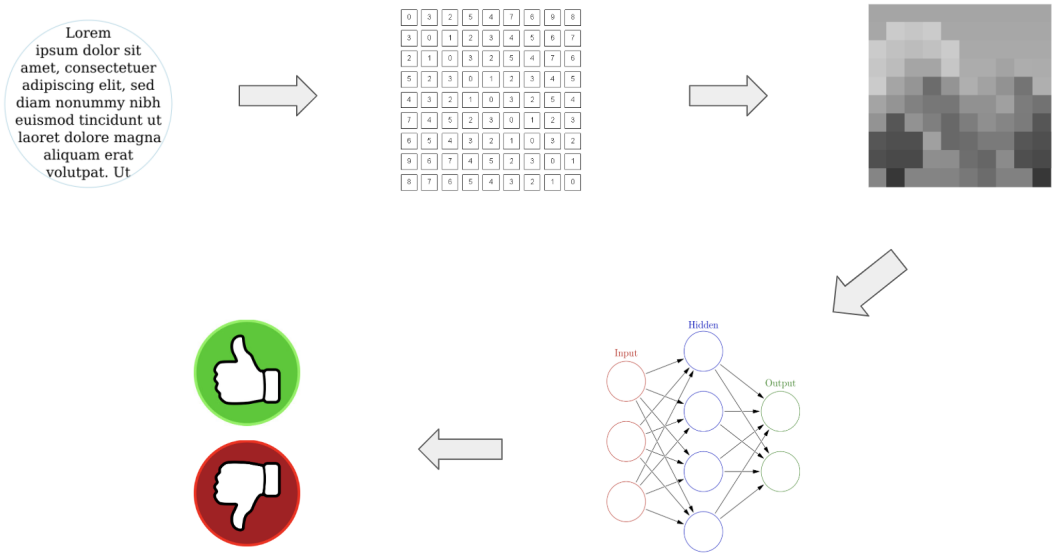
Finalmente, para a fase de testes e validação, um conjunto de documentos será separado para esse fim. Fazemos uso de um conjunto de teste e validação previamente separados e amplamente utilizados em diversos outros trabalhos acadêmicos (THONGTAN; PHIENTHRAKUL, 2019) (RAFFEL et al., 2020), o que padroniza e facilita a comparação dos resultados obtidos.

O escopo geral do trabalho, portanto, é propor uma nova abordagem na maneira de se tratar documentos de textos antes destes serem submetidos ao treinamento de um classificador de sentimentos e, então, validar a proposta para um caso de uso aplicado. Uma ilustração das etapas desse processo pode ser visto na Figura 2.

1.1 Problema

O principal problema a ser resolvido é o desenvolvimento de um classificador que possa substituir a necessidade de utilizar um ser humano para designar uma tarefa repetitiva,

Figura 2 – Concepção macro das etapas implementadas no trabalho



Especificação de cada etapa implementada no trabalho. Desde o texto sem processamento, passando pelos métodos propostos, até a classificação do sentimento relativo ao texto. Fonte: Feito pelo autor

que envolve analisar e interpretar documentos de texto, com o objetivo de responder se os documentos possuem um sentimento positivo ou negativo (entenda sentimento positivo ou negativo como uma crítica positiva ou negativa).

1.2 Objetivos

1.2.1 Objetivo Geral

Propor uma nova abordagem que substitui a representação matricial textual por uma imagem na arquitetura de implementação de um modelo treinado para classificar o sentimento de documentos de texto. Neste sentido, iremos verificar sua performance, em termos de assertividade, quando comparado a modelos de PLN já existentes na atualidade.

1.2.2 Objetivo Específico

Para que o objetivo geral seja cumprido, é necessário a execução dos itens descritos a seguir:

a) Estudar técnicas de processamento de linguagem natural, processamento de imagens, redes neurais convolucionais para classificação de imagens e representação de palavras no espaço algébrico matemático;

b) Buscar bases de dados de documentos de texto em português para serem utilizadas nos treinamentos dos modelos de aprendizado de máquina;

c) Propor uma representação de texto baseada em imagens;

d) Desenvolver uma rede neural convolucional para classificação das imagens geradas;

e) Verificar se uso das camadas de convoluções contribuem para uma melhora na classificação de sentimento;

f) Treinar um classificador utilizando métodos conhecidos de processamento de linguagem natural para posterior análise;

g) Comparar os resultados com classificadores já existentes para análise de sentimento.

2 Aprendizado de Máquina

Para a elaboração e implementação do trabalho foram utilizados diversos métodos e algoritmos de classificação de dados, desde os mais simples como a Regressão Logística (BISHOP, 2006) e *Naive Bayes* (JURAFSKY; MARTIN, 2009), até métodos mais complexos como o LGBM (*Light Gradient Boosting Machine*) (KE et al., 2017), Redes Neurais (HAYKIN, 2009) e Redes Neurais Convolucionais (HAN, 2017).

Para uma melhor compreensão do funcionamento dos algoritmos citados é necessária uma contextualização histórica dos elementos que os compõem e permitem o seu funcionamento, apresentada nas subseções a seguir.

2.1 Contexto Geral

Para um bom entendimento do presente trabalho, se faz necessário o estabelecimento de alguns conceitos e definições que servirão de base para tal. A primeira definição é o conceito do que é um dado e qual a sua finalidade. Um dado nada mais é do que um conjunto de informações que identificam e caracterizam um determinado artefato, entidade ou objeto. O dado, portanto, se constitui das características de determinado objeto que o identifica. O dicionário Michaelis possui uma definição bem objetiva sobre dado: "Representação de fatos, conceitos e instruções, por meio de sinais, de maneira formalizada, possível de ser transmitida ou processada pelo homem ou por máquinas."

Para concretizar o conceito vejamos o exemplo de uma casa. Seu conceito é bastante amplo no sentido de que qualquer pessoa pode imaginar o que é uma casa. Porém, sem informações mais detalhadas, é muito provável que cada indivíduo imagine uma casa diferente em seu consciente. Para solucionar tal problema utiliza-se um conjunto de características que têm como objetivo identificar a determinada casa, tornando-a mais específica e única no imaginário de cada pessoa. Para este conjunto de características se dá o nome de dados.

Outro conceito importante a ser estabelecido é o de conjunto de dados. O conjunto de dados é o conjunto de informações que identificam diversos objetos ou entidades de um mesmo tipo. Por exemplo, um conjunto de dados de casas nada mais é do que um conjunto de informações que identificam uma ou mais casas distintas. Estes conceitos podem ser extrapolados para diversos tipos de dados, não apenas de casas, como por exemplo dados que identificam pessoas, animais, fotos e até mesmo textos.

Dados e conjuntos de dados, portanto, são o alicerce deste trabalho. É a partir deles que é possível a utilização de estatística inferencial e aprendizado de máquina para classificação de documentos de texto, como por exemplo de conteúdos negativos, com críticas ou xingamentos

sobre determinado assunto, e como de conteúdos positivos, como elogios sobre determinado assunto.

Para o desenvolvimento desse trabalho levou-se em consideração diversas áreas do conhecimento computacional, sendo elas o aprendizado de máquina e aprendizado em profundidade, o processamento de linguagem natural e o processamento de imagens. Cada área será melhor definida e aprofundada abaixo em seus respectivos subseções.

O aprendizado de máquina e seus algoritmos estão atrelados ao ramo da estatística inferencial, que tem como objetivo utilizar dados para tarefas de classificação, agrupamento ou regressão, de acordo com as características presentes no próprio conjunto de dados. A tarefa de encontrar padrões, ou aprender a partir de um conjunto de dados faz parte do aprendizado estatístico (HASTIE; TIBSHIRANI; FRIEDMAN, 2001).

Quando se trata destes tipos de algoritmos recai-se inexoravelmente sob a questão da representação dos dados. Em outras palavras, para que um algoritmo consiga aproximar uma função que represente uma separação entre os dados localizados em determinado espaço (ou hiperespaço, quando o espaço a ser utilizado para representar os dados visualmente possui mais de três dimensões), esses precisam estar bem definidos em termos de suas características, para que de fato exista um padrão em sua representação (GOODFELLOW; BENGIO; COURVILLE, 2016).

Uma das partes mais importantes do presente trabalho é a utilização de métodos de classificação. Algoritmos de classificação podem ser divididos em dois subgrupos: algoritmos de aprendizado supervisionado e algoritmos de aprendizado não-supervisionado. A principal diferença entre os dois tipos está nas informações de entrada que são submetidas a esses algoritmos.

Para tornar o conceito concreto, tomemos como exemplo o subconjunto de dados definidos pela Tabela 1

Tabela 1 – Subconjunto do conjunto de dados Iris (medidas em cm)(FISHER, 1936)

ComprimentoSepala	LarguraSepala	ComprimentoPetala	LarguraPetala	Tipo
4.4	2.9	1.4	0.2	iris-setosa
6.7	3.1	4.7	1.5	iris-versicolor
7.2	3.6	6.1	2.5	iris-virginica

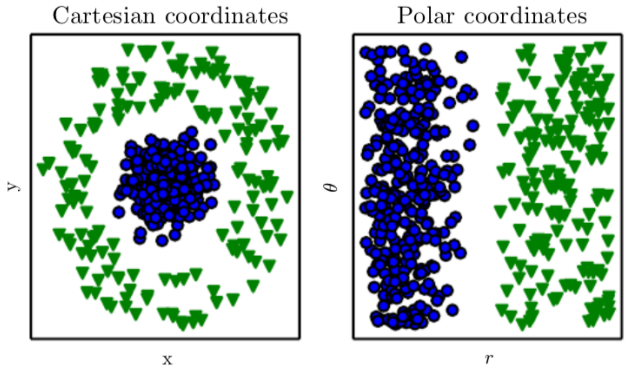
Fonte: Feito pelo autor

A Tabela 1 representa um subconjunto de dados que define diversos tipos de flores. Os dados foram retirados de um famoso conjunto de dados denominado Iris, possuindo 150 dados distintos de 3 tipos diferentes de flores, e cada dado sendo identificado por 4 características: comprimento da sépala, largura da sépala, comprimento da pétala e largura da pétala (FISHER, 1936). Como podemos observar, cada dado, ou cada flor, possui características específicas que as identificam, além da existência da coluna "Tipo". Esta coluna também é concebida com

o nome de coluna alvo ou variável dependente e é justamente esta coluna que diferencia um algoritmo supervisionado de um algoritmo não-supervisionado. Ela possui este nome pois é a partir dela que os algoritmos de classificação supervisionados serão processados. Em outras palavras, para algoritmos supervisionados os dados de treinamento precisam ser identificados por uma coluna alvo e para algoritmos não-supervisionados essa identificação prévia não é necessária.

Em termos gerais, o objetivo de um algoritmo de classificação supervisionado, alimentado com o conjunto de dados Iris, é gerar um modelo que, dado um argumento de entrada como um vetor de características de flores (também conhecido como variáveis independentes) e um valor categórico (ou alvo que identifica cada flor), seja capaz de dizer de qual tipo é a flor em questão. A Figura 3 demonstra uma possível representação de dados e mostra visivelmente uma aglomeração no momento de plotagem. Dados que se localizam perto de outros dados, são ditos similares e, por consequência, pertencem a um mesmo grupo.

Figura 3 – Representação de dados no plano cartesiano.



Demonstração visual da posição cartesiana de cada dado, sendo possível perceber uma região explícita circular que separa os dois tipos de dados. Na segunda figura o mesmo conceito da figura anterior é demonstrado sendo possível utilizar uma reta para separar os dois tipos de dados. Fonte: (GOODFELLOW; BENGIO; COURVILLE, 2016)

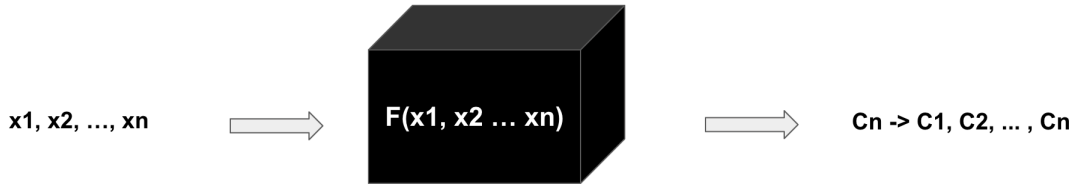
O que um algoritmo de classificação busca encontrar é alguma maneira de separar estes grupos (GOODFELLOW; BENGIO; COURVILLE, 2016). Contextualizando essa questão com o assunto do trabalho, o classificador implementado busca, na etapa de treinamento, categorizar os dados a ele submetidos, no caso do trabalho textos relacionados a críticas, em duas categorias: positivo e negativo. Para tal feito é necessário utilizar conjuntos de dados que contenham diversas frases ou documentos de texto anotados por um ser humano com as duas categorias citadas. Dois conjuntos de dados amplamente conhecidos e utilizado na literatura e que também serão utilizados por nós são o conjunto de dados IMDB (MAAS et al., 2011) e o SST-2 (*Stanford Sentiment Treebank Binary*) (SOCHER et al., 2013).

Outro aspecto importante de algoritmos de aprendizado estatístico é o processo de aprendizado, também conhecido como treinamento. É nesse momento que acontece o apren-

dizado e reconhecimento de padrões através dos dados. Por uma questão de objetividade e simplicidade inicial, podemos conceber os algoritmos de classificação como uma caixa preta (ilustrada pela Figura 4) ou como uma função qualquer, que dado o parâmetro de entrada como sendo um conjunto X contendo as n características de um determinado conjunto de dados, obtém-se o resultado $F(Xn) = W^t.Xn = C$ como sendo a classe cujo dado de entrada pertence, e W representando o vetor de pesos, ou coeficientes responsáveis por determinar a forma da função em questão, e consequentemente responsáveis pela generalização da classificação.

Este conceito servirá de base para explicações mais profundas e detalhadas dos algoritmos utilizados no trabalho, presentes nos capítulos a seguir.

Figura 4 – Ilustração da concepção de algoritmos de aprendizado



Representação visual do conceito de caixa preta para concepção simplificada de como um algortimo de aprendizado de máquina trata os dados submetidos a ele. Aqui, $x1, x2, ..., xn$ representa o vetor de características utilizado como dado de entrada; $F(x1, ..., xn)$ representa o modelo aplicado ao vetor de características e Cn representa o dado de saída que, no caso de um modelo de classificação, define a qual classe o dado de entrada pertence.

Um algoritmo de aprendizado, portanto, é um algoritmo que consegue aprender, ou encontrar os os valores ótimos dos pesos (W) de acordo os valores numéricos que caracterizam determinados dados (X), para então classificá-los de forma correta. Mitchell possui uma definição concisa sobre o assunto: "Um programa de computador aprende se, de acordo com uma experiência E, aliado a um conjunto de tarefas T e medidas de performance P, a performance P sobre as tarefas T, aumenta de acordo com a experiência E"(MITCHELL, 1997). Em outras palavras, um software é dito que aprende se a acurácia de seu resultado de saída aumenta e tende a resultados próximos da realidade conforme mais tarefas de treinamento são executadas. Uma tarefa de aprendizado de máquina normalmente é descrita em termos de como um sistema deve processar um dado. Esses dados são definidos em termos de características. Um dado é normalmente representado por um vetor X pertencente a Rn, onde $X = \{x1, x2, \ldots, xn\}$. É com os dados deste X que o algoritmo irá trabalhar (GOODFELLOW; BENGIO; COURVILLE, 2016).

Existem muitos tipos de tarefas que um algoritmo de aprendizado pode executar e a que se encaixa com o contexto dessa dissertação é a tarefa de classificação. Como dito, este é um programa que tem como objetivo categorizar um determinado dado Z em uma determinada classe k (GOODFELLOW; BENGIO; COURVILLE, 2016). Para que isso seja

possível o algoritmo normalmente produz uma função $f : Rn \rightarrow \{1, \ldots, k\}$, ou seja, $y = f(X)$, onde X é um vetor de características que aplicado a f, produz um valor y mapeado para uma das classes de $\{1, \ldots, k\}$.

Existem outros tipos de abordagens de classificação, como por exemplo abordagens probabilísticas que calculam a probabilidade de determinadas características do conjunto de dados pertencerem a uma classe ou não. De forma geral, $P(Y|X)$, em outras palavras, qual a probabilidade de o dado ser da classe Y dado o conjunto características X. Tal abordagem é utilizada pelo método Naive Bayes (JURAFSKY; MARTIN, 2009), utilizado no trabalho.

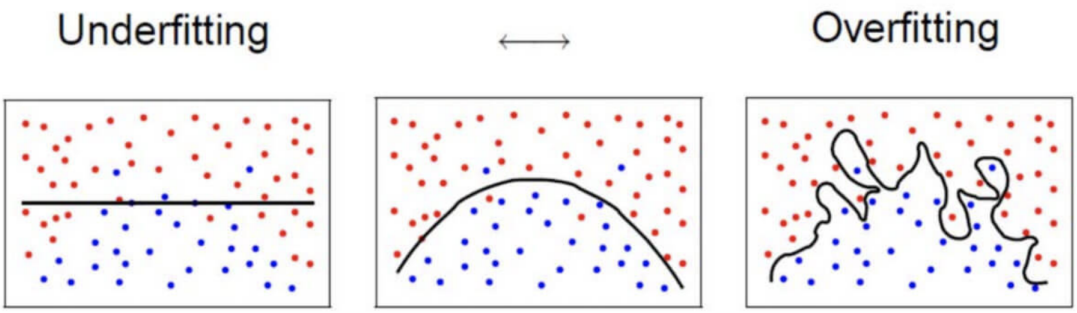
2.1.1 Generalização do modelo

O principal objetivo do aprendizado de máquina é conseguir encontrar o melhor conjunto de pesos, ou coeficientes, através dos diversos algoritmos de aprendizado e métodos de otimização, relacionado à um determinado conjunto de dados, para assim gerar um modelo matemático capaz de generalizar e caracterizar os dados analisados.

Os algoritmos de aprendizagem de máquina se generalizam, em suma, introduzindo pesos, ou coeficientes, diferenciados às características de entrada. Um fator importante sobre essa generalização é a forma que medimos a performance do modelo em questão. De maneira geral, um modelo de aprendizado de máquina sempre apresentará erros, de forma que os algoritmos sempre buscarão encontrar os melhores pesos a serem ajustados, tendo como base a minimização do erro intrínseco (HAYKIN, 2009). A generalização do modelo, então, é medida pela distância do valor esperado em relação a uma nova entrada. Quanto mais perto de zero menor é o erro de generalização (GOODFELLOW; BENGIO; COURVILLE, 2016). Vale enfatizar porém que tal afirmação remete à uma conclusão incorreta sobre a qualidade do modelo, pois o melhor modelo não é aquele que possui o menor erro, mas sim aquele que melhor generaliza os dados analisados. É neste momento que recai-se sobre os problemas de *overfitting* e *underfitting*. O primeiro acontece quando o treinamento do modelo generaliza demais os dados, a ponto de tornar o aprendizado específico aos dados utilizados no momento do treinamento, fazendo assim com que o modelo perca seu poder de generalização. O segundo, por sua vez, é justamento o contrário do *overfitting*, e acontece quando o modelo é incapaz de ajustar os parâmetros de maneira suficiente para a generalização, gerando assim resultados com um alto erro na saída, quando comparado ao valor esperado. Tais conceitos são representados pela Figura 5.

É razoável indagar, portanto, como saber se um modelo atingiu um aprendizado adequado. Para isso existem algumas propostas de visualização que auxiliam nessa análise, como demonstrado na Figura 6. É possível perceber que ao decorrer das iterações de aprendizado o erro do treinamento tende a diminuir assintoticamente, enquanto o erro de generalização possui um valor ótimo, onde ocorre justamente a menor distância entre os erros de treinamento e generalização (GOODFELLOW; BENGIO; COURVILLE, 2016). Com isso é possível escolher

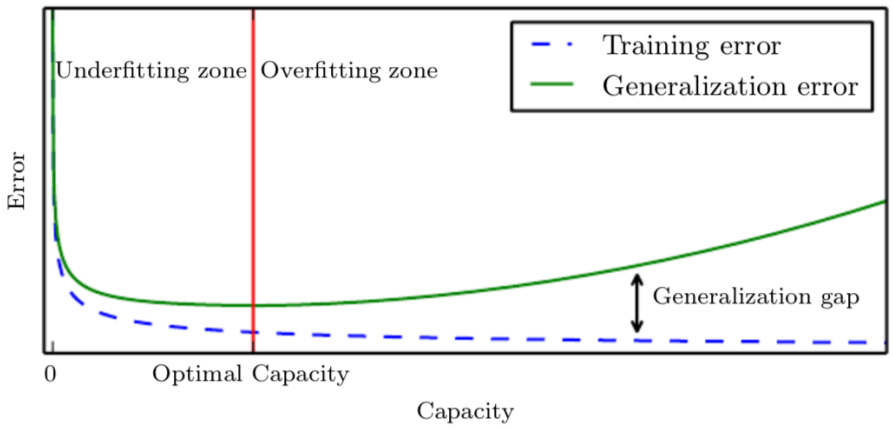
Figura 5 – Representação gráfica de exemplos de *underfitting* e *overfitting*



Na primeira figura é apresentado o conceito de underfitting utilizando uma curva cujos parâmetros não foram suficientemente ajustados para separar os dados de maneira assertiva. Na segunda figura a curva apresentada possui parâmetros ajustados corretamente, conseguindo separar os dados de maneira generalizada. Na última figura é apresentado o conceito de overfitting onde a curva em questão consegue separar os dados de maneira assertiva, porém sem nenhuma generalização. Fonte: (ROBERTSHAW, 2015)

o melhor conjunto de parâmetros e, consequentemente, a melhor versão do modelo.

Figura 6 – Curvas que representam os erros de treinamento e generalização, e o ponto ótimo do modelo



Demonstração do ponto ótimo de treinamento. A curva em verde representa a performance do modelo em função de sua capacidade de classificação. Podemos notar que a partir de determinado momento o erro de generalização começa a aumentar por conta do overfitting, enquanto o erro de treinamento permanece normal tendendo a zero. Fonte: (GOODFELLOW; BENGIO; COURVILLE, 2016)

2.1.2 Hiperparâmetros e conjuntos de validação

A grande maioria dos algoritmos e aprendizado de máquina possuem diversos parâmetros que podem ser ajustado e que influenciam na execução, nos cálculos e, consequentemente, no resultado gerado pelo algoritmo. Tais parâmetros são comumente denominados de hiperparâme-

tros. Diferentemente dos pesos ajustados pelos processos de aprendizagem, os hiperparâmetros não possuem essa abordagem (não são ajustados automaticamente), a não ser que isso seja feito explicitamente. Pela escolha dos hiperparâmetros é possível esbarrar nas zonas de *underfitting* e *overfitting*, de forma que os valores a serem escolhidos precisam condizer com a capacidade do modelo cujo erro de generalização seja o menor possível. Para que essa questão seja resolvida e testada é necessário um conjunto de validação contendo exemplos de dados que não foram utilizados para o treinamento do modelo. Com isso é possível testar se a qualidade no modelo está razoável, pela comparação dos resultados obtidos com os resultados esperados (GOODFELLOW; BENGIO; COURVILLE, 2016). Para ilustrar tal peculiaridade, ilustramos na Figura 6 a relação entre a performance do modelo e o erro de generalização. Vale frisar que o conjunto de validação, provém de uma parte do conjunto de treinamento, que são separados antes do momento de aprendizado. Normalmente utiliza-se 80% para treinamento e 20% para validação mas, certamente, essa não é uma regra. Os 20% separados são reservados apenas para validação do modelo e não serão utilizados em nenhum momento da etapa de treinamento. São dados, portanto, nunca antes vistos pelo algoritmo, maximizando a realidade do problema em seu desempenho. Essa representação é ilustrada na Figura 7.

Figura 7 – Representação de como são divididos os conjuntos de treinamento, de validação e de teste (fora de proporção)

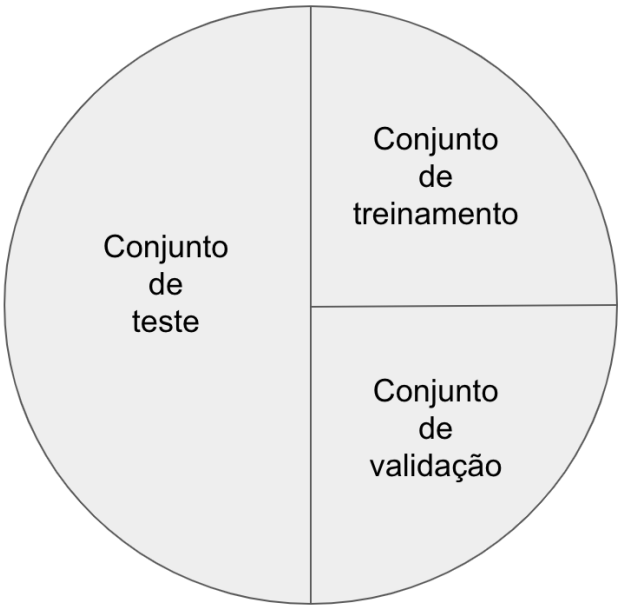


Gráfico que ilustra os diferentes tipos de conjuntos de dados utilizados nas etapas do aprendizado de máquina. O conjunto de treinamento é utilizado na etapa de treinamento do modelo, estes dados não devem ser utlizados nas outras etapas para não enviesar o resultado final do modelo; o conjunto de validação é utilizado para validar se os hiperparâmetros utilizados estão coerentes; e o conjunto de teste é utilizado para verificar a capacidade de classificação do modelo, este conjunto é apenas utilizado nesta etapa. Fonte: Feito pelo autor.

2.2 Algoritmos de Aprendizado

2.2.1 Regressão Logística

Um dos algoritmos de classificação utilizados no trabalho é o Regressor Logístico, que toma como base para a classificação a função logística, também conhecida como função sigmóide, primeiramente utilizada num contexto de análise de taxa de crescimento populacional (PEARL; REED, 1920), mas que posteriormente se mostrou como um método promissor para tarefas de classificação em geral. Apesar de estar presente em seu nome o termo 'Regressão' o método é utilizado como um classificador (BISHOP, 2006). Nesta seção será definido e explicado em detalhes o funcionamento do algoritmo.

A função sigmóide é definida como

$$f(a) = \frac{1}{1 + e^{-a}}, \tag{1}$$

onde a é tido como um vetor de caracrterísticas em combinação linear com os parâmetros de peso, assim $a = (w0 + w1.a1 + w2.x2 + ... + wn.xn)$. Os valores de a também podem ser definidos como uma multiplicação pontual de dois vetores,

$$a = X.W = \begin{pmatrix} 1 \\ x1 \\ x2 \\ ... \\ xn \end{pmatrix} . \begin{pmatrix} w0 \\ w1 \\ w2 \\ ... \\ wn \end{pmatrix}, \tag{2}$$

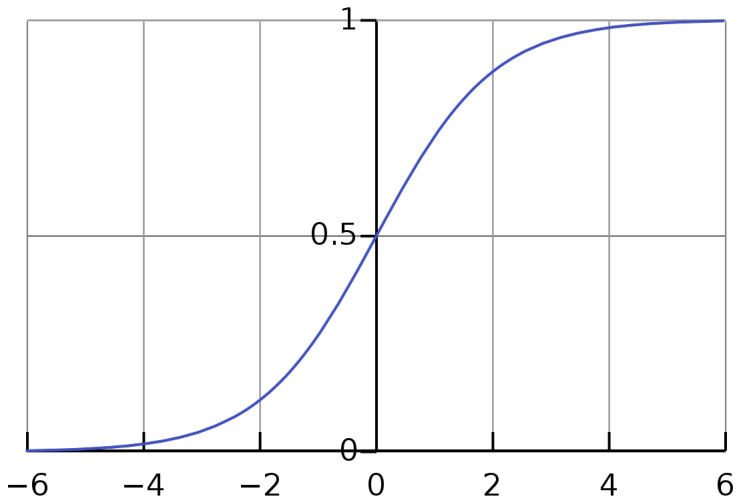
onde X é o conjunto de características e W é conjunto de pesos a ser otimizado. A função logística possui uma forma descrita pela Figura 8.

O regressor logístico, portanto, pode ser concebido como uma função que dado um conjunto de características de determinado dado, esta função produz um resultado binário $[0, 1]$. Em termos matemáticos, dado $X = x0, x1, x2, \ldots xn$ como sendo o conjunto de entrada para a função, tem-se $f(X) = [0, 1]$, ou $P(X) = P(Y = 1|X)$ em termos probabilísticos, lendo-se como a probabilidade de $Y = 1$, dado o conjunto de características X. O algoritmo recebe este nome justamente por se utilizar da função logística, também conhecida como sigmoid, para o cálculo do resultado de saída.

2.2.1.1 Otimização de parâmetro

Para o ajuste dos coeficientes W é comumente utilizado o método do estimador de máxima probabilidade (BISHOP, 2006), onde este ajuste é feito a partir dos dados utilizados para treinamento. O estimador de máxima probabilidade é um algoritmo de otimização utilizado por uma miríade de algoritmos de aprendizado de máquina. Seu principal objetivo é encontrar um conjunto de valores de coeficientes W cujo valor de saída da função logística se aproxime ao

Figura 8 – Gráfico da função logística



Ilutração da forma da função logística. Fonte: Research Gate. Disponível em: <https://www.researchgate.net/figure/Logistic-function-curve-This-is-a-standard-logistic-function-and-it-has-a-typical-S_fig1_336974843>

máximo da classe principal utilizada como base, aquela definida por 1, e para dados pertencentes à classe 0, valores que se aproximem ao máximo de 0. Este algoritmo pertence à classe dos métodos numéricos de otimização Quasi-Newton (BISHOP, 2006).

2.2.2 *Naive Bayes*

O método *Naive Bayes* é outro utilizado no trabalho para a tarefa de classificação binária de texto por ser comumente utilizado em tarefas desta natureza. Tal método possui uma abordagem diferente utilizada pelo regressor logístico, sendo esta uma abordagem probabilística tomando como base o Teorema de Bayes. Tal método apresenta resultados interessantes em tarefas de identificação de spam em e-mails (Luo et al., 2010). Dado um conjunto de características $X(x0, x1, \ldots, xn)$, o objetivo do método é retornar a probabilidade deste conjunto pertencer a uma determinada classe, por exemplo, à classe 1. Em outra abordagem, pode-se definir o modelo como sendo $(Y = y|X = (x0, x1, \ldots, xn))$, ou ainda em termos da probabilidade condicional $P(Y|X)$, onde este pode ser interpretado pelo teorema de Bayes, conforme definido pela equação abaixo:

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}. \tag{3}$$

O cálculo das probabilidade condicionais são simples e podem ser demonstradas com o seguinte exemplo. Suponha um subconjunto de dados fictícios derivado do conjunto de dados aberto denominado Golf, descrito na Tabela 2. Aqui, $X =$[Tempo, Temperatura, Umidade, Vento] e $Y =$ [Favorável]. Para calcularmos a probabilidade de termos um ambiente não favorável, ou seja, $P(Y =$ não$)$, usa-se a quantidade de ocorrências de 'não' sobre a quantidade total

Tabela 2 – Subconjunto do conjunto de dados Golf

Tempo	Temperatura	Umidade	Vento	Favorável
ensolarado	quente	alta	falsa	não
ensolarado	quente	alta	verdadeira	não
nublado	quente	alta	falsa	sim
chuvoso	ameno	alta	falsa	sim
chuvoso	frio	normal	falsa	sim

Fonte: Feito pelo autor

de ocorrências, de forma que $P(Y = \text{não}) = 2/5$. Para se calcular $P(X = \text{quente}|Y = \text{sim})$ usa-se a quantidade de ocorrencias de 'quente' e 'sim' sobre a quantidade total de 'sim', de forma que $P(X = \text{quente}|Y = \text{sim}) = 1/3$. Cada combinação de todos os possíveis $P(X|Y)$ é tido como um parâmetro do método Naive Bayes de forma que, partindo do princípio que as características do conjunto de dados são dependentes umas das outras, conclui-se que quanto maior a quantidade de características de um conjunto de dados, a quantidade de possibilidades de parâmetros é exponencialmente maior. Mais especificamente, $2^{(k+1)} - 1$ parâmetros, onde k é quantidade de características do conjunto de dados. Neste caso, a alta quantidade de parâmetros se torna um problema pois exige proporcionalmente recursos computacionais que pode acarretar num aumento de custos.

Para resolver o problema de custo computacional costuma-se utilizar uma abordagem "ingênua", onde se considera que as características do conjunto de dados são independentes umas das outras. De fato, isso é algo que raramente acontece, mas diminui consideravelmente a quantidade de parâmetros utilizados pelo algoritmo, baixando a quantidade para $2k$ parâmetros (ZHANG, 2004). Isso se torna possível pois se as características forem independentes pode-se assumir que $P(X = (x1, x2)|Y) = P(X = x1|Y).P(X = x2|Y)$ o que, em termos gerais, simplifica as probabilidades condicionais conforme equação abaixo,

$$P(Y|X) = \frac{P(Y)\prod_{i=1}^{n} P(X_i|Y)}{P(X)}, \tag{4}$$

onde a regra de classificação pode ser definida por

$$y = \arg\max_{y} P(Y)\prod_{i=1}^{n} P(X_i|Y) \tag{5}$$

Deste modo, como estamos considerando que as características do conjuntos são independentes umas das outras, quantidade de operações computacionais são reduzidas, e assim reduzindo o custo computacional significativamente.

2.2.3 *Light Gradient Boosting Machine*

Outro modelo de classificação utilizado no trabalho é o chamado *LightGBM*. Este algoritmo é relativamente novo e pertence à classe dos métodos de aprendizado supervisionado

baseado em árvores.

Os algoritmos baseados em árvore apresentados por (BREIMAN et al., 2017), abordam tanto problemas de regressão quanto de classificação. Alguns modelos são popularmente conhecidos, tais como árvores de decisão, Random Forest (STATISTICS; BREIMAN, 2001) e Gradient Boosting (AYYADEVARA, 2018).

Um modelo de classificação baseado em árvores pode ser interpretado como um classificador baseado em partições, onde tais partições são criadas de acordo com a similaridade das características do conjunto de dados analisado, resultando em regiões homogêneas criadas a partir de sucessivas divisões, ou particionamentos, binários dos dados (BREIMAN et al., 2017).

O conjunto de dados sem nenhuma partição é denominado raíz e a cada divisão as novas partições são denominadas nós. Cada nó é criado de acordo com uma característica relevante do conjunto. Se um nó não possuir mais nenhuma divisão, este então é denominado nó terminal, ou simplesmente folha, e a ele é atribuída uma classe. Em termos gerais, um classificador baseado em árvores possui k folhas, onde cada folha possui uma característica específica do conjunto de dados, e cada nó terminal pode ser interpretado como um subconjunto, onde os dados pertencentes a ele são similares de acordo com suas características. Na etapa de treinamento. a cada dado submetido ao modelo, este será classificado e direcionado a um subconjunto (folha) específico. A maneira mais comum de se interpretar um subconjunto de um classificador baseado em árvores é em termos de sua homogeneidade ou também pureza. Cada folha de uma árvore tende a ser mais homogênea ou possuir uma maior pureza.

Outro aspecto importante deste tipo de classificador é maneira como é obtida e calculada a melhor subdivisão dos subconjuntos. Para isso existem diversos algoritmos amplamente utilizados, tais como: Gini, Chi Quadrado, Ganho de Informação, entre outros. Todos estes têm como objetivo encontrar o melhor tipo de divisão dos dados em subconjuntos onde cada divisão é feita sobre uma característica do conjunto de dados.

Por exemplo, considerando o conjunto de dados Golf representado pela Tabela 2, o mesmo possui a variável dependente (ou alvo) como sendo a coluna Favorável e o restante das colunas representando as características dos dados. O algoritmo então, para decisão da melhor divisão em subconjuntos, levará em consideração todas a possibilidades de divisão em subconjuntos possíveis. Uma possível divisão, tendo como base a coluna Tempo, é separar os dados em ensolarado e não ensolarado, posteriormente em nublado e não nublado, e assim sucessivamente para todas as possibilidades. Com isso é possível identificar qual a divisão que melhor identifica os subconjuntos em termos de homogeneidade.

É importante ressaltar que as subdivisões binárias sucessivas em subconjunto tende à criação de infinitos subconjuntos com apenas 1 elemento, ocasionando um *overfit* do modelo. Para resolver esse problema o parâmetro de quantidade mínima de elementos por subconjunto

é fornecido e é, então, utilizado como critério de parada. Também são utilizados métodos de poda ou *pruning* para resolver o problema do *overfitting*. Métodos de podas consistem em remover partes desnecessárias ou redundantes da árvore em questão, permitindo assim uma redução na complexidade do classificador.

Como uma maneira de aperfeiçoar ainda mais os resultados dos modelos de classificação baseados em árvores, é utilizada também uma técnicas denominada Boosting, que combina o resultado de diversos classificadores fracos em sequência para criar um modelo forte de classificação, onde cada classificador utilizado minimiza os erros cometidos pelo modelo anterior. No caso, cada classificador fraco é uma árvore de decisão. Este método é o precursor do classificador LGBM utilizado em nosso trabalho. Como pontos positivos pode-se ressaltar que tais modelos possuem uma boa performance para tanto tarefas de classificação quanto de regressão. Além disso, também conseguem lidar com conjuntos de dados que possuem tanto dados categóricos quanto contínuos, com variáveis independentes, e não é necessário uma camada de pré-processamento destes dados. Como lado negativo, tem-se que é necessário um maior cuidado na escolha dos hiperparâmetros e, por consequência, tende a facilitar a ocorrências de *overfit*.

Definidos os conceitos envolvidos nos principais de métodos de classificação baseado em árvores, a definição do método LGBM se torna mais direta e simples. O algoritmo LGBM (KE et al., 2017) é um método baseado em *boosting* que utiliza algoritmos de aprendizado baseado em árvores. Sua principal característica que difere dos métodos tradicionais é que, ao invés de escolher o melhor critério de divisão de nós e crescer a árvore em níveis, o LGBM cresce sua árvore em termos de suas folhas. Assim, o algoritmo escolhe a folha (ou nó terminal) que contenha a maior variação de perda para crescer, ou seja, a folha mais heterogênea da árvore. Neste sentido, o método foca em melhorar o pior conjunto de divisões sucessivas que caracterizou determinada classe do conjunto de dados. Tal algoritmo se tornou popular ao longo dos últimos tempos por conseguir lidar com grandes quantidades de dados em um tempo de processamento hábil. O crescimento de uma árvore em termos de suas folhas é ilustrado pela Figura 9 e o crescimento de uma árvore em níveis é ilustrado na Figura 10.

2.3 Redes Neurais

Os primeiros resquícios de pesquisa relacionada a redes neurais são datados da década de 40, com os neurônios de McCulloch-Pitts, sendo este considerado o primeiro modelo baseado no cérebro do ser humano. Basicamente, se trata de um modelo linear que reconhece duas categorias diferentes de entrada, fazendo um teste se $f(x, w)$ é positivo ou negativo. Na ocasião, e para que de fato o modelo consiga operar de maneira correta, os pesos eram ajustados manualmente por um humano (MCCULLOCH; PITTS, 1943).

Na década seguinte surgiu o *Perceptron*, um modelo desenvolvido por Rosenblatt,

Figura 9 – Ilustração do crescimento de árvores em termos de suas folhas

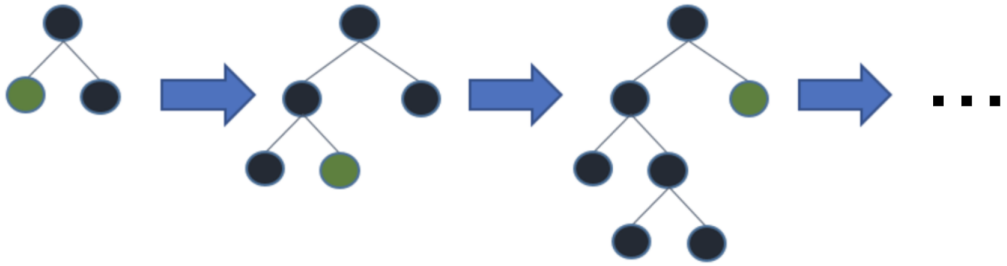


Imagem que define em etapas qual o comportamento do crescimento de uma árvore quando utilizado um algoritmo baseado em árvores que tem como regra o crescimento da árvore em termos de suas folhas. Um exemplo de algoritmo que utiliza esta abordagem é o Random Forest (STATISTICS; BREIMAN, 2001). Imagem disponível em: <https://medium.com/@pushkarmandot/https-medium-com-pushkarmandot-what-is-lightgbm-how-to-implement-it-how-to-fine-tune-the-parameters-60347819b7fc>

Figura 10 – Ilustração do crescimento de uma árvore baseado em níveis

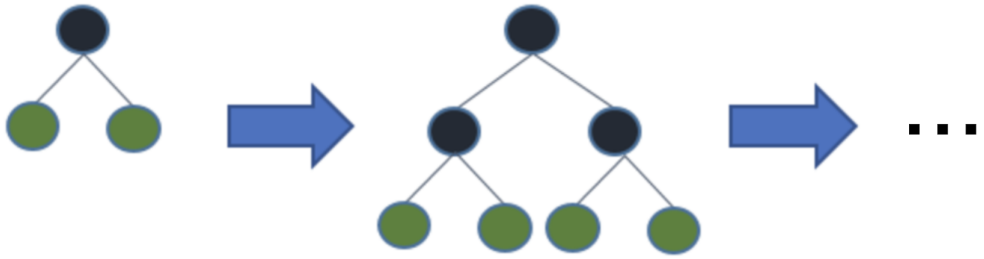
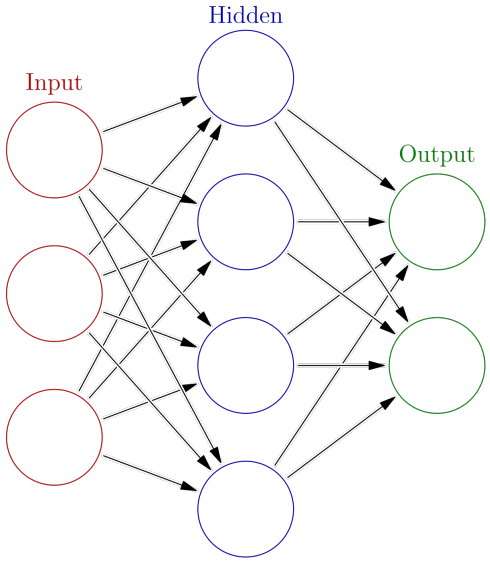


Imagem que define em etapas o comportamento do crescimento de uma árvore quando utilizado um algoritmo baseado em árvores que tem como regra o crescimento da árvore em níveis. Um exemplo de algoritmo que utiliza esta abordagem é o LGBM (KE et al., 2017)

que conseguia aprender os valores dos pesos, ou seja, o modelo possibilitava o ajuste de parâmetros dado exemplos de entrada com suas respectivas categorias (ROSENBLATT, 1958). Isso somado aos trabalhos de Lecum, Hummerhart (LECUN et al., 1989), sobre o algoritmo de *backpropagation* deu origem então a um modelo mais complexo denominado *Multilayer Perceptron*, ou *Perceptron* em multi-camadas, conforme ilustrado na Figura 11. Neste caso, o método exige no mínimo três camadas de neurônios, uma de entrada, uma escondida e uma de saída.

Estes são modelos lineares que serviram de base para a construção de modelos mais complexos que dão possibilidade para resolução de problemas proporcionalmente mais difícies. Um exemplo foi a implementação do Neocognitron, desenvolvido Fukushima (FUKUSHIMA, 1980), onde um modelo de arquitetura mais robusta e complexa, baseado no sistema visual dos mamíferos, foi utilizado para processamento de imagens. Tal método, posteriormente, serviu de base para as tão conhecidas redes neurais convolucionais (LECUN et al., 1998).

Figura 11 – Arquitetura *Multilayer Perceptron* (MLP)



Representação visual da abstração utilizada na arquitetura multilayer perceptron. A camada de input representa os valores dos dados analisados e a camada de output representa qual classe determinado dado pertence. Fonte: Feito pelo autor

Na década de 80, o conceito de representação distribuída ascendeu, com a ideia de que cada dado de entrada de uma rede deveria ser representado por muitas características, e cada uma dessas características deveria ser levada em consideração na representação de outros possíveis dados de entrada (HINTON, 1986). Além disso, devido a popularização do algoritmo de *backpropagation* (LECUN et al., 1989), também ascendia o conceito de propagação do erro através dos neurônios, tendo sucesso nas utilizações em redes neurais profundas, um conceito fundamental para o desenvolvimento do modelo do presente trabalho.

Em suma, para um entedimento profundo sobre o funcionamento de uma rede neural é necessário a definição de alguns conceitos básicos sendo eles o *Forwad Pass*, os pesos, as funções de ativação, e o algoritmo de *backpropagation*, os quais discutimos brevemente a seguir. Além disso, como mencionado acima, para uma estrutura mínima de uma rede neural multi camadas é necessário também três partes, ou camadas: a entrada, a camada escondida e a saída, conforme ilustrado na Figura 11. O algoritmo de redes neurais, portanto, se trata de um algoritmo de classifiação supervisionado, assim como os algoritmos anteriormente mencionados. Na prática, cada nó, ou neurônio do grafo em questão, representa um recipiente que armazena valores pertencentes ao conjunto dos números reais. Especificamente, os neurônios da camada de entrada representam os valores pertencentes a cada dado do conjunto de dados e os valores dos demais neurônios são calculados conforme a execução do algoritmo, assim como cada aresta, ou pesos, que também são recipientes que armazenam números pertencentes ao conjunto dos reais. Ao conjunto de pesos se denomina a letra W que são inicializados aleatoriamente antes da execução do algoritmo.

Com essas duas entidades (valores de entrada e valores dos pesos) já é possível calcular os valores dos demais neurônios. Para que isso seja possível é feita uma somatória ponderada dos valores de entrada com os pesos referentes a cada neurônio da camada escondida, ou seja

$$\sum_{i}^{n} x_i w_i. \tag{6}$$

Por exemplo, vejamos o caso onde temos uma rede neural que possui 10 neurônios em sua camada de entrada, 3 neurônios em sua camada escondida e 1 neurônio em sua camada de saída. Para calcular o valor do primeiro neurônio da camada escondida, multiplica-se os valores de entrada pelos valores dos pesos das arestas que apontam para o neurônio em questão; e assim por diante. Com isso é possível abstrair os valores dos pesos em termos de matrizes. Como existem 10 neurônios de entrada e 3 neurônios na camada escondida uma matriz de 10x3 é capaz de armazenar os valores dos pesos, onde cada coluna representa um neurônio da camada escondida. Para cada cálculo dos valores dos neurônios posteriores existe uma matriz responsável por armazenar os valores dos pesos. No entato, apenas calcular os valores de cada naurônio não é suficiente de forma que para cada neurônio é aplicada uma função de ativação que é escolhida previamente. Existem diversas funções de ativação, sendo elas: a sigmóide, tanh, e ReLu. Finalmente, a última camada da rede neural é responsável pela decisão de classificação do dado em questão, tendo a quatidade de neurônios atrelada à quantidade de classes pertencentes ao conjunto de dados. Aos cálculos dos valores do neurônios até a última camada se dá o nome de *Forward Pass*.

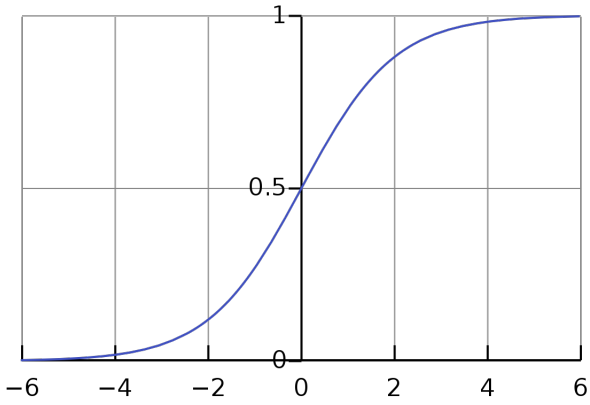
2.3.1 Função de ativação

Como mencionado anteriormente, as funções de ativação fazem parte da etapa *Forward Pass* em um treinamento de uma rede neural. Elas são aplicadas a todos os neurônios da camada escondida e da camada de saída, com o objetivo de normalizar os valores de cada neurônio para um domínio específico, por exemplo o domínio $[0, 1]$. Não neccessariamente deve existir apenas uma função de ativação para todas as camadas, sendo possível escolher quais funções de ativação serão utilizadas em cada camada.

Duas são as funções de ativação mais utilizadas na literatura: a sigmóide, ilustrada pela Figura 12 e a ReLu (*Rectified Linear Unit*), ilustrada pela Figura 13.

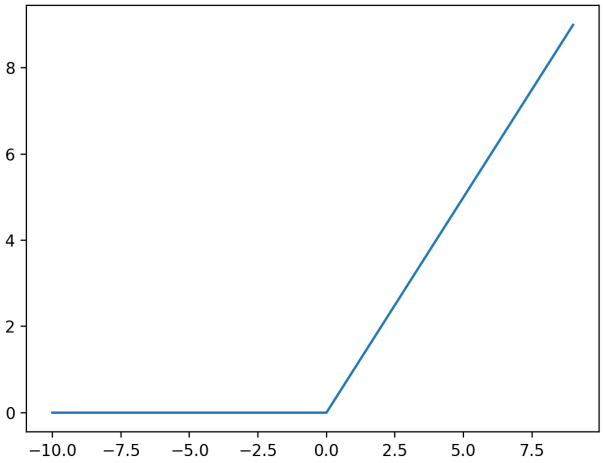
Existem basicamente dois tipos de função de ativação, as lineares e as não-lineares. Em termos gerais, as funções lineares são mais simples e proporcionam uma melhor performance no treinamento da rede, porém, diferente das funções não lineares, são incapazes de aprender estruturas mais complexas do conjunto de dados (GOODFELLOW; BENGIO; COURVILLE, 2016). Por outro lado, o uso de funções não lineares também possui alguns pontos negativos, sendo um deles relacionado a valores muito altos (ou muito baixos) obtidos pela somatória ponderada em cada neurônio. Assim, tomando como exemplo a função sigmóide demonstrada

Figura 12 – Gráfico da função logística



Ilutração da forma da função logística, também conhecida como sigmóide. Fonte: Wikipedia. Disponível em: <https://pt.wikipedia.org/wiki/Função_Logística>

Figura 13 – Gráfico da função de ativacão ReLu



Ilutração da forma da função de ativação ReLu (*Rectified Linear Unit*). Disponível em: <https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/>

pela Figura 12, quando existem valores extremos, no caso da Figura 12 menores que -6 ou maiores que 6, por conta das características da função, tais valores se tornariam 0 ou 1, respectivamente, independentes de sua grandeza. A isso se dá o nome de saturação, algo não interessante quando se trata de treinamento de redes neurais, pois a perda (ou saturação) destes valores dificulta no processo de aprendizagem, e também em como os neurônios se relacionam com os dados de entrada da rede (GOODFELLOW; BENGIO; COURVILLE, 2016).

Os valores presentes em cada neurônio, portanto, são de suma importância, pois é através deles que é decido se o neurônio será ativado ou não no momento da classificação.

2.3.2 *Backpropagation*

Até o momento nós definimos os métodos de funcionamento básico de uma rede neural, além de mostrarmos como são calculadas e armazenadas as estruturas necessárias para o funcionamento do algoritmo. Nesta seção será definido o funcionamento do algoritmo *backpropagation* que é responsável pela atualização dos pesos e valores dos neurônios e, consequentemente, o aprendizado da rede em geral. Existem dois conceitos principais que são fortemente atrelados ao algoritmo em questão: a função de custo e o método de gradiente descendente. A função de custo está presente na maioria dos algoritmos de aprendizado supervisionado, sendo através dela que calculamos o erro atrelado ao modelo. De maneira simples a função custo pode ser definida como

$$C(W) = \frac{1}{N}\sum_{i}^{n}(Y_i - A_i)^2, \tag{7}$$

onde Y representa o vetor da camada de saída da rede, que armazena os valores preditos pela mesma, A representa o vetor que armazena o resultado real de classificaçõa dos dados em questão e W representa conjuntos de pesos da rede. Ao se cálcular a distância entre o resultado real e o resultado ideal obtém-se o erro do modelo, ou o custo do modelo. Idealmente, um modelo com custo 0 (zero) é tido como o modelo perfeito, ou seja, um modelo que consegue generalizar os dados ao ponto de acertar todas as vezes. Naturalmente, tal resultado é a situação ideal de forma que na prática a ideia de um algoritmo de aprendizado é obter um modelo que minimize o erro. No caso de redes neurais, por exemplo, o aprendizado acontence pela otimização, ou minimização da função de custo, sendo que para isso utiliza-se o método do gradiente, já bem estabelecido no campo do cálculo. No ramo do aprendizado de máquina a este método se dá o nome de Gradiente Descendente, definido por

$$\Theta^1 = \Theta^0 - \alpha \nabla J(\Theta). \tag{8}$$

Como o cálculo do gradiente descendente é feito sobre a função de custo, que possui como parâmetro de entrada os pesos da rede, o resultado do gradiente informa para qual direção os pesos devem ser atualizados e com qual intensidade.

O algoritmo de *backpropagation*, por sua vez, utiliza os dados gerados pelo método do gradiente descendente e, de fato, atualiza (ou propaga) os pesos e valores de neurônios que devem ser atualizados. Este processo é feito recursivamente para todas as camadas da rede neural (RUMELHART; HINTON; WILLIAMS, 1988). Existem diversas maneiras de se propagar os ajustes necessários para os pesos. Uma delas é o chamado Gradiente Descendente Estocástico, um método bastante utilizada para o treinamento de uma rede neural. Neste método os pesos são atualizados logo após a etapa de *Forward Pass*. Em outras palavras, para cada dado submetido à rede neural, o custo e a atualização dos pesos são feito logo em seguida. Vale frisar que este é um processo custoso computacionalmente, de forma que técnicas auxiliares podem ser utilizadas para minimizar o problema. Em especial, a técnica denominada

Mini-batch embaralha o conjunto de treinamento e o divide em lotes (ou *batches*) de forma que os pesos e neurônios são atualizados somente depois que todos os dados pertencentes ao lote forem processados pela rede neural (ZHANG et al., 2020).

2.4 Camadas de Convoluções

Antes de definirmos o conceito de camadas de convoluções, primeiro é necessário falarmos de Redes Neurais Convolucionais. As camadas de convoluções, portanto, fazem parte das denominadas redes neurais convolucionais (CNNs). Uma CNN é definida como a junção de uma camada de convoluções, também conhecida como extrator de características, mais a utilização do algoritmo de Redes Neurais, explicado anteriormente. Essa arquitetura é comumente utilizada no reconhecimento de imagens. Sua arquitetura foi baseada em processos biológicos de mamíferos (FUKUSHIMA, 1980).

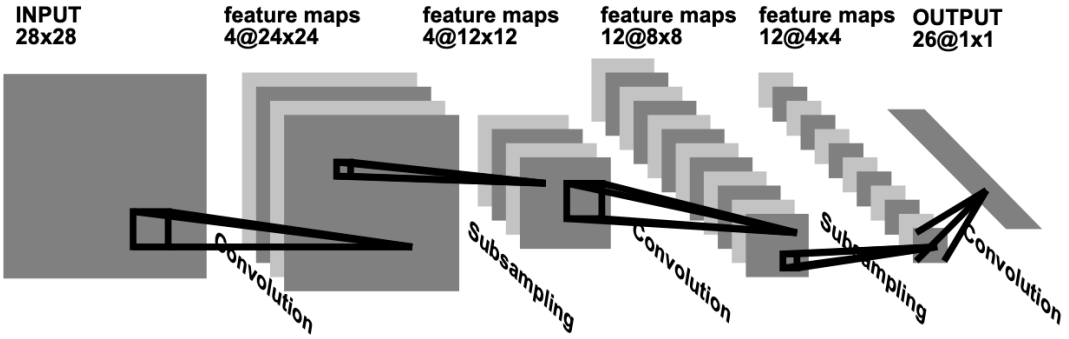
As CNNs são otimizadas e utilizam pouco pré-processamento quando comparada com outros algoritmos de classificação de imagens. Isso significa que a rede consegue aprender de forma automatizada filtros, ou características da imagem processada, que em outros algoritmos são feitos à mão. Tais filtros são capazes de extrair características elementares das imagens, como bordas e limites. Somado a isso, se essas características detectadas têm grande importância em uma parte da imagem, é provável que essa importância se estenda por toda a imagem (RUMELHART; HINTON; WILLIAMS, 1986). As saídas dos neurônios responsáveis por essas extrações são denominadas mapa de características. Em cada posição da imagem existem diferentes tipos de unidades e diferentes tipos de mapa de características que computam diferentes tipos de características. Uma possível implementação para esse processo seria para cada mapa de característica, escanear a imagem de entrada com um único neurônio e, posteriormente, armazenar os estados deste neurônio na posição correspondente do mapa de características. Essa operação é denominada convolução (LECUN; BENGIO, 1998), e o conjunto dessas convoluções é denominado camada de convoluções.

A camada de convoluções é normalmente composta por diversos mapas de características contendo diferentes vetores de pesos, para que diversas características possam ser extraídas de cada posição (LECUN; BENGIO, 1998). A Figura 14 demonstra o *pipeline* de processamento e utilização dos mapas de características em cada momento de processamento das camadas de convoluções.

Porém, é importante ressaltar que as camadas de convoluções também podem ser utilizadas com outros algoritmos de classificação, não se limitando apenas à Rede Neural. A utilização de outros algoritmos de classificação faz parte da proposta deste trabalho.

Especificamente falando sobre as operações realizadas pela de camada de convoluções pode-se citar os filtros e as camadas de *pooling*.. Os filtros são baseados nas operações provenientes da computação gráfica responsáveis pela identificação de regiões e características

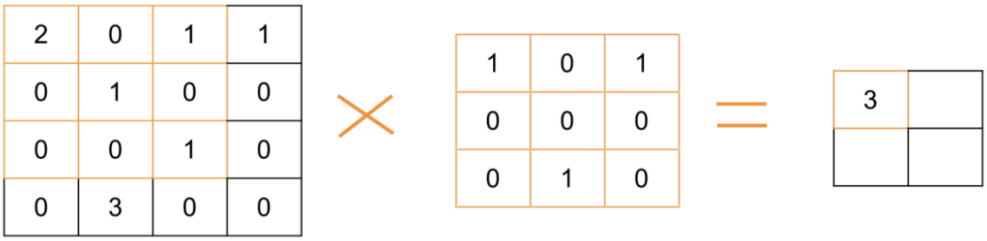
Figura 14 – Pipeline de processamento das camadas de convoluções de uma CNN



Representação visual de como o dado é tratado pelas convoluções da CNN. Cada região destacada representa uma operação de filtro, reduzindo os valores da sub-matriz destacada à apenas um valor. Fonte: (LECUN; BENGIO, 1998)

relevantes do dado em questão, como por exemplo a identificação de bordas e limites de uma determinada imagem. Já as operações de *pooling* são responsáveis por diminuir a dimensionalidade da estrutra de mapa de características gerada pelas operações de filtros. Matematicamente, uma operação de filtro pode ser representada por matriz quadrada nxn como mostrado na Figura 15. Esta matriz quadrada é então sobreposta a uma região específica da imagem e os valores dos pixels, tanto da imagem quanto do filtro, são utilizados para o cálculo.

Figura 15 – Exemplo de uma operação de filtro.



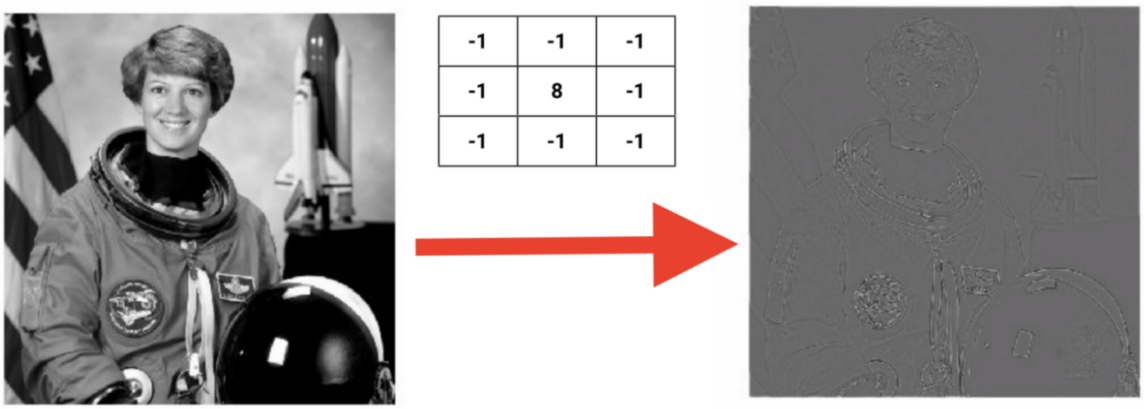
Representação visual de como uma operação de filtro é realizada. Disponível em: <https://towardsdatascience.com/a-beginners-guide-to-convolutional-neural-networks-cnns-14649dbddce8>

O cálculo representado pela Figura 15 é representado pela Equação 9, onde pf representa o valor do pixel do filtro e pi representa o valor do pixel da imagem. Por exemplo, o valore mostrado pela Figura 15 é calculado como: $(2x1) + (0x0) + (1x1) + (0x0) + (1x0) + (0x0) + (0x0) + (0x1) + (1x0) = 3$

$$\sum pfpi \tag{9}$$

A titulo de visualização a aplicação de um determinado filtro a uma imagem gera uma nova imagem, que também pode ser representada por uma matriz com outros valores. A Figura 16

Figura 16 – Exemplo de uma operação de filtro aplicado a uma imagem.



Representação visual de como uma operação de filtro identifica regiões relevante numa imagem. Disponível em: <https://www.cs.columbia.edu/education/courses/course/COMSW4995-7/26050/>

De modo geral, ao se utilizar camada de convoluções juntamente com algoritmos de classificação para o reconhecimento de imagens, a necessidade de um extrator de características feito à mão é abolida. Os pesos compartilhados e as sub amostras que são geradas ao passar das convoluções causa uma invariância em relação a pequenas transformações geométricas ou distorções (LECUN; BENGIO, 1998).

2.4.1 Classificação de imagens

A classificação de imagens, portanto, leva em consideração dois conceitos previamente explicados, a utilização de modelos de aprendizado de máquina para classificação e a utilização da camada de convolução para extração de características da imagem, que é utilizada como entrada no algortimo de classificação utilizado. De forma geral, uma CNN se utiliza do algoritmo de Redes Neurais para a camada de classificação, porém como proposta do trabalho, nós não nos limitamos a apenas a este algoritmo, assim utilizamos outros algoritmos de classificação alimentados pelo resultado da camada de convoluções.

Neste processo, características de imagens são extraídas para que um algoritmo de classificação receba como entrada o mapa de características da imagem já processada e retorne como saída o grupo ao qual a imagem pertence. Observa-se que a teoria que embasa esse assunto é relativamente recente, juntamente com os modelos que obtiveram resultados satisfatórios utilizando tal arquitetura (LECUN; BENGIO, 1998). Justamente por esse fato que o número de trabalhos relacionados é alto, além de apresentarem resultados relevantes para a melhora da área de classificação de imagens. Como exemplos podemos citar os trabalhos de

Han e Lee que utilizaram placas gráficas e multithreads para acelerar a performance de tempo de execução do treinamento de uma CNN (HAN, 2017), sendo isso relevante pois para um bom ajuste de parâmetros é necessária uma grande quantidade de amostras de dados, algo que influencia direta e proporcionalmente no tempo de execução do treinamento. De fato, quanto mais dados, maior o tempo de execução (GOODFELLOW; BENGIO; COURVILLE, 2016). Como um segundo exemplo, temos também os trabalhos de (ZHANG, 2017) que, utilizando CNN, desenvolveram um método de classificação de imagens de altíssima resolução provenientes de satélites, obtendo resultados satisfatórios quando comparados com métodos de aprendizado supervisionado de classificação (ZHANG, 2017). Finalmente, os trabalhos de Lee e Kwon, abordam a questão teórica conceitual sobre o assunto, propondo um novo modelo de rede neural convolucional que leva em consideração características tanto espaciais quanto espectrais de imagens hiperespectrais (KWON, 2016).

2.5 Análise Sentimental

Quando se fala sobre análise sentimental, obrigatoriamente fala-se também sobre linguagem, emoções e opiniões sobre determinado tópico. Assim, contextualizando tal assunto com as tecnologias da atualidade, como por exemplo a ascensão das mídias sociais e da inteligência artificial, a quantidade de opiniões e emoções referentes a isso é alta. Isso evidencia a importância da análise sentimental em documentos de texto, além de auxiliar na tomada de decisões (LIU, 2007). Com isso, observa-se que sistemas que utilizam dessa análise encontraram suas aplicações em quase todo domínio social e de negócio (LIU, 2012). Tem-se como exemplo o trabalho de O'Connor que analisou a base de dados em texto do Twitter e encontrou uma ligação entre os sentimentos dos tweets e resultados de enquetes públicas (O'CONNOR et al., 2010).

Existem diversas abordagens para analisar um documento de texto com relação ao seu sentimento. A primeira delas é a nível de documento de texto, i.e. um conjunto de sentenças que possui um sentimento, como por exemplo positivo ou negativo (HASSAN; QAZVINIAN; RADEV, 2010). Neste caso, partimos do princípio que cada documento expressa uma opinião única a um determinado assunto específico, também chamado de entidade, não sendo aplicável a documentos que expressem opiniões sobre múltiplas entidades (LIU, 2012). Outra abordagem é a nível de sentença, ao qual a tarefa de analisar o sentimento é específica por cada sentença do documento de texto, determinando se elas são positivas, negativas ou neutras. A este nível, a análise está intimamente relacionada à uma análise subjetiva do documento, sendo possível distinguir se uma sentença está relacionada a visões objetivas e fatuais sobre determinado assunto, ou opiniões e pontos de vista pessoais sobre o assunto (LIU, 2012). Finalmente, existe também a abordagem a nível de entidade, esta sendo a análise mais minuciosa. Ao invés de analisar os construtores da linguagem, como por exemplo parágrafos e sentenças, este nível leva em consideração a opinião diretamente, se baseando na ideia que uma opinião consiste de

um sentimento e um alvo para o sentimento. O objetivo desse nível é descobrir sentimento em entidades e em seus aspectos. Para ilustrar, na sentença "a qualidade de chamada do meu celular é boa, porém sua bateria dura pouco."existem dois aspectos distintos, a qualidade da chamada (positivo) e a duração da bateria (negativo). Baseado nesse nível de análise, uma estrutura de opiniões sobre entidade e seus aspectos é formada, que torna os textos não estruturados em dados estruturados que posteriormente poderão ser utilizados para diversos tipos de análises qualitativas e quantitativas (LIU, 2012).

2.5.1 Classificação de sentimento em sentença

A nível de algoritmo em aprendizado supervisionado para classificação, uma sentença e um documento podem ser considerados a mesma coisa. Para uma questão de ordem, assume-se que uma sentença expressa um único sentimento proveniente de uma única opinião. Por exemplo em (HASSAN; QAZVINIAN; RADEV, 2010), um método foi proposto para identificar atitudes sobre participantes em uma discussão online. Deste modo, um modelos baseado em aprendizado supervisionado foi desenvolvido para identificar se as atitudes dos participantes de determinada discussão eram, no geral, positivas ou negativas.

3 Processamento de linguagem natural

3.1 Conceito

Como o objeto de estudo do trabalho permeia as bases da comunicação através da linguagem, a necessidade de estudar métodos de processamento de linguagem natural é impreterível. Um dos fatores mais interessante sobre isso é a necessidade de junção de duas grandes áreas, computação e linguística, que antigamente eram consideradas imiscíveis. Estudar e compreender as estruturas da ciência da linguagem é a chave para a criação de modelos computacionais que sejam capazes de modelar a comunicação em termos de estruturas inteligíveis pela máquina, tornando assim possível o entendimento artificial de conversações e escritas. Grande parte disso também deve levar em consideração a parte cognitiva de como os seres humanos adquirem, produzem e entendem a linguagem. Para que tudo isso seja levado em consideração, estudos propuseram que existem regras que são utilizadas para estruturar expressões linguísticas (MANNING; SCHüTZE, 1999).

Existe um problema intrínseco com o conceito de estruturação linguística, justamente pelo fato da comunicação não se restringir apenas ao texto falado ou escrito, e sim por uma união disso com expressões cognitivas e emocionais intrínsecas do ser humano. De acordo com (SAPIR, 1921), não é possível fornecer uma caracterização exata das formas de linguagem, mas é possível encontrar determinados padrões nas regras sintáticas da linguagem que ajudam na representação computacional final. Como exemplo, é uma regra que uma sentença no português normalmente é constituída de um sujeito e um predicado, e estes por suas vezes são constituídos por outros morfemas. Isso facilita a identificação de padrões e, por consequência, melhoram a caracterização da linguagem (MANNING; SCHüTZE, 1999).

A análise sintática da língua sozinha não sustenta uma boa caracterização computacional da linguagem, de forma que modelos estatísticos são utilizados (MANNING; SCHüTZE, 1999). Partindo deste princípio, existe um forte argumento de que a probabilidade faz parte do entendimento científico da linguagem, a cognição humana sendo probabilística e, consequentemente, a linguagem também. Com isso a teoria da probabilidade se torna a parte central da teoria da linguagem (MANNING; SCHüTZE, 1999). Para sustentar o argumento probabilístico da cognição, tem-se o fato que de a sociedade vive num mundo repleto de incertezas e informações incompletas e, portanto, para que se viva nesse mundo é necessário levar isso em consideração (MANNING; SCHüTZE, 1999).

Processar as palavras formando uma ideia com significado geral da sentença e ponderar sobre isso para se tomar uma decisão é justamente a linha de raciocínio geral utilizada pelos seres humanos para a compreensão de determinados fenômenos. Em outras palavras, os processo

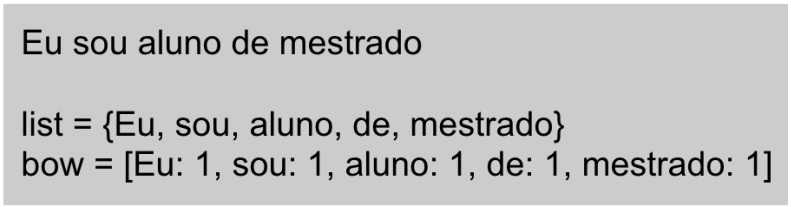
cognitivos utilizados pela linguagem são muito similares àqueles utilizados para processar outras formas de conhecimento. Esses processos cognitivos então são melhores formalizados como processos probabilísticos que são capazes de lidar com incertezas e informações incompletas (MANNING; SCHüTZE, 1999). Aliado a isso, diversas abordagens e métodos foram propostas e obtiveram resultados satisfatórios em relação às suas representações, e serão abordados nas subseções a seguir.

3.2 Métodos de representação

3.2.1 Bag of Words

O modelo de *bag of words* (BoW), ou saco de palavras, é o modelo mais simples utilizado para a representação de sentenças em NLP. Com ele cada sentença é representada por um vetor de tamanho n, com n sendo a quantidade de palavras distintas presentes no corpo do texto a ser analisado. Com isso, infere-se que cada posição do vetor representa uma palavra distinta, e o conteúdo de cada posição é preenchida com a quantidade de ocorrência da palavra em questão na sentença (HARRIS, 1954).

Figura 17 – *Bag of Words (BOW)*



A imagem resume de forma direta qual o funcionamento do método BOW, ou saco de palavras em tradução direta, nele cada palavra única do conjunto de dados é separada em um vetor de tamanho N, onde cada posição representa uma palavra. Cada frase do conjunto é definida como vetor esparço de tamanho N, onde cada posição possui o valor referente a quantidade de palavras presente na frase que a posição em questão representa. Fonte: Feito pelo autor.

A Figura 17 demonstra como uma sentença é representada em forma de vetor. As posições preenchidas com 0 são omitidas, tornando o vetor esparso para economia de memória e processamento. Esse modelo não leva em consideração a posição das palavras, porém apresenta resultados satisfatórios em classificação de documentos de texto, como visto em (KO, 2012). Uma alternativa para melhorar a representação das sentenças é levar em consideração não só palavras isoladas mas sim conjuntos de palavras. Por exemplo, a expressão "Eu sou"tomaria uma nova posição no vetor de representação, aumentando a complexidade do modelo e, possivelmente, diminuindo a distância do resultado esperado.

3.2.2 Word2Vec

3.2.2.1 *One Hot Encoding*

Quando trata-se análise de dados recai-se recorrentemente em suas representações numéricas. Muitos conjuntos de dados possuem características que são denominadas categóricas, por exemplo, levando em consideração a Tabela 2 que contém um subconjunto do conjunto de dados Weather, as possibilidades relacionadas a característica 'Temperatura' são: quente, frio e ameno. Assim, para que esse dado seja utilizável em análises é necessária sua conversão em formato númerico.

Para dados de textos essa conversão é também necessária de forma que o método Word2Vec utiliza um tipo específico de conversão de texto para número, denominada *One-hot-encoding*, ou vetores binários de palavras (ZHENG; CASARI, 2018). O método *One-hot-encoding* funciona da seguinte maneira. Dado um conjunto de palavras qualquer, é definido um vocabulário V e tamanho Tv contendo todas as ocorrências únicas de palavras em ordem alfabética. Dessa maneira, cada palavra contida no conjunto de palavras está mapeada e possui uma posição específica definida. Para cada palavra atribui-se um vetor binário específico de tamanho fixo Tv onde na posição da palavra em questão coloca-se o valor 1 e nas demais posições coloca-se o valor 0. Com isso, cada palavra possui uma representação numérica binária e única. Exemplificando, dada a frase: 'Tenha um ótimo dia', é possível criar um vocabulário $V =$ (dia, ótimo, tenha, um) de tamanho $Tv = 4$. Com a utilização do método *one-hot-encoding* a frase em questão será convertida para a seguinte matriz: $[[0, 0, 1, 0], [0, 0, 0, 1], [0, 1, 0, 0], [1, 0, 0, 0]]$ de maneira que obtém-se uma representação númerica para a frase em questão.

3.2.2.2 Definição

Word2Vec é um método para representação de palavras que utiliza uma rede neural relativamente simples, com apenas duas camadas que é treinada para reconstruir contextos linguísticos e semânticos de palavras (MIKOLOV et al., 2013). O método foi desenvolvido por uma equipe do Google, liderada por Mikolov. Como dado de entrada recebe um corpo de texto e produz um vetor de centenas de dimensões que representa cada palavra no hiperespaço. Esses vetores possuem posições no hiperespaço de tal maneira que palavras que compartilham do mesmo contexto estão mais próximas umas das outras (MIKOLOV et al., 2013).

A Figura 18 apresenta graficamente com dimensionalidade reduzida a questão do posicionamento das palavras no espaço de acordo com suas características contextuais. O Word2Vec possui duas abordagens principais para a geração dos vetores que representam as palavras: *Continuos Bag of Words* (CBOW) e *Skip-gram*. Na arquitetura CBOW o modelo gera o vetor de representação de acordo com as palavras que estão ao redor das palavras a serem analisadas, ao contrário da arquitetura *skip-gram* que utiliza as palavras a serem analisadas para gerar representações das palavras que estão ao redor (MIKOLOV et al., 2013). A abordagem

Figura 18 – Posição semantica das palavras

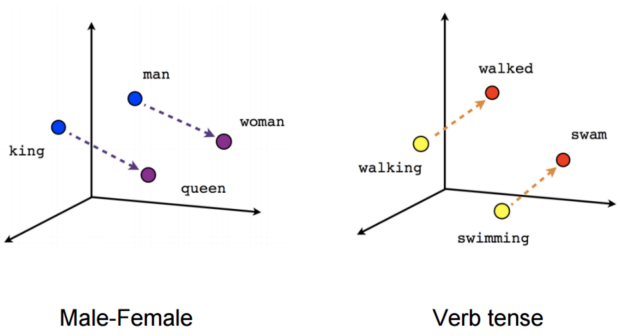


Figura retirada do artigo que deu origem ao método Word2Vec. Nela é ilutrado o funcionamento da representação das palavras geradas ao fim do treinamento do modelo. Nela é possível observar, em dimensão reduzida, que palavras que possuem características semânticas semelhantes estão localizadas geometricamente em uma mesma região. Fonte: (MIKOLOV et al., 2013)

CBOW é mais rápida, porém o *skip-gram* apresenta melhor performance quando se trata de palavras que não aparecem com frequência no corpo do texto (MIKOLOV et al., 2013). O método, portanto, tem como objetivo principal gerar um modelo de vetorização de palavras que leve em consideração características semânticas e sintáticas de determinado conjunto de dados textual (MIKOLOV et al., 2013).

Como definição mais detalhada do método CBOW utiliza-se todas as frases do conjunto de dados utilizado para gerar o vocabulário e posteriormente o modelo de vetorização de palavras. A característica mais importante deste método está na arquitetura de redes neurais que é utilizada para possibilitar o aprendizado de uma matriz de pesos, genérica o suficiente para representar as palavras pertencentes ao conjunto de dados, levando em consideração contexto e características semânticas. É importante frisar que antes da execução do algoritmo de treinamento do modelo, todo corpo de texto em questão é convertido em número pelo método *one hot encoding*, assim todas as palavras do vocabuário passam a possuir uma representação única binária que então será utilizada pelo algoritmo.

No momento da etapa de treinamento escolhe-se uma janela de palavras, representada pela Figura 19, através de um número N e estas são submetidas à rede neural. O objetivo da escolha de uma janela de palavras se dá pela necessidade de criar relações entre todas as palavras do conjunto de dados, visto que o conjuntos dados de texto em geral, quando utilizados para geração de modelos de vetorizaçào de palavras, não possuem anotação específica para cada palavra, tornando assim o modelo Word2Vec um algoritmo não supervisionado. Assim, para se criar uma referência verdadeira na etapa de treinamento constrói-se a tupla (contexto, alvo) onde o alvo é utilizado pela rede neural como verdade e é usado como referência pelo algoritmo de *backpropagation*.

O objetivo da rede, portanto, é predizer qual a palavra que mais se relaciona com o

Figura 19 – Definição do funcionamento do parâmetro 'janela de palavras'

A figura ilustra o funcionamento de como cada palavra do conjunto de dados é anotada para que seja levado em consideraçào o contexto na etapa de treinamento da rede neural. Fonte: Feito pelo autor

conjunto de palavras submetido a ela, e através do algoritmo de *backpropagation* determinar os valores da matriz de peso. A janela de palavras é deslocada até o fim de todo corpo do texto do conjunto de dados. A quantidade de épocas necessárias, taxa de aprendizagem e, principalmente, o tamanho da estrutura da camada escondida, são definidos antes da execução do algoritmo. A quantidade de neurônios da camada escondida é importante pois representa o tamanho do vetor que representa cada palavra.

A arquitetura da rede utilizada para o método pode ser visualizada pela Figura 20. Tanto a entrada quanto a saída da rede possuem o mesmo tamanho, e este tamanho é justamente o tamanho Tv do vocabulário gerado previamente. Assim, se o tamanho da entrada é Tv e o tamanho da camada escondida é Th, a matriz de pesos $W0$ terá dimensões de $Tv.Th$. Os valores dos neurônios da camada escondida H, portanto, são calculados através da multiplicação do vetor de entrada pela matriz de pesos, ou seja,

$$H = \begin{bmatrix} x0 \\ x1 \\ ... \\ xn \end{bmatrix}^T . \begin{bmatrix} w00 & w01 & ... & w0n \\ w10 & w11 & ... & w1n \\ ... & ... & ... & ... \\ wm0 & wm1 & ... & wmn \end{bmatrix}. \tag{10}$$

Vale frisar que a matriz de peso é iniciada com valores aleatórios no início da execução do algoritmo e estes são atualizados no decorrer das iterações.

Como o vetor de saída tem o mesmo tamanho do vetor de entrada, para que os valores de saída sejam calculados a matriz de pesos $W1$ existente entre essas duas camadas possuirá dimensões $Th.Tv$, o oposto da matriz de pesos antecedente, e o resultado da camada de saída é calculado pela multiplicação dos valores dos neurônios da camada escondida H pela matriz

Figura 20 – Arquitetura de rede usado no método CBOW

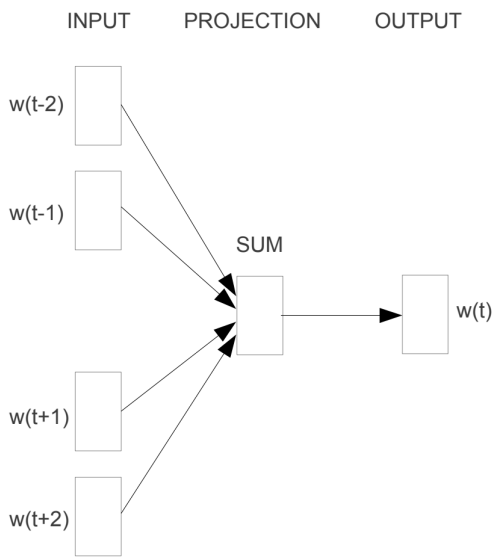


Figura retirada do artigo que deu origem ao método Word2Vec. Nela é ilutrada a arquitetura utilizada pelo método CBOW. É possível verificar que para o método são submetidas diversas palavras como entrada e apenas uma palavra é gerada como saída. Fonte: (MIKOLOV et al., 2013)

de peso $W1$. A camada de saída é concebida como uma camada de *softmax*, que nada mais é do que uma camada padrão de uma rede neural que possui como função de ativação a *softmax*, definida pela Equação 11, que transforma um vetor de K valores reais em um vetor de K valores reais que somados são iguais a 1. Cada valor, portanto, pode ser interpretado como uma probabilidade. Como cada posição da camada de saída representa uma palavra distinta do conjunto de dados, a posição com maior probabilidade deve seguir a regra criada pela janela de palavras (contexto, alvo).

$$\sigma(\vec{X})_i = \frac{e^{Xi}}{\sum_{j=1}^{K} e^{Xj}} \tag{11}$$

Como no método CBOW são submetidas à rede um conjunto de palavras como entrada, os valores dos neurônios da camada escondida são gerados a partir da média dos valores obtidos pela multiplicação dos N vetores de entrada pela matriz peso $W0$. Intuitivamente, cada linha da matriz de pesos $W0$ representa uma palavra distinta e os valores das colunas são os vetores de palavras obtidos.

Pode-se dizer que o método *Skip-gram* é o exato oposto do método CBOW quando comparado em termos da arquitetura de rede neural utilizada e lógica de implementação para geração da matriz de representação. Enquanto o método CBOW tem como objetivo prever qual

a palavra pertencente à um contexto específico, o método *Skip-gram* tem como objetivo prever quais são as palavras do contexto dado uma palavra específica. Tal método também necessita dos dados convertidos em números pelo método *one hot encoding* e seu funcionamento pode ser melhor visualizado pela Figura 21.

Figura 21 – Arquitetura de rede usado no método Skip-gram

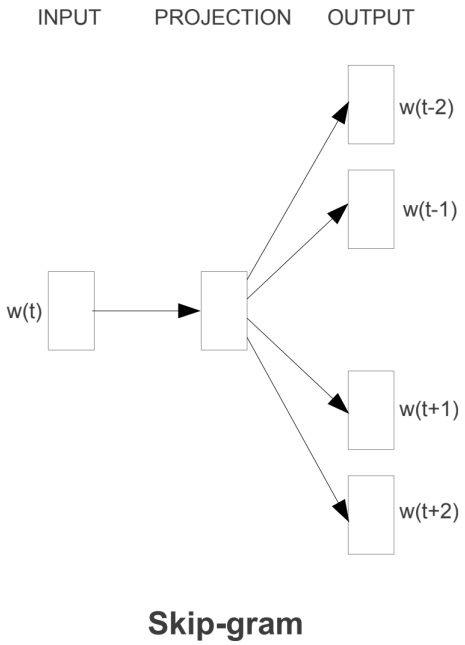


Figura retirada do artigo que deu origem ao método Word2Vec. Nela é ilutrada a arquitetura utilizada pelo método Skip-gram. É possível verificar que para o método é submetida apebas uma palavra como entrada e como saída é conjunto de palavras que represeta o contexto da palavra em questão. Fonte: (MIKOLOV et al., 2013)

É importante ressaltar que o método, no geral, constrói uma representação vetorial apenas para palavras únicas, levando em consideração o conjunto a qual cada palavra pertence como contexto. Para representação de frases inteiras utiliza-se o método Doc2Vec (LE; MIKOLOV, 2014), explicado na seção seguinte.

3.2.3 Doc2Vec

O método Doc2Vec, como o próprio nome sugere, é uma derivação do método Word2Vec, inclusive sendo construído também por Mikolov autor de ambos os métodos portanto. O método em questão utiliza das mesmas premissas das arquitetura das redes neurais utilizadas pelo Word2Vec, tendo como diferencial a utilização da uma estrutura de dados, além das estruturas responsáveis por identificar cada palavra. Aqui, a estrutura de dados é responsável por identificar cada frase única do conjunto de dados, denominada Identificador do Parágrafo. Sua utilização pode ser melhor visualizada pela Figura 22.

Figura 22 – Arquitetura de rede usado no método DM

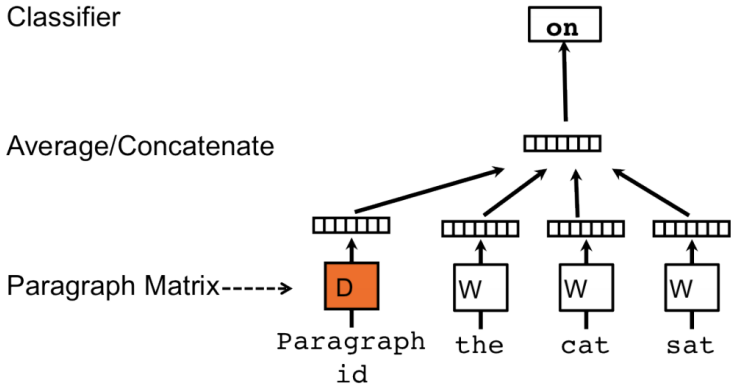


Figura retirada do artigo originário do método Doc2Vec (LE; MIKOLOV, 2014). Nela é possível visualizar a utilização do Identificador de parágrafo para a abordagem DM. Fonte: (LE; MIKOLOV, 2014)

Existem também duas abordagens distintas para a geração do modelo, sendo elas a DM (*Distributed Memory*) e a DBOW (*Distributed Bag of Words*). A primeira segue as mesmas premissas da abordagem CBOW do método Word2Vec, e a segunda segue as mesmas premissas da abordagem *Skip-Gram* (LE; MIKOLOV, 2014). Para o método DBOW, em contraposição com método *Skip-Gram*, ao invés de se utilizar a estrutura que define determinada palavra, é utilizado o Identificador de Parágrafo e, a partir disso, a rede tenta predizer quais as palavras relacionadas com o Identificador de Parágrafos em questão (LE; MIKOLOV, 2014). Tal arquitetura pode ser melhor visualizada pela Figura 23

Figura 23 – Arquitetura de rede usado no método DBOW

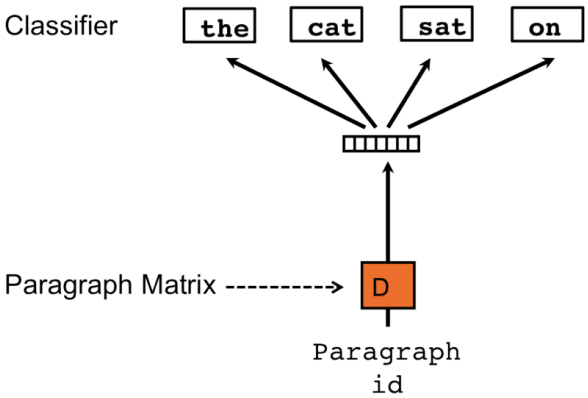


Figura retirada do artigo originário do método Doc2Vec (LE; MIKOLOV, 2014). Nela é possível visualizar a utilização do Identificador de parágrafo para a abordagem DBOW. Fonte: (LE; MIKOLOV, 2014)

4 Desenvolvimento

4.1 Implementação

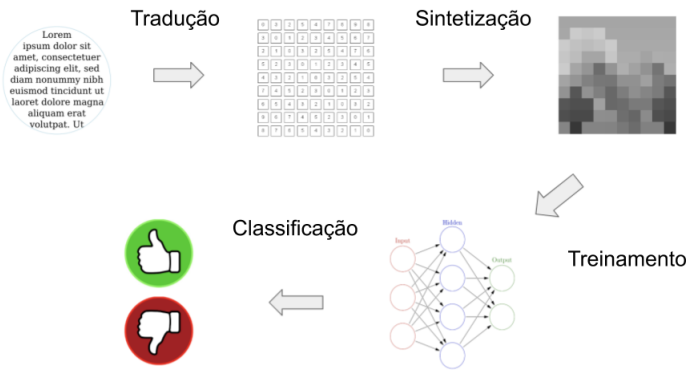
4.1.1 Linguagem

Para um melhor entendimento da implementação feita do objeto de estudo do trabalho é necessário especificar quais tecnologias foram utilizadas, assim como os métodos e técnicas que permitiram a sintetização do resultado final, descritos em tabelas e gráficos.

A base para qualquer desenvolvimento de um sistema computacional se dá pela linguagem que será utilizada. No ramo do aprendizado de máquina algumas linguagens se destacam, tanto pela sua performance e facilidade de manipulação de dados, quanto pela variedade de bibliotecas que facilitam o desenvolvimento de modelos inteligentes. Como exemplo, destacam-se as linguagens R, Matlab e Python. Assim, a linguagem utilizada para a implementação do trabalho foi o Python (RASCHKA, 2015).

O sistema proposto, como um todo, pode ser dividido em sub-sistemas, onde cada sub-sistema é definido por técnicas computacionais específicas, assim como a utilização de bibliotecas específicas que juntas definem o comportamento final do sub-sistema e garantem a compatibilidade entre eles. Por uma questão de organização e facilidade de entendimento foi proposto pelo autor a nomeação de cada sub-sistema evidenciado pela Figura 24 abaixo.

Figura 24 – Pipeline de desenvolvimento nomeado



A figura ilustra todas as etapas necessárias para a implementação do projeto identificando desde o texto cru do conjunto de dados até a classe deste em sentimento positivo ou negativo. As etapas foram criadas para facilitar a explicação do desenvolvimento. As etapas são: Tradução, Sintetização, Treinamento e Classificação. Fonte: Feito pelo autor

4.2 Sub-sistemas

4.2.1 Tradução

A etapa de tradução tem como principal característica a utilização de técnicas de processamento de linguagem natural, mais especificamente a utilização do método word2vec para a representação de frases num espaço N-dimensional levando em consideração o contexto, agrupando frases e palavras por suas similaridades inferidas a partir do modelo proposto por (MIKOLOV et al., 2013). Mais especificamente, utilizou-se uma variação do Word2Vec, denominada Doc2Vec para a representação de frases como um todo, e não de palavra por palavra como pensado incialmente na implementação do Word2Vec (MIKOLOV et al., 2013) (LE; MIKOLOV, 2014). Assim, como feito por (LE; MIKOLOV, 2014) a análise sentimental no conjunto de dados de críticas de cinema IMDB, assim como o conjunto de dados SST2, possuem uma melhor performance quando se comparado à utilização do *Bag of Words* (BOW) (HARRIS, 1954) para representação de frases. É justamente sob esta ótica que o motivo e objetivo do trabalho se sustenta.

O modelo, portanto, gerado pelo método Doc2Vec, espera como entrada uma frase ou um parágrafo em formato inteligível pelo ser humano (linguagem natural) e produz um vetor de dimensão N, onde N é fornecido como parâmetro de entrada antes do treinamento do modelo. Se nenhum parâmetro de entrada for fornecido, valores padrões serão utilizados (LE; MIKOLOV, 2014). Para a implementação foram gerados modelos que produziam vetores de $N = 2304$ posições, que posteriormente foram redimensionadas para matrizes de 48x48, que será utilizada no próximo sub-sistema. O redimensionamento da estrutura gerada pelo modelo Doc2Vec é necessário pois a matriz será posteriormente transfomada em uma imagem para a geração do modelo de classificação.

Em termos de linguagem computacional foi utilizado a implementação do modelo Doc2Vec da biblioteca *gensim* para Python (REHUREK, 2011).

4.2.2 Sintetização

A etapa de sintetização caracteriza o diferencial do trabalho, é justamente nesse momento que ocorre a transformação da matriz gerada pelo modelo doc2vec em uma foto em escala de cinza, que posteriomente será utilizada como conjunto de treinamento na etapa seguinte. Para a geração da foto foram utilizados os métodos das bibliotecas PIL (*Python Image Library*) e Pillow para a linguagem Python.

A matriz gerada pelo modelo possui um padrão específico que foge do padrão utilizado por imagens em escala de cinza. Basicamente, imagens em escala de cinza são definidas por matrizes quadradas com valores entre 0 e 255 em cada posição, onde 0 representa a cor preta e 255 representa a cor branca, e os números entre esse limite uma escala de cinza proporcional

ao seu valor (FOLEY et al., 1990).

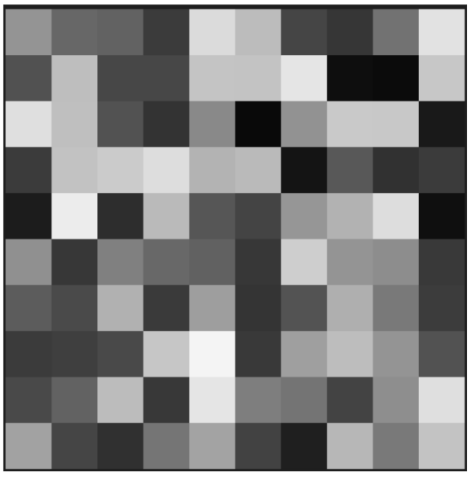
Para o ajuste dessa incongruência, as matrizes geradas foram todas normalizadas pelo método de normalização min-max (HAYKIN, 2009), definido pela seguinte fórmula:

$$z = \frac{x_i - min(x)}{max(x) - min(x)}, \tag{12}$$

onde x representa a matriz e x_i cada posição da matriz, representada no caso por um pixel.

A Figura 25 abaixo foi gerada utilizando o método proposto, e demonstra as características da imagem produzida e qual frase ela representa, classificada com polaridade positiva. Para a geração da imagem foram aplicadas técnicas de pré-processamento de texto para que informações irrelevantes fossem eliminadas, porém o texto apresentado na figura não possui nenhum pré-processamento para um melhor entendimento da ideia.

Figura 25 – Representação visual



For a movie that gets no respect there sure are a lot of memorable quotes listed for this gem. Imagine a movie where Joe Piscopo is actually funny! Maureen Stapleton is a scene stealer. The Moroni character is an absolute scream. Watch for Alan 'The Skipper' Hale jr. as a police Sgt.

A figura ilustra a representação visual do vetor de palavras gerado pelo método Doc2Vec de uma frase específica do conjunto de dados. Todas a frases do conjunto de dados possui uma representação desta natureza. A partir destes vetores de palavras que o modelo de binário de classificação de sentimento é gerado. A representação é possível pois o vetor de palavras possui em cada posição um número cujo domínio é [0, 1], portanto cada posição do vetor poder representada como um pixel e cada valor pode ser representado como uma cor na escala de cinza, onde 0 seria preto e 1 seria braco. Fonte: Feito pelo autor

4.2.3 Treinamento

Na etapa de treinamento foram utilizados diversos algoritmos de aprendizado supervisionado com o intuito de enriquecer os resultados obtidos e possibiltar uma análise mais

detalhada sobre o método proposto. Para isso foram utilizados os métodos: Regressão Logística, *Naive Bayes*, LGBM e Redes Neurais MLP. Nesta etapa, também, é onde os parâmetros das camadas de convolução (CC) serão otimizados de acordo com as imagens contidas no conjunto de treinamento que será utilizado. Como o algoritmo utilizado para o treinamento do modelo é supervisionado, os dados do conjunto de treinamento necessitam estar nomeados (HAYKIN, 2009). Desse modo, o algoritmo utilizado encontra o melhor conjunto de parâmetros, definindo um hiperplano que é utilizado como região de decisão para a classificação generalizada de imagens submetidas ao modelo (GOODFELLOW; BENGIO; COURVILLE, 2016).

A Figura 26 demonstra, em dimensionalidade reduzida, a utilização de um hiperplano para a classificação de dados, apenas para ilustração e compreensão do funcionamento de um classificador, onde cada parâmetro pode ser concebido como uma dimensão diferente. Além

Figura 26 – Hiperplano em dimensão reduzida

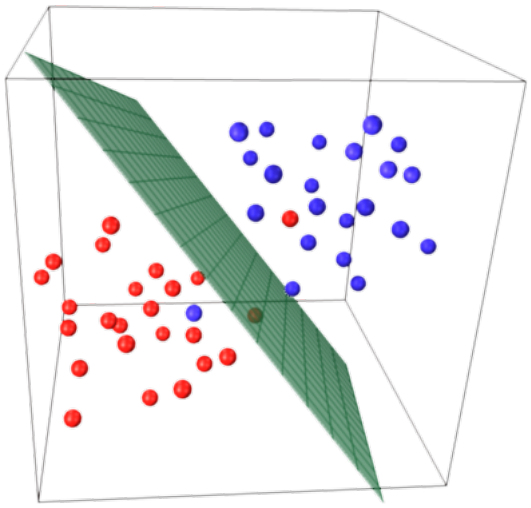


Ilustração de um hiperplano em dimensão reduzida sendo utilizado como fator de classificação de dados. O hiperplano em questão divide os dados de maneira visualmente correta em duas classes bem definidas. Fonte: Feito pelo autor

disso, as camadas de convoluções foram também utilizadas como um extrator de características do conjunto de dados, ideia esta extraida dos trabalhos de (Wang; Gang, 2018). Desse modo, os modelos de Regressão Logística, *Naive Bayes* e LGBM também foram treinados utilizando as caracterísitcas extraída pela CC. Assim, é possível analisar como uma camada de convolução influencia no resultado final da classificação.

Por fim, foi utilizada também a técnica de otimização de parâmetros (BERGSTRA; BENGIO, 2012) para encontrar o melhor conjunto de parâmetros, ou seja, o conjunto que possui a melhor métrica, para cada algoritmo utilizado. Para a validação dos modelos treinados foram utilizadas técnicas de validação cruzada (*cross-validation*), mais especificamente o *K-fold cross-validation*, onde o conjunto de dados é divido em K partes iguais e $\frac{K-1}{K}$ partes são utilizadas no treinamento, e $\frac{1}{K}$ partes são usadas para validação, iterando sobre cada parte K

vezes. Para a geração dos resultados foi utilizado $K = 5$.

Em termos de linguagem de programação foram utilizadas duas bibliotecas importantes para o desenvolvimento dos modelos de aprendizado de máquina. A primeira sendo o *scikit-learn* e a segunda sendo o *Tensorflow* juntamente com o *Keras*. Ambas fornecem uma abstração simples para a implementação de modelos complexos de aprendizado de máquina. A primeira foca em modelos mais simples, como por exemplo regressão logística e Naive-Bayes, e a segunda tendo como objetivo a construção de redes neurais e as camadas de convoluções, um modelo mais complexo devido a grande quantidade de camadas e filtros utilizados para extração das características mais importantes de cada foto (LECUN et al., 1998).

4.2.4 Classificação

Depois da etapa de treinamento e de validação do modelo, segue a etapa de classificação. É nessa etapa que o modelo gerado será posto à prova, onde testes reais serão realizados com o objetivo de confirmar se o modelo é abstrato o suficiente ao ponto de reproduzir da maneira mais próxima a capacidade de classificação humana. É a partir da classificação que os resultados finais são gerados. No próximo capítulo explicamos essa etapa de nosso trabalho.

5 Resultados

Nesta seção será apresentado os resultados finais da dissertação. Como dito anteriormente, foram utilizados dois conjuntos de dados a fim de se avaliar os estudos realizados, o conjunto de dados do IMDB (*Internet Movie Database*) e do SST2 (*Stanford Sentiment Treeback Binary*). Como exposto anteriormente, o objetivo é mostrar que o tratamento dos dados fazendo uso de imagens para análise de sentimentos, pode trazer resultados superiores quando comparados aos dados matriciais puros, onde este tratamento inicial não é realizado. Abaixo são descritos em mais detalhes os conjuntos de dados utilizados, os resultados esperados e os obtidos no trabalho.

5.1 Conjuntos de dados utilizados

Uma compilação das características dos conjuntos de dados utilizados é apresentada na Tabela 3. As subseções a seguir explicam em detalhes as informações contidas nessa tabela. Como dito, foram utilizados dois conjuntos de dados. O primeiro é o conjunto de dados do IMDB (MAAS et al., 2011), que apresenta críticas de diversos filmes presentes no site de mesmo nome, e o segundo o conjunto de dados SST2 (SOCHER et al., 2013), que também apresenta críticas relacionadas a filmes, porém estes provenientes da universidade de Stanford.

Tabela 3 – Detalhes dos conjuntos de dados

Conjunto de dados	Train-POS	Train-NEG	Test-POS	Test-NEG	Não anotados	Total
IMDB	12500	12500	12500	12500	50000	100000
SST2	55830	42250	14701	11177	0	123958

Fonte: Feito pelo autor

5.1.1 IMDB

O conjunto de dados IMDB possui um total de 100.000 dados, onde 50.000 são dados anotados e 50.000 são dados não anotados. Dos 50.000 dados anotados, 25.000 são utilizados para treinamento e 25.000 são utilizados para teste. Desses 25.000 dados, tanto para o conjunto de testes quanto para o conjunto de treinamento, 12500 dados são anotados como positivo (Train-POS e Test-POS), e 12.500 são anotados como negativo (Train-Neg e Test-NEG).

5.1.2 SST2

O conjunto de dados SST-2 possui 123.958 dados, onde todos os dados estão anotados. Deste total, 98.080 são dados de treinamento e 25.878 são dados de teste. Dos 98.080 dados de

treinamento 55.830 são dados anotados como positivo (Train-Pos)e 42250 são dados anotados como negativos (Train-NEG). Do conjunto de testes, por sua vez, 14.701 são dados anotados como positivo (Test-POS) e 11.177 são dados anotados como negativo (Test-NEG).

5.2 Resultados esperados

A fim de se comparar os resultados resultados obtidos pelo método apresentado com os já obtidos na literatura, utilizou-se como padrão as contribuições de (LAU; BALDWIN, 2016) e (LE; MIKOLOV, 2014). Como objetivo principal do trabalho, pretende-se elucidar como o uso de imagens geradas a partir dos resultados retornados pelo modelo doc2vec pode melhorar a performance dos modelos de aprendizagem de máquina. Mais do que comparar os resultados com o estado da arte, o foco é mostrar que as imagens podem trazer benefícios à acurácia dos métodos quando comparado ao modelo sem este tratamento. O objetivo, como já dito anteriormente, é fazer uso dessas imagens a fim de submetê-las como entrada de um algoritmo de aprendizagem de máquina que, em um primeiro momento, será treinado para classificar as imagens a ele submetido. Como mostrado pela Seção 5.3, o ganho em alguns casos é significativo o que mostra que tal técnica possui vantagens expressivas quando comparadas ao aprendizado de máquina sem este tratamento.

5.3 Resultados obtidos

O método para vetorização de palavras utilizado no trabalho foi o Doc2Vec (LE; MIKOLOV, 2014) que possui dois métodos distintos para o cálculo da matriz de pesos e, consequentemente, para a geração dos vetores finais: o DBOW (*Distributed Bag of Word*) e DM (*Distributed Memory*). Tais métodos, como dito anteriormente, se distinguem pela utilização do chamado Identificador de Parágrafos. No método DM, além da necessidade de representação única de cada palavra do corpo de texto, também utiliza-se uma representação para cada sentença do conjunto de dados. Já no método DBOW, apenas utiliza-se a representação de cada sentença.

O tamanho do vetor de saída utilizado foi de 2304 posições, onde cada valor foi normalizado para o domínio [0, 1]. Este vetor também pode ser concebido como uma imagem em escala de cinza, pois como cada posição do vetor possui um número entre 0 e 1, ao se gerar uma imagem, cada posição se torna um pixel com uma cor em escala de cinza, onde 0 representa a cor preta e 1 representa a cor branca. Para cada um desses métodos e para cada conjunto de dados foram aplicados diversos algoritmos de classificação, sendo eles: *Naive Bayes*, LGBM, Regressão Logística e Redes Neurais, com o intuito de maximizar a acurácia de nossos resultados.

Por uma questão de padronização, estes mesmos algoritmos também foram utilizados

para tratar agora, não os dados de forma matricial pura, mas sim utilizando as imagens como dados de entrada. O diferencial do modelo proposto está no uso de camadas de convoluções que tem o propósito de inferir um tratamento extra às matrizes de entrada. As camadas de convoluções são utilizadas para extrair características relevantes de uma imagem. Por exemplo, numa primeira camada de convolução, esta tenta detectar alguns padrões iniciais e as bordas das imagens. Numa segunda camada, esta tenta entender a forma, as cores entre outras características relevantes. Finalmente, uma camada final, que geralmente é chamada de camada de recurso, tenta classificar a imagem em si.

Para geração dos conjuntos de treinamento, validação e testes foram utilizadas 2 abordagens. A primeira sendo a utilização de 80% dos dados para treinamento e validação e 20% para teste. A segunda, os conjuntos de treinamento e validação são dividos igualmente, 50% para cada conjunto. No caso do banco de dados do IMDB, os dados utilizados como referência na literatura, usam um subconjunto de banco de dados pré estabelecidos na referência (MAAS et al., 2011) cuja divisão é 50% (treino) e 50% (teste). Esta base é a divisão padrão utilizada por diversos estudos da literatura, de forma que podemos fazer uso deste como base de comparação. Já no caso do banco de dados do SST2, a base de comparação é dado pela divisão 80% (treino) e 20% (teste) sendo que agora esta é a divisão padrão utilizada nos estudos que envolvem esta base de dados.

As médias e os desvios padrões são obtidos através da técnica denominada *K-fold cross-validation*. Nesta técnica, o conjunto de treinamento é divido em k partes e o treinamento é efetuado independentemente utilizando $k-1$ destas, sendo que a sobressalente é utilizada para teste. O processo é repetido utilizando todas as k partes como conjunto sobressalente, de forma que a média e o desvio padrão é enfim calculado. Por exemplo, em nossos estudos divimos o conjunto em 5 partes, ou seja, tomamos $k = 5$. Inicialmente, utilizando as partes 1, 2, 3 e 4 como base de treinamento e a parte 5 como teste. Na sequência, tomamos as partes 2, 3, 4 e 5 como base de treinamento e a parte 1 como teste e assim sucessivamente. Vale frisar que estes subconjuntos são tomados usando como base o conjunto de treinamento de forma que a validação é feita à posteriori utilizando o classificador que obteve a melhor performance na fase de teste e treinamento.

Com o estado da arte em tarefas de análise sentimental sobre o conjunto de dados IMDB tem-se os resultados obtidos por (THONGTAN; PHIENTHRAKUL, 2019) conseguindo uma acurácia de $97,4\%$. Como estado da arte para o conjunto de dados SST2 tem-se os resultados obtidos por (RAFFEL et al., 2020) conseguindo uma acurácia de também $97,4\%$, se igualando aos resultados obtidos com o conjunto de dados IMDB.

Quando se tratando do estado da arte, referência (THONGTAN; PHIENTHRAKUL, 2019) e (RAFFEL et al., 2020), vale frisar que nos respectivos estudos nenhuma menção é feita sobre os subconjuntos utilizados, de forma que supusemos que a acurácia obtida nestes artigos é referente aos subconjuntos de testes/treinamento pré estabelecidos no (MAAS et al.,

2011) e (SOCHER et al., 2013).

Os resultados podem ser melhores visualizados pela Tabela 4.

Tabela 4 – Resumo dos resultados do estado da arte

Conjunto de dados	Modelo	Dados de Treinamento Extra	Acurácia (%)
IMDB	NB-weighted-BON + dv-cosine	sim	97,4
SST2	T5-3B	não	97,4

Fonte: Feito pelo autor

Como primeira abordagem foi utilizado o método DBOW para vetorização das frases pertencentes ao conjunto de dados IMDB. Após a geração dos vetores de palavras os dados foram submetidos a diversos treinamentos com diversos algoritmos diferentes. Inicialmente, como dito anteriormente, as matrizes foram utilizadas em seu estado puro, sem a utilização de uma camada de convolução para extração de características. Na sequência, utilizamos então as camada de convolução, já explicadas anteriormente, a fim de gerarmos um novo conjunto de características, sendo que as matrizes são então agora tratadas como imagens. Por uma questão de padronização, os resultados que contenham a iniciais CC antes do algoritmo de classificação, foram aqueles gerados a partir da utilização da camada de convoluções como um extrator de características. Esta camada, como já explicamos anteriormente, funciona como uma espécie de filtro que consegue identificar as características mais importantes dos dados de entrada. Os dados de entrada para este modelo de CC são concebidos como imagens e, por conta disso, as camadas de convoluções se tornam uma etapa importante do processo.

Os resultados presentes na Tabela 5 foram obtidos com o uso de 80% dos dados para treinamento e 20% para teste.

Tabela 5 – Resultados aplicado ao conjunto de dados IMDB com o método DBOW

Método	Acurácia	Teste
LGBM	0.868111 ± 0.004559	0.8732
Logistic Regression	0.874262 ± 0.003441	0.8828
Naive Bayes	0.806281 ± 0.008185	0.8119
Neural Network	0.867712 ± 0.008206	0.8658
CC + LGBM	0.869361 ± 0.004610	0.8752
CC + Logistc Regression	0.963746 ± 0.002534	0.9677
CC + Naive Bayes	0.518451 ± 0.001913	0.5195
CC + Neural Network	0.953476 ± 0.011361	0.8462

Fonte: Feito pelo autor

Também foram gerados resultados levando em consideração a técnica DM do método Doc2Vec utilizando-se das mesmas condições e dos mesmos algoritmos obtidos com o método

DBOW. Os resultados presentes na Tabela 6 foram obtidos com o uso de 80% dos dados para treinamento e 20% para teste.

Tabela 6 – Resultados aplicado ao conjunto de dados IMDB com o método DM

Método	Acurácia	Teste
LGBM	0.870337 ± 0.004360	0.8723
Logistic Regression	0.881163 ± 0.004727	0.8842
Naive Bayes	0.842084 ± 0.016526	0.8440
Neural Network	0.866936 ± 0.011017	0.8423
CC + LGBM	0.979647 ± 0.000804	0.9795
CC + Logistc Regression	0.986224 ± 0.001020	0.9870
CC + Naive Bayes	0.709545 ± 0.046845	0.7779
CC + Neural Network	0.979418 ± 0.001303	0.8624

Fonte: Feito pelo autor

Analisando os resultados de ambos os métodos utilizados, é possível observar resultados significativos provenientes do algoritmo de Regressão Logística, e também pela utilização da rede neural com camada de convolução onde uma rede neural MLP padrão foi utilizada como algoritmo de classificação.

Foram também gerados resultados sobre o conjunto de dados IMDB utilizando o conjunto de testes pré estabelecido em 50% dos dados. Tais resultados podem ser observados pelas Tabelas 7 e 8, utilizando os métodos DBOW e DM respectivamente.

Tabela 7 – Resultados aplicado ao conjunto de dados IMDB com o método DBOW

Método	Acurácia	Teste
LGBM	0.866789 ± 0.002954	0.847627
Logistic Regression	0.878710 ± 0.003848	0.867749
Naive Bayes	0.781462 ± 0.009980	0.771381
Neural Network	0.869709 ± 0.010379	0.864389
CC + LGBM	0.866069 ± 0.003791	0.830786
CC + Logistc Regression	0.876950 ± 0.002992	0.837827
CC + Naive Bayes	0.554524 ± 0.002719	0.574685
CC + Neural Network	0.869709 ± 0.006623	0.826586

Fonte: Feito pelo autor

Visando a diversificação dos resultados e o teste do método proposto, os mesmo algoritmos utilizados nos resultados anteriores também foram aplicados a um conjunto de dados diferente, o SST2 (*Stanford Sentiment Treebank Binary*). Aqui nos limitamos ao estudo da base pré-estabelecida, sendo que a divisão é de 80% para treinamento e 20% para teste. A Tabela 9 explicita os resultados obtidos com a utilização do método DBOW para vetorização de frases.

Tabela 8 – Resultados aplicado ao conjunto de dados IMDB com o método DM

Método	Acurácia	Teste
LGBM	0.857708 ± 0.003423	0.803344
Logistic Regression	0.872869 ± 0.001770	0.838147
Naive Bayes	0.840627 ± 0.009873	0.792023
Neural Network	0.851108 ± 0.029925	0.818985
CC + LGBM	0.857228 ± 0.003063	0.815265
CC + Logistc Regression	0.865949 ± 0.002542	0.777982
CC + Naive Bayes	0.606328 ± 0.013164	0.660772
CC + Neural Network	0.855748 ± 0.006623	0.827946

Fonte: Feito pelo autor

Tabela 9 – Resultados aplicado ao conjunto de dados SST2 com o método DBOW

Método	Acurácia	Teste
LGBM	0.882228 ± 0.002111	0.721037
Logistic Regression	0.803028 ± 0.002965	0.569363
Naive Bayes	0.583370 ± 0.002661	0.541850
Neural Network	0.705342 ± 0.007660	0.686838
CC + LGBM	0.876172 ± 0.001497	0.732320
CC + Logistc Regression	0.832962 ± 0.002784	0.648852
CC + Naive Bayes	0.610134 ± 0.002598	0.613532
CC + Neural Network	0.879435 ± 0.004548	0.669139

Fonte: Feito pelo autor

A Tabela 10 mostra os resultados obtidos com a utilização do método DM para vetorização de palavras.

Tabela 10 – Resultados aplicado ao conjunto de dados SST2 com o método DM

Método	Acurácia	Teste
LGBM	0.792852 ± 0.003330	0.669989
Logistic Regression	0.730506 ± 0.002576	0.648581
Naive Bayes	0.566088 ± 0.001312	0.431911
Neural Network	0.569229 ± 0.004049	0.568088
CC + LGBM	0.793352 ± 0.002632	0.648465
CC + Logistc Regression	0.743352 ± 0.002738	0.617319
CC + Naive Bayes	0.573715 ± 0.002250	0.431988
CC + Neural Network	0.778813 ± 0.004101	0.641703

Fonte: Feito pelo autor

Com o conjunto de dados SST2 é possível observar que o algoritmo LGBM apresenta resultados mais relevantes, assim como o uso de uma rede neural MLP com uma camada de

convolução como algoritmo de classificação. Em termos dos métodos de vetorização, por sua vez, o método DBOW apresentou resultados gerais mais significativos em comparação com o método DM. É importante ressaltar que ambos os conjuntos de dados utilizados para geração dos resultados também são utilizados no artigo (LE; MIKOLOV, 2014).

Inicialmente todos os resultados gerados relacinados ao modelo baseado em camadas de convolução utilizaram imagens de dimensão 48x48. Com o intuito de enriquecer a análise foram gerados também resultados orientados a imagens de outros tamanhos, para modelos baseado em CC. A Tabela 11 mostra os resultados obtidos ao executarmos o modelo baseado em CC que obteve melhor performance de acordo com a Tabela 6 sobre o conjunto de dados IMDB, para diversos tamanho de imagens. A Tabela 12 mostra os resultados obtidos pela execução do modelo baseado em CC que obteve melhor performance de acordo com a Tabela 9 sobre o conjunto de dados SST2, também para diversos tamanhos de imagens diferentes. Com isso, é possível perceber que a utilização de imagens de dimensões 48x48 apresentam os melhores resultados para ambos os conjunto de dados utilizados. É importante ressaltar que para o processo de decisão da dimensão da imagem a ser utilizada foram testadas diversas dimensões desde 10x10 até 52x52, porém como é possível observar pela Tabela 11, apenas a partir da dimensão 44x44 que é possível observar resultados relevantes.

Tabela 11 – Resultados de acordo com a dimensão das imagens - IMDB

Método	Dimensão	Resultado	Teste
DM + CC + Logistic Regression	44x44	0.873487 ± 0.003926	0.873500
	46x46	0.876587 ± 0.000635	0.872300
	48x48	0.986224 ± 0.001020	0.987000
	50x50	0.877337 ± 0.003083	0.879100
	52x52	0.874762 ± 0.003109	0.884500

Fonte: Feito pelo autor

Tabela 12 – Resultados de acordo com a dimensão das imagens - SST2

Método	Dimensão	Resultado	Teste
DBOW + CC + LGBM	44x44	0.876447 ± 0.001813	0.714622
	46x46	0.874887 ± 0.001457	0.737846
	48x48	0.876172 ± 0.001497	0.732320
	50x50	0.875560 ± 0.002385	0.739199
	52x52	0.874938 ± 0.001208	0.734021

Fonte: Feito pelo autor

Apesar dos resultados obtidos não atingirem os resultados proveniente do estado da arte, conforme podemos observar nas Tabelas 7, 8, 9, e 10, é possível perceber resultados promissores quando comparamos os modelos gerados com a utilização das camadas de convolução como

filtros, ou extrator de características. Assim, é possível concluir que o uso de camadas de convoluções influenciam positivamente os resultados finais de classificação. Em especial, o uso do método de vetorização DM *Distributed Memory* e o classificador de Regressão Logística apresentaram uma acurácia de 98%, o melhor resultado dentre os demais. Neste sentido, a utilização das camadas de convoluções para classificação de texto se torna relevante pois herda conceitos importantes do ramo de classificação de imagens, que são utilizados amplamente nas redes neurais convolucionais. Com isso, abre-se caminho para uma grande quantidade de possibilidades de pesquisa no campo da classificação de sentimentos.

6 Conclusão

Como conclusão podemos perceber que o modelo proposto se mostrou versátil, abrindo espaço para o teste com diversos tipos de classificadores de aprendizado de máquina. É possível concluir que o uso das camadas de convoluções apresentaram resultados promissores quando utilizadas em conjunto com outros algoritmos de classificação.

A utilização de imagens para classificação de documentos de texto, portanto, pode melhorar a performance dos modelos gerados com essa abordagem, principalmente pelo uso do algoritmo Regressão Logística e LGBM em conjunto com as camadas de convoluções. A metodologia que se mostrou promissora neste sentido, foi o uso do método de vetorização de texto DM, para o caso do conjunto de dados do IMDB, e o método de vetorização de texto DBOW, no caso do conjunto de dados do SST2. O algoritmo de Regressão logística, juntamente com o uso das camadas de convoluções foi o método que se sobressaiu no estudo do conjunto de dados do IMDB, enquanto que o método de LGBM, em conjunto com as camadas de convoluções, se sobressaiu no estudo do conjunto de dados do SST2.

Vale frisar que o modelo proposto não depende de nenhum conjunto de dados específico, sendo que a metodologia apresentada nesta dissertação pode ser utilizada na tarefa de classificação de texto para qualquer outro conjunto de dados.

Como trabalhos futuros existem diversas possibilidades de melhoria aos métodos propostos. Começando pela vetorização e representação de documentos de texto, existem métodos bastantes relevantes na atualidade que podem melhorar o resultado final. Estes métodos, denominados *Transformers*, utilizam mecânismo de atenção e redes neurais para gerar uma representação de texto mais precisa, e podem ser utilizados para geração das imagens do método proposto. A utilização das camadas de convoluções fazendo uso dessa nova representação de texto é algo que será explorado em trabalhos futuros.

Além disso, abordagens relacionadas a outros tipos de aprendizado, como aprendizado não-supervisionado, onde o conjunto de dados não possui anotações prévias, ou aprendizado semi-supervisionado, onde apenas uma parcela do conjunto de dados está anotada, também podem ser abordados utilizando a mesma metodologia exposta nesse trabalho.

Acreditamos que nossos resultados podem contribuir de forma significativa na tarefa de análise de sentimento de texto, sendo a estratégia aqui desenvolvida promissora em estudos futuros.

Referências

AYYADEVARA, V. Gradient boosting machine. In: _____. [S.l.: s.n.], 2018. p. 117–134. ISBN 978-1-4842-3563-8.

BERGSTRA, J.; BENGIO, Y. Random search for hyper-parameter optimization. J. Mach. Learn. Res., JMLR.org, v. 13, n. null, p. 281–305, fev. 2012. ISSN 1532-4435.

BISHOP, C. M. Pattern Recognition and Machine Learning (Information Science and Statistics). Berlin, Heidelberg: Springer-Verlag, 2006. ISBN 0387310738.

BREIMAN, L. et al. Classification And Regression Trees. [S.l.: s.n.], 2017. 1-358 p. ISBN 9781315139470.

FISHER, R. A. The use of multiple measurements in taxonomic problems. Annals of Eugenics, v. 7, n. 2, p. 179–188, 1936. Disponível em: <https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1469-1809.1936.tb02137.x>.

FOLEY, J. D. et al. Computer Graphics: Principles and Practice (2Nd Ed.). Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1990. ISBN 0-201-12110-7.

FUKUSHIMA, K. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biological cybernetics, Springer, v. 36, n. 4, p. 193–202, 1980.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. Deep Learning. [S.l.]: MIT Press, 2016. <http://www.deeplearningbook.org>.

HAN, K.-Y. L. S.-H. Implementation of image classification cnn using multi thread gpu. Proceedings of the IEEE, 2017.

HARRIS, Z. Distributional structure. Word, v. 10, n. 23, p. 146–162, 1954.

HASSAN, A.; QAZVINIAN, V.; RADEV, D. What's with the attitude?: Identifying sentences with attitude in online discussions. In: Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing. Stroudsburg, PA, USA: Association for Computational Linguistics, 2010. (EMNLP '10), p. 1245–1255. Disponível em: <http://dl.acm.org/citation.cfm?id=1870658.1870779>.

HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. The Elements of Statistical Learning. New York, NY, USA: Springer New York Inc., 2001. (Springer Series in Statistics).

HAWKINGS, S. Inside a Great Mind. [Entrevista concedida a] Parade Magazine. 2010.

HAYKIN, S. S. Neural networks and learning machines. Third. [S.l.]: Pearson Education, 2009.

HINTON, G. E. Learning distributed representations of concepts. In: AMHERST, MA. Proceedings of the eighth annual conference of the cognitive science society. [S.l.], 1986. v. 1, p. 12.

JURAFSKY, D.; MARTIN, J. H. Speech and Language Processing (2nd Edition). USA: Prentice-Hall, Inc., 2009. ISBN 0131873210.

KE, G. et al. Lightgbm: A highly efficient gradient boosting decision tree. In: GUYON, I. et al. (Ed.). Advances in Neural Information Processing Systems. Curran Associates, Inc., 2017. v. 30, p. 3146–3154. Disponível em: <https://proceedings.neurips.cc/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf>.

KO, Y. A study of term weighting schemes using class information for text classification. In: Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval. New York, NY, USA: ACM, 2012. (SIGIR '12), p. 1029–1030. ISBN 978-1-4503-1472-5. Disponível em: <http://doi.acm.org/10.1145/2348283.2348453>.

KWON, H. L. . H. Contextual deep cnn based hyperspectral classification. IEEE, 2016.

LAU, J. H.; BALDWIN, T. An empirical evaluation of doc2vec with practical insights into document embedding generation. CoRR, abs/1607.05368, 2016. Disponível em: <http://arxiv.org/abs/1607.05368>.

LE, Q. V.; MIKOLOV, T. Distributed representations of sentences and documents. CoRR, abs/1405.4053, 2014. Disponível em: <http://arxiv.org/abs/1405.4053>.

LECUN, Y.; BENGIO, Y. The handbook of brain theory and neural networks. In: ARBIB, M. A. (Ed.). Cambridge, MA, USA: MIT Press, 1998. cap. Convolutional Networks for Images, Speech, and Time Series, p. 255–258. ISBN 0-262-51102-9. Disponível em: <http://dl.acm.org/citation.cfm?id=303568.303704>.

LECUN, Y. et al. Backpropagation applied to handwritten zip code recognition. Neural Computation, MIT Press, Cambridge, MA, USA, v. 1, n. 4, p. 541–551, 1989.

LECUN, Y. et al. Gradient-based learning applied to document recognition. Proceedings of the IEEE, v. 86, n. 11, p. 2278–2324, 1998.

LIU, B. Sentiment Analysis and Opinion Mining. [S.l.]: Morgan & Claypool Publishers, 2012. ISBN 1608458849, 9781608458844.

LIU, H. Social network profiles as taste performances. J. Comp.-Med. Commun., John Wiley & Sons, Inc., New York, NY, USA, v. 13, n. 1, p. 252–275, out. 2007. ISSN 1083-6101. Disponível em: <https://doi.org/10.1111/j.1083-6101.2007.00395.x>.

Luo, Q. et al. Research of a spam filtering algorithm based on naïve bayes and ais. In: 2010 International Conference on Computational and Information Sciences. [S.l.: s.n.], 2010. p. 152–155.

MAAS, A. L. et al. Learning word vectors for sentiment analysis. Association for Computational Linguistics, Portland, Oregon, USA, p. 142–150, June 2011. Disponível em: <http://www.aclweb.org/anthology/P11-1015>.

MANNING, C. D.; SCHüTZE, H. Foundations of Statistical Natural Language Processing. Cambridge, MA, USA: MIT Press, 1999. ISBN 0-262-13360-1.

MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. The bulletin of mathematical biophysics, Springer, v. 5, n. 4, p. 115–133, 1943.

MIKOLOV, T. et al. Efficient estimation of word representations in vector space. CoRR, abs/1301.3781, 2013. Disponível em: <http://arxiv.org/abs/1301.3781>.

MITCHELL, T. M. Machine Learning. 1. ed. New York, NY, USA: McGraw-Hill, Inc., 1997. ISBN 0070428077, 9780070428072.

O'CONNOR, B. et al. From tweets to polls: Linking text sentiment to public opinion time series. In: COHEN, W. W.; GOSLING, S. (Ed.). ICWSM. [S.l.]: The AAAI Press, 2010.

PEARL, R.; REED, L. J. On the rate of growth of the population of the united states since 1790 and its mathematical representation. Proceedings of the National Academy of Sciences of the United States of America, National Academy of Sciences, v. 6, n. 6, p. 275–288, 1920. ISSN 00278424. Disponível em: <http://www.jstor.org/stable/84343>.

RAFFEL, C. et al. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. 2020.

RASCHKA, S. Python Machine Learning. [S.l.]: Packt Publishing, 2015. ISBN 1783555130, 9781783555130.

REHUREK, R. Scalability of semantic analysis in natural language processing. In: . [S.l.: s.n.], 2011.

ROBERTSHAW, T. Underfitting e Overfitting. 2015. Disponível em: <https://tomrobertshaw.net/2015/12/introduction-to-machine-learning-with-naive-bayes/>.

ROSENBLATT, F. The perceptron: a probabilistic model for information storage and organization in the brain. Psychological review, American Psychological Association, v. 65, n. 6, p. 386, 1958.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1. In: RUMELHART, D. E.; MCCLELLAND, J. L.; GROUP, C. P. R. (Ed.). Cambridge, MA, USA: MIT Press, 1986. cap. Learning Internal Representations by Error Propagation, p. 318–362. ISBN 0-262-68053-X. Disponível em: <http://dl.acm.org/citation.cfm?id=104279.104293>.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning representations by back-propagating errors. In: _____. Neurocomputing: Foundations of Research. Cambridge, MA, USA: MIT Press, 1988. p. 696–699. ISBN 0262010976.

SAPIR, E. Language, An Introduction to the Study of Speech. New York: Brace, 1921.

SOCHER, R. et al. Recursive deep models for semantic compositionality over a sentiment treebank. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing. Seattle, Washington, USA: Association for Computational Linguistics, 2013. p. 1631–1642. Disponível em: <https://www.aclweb.org/anthology/D13-1170>.

STATISTICS, L. B.; BREIMAN, L. Random forests. In: Machine Learning. [S.l.: s.n.], 2001. p. 5–32.

THONGTAN, T.; PHIENTHRAKUL, T. Sentiment classification using document embeddings trained with cosine similarity. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop. Florence, Italy: Association for Computational Linguistics, 2019. p. 407–414. Disponível em: <https://www.aclweb.org/anthology/P19-2057>.

TURING, A. M. Computing machinery and intelligence. Mind, Oxford University Press on behalf of the Mind Association, v. 59, n. 236, p. 433–460, 1950. ISSN 00264423.

Wang, W.; Gang, J. Application of convolutional neural network in natural language processing. In: 2018 International Conference on Information Systems and Computer Aided Education (ICISCAE). [S.l.: s.n.], 2018. p. 64–70.

ZHANG, A. et al. Dive into Deep Learning. [S.l.: s.n.], 2020. <https://d2l.ai>.

ZHANG, H. The optimality of naive bayes. In: . [S.l.: s.n.], 2004. v. 2.

ZHANG, J. L. C. W. S. W. H. Z. B. Classification of very high resolution sar image based on convolutional neural network. IEEE, 2017.

ZHENG, A.; CASARI, A. Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists. 1st. ed. [S.l.]: O'Reilly Media, Inc., 2018. ISBN 1491953241.

UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
Campus de São José do Rio Preto

TERMO DE REPRODUÇÃO XEROGRÁFICA

Autorizo a reprodução xerográfica do presente Trabalho de Conclusão, na íntegra ou em partes, para fins de pesquisa.

São José do Rio Preto, 04 / 03 / 2021

Assinatura do autor