ESPECIALIZAÇÃO EM COMPUTAÇÃO
CIENTÍFICA PARA A INDÚSTRIA

COLETÂNEA DE ARTIGOS CIENTÍFICOS TRABALHOS DE CONCLUSÃO DE CURSO

TURMA 2023

PUBLICAÇÃO DO IFSC
GitHub
creative commons
pecce

Reitor
Maurício Gariba Júnior

Diretor do Campus Florianópolis
Zízimo Moreira Filho

Chefe do Departamento Acadêmico de Eletrotécnica
Edison Antonio Cardoso Aranha Neto

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
DE SANTA CATARINA – IFSC – CAMPUS FLORIANÓPOLIS
AV. MAURO RAMOS N. 950, FLORIANÓPOLIS, SC.

Especialização em Computação Científica para a Indústria

Coletânea de Artigos Científicos
Trabalhos de Conclusão de Curso
Turma 2023

Florianópolis
2024

Revisão: Cláudia Regina Silveira, Doutora em Letras - Literatura Brasileira.

Catalogação na fonte elaborada por
Karla Viviane Garcia Moraes – CRB14/1002

C694 Coletânea de artigos científicos : trabalhos de conclusão de curso : turma 2023 / Organizador Sérgio Luciano Ávila. – Florianópolis : Publicações do IFSC, 2024.
105 p.

Curso de Especialização em Computação Científica para Indústria

ISBN: 978-65-996422-9-6

1. Inteligência computacional. 2. Inteligência artificial. 3. Pesquisa científica. I. Ávila, Sérgio Luciano.

CDD 006.3

AUTORES

ORGANIZADOR PRINCIPAL

Prof. Sérgio Luciano Avila
Coordenador do Curso

COMITÊ ORGANIZADOR

Prof. Cesar Alberto Penz
Prof. Maurício Edgar Stivanello
Prof. Pedro Giassi Junior
Prof. Sérgio Luciano Avila

ALUNOS

Alexsandro Gehlen
Bruna Martini
Fernando Vasata
Jonathan Tenório Lima
Lucas Gabriel Coliado Bandeira
Thiago Haigerti Bertoldi

GITHUB

GITHUB.COM/PECCE-IFSC/CCI

IFSC.EDU.BR/ESPECIALIZACAO

GRUPO DE PESQUISAS EM COMPUTAÇÃO CIENTÍFICA PARA ENGENHARIA

IFSC.EDU.BR/PECCE

Coletânea de Artigos Científicos Trabalhos de Conclusão de Curso Turma 2023

Curso de Especialização em Computação Científica para a Indústria

Área de concentração	:	Engenharias
Público	:	Graduandos e pós-graduandos
Palavras-chave	:	Inteligência computacional. Pesquisa científica. Problemas da Indústria. Artigos

O objetivo geral do curso de Especialização em Computação Científica para a Indústria é oportunizar aprimoramento profissional aos interessados nas teorias e metodologias computacionais para resolução de problemas da engenharia, buscando transformar dados em informações que possam basear tomadas de decisão mais assertivas. São objetivos específicos:

a) capacitar os profissionais em tema atual, relevante e de larga aplicabilidade;

b) apresentar alternativas tecnológicas que possam contribuir para a promoção do desenvolvimento técnico, econômico e social;

c) ampliar a empregabilidade e o empreendedorismo do egresso; e

d) suprir a ausência de profissionais capacitados em computação científica.

Os conhecimentos que fazem parte da formação do profissional e que constituem, portanto, as contribuições e o perfil do egresso do curso de especialização em computação científica para engenharia são:

a) identificação e análise dos problemas de engenharia;

b) escolha e aplicação de ferramentas computacionais mais adequadas para encontrar soluções para problemas de engenharia;

c) produção científica e de articulação entre ensino, pesquisa aplicada e extensão tecnológica;

d) desenvolvimento da comunicação oral, escrita e de uso dos recursos tecnológicos disponíveis.

O público-alvo são graduados nos bacharelados de Física, Matemática, Computação e Engenharias, bem como os graduados em cursos superiores de tecnologia correlatos, interessados em se especializar em computação científica para a indústria.

Neste contexto, este livro é organizado em capítulos, sendo cada capítulo um artigo científico – os trabalhos de conclusão de curso da turma 2023 – que apresentam problemas da indústria e as ferramentas para a sua resolução. As defesas públicas ocorreram em 25 de outubro de 2023.

Reforça-se que o propósito maior deste livro é abrir portas, nunca esgotar assuntos. Para obter maior detalhamento dos métodos, bem como suas validações, sugere-se a busca pelas obras específicas de cada tema.

1.

Técnicas de Inteligência Artificial para detecção de fraudes em transações de cartões de crédito

Discente: Alexsandro Gehlen

Prof. Orientador: Pedro Giassi Júnior

RESUMO As transações de cartão de crédito são uma parte essencial da vida moderna, mas também são um alvo frequente de fraudes, as quais, podem causar grandes prejuízos aos envolvidos, tanto financeiramente quanto em termos de danos à reputação. Técnicas de inteligência artificial estão sendo utilizadas para aumentar o combate à fraude realizando a classificação das transações. Neste trabalho foram implementados os modelos Random Forest, Rede Neural Artificial, Autoencoder, Naive Bayes e um Ensemble com a junção de todos os modelos. Para realizar as métricas de validação do modelo foram utilizadas matriz de confusão, acurácia e curva ROC. Os modelos tiveram acurácia acima de 95%, mostrando que conseguem identificar transações fraudulentas.

PALAVRAS-CHAVE: cartão de crédito, classificação, fraude, inteligência Artficial, *machine learning*.

I. INTRODUÇÃO

Uma pesquisa feita pela Confederação Nacional de Dirigentes Lojistas (CNDL) e pelo Serviço de Proteção ao Crédito (SPC Brasil) mostrou que 46% dos internautas brasileiros foram vítimas de algum tipo de golpe financeiro nos 12 meses anteriores ao estudo, o que equivale a um universo aproximado de 12,1 milhões de pessoas. E estima-se que o prejuízo total decorrente dessas fraudes financeiras chegue a cerca de R$ 1,8 bilhão (CNDL, 2022). O Brasil registrou 4,1 milhões de movimentações suspeitas no ano de 2021, segundo o Indicador de Tentativas de Fraude da Serasa Experian. O número representa crescimento de 16,8% em relação a 2020 e é o maior desde 2011, quando foi iniciada a série histórica (Experian, 2021).

A fraude explorada neste trabalho pode estar relacionada a compras não identificadas por parte dos usuários legítimos, clonagem de cartões de crédito, roubos de contas de usuários ou clonagem de sites de venda.

Existe uma variedade de tipos de fraudes, mas as mais comuns são do tipo *phishing* e *pharming*. *Pishing* é um ataque que tenta roubar o dinheiro ou a identidade da vítima fazendo com que ela revele informações pessoais, tais como números de cartão de crédito, informações bancárias ou senhas em sites que fingem ser legítimos. Já o *pharming* explora vulnerabilidades em sites ou aplicações, a fim de permitir que um *cracker* redirecione os usuários para um site ou aplicativo falso, possibilitando que o fraudador receba informações e dados dos usuários e com isso possa efetuar transações online utilizando as informações roubadas (ACFE, 2008). Estas fraudes, que geram prejuízos financeiros, não podem ser arcadas por clientes e comerciantes, pois o uso do cartão de crédito é tido como uma atividade de risco, conforme o artigo 927 do Código Civil (CONCIL, 2018). Uma vez que as fraudes no cartão não tenham sido causadas pelo titular nem pelo comerciante, cabe à operadora responsável reparar os danos. O processo de autorização de cartão de crédito envolve pelo menos cinco partes: o portador do cartão, o estabelecimento, o adquirente, a bandeira e o emissor (Santiago, 2014), e cada um é, respectivamente, responsável por uma etapa da autorização.

Um sistema tipicamente utilizado para detecção de fraudes é o Sistema de Detecção de Fraudes (FDS), que geralmente é composto por cinco etapas: as primeiras duas etapas são executadas em tempo real, que fazem as requisições de identificação do cartão, limite e algumas checagens de regras desenhadas por especialistas em fraudes; as próximas duas etapas não acontecem em tempo real, são sistemas de pontuação para cada transação e utilizam modelos estatísticos para uma possível detecção de fraudes; por fim, na última etapa entram os investigadores que são pessoas especializadas em fraudes (Borgne, 2021).

Atualmente existem algumas técnicas de IA (Inteligência Artificial) que são utilizadas para auxiliar na detecção de fraudes. Algoritmos computacionais como regressão logística, KNN (*K-nearest neighbors*), árvore de decisões (RF - *random forest*), K-MEANS, também redes neurais artificiais (RNA) têm sido implementados e por último métodos de probabilidade *Naive Bayes*. Em pesquisa realizada no *IEEE Xplore* foram encontrados 251 artigos buscando por *detection fraud in credit card transaction* a partir de 2019.

O avanço constante da tecnologia tem impulsionado o aumento do poder computacional, possibilitando a criação e implementação de soluções mais sofisticadas em diversas áreas. Uma dessas áreas é a detecção e classificação de fraudes, onde a aplicação de técnicas de Inteligência Artificial tem se mostrado extremamente eficaz. Neste estudo, o objetivo é aplicar diversos modelos para avaliar as suas capacidades em classificar transações de cartão de crédito como legítimas ou fraudulentas.

Na seleção dos modelos, foram considerados diversos métodos distintos de aprendizado de máquina visando a possibilidade de comparação dos resultados. Os modelos escolhidos são: RF, RNA, Autoencoder e o método probabilístico Naive Bayes. Após a implementação e treinamento desses modelos, de forma independente, foi criado um conjunto que os englobasse, permitindo assim uma comparação abrangente dos resultados obtidos por cada abordagem. Para analisar os resultados de todos os modelos implementados, foram utilizadas métricas para avaliar sua assertividade.

II. TRABALHOS RELACIONADOS

Para fundamentar o trabalho em questão, foram conduzidas pesquisas sobre estudos previamente realizados no âmbito da utilização de inteligência artificial na classificação de transações de cartões de crédito.

Madhuryaa (2022) realizou um estudo de comparação de diferentes técnicas de regressão logística, KNN, RF, K-MEANS para a detecção de fraudes em transações de cartões de crédito. Para a avaliação de desempenho e assertividade dos modelos usados foram utilizados: F1, acurácia, precisão e sensibilidade. Nos resultados é possível observar uma acurácia e sensibilidade de mais de 90% nos modelos, mas se alerta que, o número total de fraudes é sempre menor do que o total de dados considerados, portanto, a variação do subconjunto é alta e pode variar em diferentes instâncias, fazendo com que um modelo falhe em uma determinada situação.

Ileberi (2021) realizou um estudo com os algoritmos KNN, árvore de decisões, floresta de decisão, K-MEANS. Adicionou *adaboost* para aumentar a performance e cita a utilização de uma técnica chamada de *smote* que é para balanceamento dos dados criando um dataset sintético. A técnica *smote* aumenta o número da classe minoritária para assim melhorar o aprendizado dos modelos.

Azevedo, Komati e Júnior (2021) desenvolveram uma rede neural *autoencoder* para detecção de anomalias. Eles utilizaram um *dataset* disponibilizado por uma *fintech* brasileira. Para uma rede neural *autoencoder* ter um bom desempenho é necessário ter uma grande quantidade de dados. A rede neural *autoencoder* é separada em bloco de codificação e decodificação, que quando uma transação é fraudada a sua decodificação irá gerar anomalias que é diferente do que ela conhece e assim classificando a transação como anomalia/fraude. Um levantamento feito neste trabalho é em relação ao impacto ao usuário com a classificação do falso negativo, o qual barraria transações legítimas.

Kim (2019) faz a comparação entre dois modelos: um conjunto híbrido e uma *deep learning*. O modelo híbrido é mais utilizado no mercado e considerado até o momento mais estável. Esse modelo

híbrido apresentado é constituído por árvore de decisão, regressão logística e uma rede neural de pouca profundidade. Já no modelo de deep learning foram testadas algumas configurações e hiperparâmetros. O modelo que apresentou melhor resultado foi o *deep learning*.

Awoyemi, Adetunmbi e Oluwadare (2017) apresentaram métodos de detecção de fraude em transações de cartão de crédito. As técnicas utilizadas foram Naive Bayes, KNN e regressão logística e os resultados de acurácia encontrados foram respectivamente 97.92%, 97.69% e 54.86%. Também mostrou que uma variação das amostras pode alterar a capacidade do modelo implementado sendo que seu *dataset* é altamente desbalanceado.

III. DESENVOLVIMENTO

As implementações de todos os modelos foram realizadas na plataforma *Google Colab,* com a linguagem Python e com as bibliotecas *Scikit Learn, Tensorflow e Matplotlib.*

A. DATASET

O dataset utilizado consiste em transações financeiras realizadas em 2013 por portadores de cartão de crédito europeus, disponibilizados na plataforma *Kaggle* (Kaggle, 2023).

O dataset passou por uma Análise de Componentes Principais (PCA) que mascara os dados por se tratar de dados sensíveis. Além disso, um desafio pertinente aos dados é o desbalanceamento do dataset, pois o número de transações legítimas é muito maior do que o transações fraudulentas.

O dataset foi separado igualmente, contendo os mesmos dados para treinamento e testes, a divisão foi de 80% de treinamento e 20% de testes. A Figura 1.1 expõe o desbalanceamento do dataset com 56.962 transações legítimas e apenas 98 fraudulentas.

B. RANDOM FOREST

Este modelo opera mediante a criação de um conjunto de árvores de decisão, com um funcionamento diferenciado em relação à árvore de decisão convencional. Na abordagem tradicional da árvore de decisão, é estabelecida uma estrutura de nós, onde as condições são verificadas, direcionando o fluxo para ramos específicos com base na satisfação ou não das condições. No caso do *Random Forest*, o processo inicial envolve a seleção aleatória de uma parte das amostras dos dados de treinamento ao invés de utilizar o conjunto completo. O algoritmo realiza os cálculos necessários para determinar o fluxo dos dados em cada nó. Em seguida, as próximas escolhas de nós são feitas, excluindo as variáveis já empregadas. Essa abordagem pode não selecionar as variáveis ótimas em cada iteração, mas, devido à construção de múltiplas árvores, o modelo alcança resultados melhores. (Didática Tech, 2023). No escopo deste trabalho, foram implementadas dez árvores no modelo, cada uma com uma profundidade contendo sete nós. Esse arranjo específico visa à obtenção de um equilíbrio entre a complexidade do modelo e a capacidade de generalização, resultando em um desempenho eficaz na classificação de transações de cartões de crédito.

C. REDE NEURAL ARTIFICIAL

As Redes Neurais Artificiais (RNAs), por outro lado, possuem uma estrutura composta por várias camadas de neurônios. Essa configuração engloba uma camada de entrada, uma ou mais camadas intermediárias (ocultas) e uma camada de saída. Cada neurônio em uma camada está interconectado com outros neurônios e é caracterizado por um peso e um limiar associado. Quando a saída de um neurônio ultrapassa seu limiar, ele é ativado e transmite os dados para a camada subsequente. A RNA é um sistema adaptativo que utiliza informações sobre seus erros para melhorar seu desempenho ao longo do tempo. Isso possibilita que a rede aborde de maneira ágil problemas complexos e não lineares. As RNAs são treinadas com conjuntos de dados conhecidos, permitindo que seus pesos sejam ajustados de acordo. Posteriormente, elas são testadas com dados desconhecidos para avaliar sua capacidade de

generalização. As RNAs têm aplicabilidade em diversas áreas, incluindo classificação e previsão (Amazon, 2020). Neste sistema, a abordagem de Redes Neurais Artificiais foi empregada para a tarefa de classificação. Foram utilizadas duas camadas intermediárias (ocultas), cada uma com trinta neurônios. O processo de treinamento compreendeu trinta épocas, nas quais a rede neural se adaptou aos padrões presentes nos dados conhecidos, ajustando seus pesos de acordo com a aprendizagem obtida. Essa configuração foi escolhida para maximizar a capacidade da rede de identificar padrões complexos nas transações de cartões de crédito.

D. REDE AUTOENCODER

A rede *Autoencoder* é um tipo de rede neural de aprendizado não supervisionado, cujo propósito é replicar a entrada na saída, ao mesmo tempo em que realiza uma compactação da entrada em uma representação de espaço latente e, subsequentemente, reconstrói a saída a partir dessa representação. Essa arquitetura de rede é dividida em duas partes distintas: a etapa de codificação e a etapa de decodificação (DEEP, 2022). No contexto específico abordado, o foco da rede autoencoder é a detecção de anomalias. A rede é treinada utilizando transações legítimas. Dado que o conjunto de dados apresenta um desequilíbrio significativo, com menos de 1% das transações sendo fraudulentas, a rede foi treinada utilizando 80% do conjunto de dados. Durante a fase de teste, a saída da rede classifica as transações como fraudulentas se o erro na reconstrução exceder um limite predefinido. Esse limite é definido com base em critérios pré-estabelecidos. Também é possível selecionar atributos específicos para essa tarefa. No cenário, a rede é composta por uma camada de entrada com 30 neurônios, seguida por duas camadas ocultas com 15 e 5 neurônios, respectivamente, nas etapas de codificação e decodificação. Essas camadas ocultas são simétricas, refletindo a estrutura de compactação e reconstrução da rede autoencoder. O treinamento é realizado ao longo de 30 épocas, permitindo à rede aprender representações latentes eficazes para a detecção de anomalias em transações de cartões de crédito.

E. NAIVE BAYES

O modelo *Naive Bayes* é um algoritmo probabilístico fundamentado no Teorema de Bayes, que emprega uma fórmula matemática para calcular probabilidades condicionais, resultando na criação de uma tabela de respostas. Uma das suas vantagens é a escalabilidade linear em termos de complexidade temporal, de acordo com o aumento do número de recursos. (Fontana, 2020).

F. ENSEMBLE

Por fim, foi treinado um *ensemble*, que se trata de um conjunto de modelos construídos de forma que a saída de um modelo seja usada para treinar o modelo subsequente. Neste caso, o *ensemble* foi formado pela combinação dos quatro modelos mencionados anteriormente, atribuindo pesos iguais a cada um deles. A classificação final é determinada quando três ou mais modelos apresentam a mesma classificação. Nesse contexto, a decisão de classificação adotada pelo *ensemble* é aquela que é concordante entre os modelos participantes.

G. MÉTRICAS

Além de desenvolver os modelos é importante uma avaliação precisa para validar os resultados alcançados por meio das implementações realizadas. Uma seleção inadequada de métricas de avaliação pode comprometer a correta avaliação dos modelos. Cada modelo emprega cálculos específicos para apresentar seus resultados, com métricas e custos distintos. No escopo dessa avaliação, a matriz de confusão foi empregada como uma ferramenta essencial. A partir dessa matriz, foi possível extrair a acurácia do modelo, proporcionando uma visão geral da sua eficácia. Adicionalmente, a curva ROC (*Receiver Operating Characteristics*) foi utilizada como uma medida adicional de avaliação. A curva ROC é uma representação gráfica que mostra o desempenho do modelo em termos de taxa de verdadeiros positivos em relação à taxa de falsos positivos em diferentes limiares de classificação. Dessa forma, a combinação da matriz de confusão, acurácia e da curva ROC permite

uma avaliação abrangente e robusta do modelo, considerando diferentes aspectos do seu desempenho na classificação de transações de cartões de crédito.

IV. RESULTADOS

A. RANDOM FOREST

Os rótulos presentes no dataset utilizado permitiram obter a matriz de confusão mostrada na Figura 1.2.

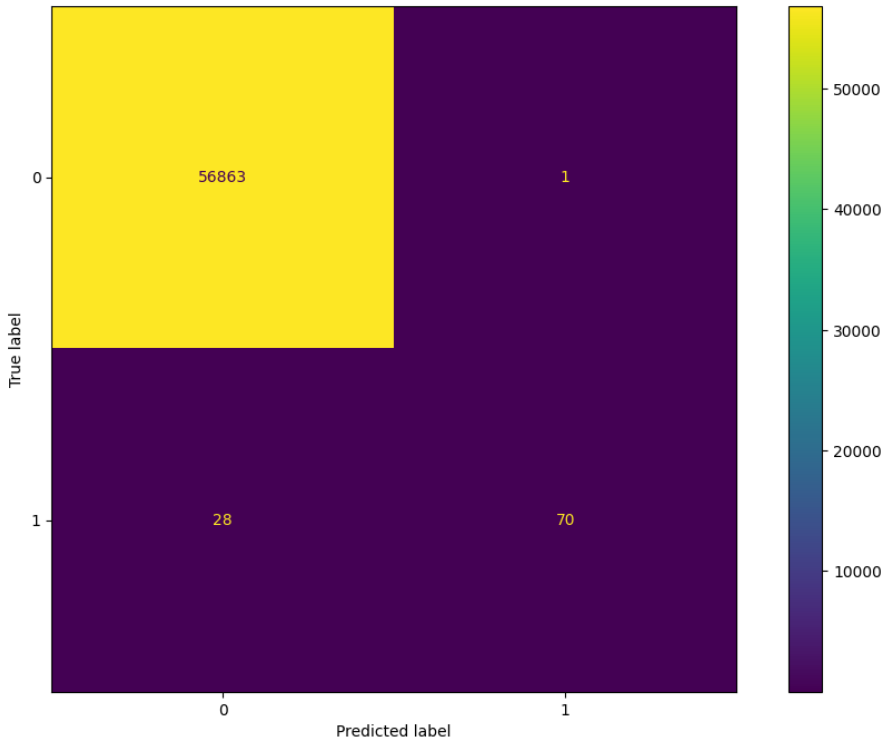


Figura 1.2. Matriz de confusão *random forest*.

A partir dessa matriz, calculou-se a acurácia, que foi de 99.94%. Também é possível verificar o número de classificações: 56863 classificadas corretamente como legítimas, 70 classificadas

corretamente como transações fraudulentas, 1 classificada erroneamente como transação fraudulenta e 28 classificadas erroneamente como transações legítimas. A Figura 1.3 apresenta a curva ROC com um desempenho de 95.27%.

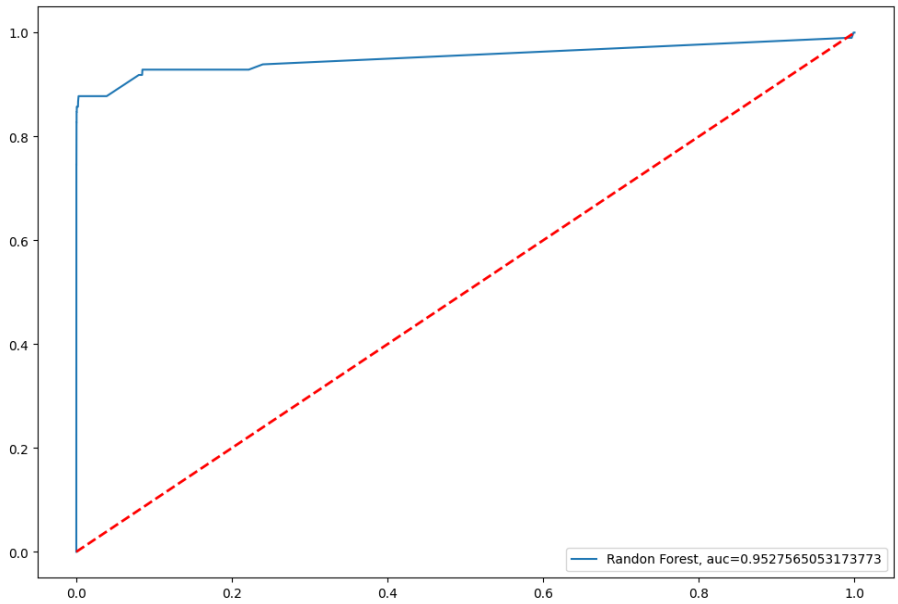


Figura 1.3. Curva ROC modelo random forest.

B. REDE NEURAL ARTIFICIAL

Os resultados das RNAs foram bons, sendo que ainda é possível variar parâmetros para tentar melhorar seus resultados devido a sua maleabilidade de implementação.

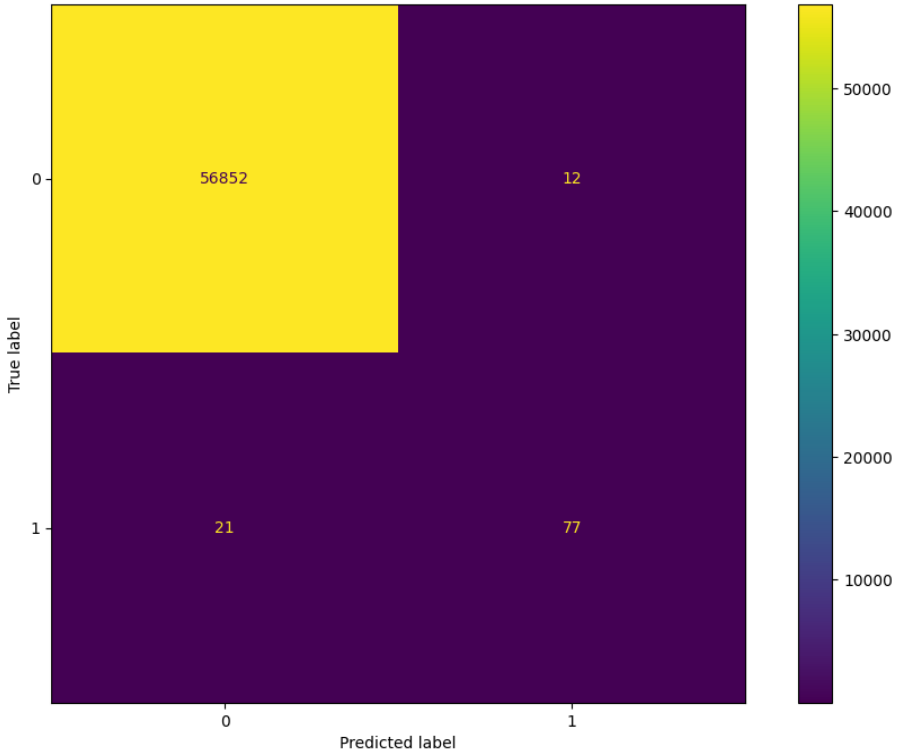


Figura 1.4. Matriz de confusão RNA.

Através dessa matriz, calculou-se a acurácia que foi de 99.94%. Também é possível verificar o número de classificações: 56852 classificadas corretamente como legítimas, 77 classificadas corretamente como transações fraudulentas, 12 classificada erroneamente como transação fraudulenta e 21 classificadas erroneamente como transações legítimas. A Figura 1.5 apresenta a curva ROC com um desempenho de 96.67%.

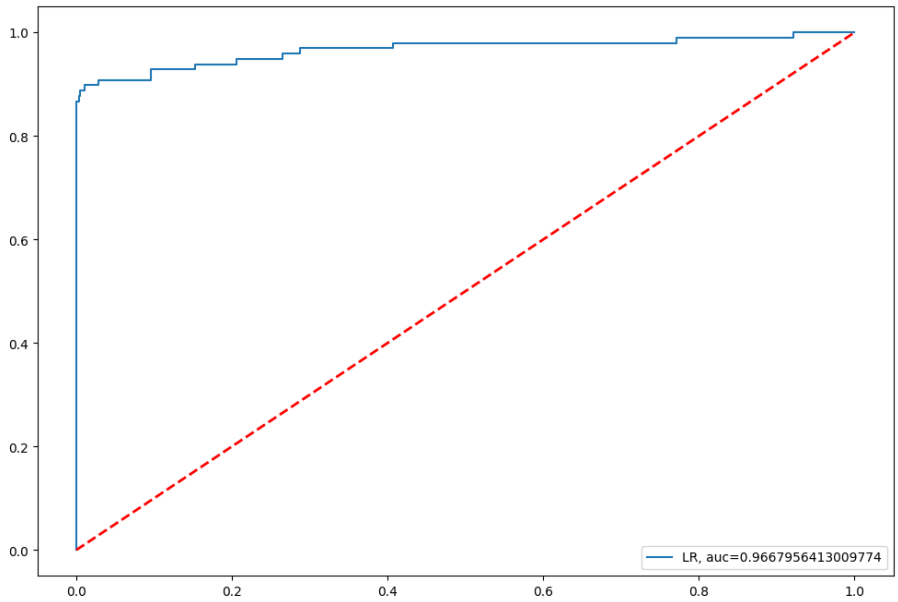


Figura 1.5. Curva ROC modelo RNA.

C. REDE AUTOENCODER

Para este modelo, foram utilizadas as mesmas métricas. Também foram geradas matrizes de confusão para o limite selecionado entre um e sete como mostrado na Tabela 1.1. Dessa forma, é possível realizar algumas considerações: diferente dos modelos apresentados anteriormente, este, apresentou uma maior classificação de resultados falsos negativos, que causariam alguns transtornos aos usuários, e uma carga adicional ao sistema onde este modelo estaria sendo executado; mas, também, mostrou como a acurácia pode ser ilusória em alguns casos, mesmo classificando muitas transações erroneamente - a pior acurácia foi de 94.93%, um resultado que, se apresentado sem outras informações, trazem a falsa ilusão de algo bom. Uma boa métrica para ver o comportamento do modelo em *datasets* desbalanceados é a curva ROC que para este modelo atingiu 77.21%, como mostrado na Figura 1.6.

Tabela 1.1. Resultados Rede Autoencoder.

Limite	VP	FN	FP	VN	Acurácia (%)
1	53982	2882	15	83	94.91
2	55565	1299	18	80	97.68
3	56023	841	26	72	98.47
4	56261	603	33	65	98.88
5	56398	466	39	59	99.11
6	56485	379	44	54	99.25

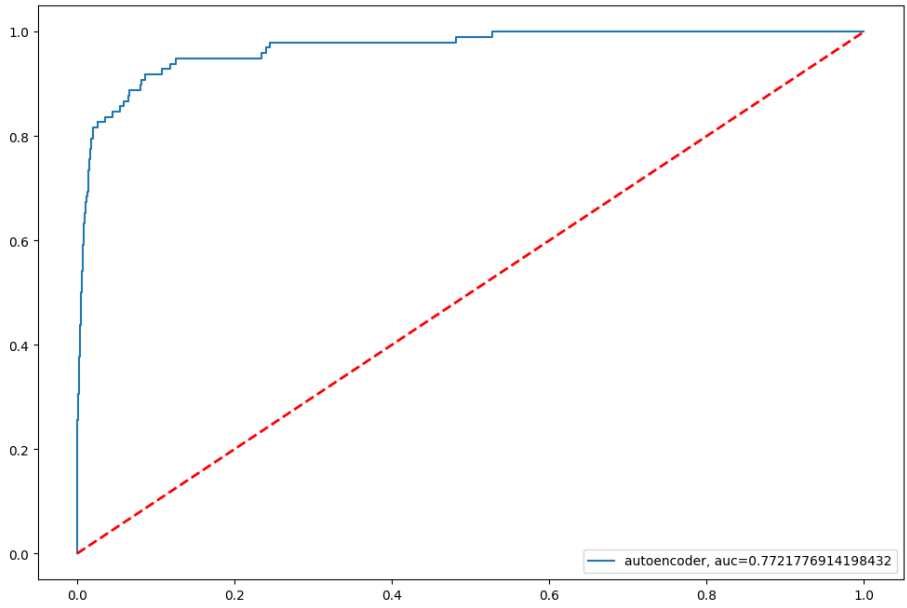


Figura 1.6. Curva ROC modelo autoencoder.

Para a implementação do *ensemble* foi escolhido o limite 4.

D. NAIVE BAYES

Este modelo tem como saída uma tupla de probabilidades, sendo que é classificada de modo binário quando seu percentual maior para transação fraudulenta (1) ou transação legítima (0). As avaliações utilizadas foram as mesmas dos modelos já apresentados. Na Figura 1.7 é possível analisar a matriz de confusão.

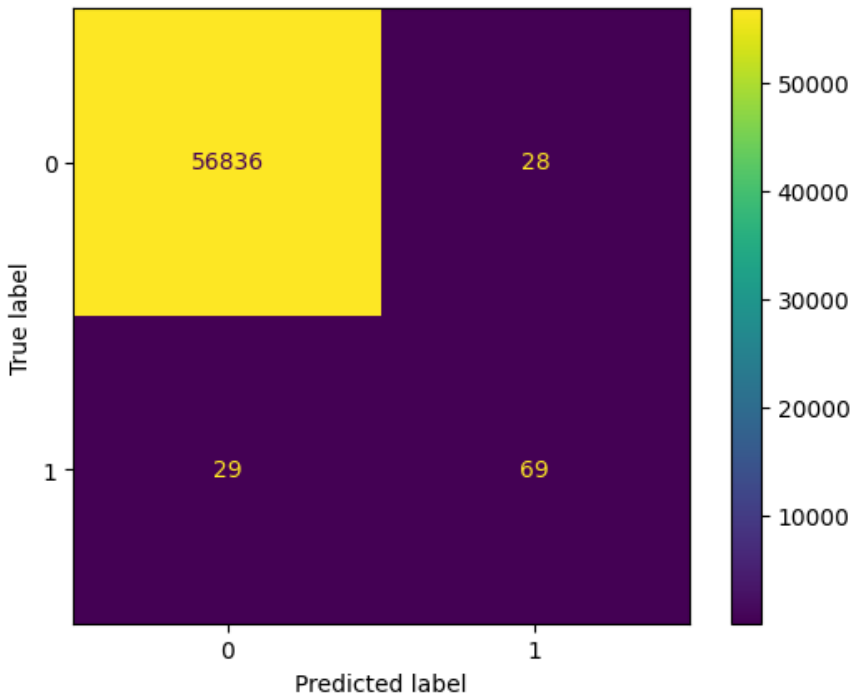


Figura 1.7. Matriz de confusão *Naive Bayes*.

A partir dessa matriz, calculou-se a acurácia, que foi de 99.89%. Também é possível verificar o número de classificações: 56836 classificadas corretamente como legítimas, 69 classificadas corretamente como transações fraudulentas, 28 classificada erroneamente como transação fraudulenta e 29 classificadas erroneamente como transações legítimas. A Figura 1.8 apresenta a curva ROC com um desempenho de 96.83%.

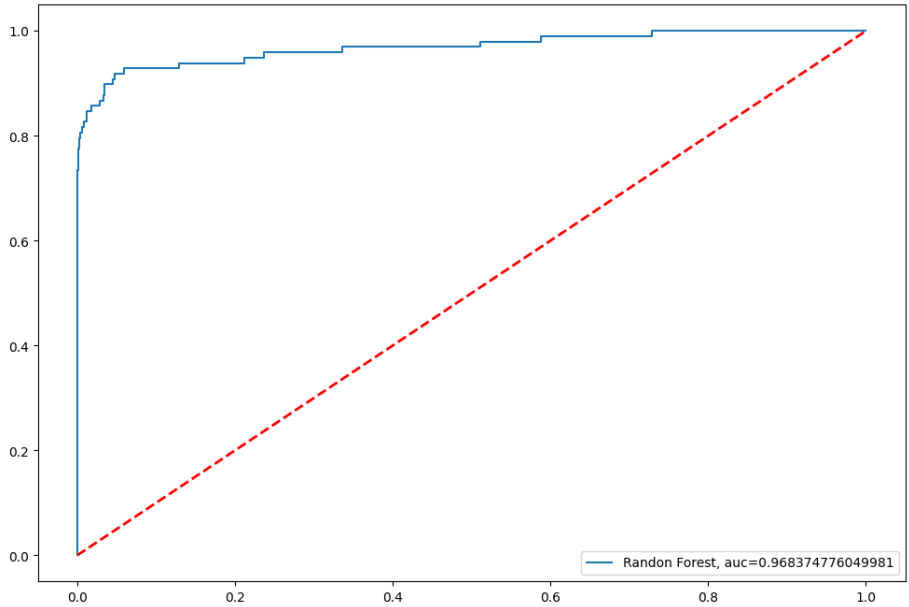


Figura 1.8. Curva ROC modelo *Autoencoder*.

F. ENSEMBLE

Por fim, apresentam-se os resultados das junções dos modelos, nos quais a saída teve peso igual para todos. As avaliações utilizadas foram as mesmas dos modelos já apresentados. Na Figura 1.9 é possível analisar a matriz de confusão. Através dela, calculou-se a acurácia que foi de 99.92%; também é possível verificar o número de classificações, 56859 classificadas corretamente como legítimas, 63 classificadas corretamente como transações fraudulentas, 5 classificada erroneamente como transação fraudulenta e 35 classificadas erroneamente como transações legítimas. Na Tabela 1.2 é possível ver os resultados individuais de cada modelo e o resultado do *ensemble*.

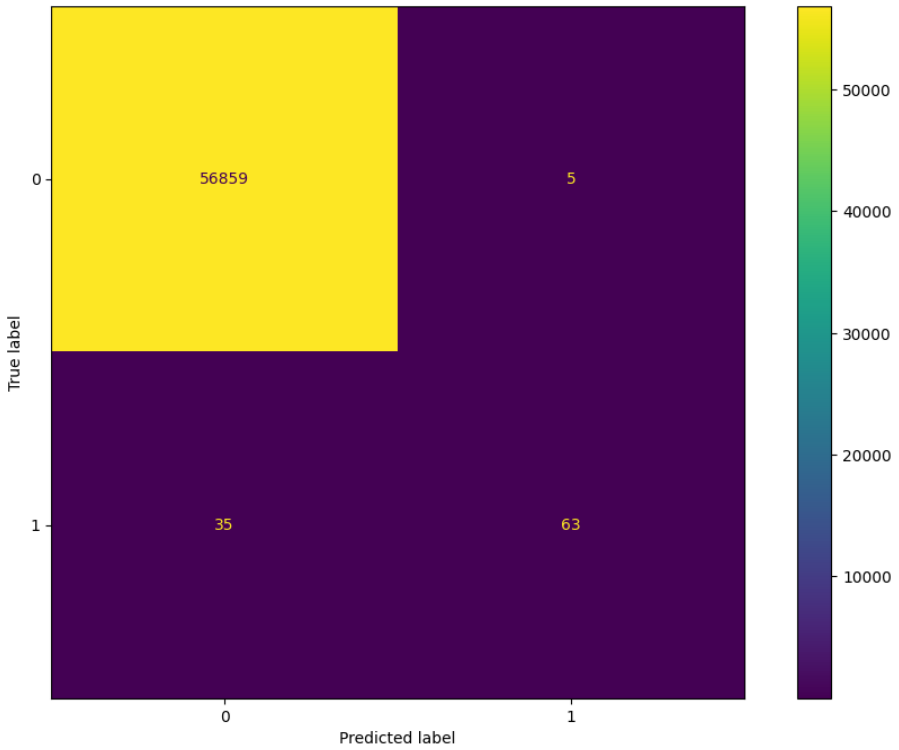


Figura 1.9. Matriz de confusão *Ensemble.*

V. DISCUSSÃO DOS RESULTADOS

Com base na Tabela 1.2, é possível realizar uma comparação abrangente dos resultados de todos os modelos. O conjunto de dados foi dividido em 80% para treinamento e 20% para testes, sendo aplicado de forma uniforme a todos os modelos, garantindo, assim, a igualdade na utilização dos dados.

Ao examinar a acurácia, observou-se que todos os modelos apresentaram resultados promissores. No entanto, para análises mais detalhadas, é necessário avaliar as colunas referentes à matriz de confusão, que proporcionam *insights* mais precisos.

Tabela 1.2: Resultados de todos os modelos.

Modelo	VP	FN	FP	VN	Acurácia %)	ROC (%)
RN	56863	1	28	70	99.94	95.27
RNA	56852	12	21	77	99.94	96.67
NB	56836	28	29	69	98.89	96.83
AUT	56261	603	33	65	98.88	75.76
ESBLE	56859	5	35	63	99.92	

Analisando a coluna de Falsos Negativos (FN), uma grande disparidade se fez notável entre os modelos *Random Forest e Autoencoder*. O modelo *Random Forest* classificou apenas uma transação legítima como fraude, o que indica que o impacto de bloqueios ou decisões equivocadas seria mínimo, evitando possíveis insatisfações dos usuários. Em contraste, o modelo *Autoencoder* classificou erroneamente 603 transações legítimas, devido à sua natureza de aprendizado não supervisionado. A Rede Neural Artificial (RNA) também obteve um desempenho sólido na coluna de Falsos Negativos (FN) e conseguiu reduzir a quantidade de Falsos Positivos (FP), ou seja, transações fraudulentas erroneamente classificadas como legítimas. Isso tem potencial para reduzir perdas financeiras. No que diz respeito ao modelo *Ensemble,* a coluna FN apresentou um resultado positivo, enquanto a FP foi um pouco mais elevada do que nos outros modelos. Vale destacar que este modelo combina todos os modelos anteriores. No entanto, ele poderia ser implementado com apenas um ou dois modelos, como o *Random Forest* e a RNA, treinados e ponderados de forma igual ou diferente. Esses dois modelos demonstraram melhores resultados para o cenário em questão. Portanto, a análise detalhada da matriz de confusão revelou-se nuances importantes sobre o desempenho e as peculiaridades de cada modelo na detecção de fraudes em transações de cartões de crédito.

VI. CONCLUSÃO

Este estudo abrangeu uma investigação diversificada de modelos destinados à detecção de fraudes em transações de cartões de crédito, cada um com suas características distintas. Os resultados obtidos através desses modelos demonstraram uma boa acurácia em geral. No entanto, um ponto importante que merece atenção é a ocorrência de falsos negativos e seus possíveis impactos sobre os usuários. Isso se deve ao fato de que transações legítimas podem ser equivocadamente bloqueadas, resultando em insatisfação por parte dos usuários.

Uma análise crítica deve ser feita sobre os falsos positivos. Falsos negativos são um problema porque transações legitimas são bloqueadas. Porém, o principal problema pode ser os falsos positivos, pois são transações fraudulentas que são autorizadas. Num universo de 98 transações fraudulentas, 35 terem sido autorizadas acaba sendo inviável de se utilizar o modelo.

Entre as abordagens modeladas, destaca-se a rede neural *autoencoder*, que representa o único modelo baseado em aprendizado não supervisionado. Essa abordagem possui potencial para exploração mais aprofundada em trabalhos futuros. Isso é especialmente relevante porque, em muitos cenários, a disponibilidade de dados rotulados para o treinamento das redes é limitada. Portanto, a investigação mais profunda dessa técnica pode revelar insights valiosos e melhorar a eficácia da detecção de fraudes em transações de cartões de crédito.

REFERÊNCIAS

ACFE. Report To the Nations on Occupational Fraud and Abuse. 2008. Disponível em: <https://www.acfe.com/uploadedFiles/ACFE_Website/Content/documents/2008-rttn.pdf>. Acesso em: 09 dez. 2022.

AMAZON. O que é uma rede neural? 2020. Disponível em: <https://aws.amazon.com/pt/what-is/neural-network/>. Acesso em: 09 dez. 2022.

AWOYEMI, John O.; ADETUNMBI, Adebayo O.; OLUWADARE, Samuel A.. Credit card fraud detection using machine learning techniques: a comparative analysis. 2017 International Conference On Computing Networking And Informatics (Iccni), out. 2017. IEEE. http://dx.doi.org/10.1109/iccni.2017.8123782. Disponível em: https://ieeexplore.ieee.org/abstract/document/8123782. Acesso em: 14 jun. 2023.

AZEVEDO, Frederico Luis de; KOMATI, Karin Satie; SEIBEL JÚNIOR, Hilário. Detection of Credit Card Fraud in a Brazilian database using Autoencoder Neural Network. Procedings do XV Simpósio Brasileiro de Automação Inteligente, [S.L.], v. 1, n. 1, p. 1-10, dez. 2021. SBA Sociedade Brasileira de Automática. http://dx.doi.org/10.20906/sbai.v1i1.2796.

BORGNE, Yann-Aël Le. Reproducible Machine Learning for Credit Card Fraud Detection - Practical Handbook. 2021. Disponível em: https://fraud-detection-handbook.github.io/fraud-detection-handbook/Chapter_2_Background/FDS.html. Acesso em: 09 dez. 2022.

CNDL. Mais de 12 milhões de consumidores sofreram alguma fraude financeira nos últimos 12 meses, aponta pesquisa CNDL/SPC Brasil. Disponível em: Mais de 12 milhões de consumidores sofreram alguma fraude ... https://www.spcbrasil.org.br › uploads › 2019/08. Acesso em: 09 dez. 2022.

CONCIL. De quem é a responsabilidade das fraudes no cartão? 2018. Disponível em: https://www.concil.com.br/blog/de-quem-e-a-responsabilidade-das-fraudes-no-cartao/. Acesso em: 09 dez. 2022.

DEEP Learning Book: Introdução aos Autoencoders. Introdução aos Autoencoders. 2022. Data Science Academy. Disponível em: https://www.deeplearningbook.com.br/introducao-aos-autoencoders/. 01 fev. 2023.

Didática Tech. "O que é e como funciona o algoritmo Random Forest." Didatica Tech, 20 jul. 2023, https://didatica.tech/o-que-e-e-como-funciona-o-algoritmo-randomforest/.

EXPERIAN, Serasa. Ano de 2021 bate recorde com mais de 4 milhões de tentativas de fraude, revela Serasa Experian. 2021. Disponível em: https://www.serasaexperian.com.br/sala-de-imprensa/analise-de-dados/ano-de-2021-bate-recorde-com-mais-de-4-milhoes-de-tentativas-de-fraude-revela-serasa-experian/. Acesso em: 09 dez. 2022.

FONTANA,Eltin. Introdução aos Algoritmos de Aprendizagem Supervisionada. 2020. Universidade Federal do Paraná, 2020.

ILEBERI, Emmanuel. Performance Evaluation of Machine Learning Methods for Credit Card Fraud Detection Using SMOTE and AdaBoost. Ieee. South Africa, p. 10-19. 2021.

KAGGLE, Credit Card Fraud Detection, fev. 2023. Disponivel em: https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud

KIM, Eunji. Champion-challenger analysis for credit card fraud detection: Hybrid ensemble and deep learning. Expert Systems With Applications. Seoul, p. 214-224. dez. 2019.

MADHURYAA, M J. Exploratory analysis of credit card fraud detection using machine learning techniques. Global Transitions Proceedings. Fisciano, p. 31-37. fev. 2022. Disponível em: https://www.keaipublishing.com/en/journals/global-transitions-proceedings/. Acesso em: 09 dez. 2022.

SANTIAGO, Gabriel Preti. Um processo para modelagem e aplicação de técnicas computacionais para detecção de fraudes em transações eletrônicas. 2014. Dissertação (Mestrado em Ciência da Computação) - Universidade de São Paulo, 2014.

2.
Previsão de Energia Natural Afluente Utilizando Redes Neurais Recorrentes

Discente: Bruna Martini

Prof. Orientador: Sérgio Luciano Avila

RESUMO Energia Natural Afluente (ENA) é a quantidade de água recebida por uma usina hidrelétrica que pode ser transformada em energia. Assim, ENA influencia diretamente no custo da energia, pois é um dado ligado à disponibilidade da água. Este artigo analisa séries temporais de dados históricos, públicos e disponíveis para capturar tendências futuras de comportamento da ENA. Para a obtenção dessa previsão, optou-se por uma rede neural recorrente (RNN) do tipo *Long-Short-Term Memory* (LSTM). A literatura demonstra que a LSTM lida de forma adequada com dependências de longo prazo nas séries temporais, bem como com previsões em comportamentos bem dinâmicos. Os dados que foram utilizados são do Operador Nacional do Sistema (ONS) e incluem como informações a precipitação e a vazão natural. As métricas de avaliação são detalhadas e explicadas ao decorrer do artigo, assim como as configurações de hiperparâmetros ajustados para otimizar o desempenho do modelo. Os resultados indicam que o modelo aqui proposto é eficaz na previsão de ENA, com acurácia superior à 95%. Além disso, existe forte correlação entre os resultados aqui obtidos com o Modelo de Previsão Chuva-Vazão (SMAP), ferramenta oficial da ONS, validando a eficácia do modelo proposto.

PALAVRAS-CHAVE: Energia Natural Afluente, Séries Temporais, Previsão, Redes Neurais Recorrentes, Redes LSTM, Dados SMAP, Ajuste de Hiperparâmetros, Validação Cruzada.

I. INTRODUÇÃO

A previsão precisa da Energia Natural Afluente (ENA) é fundamental para o setor de energia hidrelétrica, pois impacta diretamente o planejamento da operação de usinas e a tomada de decisões estratégicas (IBGE, 2018). A Energia Natural Afluente (ENA) é a quantidade de água recebida por uma usina hidrelétrica que pode ser transformada em energia. Ela é um dado importante para o acompanhamento da evolução dos recursos hidroenergéticos do país e a eficiência de cada usina. A previsão de ENA determina a capacidade de energia que a usina poderá gerar a cada período, isso impacta diretamente no preço da energia (ANEEL, 2020). Neste artigo, será explorada a aplicação de uma rede neural recorrente (RNN), em específico a rede *Long Short Term Memory* (LSTM) (Brownlee, 2020), para a previsão da ENA. Comparam-se os resultados obtidos com o Modelo de Previsão Chuva-Vazão (SMAP), modelo oficial proposto pelo Operador Nacional do Sistema (ONS) (ONS, 2019).

II. SERIES TEMPORAIS

As séries temporais têm se mostrado uma abordagem muito eficaz e apropriada para entender o comportamento de problemas, por apresentar a capacidade de modelar e compreender os padrões complexos inerentes de dados sequenciais e temporais (Moretin; Toloi, 2004). A análise de dados está cada vez mais presente no cotidiano e está se mostrando cada vez mais importante em diversos aspectos (Freund; Lavine; Schapire, 1997).

Pode-se considerar que os dados de ENA são altamente suscetíveis às variações climáticas, com grande variabilidade em curto espaço de tempo. Assim, é importante que o método utilizado para prever esse dado tenha a habilidade de lidar com essas incertezas e flutuações de dados. Na análise de séries temporais é possível criar intervalos de confiança e análises de sensibilidades.

Além disso, é possível incorporar informações auxiliares, como dados meteorológicos, topográficos e informações do SMAP (modelo

matemático chuva-vazão) na análise de ENA. A integração desses dados, caracteriza uma estrutura flexível, contribui para um modelo mais completo e eficaz na previsão de ENA, capaz de estabelecer relações mais complexas com os fatores ambientais.

III. MODELOS LSTM

As redes LSTM emergem como uma arquitetura notável na previsão de ENA, proporcionando a capacidade essencial de capturar dependências de longo prazo em dados sequenciais. Essa arquitetura superou as limitações das redes neurais tradicionais e se tornou um alicerce poderoso para a análise de dados temporais complexos (Hosseinzadeh; Dastjerdi, 2018).

As LSTM são uma variação das RNNs, projetadas para lidar com o problema do "desaparecimento do gradiente". Convém destacar que o gradiente pode se tornar muito pequeno à medida que é retro propagado para as camadas anteriores, as quais recebem atualizações de peso muito baixas, o que torna o treinamento dessas camadas ineficaz. Esse é um problema que impacta na capacidade das RNNs tradicionais capturarem correlações de longo prazo, cujas dependências temporais podem se estender com longos períodos. Na arquitetura LSTM é utilizado células de memória, que desempenham um papel fundamental no armazenamento e consulta das informações ao longo do tempo (Gers; Schmidhuber; Cummins, 2000).

Na arquitetura LSTM as seguintes equações são fundamentais (Schuster; Palla, 1997):

1. Atualização da célula – o objetivo desta equação é decidir quais informações antigas da célula de memória devem ser mantidas e quais informações novas devem ser armazenadas:

$$f_t = \sigma(w_f \cdot [h_{t-1}, x_t] + b_f) \qquad 2.1$$

onde f_t é o vetor de ativação da porta de atualização, w_f e b_f são os pesos e bias associados à porta de atualização, h_{t-1} é o estado oculto

anterior, x_t é a entrada na etapa de tempo, σ é a função sigmoid que transforma os valores em um intervalo de [0, 1].

2. Criação da célula – calcula o candidato a célula de memória:

$$c_t = tanh(w_c \cdot [h_{t-1}, x_t] + b_c) \qquad 2.2$$

onde c_t é o candidato a estado de célula na etapa de tempo, w_c e b_c são os pesos e bias associados ao candidato da célula.

3. Atualização do estado da célula: esta equação combina o estado da célula anterior com o candidato da célula para atualizar o estado da célula atual.

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \overline{C_t} \qquad 2.3$$

onde c_t é o estado da célula atual, f_t é o vetor de ativação da porta de atualização, i_t é o vetor de ativação da porta de entrada, e $\overline{C_t}$ é o novo candidato para o estado da célula no tempo.

4. Atualização do estado oculto – atualiza o estado oculto com base no estado da célula atual.

$$h_t = o_t \cdot tanh(c_t) \qquad 2.4$$

onde é o estado oculto na etapa de tempo e o_t é o vetor de ativação da porta de saída.

Essas células de memória são projetadas para manter as informações por um período prolongado, assim é possível capturar as dependências temporais a longo prazo em dados sequenciais. Além disso, as LSTMs podem controlar o fluxo de informações de maneira com que certos aspectos sejam mantidos ou descartados conforme o sistema vai se aprimorando (Pereira, 2017).

As LSTMs também utilizam de portões, de entrada, esquecimento e saída (Pereira, 2017). Os portões de entrada são destinados a

determinar quais informações novas serão armazenadas na célula de memória. Já os portões do esquecimento permitem controlar quais informações podem ser descartadas, e os portões de saída da célula de memória filtram as informações relevantes para a próxima etapa.

Os fatores que englobam a previsão de ENA, como os fatores climáticos e sazonalidades podem ter influências complexas nos padrões de fluxo de água, por isso é de suma importância habilidade das LSTMs em lidar com dependências de longo prazo. Através de suas células de memória e mecanismos de portão, as LSTMs podem recolher informações relevantes de períodos anteriores e usá-las para fazer previsões precisas.

Outra característica vantajosa das LSTMs é a capacidade de processar sequências de comprimentos distintos, tornando-as adequadas para a análise de séries temporais com variações na duração dos padrões. Além disso, a arquitetura LSTM pode ser empilhada em camadas, permitindo a construção de modelos mais complexos para capturar as nuances nas dependências temporais (PEREIRA, 2017).

Em resumo, as redes LSTM são utilizadas como base na previsão de séries temporais, como a ENA, devido à capacidade de capturar dependências de longo prazo em dados sequenciais, superar as limitações que as redes RNNs apresentam, apresentar a habilidade de guardar informações de longo prazo e processar sequências de diferentes comprimentos.

IV. COLETA E PREPARAÇÃO DOS DADOS

Para a previsão de ENA por LSTM serão utilizados os dados disponibilizados pelo ONS (ONS, 2021), que é o ENA mensal por bacia hidrográfica, o qual representa a capacidade produtiva da usina e é calculado considerando as vazões naturais descontadas das vazões vertidas nos reservatórios. Para a análise de séries temporais o principal dado levado em consideração é a carga própria de energia, medida em MWmed. Para o cálculo de ENA, considera-se que o nível da água do reservatório precisa se manter entre os valores mínimo e

máximo para operação. Cada reservatório tem uma faixa preestabelecida para a taxa de variação diária do nível de água, visto que a vazão de água que passa pelas turbinas da usina precisa ser compatível com a geração de aproveitamento.

Ao analisar os dados, observou-se uma grande variação no volume durante o ano; porém, é visível também que há semelhanças entre os dados anuais, o que leva a pensar que se pode reconhecer padrões dentro dos dados de ENA.

Para a análise, foram utilizados os dados da bacia Pimentel em um intervalo de tempo de 2 anos. Os dados foram extraídos para uma tabela onde constava a data, a precipitação diária (produto do MERGE/CPTCE-INPE referente à média espacial na área da sub-bacia Pimentel) e à vazão natural observada medida em mm/dia e em m^3/s. O conjunto total dos dados foi dividido em dados para treinamento e teste (80/20). A divisão respeitou a ordem temporal dos dados. Em seguida, o modelo foi treinado; para isso se criaram camadas de rede neural com camadas LSTM densas. Após treinar e validar o modelo, foi analisado o seu desempenho e foram exibidas as métricas de avaliação.

V. MÉTRICAS PARA MEDIR A ACURÁCIA

Para a avaliação do modelo foram usadas diferentes métricas de erro. A raiz quadrada do erro quadrático médio (RMSE) fornece uma medida da magnitude média do erro entre as previsões e os rótulos, ela fornece uma medida geral da precisão do modelo, indicando o quão próximas as previsões estão dos valores reais, isso ajuda a avaliar o desempenho geral do modelo. Essa métrica é calculada da seguinte forma (Urzagasti, 2021):

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \qquad 2.5$$

onde N é o número de exemplos no conjunto de dados, y_i são os valores reais, $\hat{y}_i$ são valores previstos pela rede.

A pontuação de variância explicada (EVS) indica a proporção da variância dos rótulos que é explicada pelas previsões, ela fornece uma visão sobre a qualidade das previsões em termos de variação dos dados, sendo utilizada para verificar se o modelo é capaz de capturar as tendências e padrões dos dados, dada por:

$$EVS = 1 - \frac{Var(y - \hat{y})}{Var(y)} \qquad 2.6$$

onde Var é a variância, y_i são os valores reais, $\hat{y}_i$ são valores previstos pela rede.

Além disso, foi incluído o erro absoluto médio (MDAE), que calcula a mediana dos erros absolutos entre as previsões e os rótulos, sendo útil quando os valores extremos podem afetar adversamente as métricas de erro médio, dado por:

$$MDAE = median(| y - \hat{y} |) \qquad 2.7$$

onde y_i são os valores reais, $\hat{y}_i$ são valores previstos pela rede.

Outras métricas mais conhecidas também foram usadas como a *Mean Squared Error* (MSE), que calcula a média dos erros ao quadrado entre as previsões e os rótulos, sendo muito utilizada para avaliar a precisão do modelo. Ela é particularmente sensível a erros maiores e pode ajudar na identificação de quando o modelo comete erros significativos. A *Mean Absolute Error* (MAE), que calcula a média dos valores absolutos dos erros entre as previsões e os rótulos. R^2 Score (R^2), coeficiente de determinação R^2 entre as previsões e os rótulos. *Mean Absolute Percentage Error* (MAPE), média da porcentagem absoluta dos erros entre as previsões e os rótulos (De Myttenaer, 2016). Cada métrica fornece uma perspectiva única sobre o desempenho do modelo, por isso a importância de analisar várias métricas. A intenção é obter uma compreensão mais completa de como o modelo está desempenhando.

VI. CONFIGURAÇÃO DE HIPERPARÂMETROS

É sempre necessário realizar ajustes nos hiperparâmetros do modelo, como o número de neurônios nas camadas LSTM, alteração na taxa de aprendizagem, *batch size* e *epochs*. *Batch size* é a quantidade de exemplos utilizados de uma só vez para ensinar o modelo a aprender, como se fosse lotes de aprendizado, e *epochs* é o número de vezes que o modelo passa por todas as informações de treinamento.

Chegou-se a um resultado satisfatório utilizando 20% dos dados para teste, um *batch size* de 512, 100 *epochs* e uma taxa de aprendizado de 0,001. Com isso, observou-se um MSE baixo, MAE satisfatório e um R^2 alto, mas ainda apresentava margens para aprimoramento do modelo. Após ajustes, como a mudança da taxa de aprendizado para 0,01, 150 *epochs* e utilizando 15% dos dados para teste, chegou-se às seguintes métricas: MSE de 0,003228, MAE de 0,039563 e R^2 de 0,997835. Os resultados são mostrados na Tabela 2.1 e nas Figuras 2.1 e 2.2.

TABELA 2.1. Métricas de Acurácia

MSE	MAE	R^2	MAPE	RMSE	EVS	MDAE
0,0032	0,0395	0,9978	0,0660	0,0568	0,9982	0,0310

A Figura 2.1 mostra a evolução da função de perda ao longo do treinamento do modelo, através dele é possível verificar que o modelo está aprendendo de forma eficaz.

A Figura 2.2 apresenta a vazão natural prevista e a vazão natural observada durante os anos de 2020 e 2021. É possível verificar que as duas linhas referentes à vazão estão caminhando juntas no decorrer do tempo. Isso significa que o modelo foi capaz de prever os dados para aquele período de forma eficiente, alcançando um resultado satisfatório.

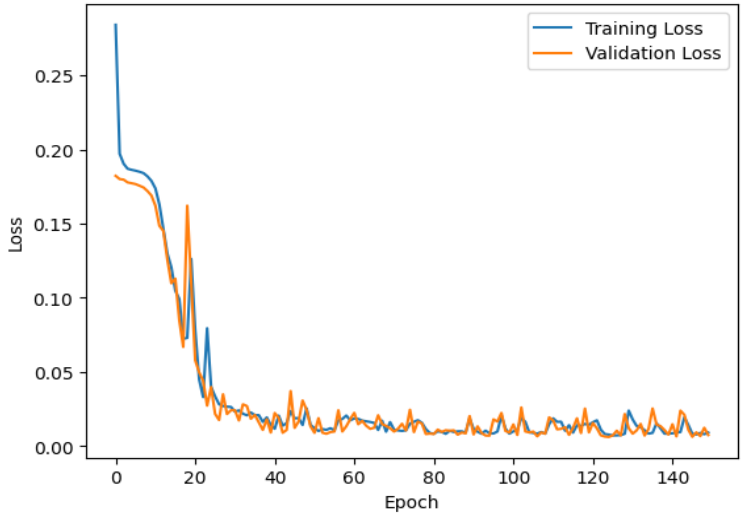


FIGURA 2.1. Gráfico da perda dos dados de treinamento e teste.

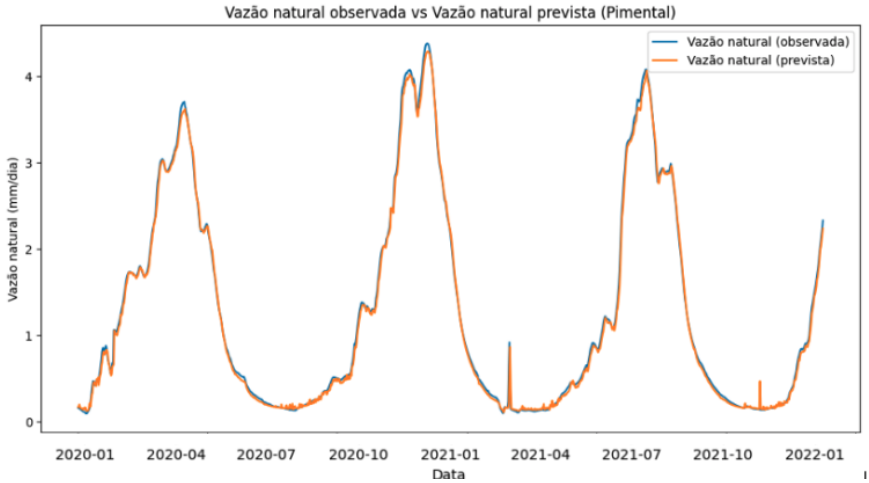


FIGURA 2.2. Gráfico que relaciona a vazão natural prevista e a vazão natural observada na bacia Pimentel.

VII. VALIDAÇÃO CRUZADA

A validação cruzada é uma técnica muito utilizada para evitar o *overfitting,* que ocorre quando um modelo de aprendizado de máquina é ajustado excessivamente e passa a capturar o ruído aleatório presente nos dados (Guimarães, 2008). Quando o modelo sofre de *overfitting* ele pode parecer muito preciso em relação aos dados de treinamento e extremamente imprecisos quando aplicados aos dados de teste, pois ele está muito ajustado aos detalhes específicos dos dados de treinamento que não se repetiram no futuro. Na validação cruzada o conjunto de dados é dividido em diversas partes e é realizado múltiplas interações de treinamento e teste, permitindo que o conjunto de dados seja testado de diversas formas e fornece uma visão mais abrangente do desempenho (Guimarães, 2008). Importante ter a preocupação de respeitar a dependência temporal entre as variáveis. Então, primeiramente os dados são divididos em subconjuntos de dados aproximadamente iguais, e em seguida são realizadas diversas interações. Em cada interação, um dos subconjuntos será utilizado como conjunto de teste e os outros subconjuntos serão usados como conjuntos de treinamento. A cada interação as métricas de validação são calculadas, e a média das métricas de desempenho em todas as interações também são calculadas, isso fornece uma estimativa mais robusta do desempenho do modelo.

Esse processo permite a análise de cenários de treinamento e teste distintos, incluindo variações nos dados de entrada, configurações nos hiperparâmetros e integração com os dados do SMAP. Dos resultados extraídos dessa validação, podem oferecer informações valiosas sobre quais abordagens são mais eficazes em diferentes contextos, bem como quais configurações do sistema produzem previsões mais assertivas. A validação cruzada é um componente de extrema importância na seleção do melhor modelo a ser utilizado. No decorrer das interações, o desempenho de modelos distintos é avaliado e comparado, o modelo que apresenta o melhor desempenho é selecionado, garantindo uma avaliação mais abrangente.

Ao implementar a validação cruzada no modelo utilizado os resultados obtidos melhoraram, porém a mudança não foi significativa nos resultados, o MSE obtido foi de 0,003194 e R^2 de 0,997858, conforme mostra a Figura 2.3.

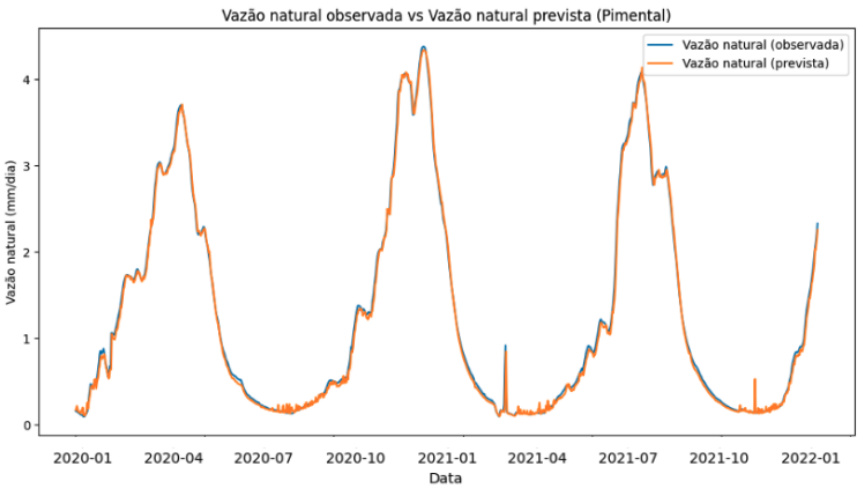


FIGURA 2.3. Gráfico que relaciona a vazão natural prevista e a vazão natural observada na bacia Pimentel.

A validação cruzada garante que os modelos sejam avaliados de maneira completa e confiável, permitindo a seleção do melhor modelo e fornecendo insights sobre a eficácia das abordagens adotadas.

VIII. COMPARAÇÃO COM OS DADOS DO SMAP

Ao comparar os resultados das previsões geradas pelo programa e os dados calculados pelo SMAP, é visível uma semelhança entre os dois conjuntos de dados, conforme pode ser visto na Figura 2.4.

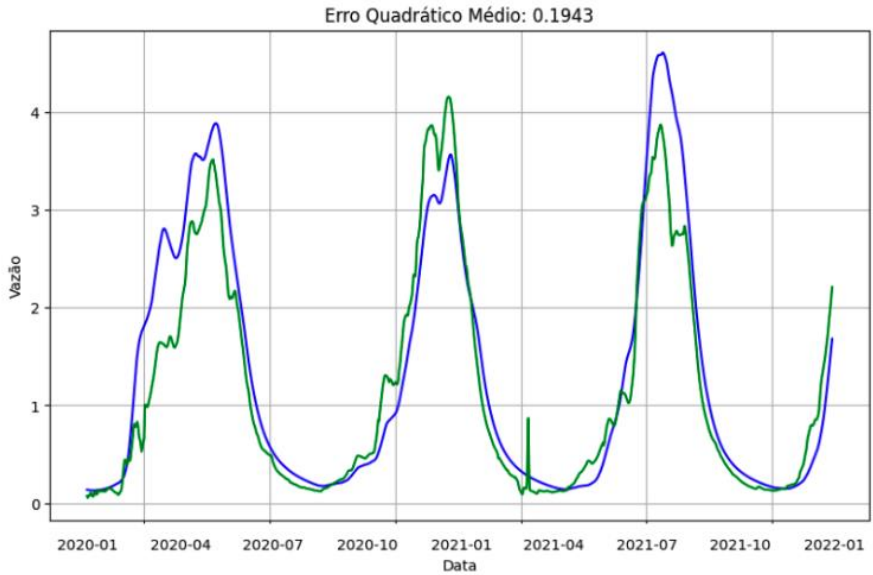


FIGURA 2.4. Gráfico referente a curva da vazão calculada pelo SMAP (Azul) e da vazão prevista (verde).

A curva em verde representa a vazão prevista, enquanto a curva em azul é referente à vazão calculada do SMAP. As curvas possuem suas semelhanças, mas notoriamente não são idênticas, o erro quadrático médio entre as curvas é de 0,1943.

A abordagem aplicada para a previsão de ENA demonstra uma consistência notável com os resultados obtidos a partir dos dados fornecidos pelo SMAP. A comparação detalhada dos valores previstos e observados revela que a diferença média entre os dois conjuntos de dados é baixa, destacando a eficácia da abordagem de modelagem. Os resultados do programa capturaram com precisão os fenômenos hidrológicos subjacentes, sugerindo que o modelo está capturando padrões relevantes de forma confiável. Por fim, a análise detalhada dos resultados apresentou uma coerência entre as previsões produzidas pelo LSTM e pelo SMAP. A semelhança encontrada reforça a confiabilidade das previsões de ENA geradas.

IX. CONCLUSÃO

A capacidade de prever com precisão a disponibilidade de recursos hídricos é de extrema importância para otimizar o uso de recursos e mitigar os riscos do setor elétrico.

Esta pesquisa teve como objetivo encontrar padrões complexos e variações sazonais nos dados de vazão de uma hidrelétrica. Isso foi possível através das técnicas utilizadas. O acréscimo de dados adicionais, como os fornecidos pelo SMAP, permitiu ajustar ainda mais a precisão da previsão.

É notável que, além da precisão, a capacidade de adaptação do modelo por LSTM foi destacada. A modelagem de séries temporais permitiu que a previsão de ENA fosse feita em escalas que variam de diárias e sazonais, assim foi possível ter uma visão holística das variações da disponibilidade da água.

Por mais que os resultados apresentados sejam satisfatórios, a previsão de ENA continua sendo um desafio complexo devido à natureza essencialmente oscilante dos recursos hídricos. À medida que a modelagem de séries temporais e o uso de dados vão evoluindo, pode-se antecipar melhorias na precisão e na confiabilidade dos resultados da previsão de ENA.

Além das análises apresentadas neste artigo, é importante destacar que existem frentes para pesquisas futuras, como, incorporar os dados em tempo real com informações meteorológicas recentes, explorar modelos de séries temporais mais avançados e técnicas de aprendizado de máquina além das LSTM, a fim de avaliar se algum modelo diferente pode aprimorar ainda mais os resultados. Pode-se, também, analisar a sensibilidade dos dados para entender qual deles tem o maior peso na hora da previsão. Por fim, realizar um estudo de impacto, para verificar como o uso das previsões de ENA afetam o planejamento e a operação das usinas, mostrando os benefícios econômicos.

REFERÊNCIAS

(ANEEL, 2023) AGÊNCIA NACIONAL DE ENERGIA ELÉTRICA - ANEEL. Hidrelétricas. Disponível em: https://www.aneel.gov.br/geracao/hidreletricas. Acesso em: 19 out. 2023.

(BARBOSA, 2020) BARBOSA, A. de G. Análise da influência dos períodos críticos na operação otimizada do sistema de usinas hidrelétricas da bacia do rio São Francisco utilizando programação não linear. 2020.

(BASTOS, 2020) BASTOS, Í. G. P. et al. Previsão de geração fotovoltaica a partir de dados meteorológicos utilizando rede LSTM. 2020.

(BOSA, 2018) BOSA, Diego Antonio. Energia natural afluente monthly forecasting using an artificial neural networks MLPs. In: 2018 Simpósio Brasileiro de Sistemas Eletricos (SBSE). IEEE, 2018. p. 1-5.

(BROWNLEE, 2023) BROWNLEE, J. Time Series Prediction with Long Short-Term Memory Networks in Python. Machine Learning Mastery, 2020. Disponível em: https://machinelearningmastery.com/time-series-prediction-lstm-recurrent-neural-networks-python-keras/. Acesso em: 19 out. 2023.

(FERREIRA, 2022) DA COSTA FERREIRA, C.; LIMA, C. H. R. XIV Simpósio de Recursos Hídricos do Nordeste, Previsão de Vazões Naturais Afluentes a um Reservatório Utilizando a Técnica de Redes Neurais Artificiais.

(MYTTENAERE, 2016) DE MYTTENAERE, Arnaud et al. Mean absolute percentage error for regression models. Neurocomputing, v. 192, p. 38-48, 2016.

(FREUND, 1997) FREUND, Y.; LAVINE, M.; SCHAPIRE, R. E. A Short Introduction to Boosting. Journal of Japanese Society for Artificial Intelligence, v. 14, n. 5, p. 771-780, 1997.

(GALIZONI, 2021) GALIZONI, T. et al. Previsão de vazões naturais afluentes médias semanais para usinas hidrelétricas. 2021.

(GUIMARÃES, 2008) GUIMARÃES, A. M. et al. Módulo de Validação Cruzada n º Treinamento de Redes neurais Artificiais com backpropagation Algoritmos e propagação resiliente. 2008.

(GERS, 2000) GERS, F. A.; SCHMIDHUBER, J.; CUMMINS, F. Learning to forget: Continual prediction with LSTM. Neural Computation, v. 12, n. 10, p. 2451-2471, 2000.

(HOSSEINZADEH, 2018) HOSSEINZADEH, M.; DASTJERDI, A. V. A Survey on the Application of Recurrent Neural Networks to Time Series Forecasting. Neural Computing and Applications, v. 29, n. 9, p. 7-16, 2018.

(IBGE, 2018) INSTITUTO BRASILEIRO DE GEOGRAFIA E ESTATÍSTICA, Anuário Estatístico de Energia Elétrica 2018. Disponível em:

https://biblioteca.ibge.gov.br/visualizacao/periodicos/91/aee_2018.pdf. Acesso em: 19 out. 2023.

(KUKI, 2020a) KUKI, C. A. C. et al. Metodologia para previsão de preço de energia considerando incertezas de vazões. 2020.

(KUKI, 2020b) KUKI, Cassia Akemi Castro et al. Estratégias de Previsão de Vazão de Curto Prazo: um estudo de caso nas Bacias do Rio Grande e Rio Paranaíba. In: Congresso Brasileiro de Automática-CBA. 2020.

(LI, 2020) Li, N., Wang, L., Li, X., & Zhu, Q. (2020). An effective deep learning neural network model for short-term load forecasting. Concurrency and Computation: Practice and Experience, 32(7), e5595.

(MORETTIN, 2004) MORETTIN, P. A.; TOLOI, C. M. C. Análise de Séries Temporais. 2. ed. Editora Blucher, 2004.

(ONS, 2023) OPERADOR NACIONAL DO SISTEMA ELÉTRICO. Modelos Hidroenergéticos - Previsões. Disponível em: https://www.ons.org.br/sistemas-informacoes/previsao-hidroenergetica. Acesso em: 19 out. 2023.

(PEREIRA, 2017) PEREIRA, M. D. M. Aprendizado profundo: redes LSTM, 2017.

(PERIN, 2020) PERIN, M. D. Ré. Deep Reservoir Computing aplicado na previsão do preço de energia elétrica no mercado de spot. 2020.

(PIVOTO, 2022) PIVOTO, M. W. et al. Estudo do modelo SMAP/ONS para a projeção de vazões de usinas hidrelétricas. 2022.

(SCHUSTER, 1997) SCHUSTER, M.; PALLA, R. Bidirectional Recurrent Neural Networks. IEEE Transactions on Signal Processing, v. 45, n. 11, p. 2673-2681, 1997.

(URZAGASTI, 2021) URZAGASTI, Carlos Alejandro et al. Comparação da Acurácia de Modelos de Redes Neurais Artificiais na Predição da Irradiância Solar e Geração de Energia Fotovoltaica. In: Anais do XVIII Congresso Latino-Americano de Software Livre e Tecnologias Abertas. SBC, 2021. p. 53-58.

(WANG, 2017) WANG, H.; RAJ, B. On the Origin of Deep Learning. ArXiv e-prints, 2017.

3.

Machine Learning Operations: fluxo de aprendizado contínuo para transcrição de áudio em atendimento ao cliente

Discente: Lucas Gabriel Coliado Bandeira

Prof. Orientador: Pedro Giassi Junior

RESUMO Em ambientes empresariais, a função de atendimento ao cliente via chamadas de áudio é uma das mais comuns e ativas. Estas atividades recebem alto investimento com o objetivo de melhoria contínua de processos e retenção de vendas. E, entretanto, as métricas e ferramentas atuais na maioria das empresas são insuficientes para levar a um entendimento integral de uma base consumidora, muitas vezes induzindo a resultados inconsistentes e erros no setor de suporte e vendas. Uma das limitações para análises deste tipo de chamadas mais a fundo são as ferramentas de transcrição e indexação relacionadas a dados sensíveis; muitas soluções atuais requerem que dados sigilosos sejam enviados a servidores de terceiros o que podem ocasionar em vazamentos, além de custos adicionais pelo serviço. O trabalho propõe uma arquitetura para utilização de modelos de *deep learning*, tendo como estudo de caso o modelo baseado em *Transformers* chamado *Whisper* criado pela *OpenAI*, que pode ser executado em servidores conhecidos e privados, introduzindo uma *pipeline* para realização de reaprendizado do modelo para o uso exclusivo da empresa. Providencia-se um modo para que a transcrição de uma multitude de chamadas seja realizada com possibilidade de melhoria contínua sem interrupção do sistema de produção.

PALAVRAS-CHAVE: *Deep Learning, Machine Learning, Operations, Fine Tune,* Áudio, *Kubernetes, Architecture,* Português.

I. INTRODUÇÃO

Em empresas de tecnologia e serviços, a atividade de atendimento e suporte ao consumidor é um setor crucial para acompanhamento e retenção de clientes, de acordo com a *Havard Business Review* (HBR, 2021). A Khomp, empresa brasileira natural de Santa Catarina, atua no mercado de Telecomunicações onde é referência desde 1996, providenciando produtos e serviços *Business-to-Business* (B2B) voltados a soluções de telefonia, habilitando operações de *call center* em diversos estados no Brasil.

Atualmente, os meios de atendimento ao consumidor utilizados são diversos. Com o crescimento do *e-commerce* na última década, entretanto, muito da interação com clientes tornou-se estritamente virtual (Rosário, 2021). Pela natureza digital, os tipos de atendimento via telecomunicações como atendimento telefônico, e-mail, chat e mídia social se dão como imperativos neste setor (Singh, 2021). Nesse contexto, a interação de um profissional de atendimento ao consumidor com um usuário é comumente registrada via dados, denominadas fontes, que são registradas em formato de banco de dados. A análise destes dados pode ser realizada para extrair métricas de performance e melhora de serviços de marketing (Ramon, 2021), porém este tipo de classificação em forma manual torna-se impraticável com o crescimento do número de atendentes e atendimentos realizados, apresentando milhares de horas em forma de dados.

No contexto de *call centers*, onde são realizados atendimentos de voz, observa-se que já existem técnicas aplicadas para a processamento natural de linguagem; é possível utilizar procedimentos de transcrição de voz-para-texto utilizando vocabulários, como no trabalho de Floriano (2022), para comunicações entre operadores de áreas técnicas e protocoladas.

A Khomp possui sistemas de gravação de chamadas internas e de suporte ao cliente, via o protocolo *Session Initiation Protocol* (SIP), através de seus serviços, mantendo um banco de gravações que servirá como fonte de dados de alta qualidade e como alvo de transcrições futuras. Um diagrama simplificado, sem detalhes de

negócio, deste sistema pode ser visto na Figura 3.1, onde Falantes cientes de que estão utilizando o sistema tem suas conversas gravadas e armazenadas no Banco de Gravações.

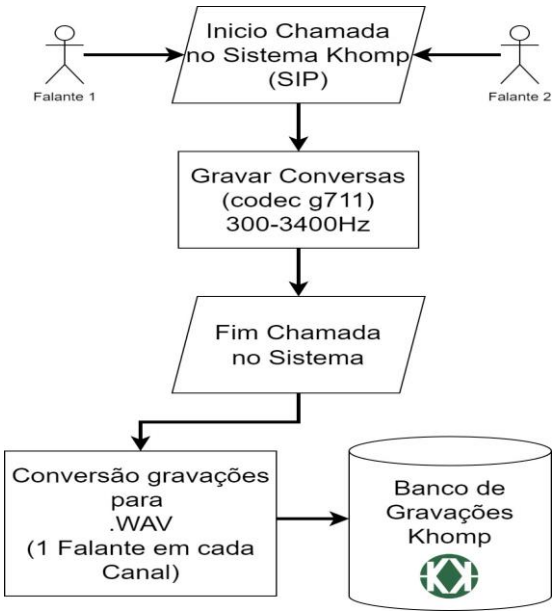


FIGURA 3.1. Fluxo simplificado de coleta de conversas no sistema de telecomunicações Khomp.

As gravações possuem uma faixa de banda curta, através do codec G.711. Entretanto, ainda existe barulho de fundo, picotes devido a conexão, e quebras de silêncio não humanas, o que apresenta desafios para atividades de transcrição. O intuito do trabalho será investigar e validar uma arquitetura capaz de realizar transcrições das conversas gravadas em um banco de gravações da Khomp. Como método, planeja-se a aplicação de redes neurais para transcrição de conversas em português brasileiro de modo que com o tempo se adeque um modelo dentro do contexto da Empresa. A análise de dados se levará em métodos de visualização e de transcrição das informações, utilizando-se de ferramentas de alta escala e disponível em português tal como OpenAPI Whisper (Radford, 2023).

O objetivo é definir uma arquitetura que implemente o procedimento para decompor chamadas telefônicas já gravadas com um formato específico, transcrevendo o áudio de vozes para texto, utilizando estado da arte em *deep learning*, e permitindo que este procedimento possa ser melhorado com o tempo dentro de um ambiente privado. O modelo é capaz de transcrever áudios cuja natureza é de baixa qualidade; faixa de espectro limitada e bi-trate baixo, dentro de uma métrica conhecida para avaliação de performance. Também é possível realizar procedimentos de melhoria do modelo de modo que a performance aumente.

II. FUNDAMENTAÇÃO TEÓRICA

A. Cultura Devops

Os requerimentos de qualidade e velocidade de entrega de software tem se tornado cada vez mais rígidos no ambiente profissional, demandando um conjunto de técnicas e habilidades específicas para possibilitar entregas razoáveis com as demandas de mercado. O agrupamento formalizado destas práticas no ambiente profissional é denominado *Development Operations* (DevOps) (Mishra, 2020); práticas de automação, testes, colaboração entre times de desenvolvimento e operação, e cultura de entregáveis que podem ser suplementados por aditivos providenciam o estado da arte em desenvolvimento de software.

B. Machine Learning

Machine Learning (ML), ou Aprendizado de máquina, é o estudo científico de algoritmos e modelos estatísticos utilizados por sistemas computacionais para a realização de tarefas específicas sem a necessidade de uma programação explícita para ela (Mahesh, 2020). Com técnicas de aprendizado de máquina, existe um maior foco em obter entendimentos e resultados a partir dos dados que, quando contrastado com implementações de regras e rotinas de um desenvolvimento de software clássico, obtém resultados satisfatórios através da generalização e estatística. Um exemplo é a classificação em figuras, onde um modelo pode ser treinado e refinado de modo supervisionado com dados de figuras previamente classificadas.

C. *Machine Learning Operations*

Projetos de ML tem como objetivo a implantação rápida de soluções. Entretanto, a operação e automação de produtos envolvendo ML é altamente desafiadora; a implantação de soluções com aprendizado de máquina muitas vezes falham ao se tornarem produtos pois apresenta um novo horizonte de práticas necessárias para os responsáveis pela arquitetura de software realizarem entregas efetivas e mensuráveis, como sugerido por (Kül, 2023). Portanto, o mercado reagiu com um novo paradigma denominado *Machine Learning Operations* (MLOps), ou Operações em Aprendizado de Máquina, onde é proposto um conjunto de melhores práticas, técnicas, e cultura de desenvolvimento ágil. A Figura 3.2. ilustra os MLOps.

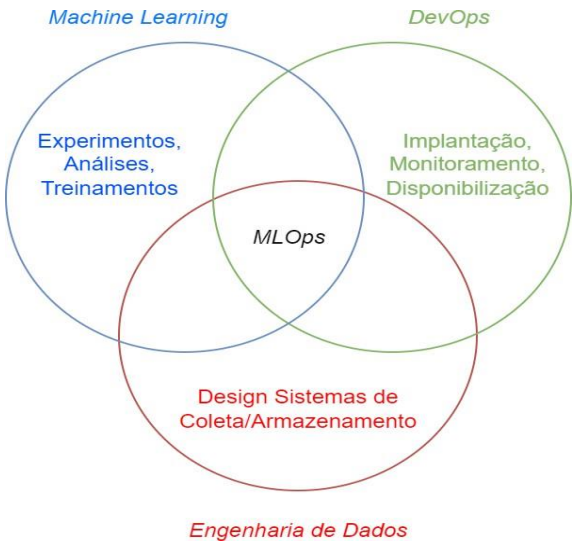


FIGURA 3.2. Diagrama de Venn descrevendo como *Machine Learning Operations* (MLOps) é a interseção de diversas disciplinas, como descrito por (Kühl, 2023).

Atualmente existem poucos exemplos em bibliografias sobre como implantar modelos de inteligência artificial profunda, entretanto a comunidade de código aberto já tem realizado esforços para disponibilizar sistemas entre tecnólogos responsáveis por utilizar

estes modelos na indústria. Diversas soluções foram abertas pela comunidade, em particular o MLFlow, que permite que cientistas de dados possam acompanhar resultados de experimentos com códigos de treino e escolher os melhores modelos para produção (Zaharia, 2018).

D. *Transformers*

O *Transformer* é uma arquitetura de rede neural identificada formalmente em 2017 (Vaswani, 2017), que diferente de outras arquiteturas, como *Long Short Term Memory* (LSTM) e *Recursive Neural Networks* (RNNs), possui excelente capacidade de paralelização e velocidade de treinamento, como também qualidade superior para atividades de Processamento natural de linguagem, ou *Natural Language Processing* (NLP), tais como atividades de tradução. Este potencial elevado se dá pelo mecanismo de atenção, que dispensa técnicas como recorrência e convolução, permitindo que a rede considere e reconheça toda a faixa de entradas através de todo o processo, mantendo o consumo de tempo e recursos razoável. A Figura 3.3 possui uma exemplificação de uma arquitetura de *transformer,* onde o bloco da esquerda ilustra o bloco lógico responsável pela atividade de *encoding* das entradas, e o bloco da direita é responsável pelo *decoding,* ou as saídas. Alguns modelos de Transformers podem ser compostos pelo *encoder, decoder* ou ambos.

E. Chamadas de Áudio

Para armazenamento de áudio, é realizado o processo de *encoding* descrito por chamados codecs. O codec G.711, por exemplo, descreve o algoritmo para *encoding e decoding* áudios na frequência de Narrowband (100Hz até 3700Hz) em uma taxa de amostragem de 8000Hz. Este codec é comumente utilizado reduzir o consumo de banda em chamadas de telefone. Na Figura 3.4 é possível ver a faixa de frequência de áudio referente a sinais *Narrowband* em comparação de outras faixas de áudio.

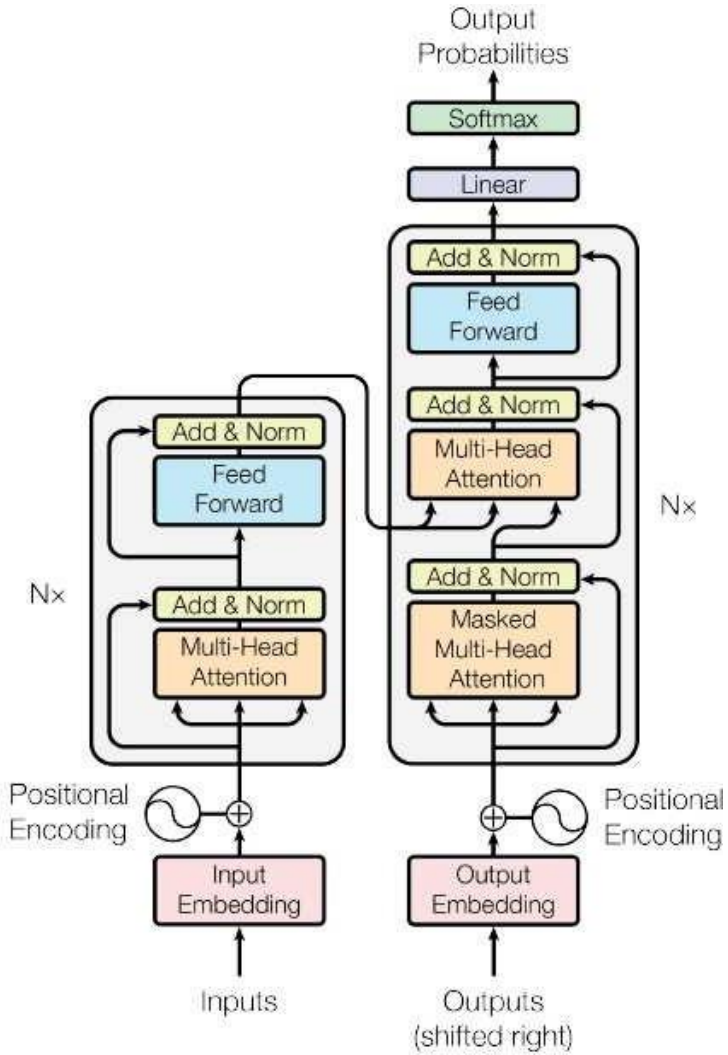


FIGURA 3.3. Diagrama de um *Transformer* e estado da arte em tecnologia de redes neurais para atividades de NLP (Vaswani, 2017).

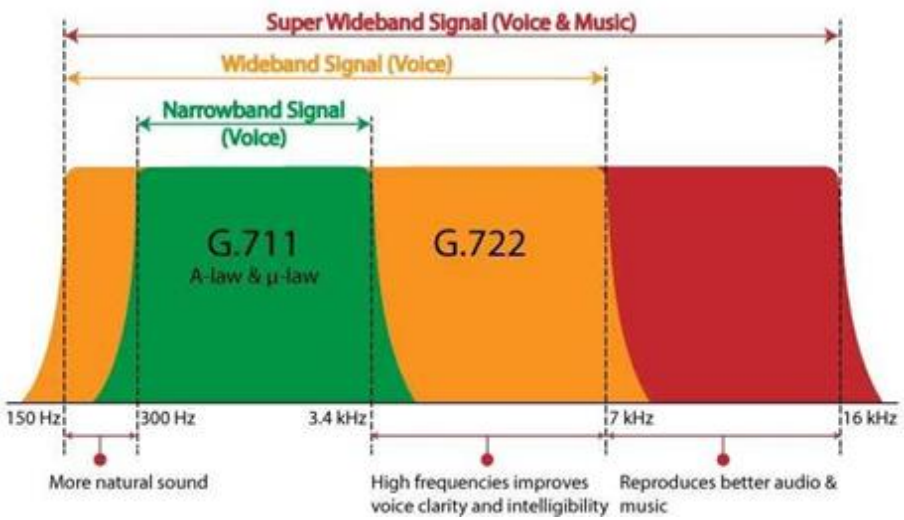


FIGURA 3.4. Gráfico de Frequência de Sinais de Áudio (Nextiva, 2020). O Codec G.711 realiza o *encoding* da faixa de 300Hz até 3.4kHz, suficiente para voz.

F. Métricas de Desempenho

1. Word Error Rate

Taxa de erro por palavra, ou *Word Error Rate* (WER) é uma métrica comum para análise de performance em atividades de reconhecimento de fala e tradução. Essa métrica está em evolução, e pode ser manipulada de diversas maneiras dependendo do resultado esperado (Ali, 2018). A equação 6.2 apresenta a fórmula onde quanto menor a taxa, melhor o resultado.

$$WER = ((S + D + I)/N) \times 100 \quad (6.1)$$

$$Accuracy = 100 - WER \quad (6.2)$$

onde S é o número de substituições, D é o número de deleções, I é o número de inserções, N é o número total de palavras de referência, e *Accuracy* é a acurácia da transcrição.

G. Procedimento *Transfer Learning*

Transfer Learning é uma técnica na área de inteligência artificial onde um modelo, pré treinado com quantidade significativa de recursos e tempo para um determinado propósito, pode ser modificado em uma atividade chamada “fine tune” de modo que um novo modelo seja desenvolvido. Isso é uma característica de grande impacto de redes neurais para o mercado, pois permite que novos modelos mais específicos sejam concebidos para tarefas similares ou mais específicas a partir de um existente (Hosna, 2022).

III. DESENVOLVIMENTO

Investigando o objetivo e escopo do projeto, encontrou-se no mercado soluções prontas de transcrição, serviços tais como GCP *Speech-to-Text, da Google, e Amazon Transcribe, da Amazon*. Entretanto, uma das preocupações principais de um projeto envolvendo dados em um contexto de relacionamento e identificação de pessoas, como no caso de atendimento ao cliente, é a questão de custo destes serviços por minuto e Lei Geral de Proteção de Dados. É de interesse então que uma empresa privada possa ter o controle dos modelos e arquitetura dedicados por lidar com este tipo de cenário.

No trabalho de Radford, 2023, foi anunciado o modelo OpenAI Whisper. Este modelo, baseado na tecnologia de estado da arte em Transformers, é aberto e de licença MIT, o que permite que este seja modificado, usado para intenções comerciais e, consequentemente, permite utilização privada. Isto contrasta com serviços dedicados em Cloud, onde não se tem o mesmo controle e ameaçam com outros riscos de projeto, como *vendor lock in* (Silva, 2013).

Uma proposta de pesquisa e desenvolvimento portanto surgiu para utilizar o modelo da *OpenAI Whisper* em uma arquitetura que comporta áudios já gravados em atividades comerciais da empresa Khomp.

A. Exploração dos dados

O maior diferencial da Khomp para o desenvolvimento desta pesquisa é sua base de dados. Atualmente, a Khomp implementa,

junto a seus serviços, uma frota de produtos eletrônicos desenvolvidos pela própria empresa. Estes produtos de telecomunicações fornecem para os clientes Khomp não apenas o serviço de gestão de chamadas telefônicas, mas também a gravação delas de forma segura.

As gravações são pré-processadas e armazenadas como arquivos do formato wave (.wav), stereo 128kbps, onde cada falante apresenta-se em um canal. Elas também possuem ruídos dentro da faixa; como eco, música etc. A intenção é ter o tempo (*timestamp*) e transcrição de áudio cada falante, ignorando outras fontes.

Em função deste cenário, há mais de 300h diárias de gravações que podem ser transcritas e disponibilizadas para análise ao próprio cliente. A proposta então se fundamenta em ter um sistema capaz de consumir das fontes de dados, realizar a transcrição, formatar os resultados, persistir os resultados, e mostrá-los em formato compreensível em uma interface.

B. Treinamento e utilização do modelo Whisper

Para Khomp, é de interesse que este modelo possa ser modificado com técnicas de fine tune para identificação futura de produtos mencionados durante as conversas. Utilizando material da *HuggingFace* (2023) para *finetune* do Modelo *Whisper*, foi realizado treino com o *dataset* do material da *Mozilla* para pronúncia de áudio em português (Radford, 2023) de modo a validar a *pipeline* de testes e obter o benchmark de recursos necessários para realizar modificações no modelo. Como descrito no trabalho de Radford [11], o modelo não aceita arquivos de áudio, mas sim Mel-Espectrogramas de trechos de 30 segundos em uma taxa de amostragem de 16kHz. Portanto, um dado de treino ou validação deve ser submetido a um processo como exemplificado na Figura 3.5, sendo fragmentado em pedaços de 30s. Foram utilizados scripts de ambiente Python para transformação do áudio e *tokenização* das transcrições.

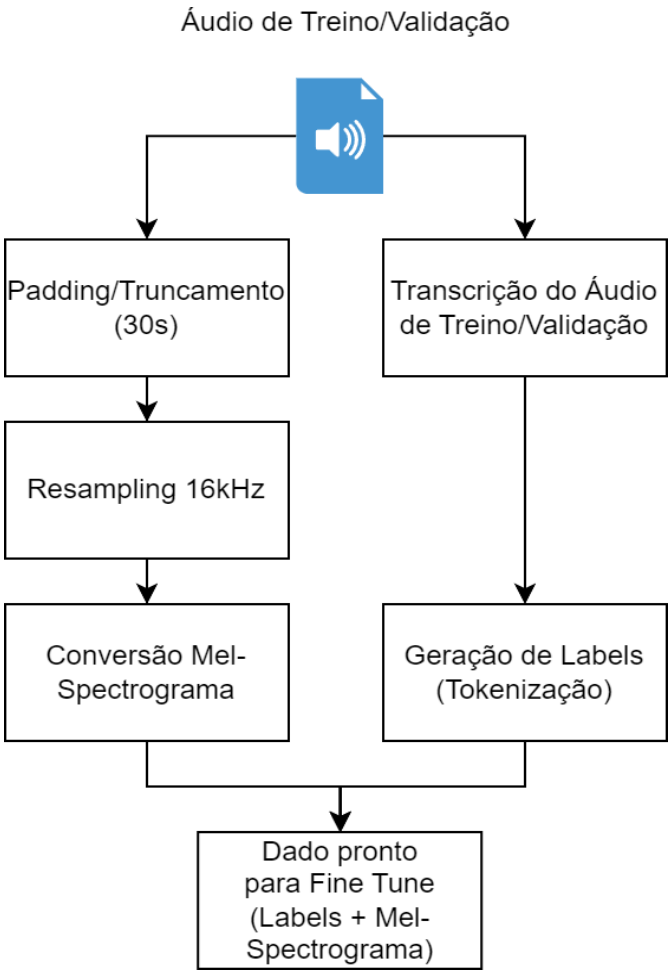


FIGURA 3.5. Fluxo de pré-processamento para dados de treino.

Este procedimento foi realizado com o *dataset* da *Common Voice 13 Mozilla* (2023) de maneira arbitrária para identificar a possibilidade de treinamento do modelo. Como métrica, foi utilizado o WER (2) padrão. Como a Khomp ainda não possuía o *dataset* para retreinamento, a etapa de *benchmarking* foi dada como satisfatória para o avanço da pesquisa, pois demonstrou a possibilidade para um investimento futuro.

Identificou-se a limitação da métrica WER, como descrito em (Radford, 2023). Para o português brasileiro não foi encontrado algum modo de normalização, onde uma mesma pronúncia ter transcrições diferentes (como a contração "né" de "não é") e, portanto, deve-se ser realizado uma padronização ou pós-processamento da transcrição de modo que a métrica não subtraia pontos do modelo. Isso foi notado como desafio para treinamentos futuros.

C. Arquitetura de software

Tendo em mãos os processos e requerimentos necessários, foi considerada uma arquitetura que implementa o padrão de arquitetura dirigida a eventos, ou *event-driven architecture.* Este tipo de arquitetura implementa diversos padrões, como consumidores de eventos, despachadores de mensagens, e brokers, enviando sinais para sistemas relevantes assincronamente (Hohpe, 2023). Este tipo de arquitetura pode ser implementado independentemente da localização da infraestrutura e tecnologia desde que sejam seguidos os padrões relevantes. Utilizando servidores feitos em linguagens de programação conhecidas pelo time, como *Python*, é possível enviar eventos para filas de mensagens, como *Apache Kafka*, que engatilham tarefas de transcrição para executores, denominados *Workers*, contendo o modelo Whisper adequado para a produção. Estes chamados *Workers* geralmente são gerenciados por orquestradores de *container*, como implementado por *Kubernetes*. Idealmente estes *Workers* são temporários, liberando recurso computacional ao executar a tarefa requisitada.

IV. APRESENTAÇÃO DOS RESULTADOS

A. Diagrama simplificado da arquitetura

Foi definido que existem times responsáveis pelo treinamento e disponibilização do modelo, denominado o time de *ML*, e um time responsável pela coleta, implantação do modelo, disponibilização do sistema, e armazenamento de resultados transcrições, denominado time de *Operations*.

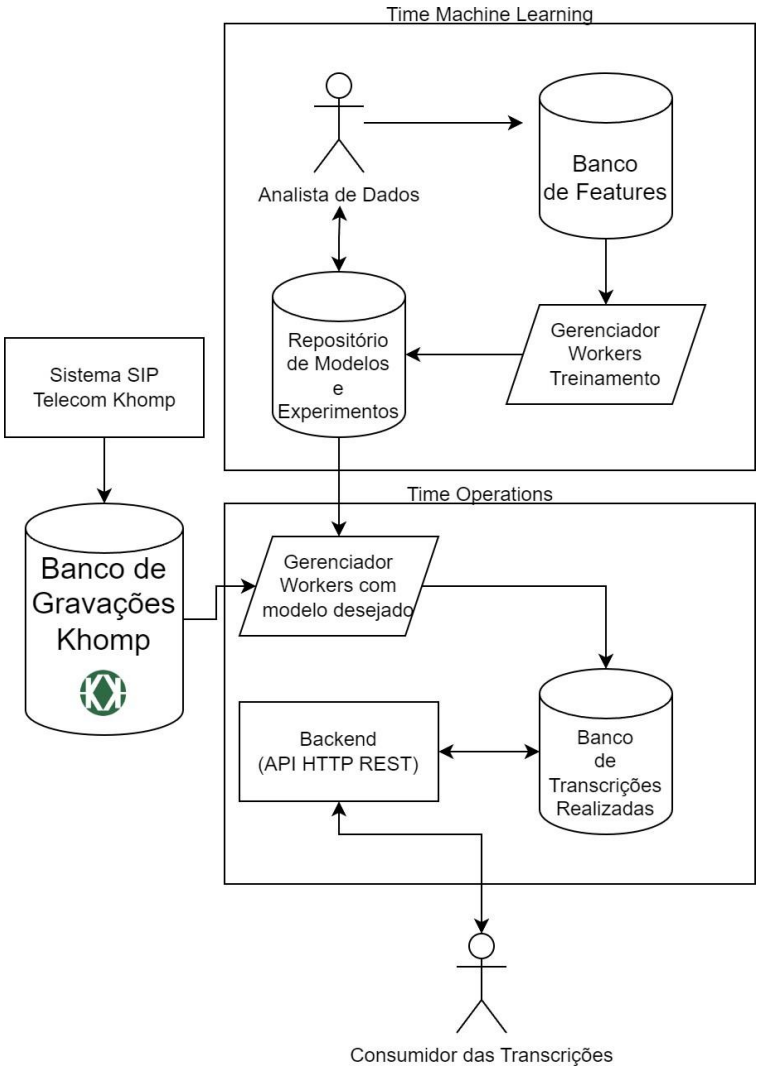


FIGURA 3.6. Diagrama simplificado da arquitetura de MLOps sugerida. Esta topologia permite que haja uma integração transparente entre os times relacionados à análise de dados e os times de operações.

As transcrições são realizadas em massa, em intervalos de tempo, por *Workers* contendo o modelo do Whisper selecionado pelo time de *ML*. Este time é responsável por experimentar, treinar, e selecionar

os modelos com melhores métricas de performance e qualidade dentro de seus ambientes de teste. O Analista é livre para modificar o Banco de *Features* onde serão mantidos treino e validação, e executar treinamentos através do orquestrador de treino.

A integração com o time de Operações e se dá pelo orquestrador de transcrições que consome o modelo apropriado definido pelo time de ML. Desta forma, o time de operações pode se preocupar em monitorar a performance dos *workers* de transcrição e no armazenamento de resultados, que são resgatados pelos usuários consumidores das transcrições. Um diagrama simplificado e agnóstico à tecnologia desta arquitetura que integra os dois times pode ser visualizado na Figura 3.6.

B. Resultado de transcrições

Para finalizar uma transcrição neste processo, os *Workers* que carregam o modelo e executam a operação sobre o áudio publicam uma mensagem em um formato específico para manter a estrutura da conversa. Para alcançar isso, foi definido um formato padronizado em JSON para descrever qual arquivo de áudio fora transcrito e as ocorrências de fala em cada canal, contendo *timestamp* em milissegundos de começo e fim de cada ocorrência, e a transcrição relevante de cada trecho identificado.

Após executar o procedimento, o *Worker* publica a transcrição realizada para o serviço consumidor relevante utilizando o formato definido, e libera seus recursos. O serviço consumidor então persiste a transcrição no banco de transcrições.

O processo e formato de transcrição foi altamente satisfatório para o objetivo da pesquisa, pois permite uma indexação simples e a apresentação dela em diversos formatos. Um modo de exibição escolhido para demonstração foi um sistema *web front-end*, capaz de consumir a transcrição de uma API Rest que a resgata do banco de transcrição e exibe-a como uma conversa sequencial na interface, estilo chat. Uma captura da tela de um consumidor de uma transcrição realizada pode ser vista na Figura 3.7. Observa-se que o

modelo transcreveu "Khomp" como "Compe", e "IP Wall", produto Khomp, como "per... Uau!", sinalizando potencial para retreinamento.

Como discutido na seção F, a métrica/pós-processamento apropriada para demonstrar a acurácia das transcrições atuais ainda está em aberto para pesquisa, e os resultados atuais se dão com o modelo Whisper genérico padrão (*checkpoint tiny*). Estudos futuros pretendem a realização de *fine tune* para identificar nomes comerciais entre outras imperfeições.

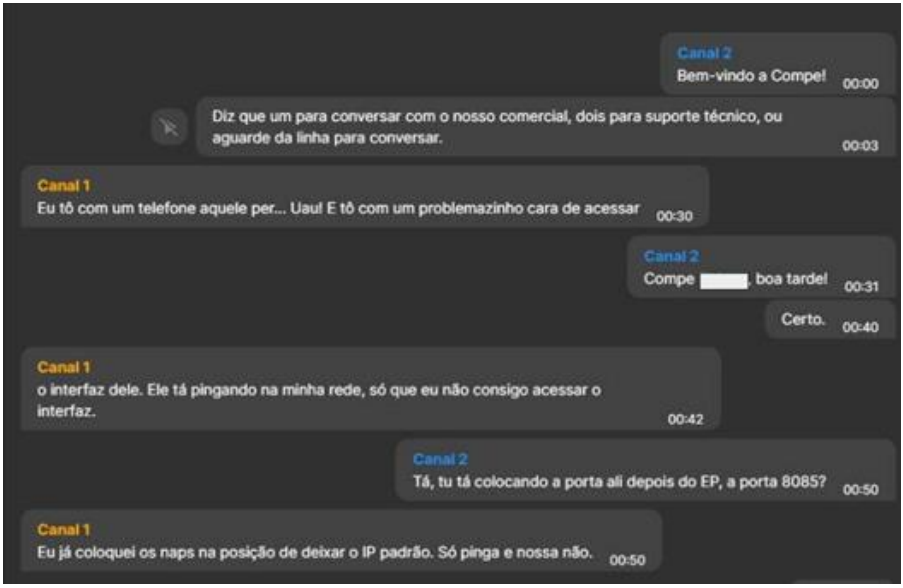


FIGURA 3.7. Interface Web com a transcrição formatada.

V. CONSIDERAÇÕES FINAIS

A cultura de *analytics* no mercado é cada vez mais relevante devido ao avanço das tecnologias de inteligência artificial e de armazenamento em massa. A Empresa Khomp viu uma futura possibilidade em inovação de mercado e estava em busca de uma arquitetura que viabilizasse essa oportunidade, enquanto cobrindo a questão de custo e segurança de dados de seus clientes.

Atualmente a habilidade de ter uma arquitetura de infraestrutura que é agnóstica ao provedor oferece uma vantagem à empresa quanto a disponibilização de funcionalidades em seus produtos B2B, entretanto a implementação comercial dos resultados requer maior exploração quanto ao comércio para ser disponibilizada como serviço.

Observou-se que são possíveis análises subsequentes em cima de transcrições, como análise de sentimento para avaliação das chamadas de suporte e identificação de dados sensíveis ou pessoais em chamadas de áudio. Uma frente de pesquisa na melhoria de métrica do WER, utilizando normalização para o português brasileiro, de modo a melhorar as ferramentas de transcrição também foi identificada. Técnicas de quantização de modelos também são possibilidades a serem exploradas, o que poderia possibilitar transcrição em ponta (embarcado nos gravadores).

REFERÊNCIAS

(ALI, 2018) Ahmed Ali and Steve Renals. “Word Error Rate Estimation for Speech Recognition: e-WER”. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers). Melbourne, Australia: Association for Computational Linguistics, July 2018, pp. 20–24. DOI:10.18653/v1/P18-2004.

(FLORIANO, 2022) Adrisson C Floriano, Sérgio L Avila, and Rubipiara C Fernandes. “Structured vocabulary specific to power operation control centers”. In: Energy Systems (2022).

(HBR, 2021) Using AI to track how customers feel in real time. https://web.archive.org/https://hbr.org/2021/05/using-ai-to-track-how-customers-feel-in- real- time. Online, Acessado: 2021-12-12. 2021.

(HOHPE, 2003) Gregor Hohpe and Bobby Woolf. Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions. Addison-Wesley Professional, Oct. 2003. ISBN: 0321200683. URL: http://www.amazon.ca/exec/obidos/redirect?tag=citeulike09-20%5C&path=ASIN/0321200683.

(HUGGINGFACE, 2020) Fine-Tune Whisper For Multilingual ASR with HuggingFace Transformers. https://web.archive.org/web/20230712012254/https://huggingface.co/blog/fine-tune-whisper. Online, Acessado: 2023-10-09. 2020.

(KUHL, 2023) Dominik Kreuzberger, Niklas Kühl, and Sebastian Hirschl. “Machine Learning Operations (MLOps): Overview, Definition, and Architecture”. In: IEEE Access 11 (2023), pp. 31866–31879. DOI: 10.1109/ACCESS.2023.3262138.

(MAHESH, 2020) Batta Mahesh. “Machine learning algorithms-a review”. In: International Journal of Science and Re- search (IJSR).[Internet] 9.1 (2020), pp. 381–386.

(MISHRA, 2020) Alok Mishra and Ziadoon Otaiwi. “DevOps and software quality: A systematic mapping”. In: Computer Science Review 38 (2020), p. 100308. ISSN: 1574-0137. DOI: https:// doi. org/10.1016/j.cosrev. 2020. 100308. URL: https://www.sciencedirect.com/science/ article/pii/S1574013720304081.

(MOZILLA, 2023) Mozilla Foundation. Common Voice 13. https://huggingface.co/datasets/mozilla-foundation/common_voice_13_0. Online, Acessado: 2023-10-09. 2023.

(NEXTIVA, 2023) What Are VoIP Codecs & How Do They Affect Call Sound Quality. https:// web.archive.org/web/20230324041438/https://www.nextiva.com/blog/voip-codecs.html. Online, Acessado: 2023-10-09. 2020.

(RADFORD, 2023) Alec Radford et al. “Robust speech recognition via large-scale weak supervision”. In: International Conference on Machine Learning. PMLR. 2023, pp. 28492–28518.

(ROSÁRIO, 2021) Albérico Rosário and Ricardo Raimundo. “Consumer marketing strategy and E-commerce in the last decade: a literature review”. In: Journal of theoretical and applied electronic commerce research 16.7 (2021), pp. 3003–3024.

(SAURA, 2021) Jose Ramon Saura. “Using data sciences in digital marketing: Framework, methods, and performance metrics”. In: Journal of Innovation & Knowledge 6.2 (2021), pp. 92–102.

(SILVA, 2013) Gabriel Costa Silva, Louis M. Rose, and Radu Calinescu. “A Systematic Review of Cloud Lock-In Solutions”. In: 2013 IEEE 5th International Conference on Cloud Computing Technology and Science. Vol. 2. 2013, pp. 363–368. DOI: 10.1109/CloudCom.2013.130.

(SINGH, 2022) Uma S. Singh et al. “A study on the revolution of consumer relationships as a combination of human interactions and digital transformations”. In: Materials Today: Proceedings 51 (2022). CMAE’21, pp. 460–464. ISSN: 2214-7853. DOI: https://doi.org/10.1016/j. matpr.2021.05.578. URL: https://www.sciencedirect. com/science/article/pii/S221478532104205X.

(VASWANI, 2017) Ashish Vaswani et al. “Attention Is All You Need”. In: CoRR abs/1706.03762 (2017). arXiv: 1706.03762.URL: http://arxiv.org/abs/1706.03762.

(ZAHARIA, 2018) Matei Zaharia et al. “Accelerating the machine learning lifecycle with MLflow.” In: IEEE Data Eng. Bull. 41.4 (2018), pp. 39–45.

4.
Detecção de defeitos em imagens de malha tecida com *deep learning*

Discente: Jonathan T. Lima

Prof. Orientador: Maurício Edgar Stivanello

RESUMO A indústria de tecidos tem se aprimorado rapidamente em virtude dos avanços tecnológicos. Todavia, a inspeção de defeitos ainda é comumente realizada por um operador humano, sujeito a variados tipos de distrações e falhas de análise. Os sistemas baseados em Visão Computacional surgiram como uma alternativa para a realização deste tipo de tarefa, originalmente, programados com base em regras matemáticas explícitas. Contudo, evoluíram nos últimos anos para incorporar as técnicas de aprendizado de máquina. O maior avanço ocorreu após a ascensão do aprendizado profundo de máquina, que permitiu às máquinas executarem tarefas de elevada complexidade, como identificar defeitos diminutos com acurácia e em altíssima velocidade. Neste trabalho são aplicados dois modelos de deep learning, ResNet18 e YOLOv8, para executar a tarefa de classificação de defeitos em imagens de malha tecida. O conjunto de dados foi obtido a partir de uma campanha experimental realizada em uma indústria real, localizada no Vale do Itajaí, Santa Catarina. Os resultados apresentaram acurácia de até 100% tanto para a validação quanto para os testes; entretanto, devido à pequena quantidade de imagens acredita-se que novos testes devem ser realizados para confirmar a aplicabilidade dos modelos ao problema real.

PALAVRAS-CHAVE: *Convolutional Neural Networks*, YOLO, Tecido de Malha, Classificação.

I. INTRODUÇÃO

A indústria têxtil é uma das principais componentes deste setor da economia e está presente em todo o mundo, representada por desde pequenas facções têxteis caseiras até a produção fabril de larga escala. O escopo desta indústria é colossal; por exemplo, estima-se que a produção mundial de algodão em 2021 ocupava uma área total de aproximadamente 32,510 milhões de hectares de terras cultiváveis (TOKEL, 2022) e o mercado têxtil foi avaliado em 1,695 trilhão de dólares em 2022 (TEXTILE, 2024).

Toda produção industrial está, invariavelmente, sujeita à ocorrência de defeitos de fabricação. Essas falhas acontecem devido à própria operação na linha produtiva, à falta de manutenção das máquinas, à inexperiência de novos operadores, à qualidade dos fios e outros fatores como a fadiga dos operadores. As técnicas clássicas de visão computacional foram empregadas com êxito nas décadas passadas para detectar esses defeitos. Todavia, as produções cresceram em escala e magnitude, assim como a diversificação dos tipos de tecidos, exigindo maior capacidade de processamento das imagens. Durante a segunda metade da década de 2010 ocorreu a ascensão das técnicas de aprendizado profundo de máquina, que rapidamente foram assimiladas pela indústria.

Para o presente trabalho foram empregados modelos de deep learning para um estudo de caso anterior, inicialmente estudado por Stivanello et al. (2016). Com o presente trabalho espera-se identificar novos padrões nas imagens e treinar modelos com acurácia mais elevada.

II. TRABALHOS ANTERIORES

Stivanello et al. (2016) construíram um sistema para a obtenção de imagens científicas durante a produção de uma malha tecida (*knitted fabric*) em uma fábrica de tecidos localizada na região do Vale do Itajaí, SC e aplicaram técnicas clássicas de visão computacional, obtendo alta de acurácia (> 80%) para um pequeno conjunto de dados. Em 2020, Vargas et al. (2020) levaram o estudo adiante e construíram um sistema para inspeção em tempo real das imagens, utilizando um

código em C++. Os autores também obtiveram resultados promissores, porém, considerando uma janela temporal curta de experimentação.

Em seu trabalho, Das et al. (2022) desenvolveram um modelo para detecção de defeitos em malhas, aplicando uma estratégia conjunta de técnicas clássicas para a extração de características (*features*) das imagens e uma combinação de duas redes neurais artificiais, o modelo de retropropagação (*feed forward backpropagation*) e Levenberg-Marquardt (LM), para a classificação entre defeito e não-defeito. Os autores atingiram uma acurácia de 88,89% para o LM.

Um sistema para a detecção de defeitos em uma fábrica de processamento de malha urdida foi concebido por Guosheng et al. (2022). Os autores empregaram os modelos R-CNN, *single shot multi-box detector* (SSD) e o YOLO com VGG-16. A acurácia dos modelos variou desde 83,5% para o SSD até 98,5% para o faster R-CNN, contudo, o YOLO apresentou os maiores valores de MAP (*Mean Average Precision*), até 66,8%.

Talu et al. (2022) desenvolveram uma solução para a inspeção in loco de defeitos em um tear. Os autores utilizaram algumas técnicas clássicas para a subdivisão (*patches*) das imagens e extração de padrões e posterior classificação do defeito com diversos modelos de aprendizado de máquina e uma CNN. Esta última arquitetura foi projetada especificamente para o projeto e gerou os melhores resultados para os testes (até 97,3% de acurácia).

Çıklaçandır, Utku e Ozdemir (2023) conduziram um estudo de visão de máquina a partir de dois *dataset*s de imagens de tecidos com defeitos. Os padrões das imagens foram extraídos com o auxílio de dois modelos de deep learning, ResNet18 e GoogleNet. Na sequência, a classificação dos defeitos foi obtida por meio de técnicas clássicas como *discrete cosine transform* (DCT) e *gray level co-occurrence matrix* (GLCM) e algoritmos de aprendizagem de máquina, PCA, KNN, SVM e *decision tree* (DT). A maior acurácia foi obtida com ResNet18, assumindo 100 features.

Hu e Jiang (2023) propuseram um modelo avançado, baseado em *Swim transformer,* um algoritmo em evidência na área de aprendizado de máquina e associado ao mecanismo de atenção (*attention*

mechanism), SWDCN. Os autores compararam os resultados de nove modelos e verificaram que o modelo proposto apresentou as melhores métricas. Também foram realizados testes com os otimizadores, onde o AdamW apresentou o melhor desempenho. Para o trabalho foram usados dois *dataset*s: um conjunto de imagens preparadas especificamente para o projeto e o *dataset* público TILDA. Outros trabalhos têm reforçado o papel crescente do mecanismo de atenção na detecção de defeitos em malhas (KAHRAMAN, 2022).

Frente aos trabalhos citados, o presente trabalho contribui por: (1) avaliar a efetividade da aplicação dos modelos ResNet18 e YOLOv8 para a classificação de defeitos em uma linha de produção de malha tecida; (2) avaliar resultados em condições próximas às existentes na linha de produção, considerando defeitos distintos; (3) disponibilizar uma base de dados adquirida em condições reais de operação com a devida rotulação. A base de dados empregada foi adaptada dos trabalhos de Vargas et al. (2020), assumindo o recorte das imagens em divisões de segmentos específicos.

III. METODOLOGIA

O sistema experimental desenvolvido por Vargas et al. (2020) permitiu identificar quatro ocorrências de defeitos predominantes entre as imagens: linhas horizontais (“LH”), linhas verticais (“LV”), rasgos (“RC”) e sem defeitos (“OK”). A Figura 4.1 contém um exemplo para cada classe de defeito.

O *dataset* original contém 30 imagens rotuladas como “LH”, 45 imagens como “LV”, 36 imagens como “RC” e 231 imagens como “OK”. Apesar de serem tratadas apenas quatro classes de defeito neste trabalho, mais de 70 defeitos já foram reportados na literatura (RASHEED et al., 2020).

A Figura 4.2 traz um esquema do sistema de aquisição construído para a obtenção das imagens. Vargas et al. (2020) criaram um sistema de inspeção em tempo real. Neste trabalho são utilizadas as imagens adquiridas com esse sistema durante a sua operação.

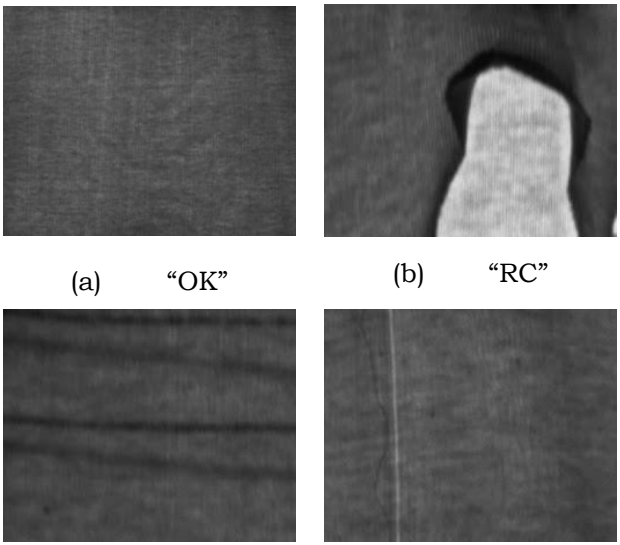


FIGURA 4.1. Exemplos de imagens para cada classe de defeito.

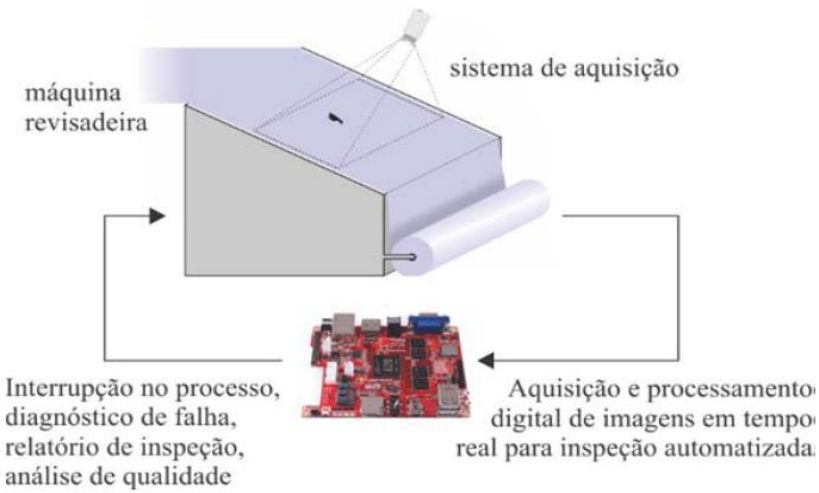


FIGURA 4.2. Sistema experimental construído por Vargas et al. (2020) para a obtenção das imagens reais de produção de malha tecida.

Para o presente trabalho, as imagens do *dataset* foram divididas em razões iguais de três, cinco e sete partes, em ambas as direções (Figura 4.3). Esse procedimento permitiu aumentar o número de instâncias da base de dados. Além disso, amplia os trabalhos desenvolvidos anteriormente, pois com a imagem dividida é possível apontar com maior resolução a localização dos defeitos e facilitar a classificação. Dessa forma, pode-se aproveitar melhor o tecido "visto". Ao invés de indicar todo o "campo de visão da câmera" como defeituoso, destaca-se uma sub-região.

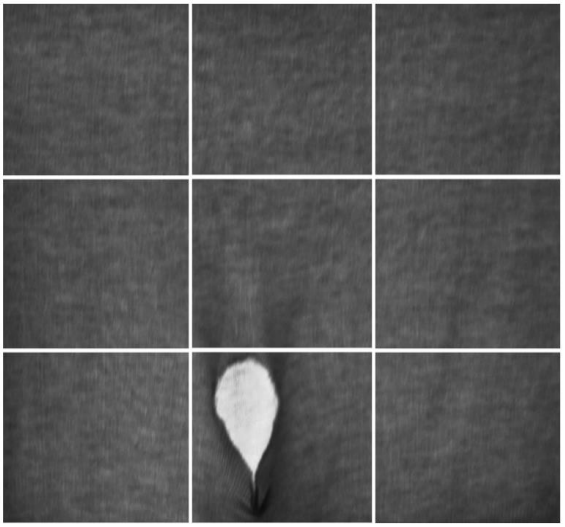


FIGURE 4.3. Exemplo de divisão das imagens para aumento de dados.

Como pode ser verificado na Figura 4.3, uma imagem original classificada como "RC", ao ser cortada gera novas instâncias da classe "OK". O mesmo pode ocorrer também para as demais classes. Assim, o *dataset* com recorte 3 (FS_3) possui um total de 2.269 imagens com dimensões de 253 x 200 pixels, o recorte 5 (FS_5) possui 6.947 imagens e dimensões de 152 x 120 pixels e o recorte 7 (FS_7) totaliza 13.413 imagens de dimensões 108 x 85 pixels. A Tabela 4.1 apresenta a quantidade de imagens para cada recorte e classe de defeito.

TABELA 4.1. QUANTIDADE DE IMAGENS POR RECORTE E CLASSE

Recorte	“RC”	“OK”	“LV”	“LH”
FS_3	102	2079	273	225
FS_5	213	5775	417	543
FS_7	300	11319	705	1089

Após análise, verificou-se que os novos *dataset* estavam desbalanceados, isto é, existe um número muito maior de instâncias pertencentes à classe “OK” em comparação com as demais classes. Logo, foram criados *datasets* balanceados. Para cada recorte, partiu-se da classe com a menor quantidade de amostras. Neste trabalho, foi a classe “RC”, em que a malha apresentou rasgos. Em seguida, a quantidade de exemplos das demais classes foi reduzida até que o total de imagens de classe estivesse na mesma ordem de grandeza. A seleção foi aleatória. A Tabela 4.2 apresenta um resumo do conjunto de dados balanceados.

TABELA 4.2. QUANTIDADE DE IMAGENS POR RECORTE E CLASSE APÓS BALANCEAMENTO

Recorte	“RC”	“OK”	“LV”	“LH”
FS_3	102	300	273	225
FS_5	213	650	417	543
FS_7	300	1300	705	1089

O desbalanceamento de classes é um problema comum de Visão Computacional. Na prática, durante o treinamento do modelo, a distribuição desbalanceada de classes gera um viés (bias) em favor da classe majoritária devido à quantidade insuficiente de amostras de treinamento para as classes minoritárias do conjunto de dados (JOHNSON, 2019).

Dadas as características do problema estudado, como a divisão de quatro classes de defeitos distintos, que podem ser identificados na extensão de cada imagem, ficou definido que a abordagem de classificação é a mais adequada. Para tanto foram empregados dois modelos pré-treinados, ResNet18 (através da API fast.ai) e o YOLOv8 (yolov8n-cls.pt, Ultralytics), empregando bases de imagens amplamente utilizadas pela comunidade e não vinculada às aplicações específicas. Foram realizados diversos testes até alcançar um arranjo adequado para os hiperparâmetros, como pode ser verificado na Tabela 4.3.

TABELA 4.3. HIPERPARÂMETROS DOS MODELOS

Modelo	Framework	Batch size	Épocas	Taxa de aprendizado
ResNet18	Pytorch	32	5	0,001
YOLOv8	Pytorch	16	10	0,01

O modelo ResNet18 é uma referência ao termo residual network, que é uma estratégia de treinamento para redes neurais profundas introduzido por He et al. [13]. O número 18 corresponde a quantidade de camadas profundas do modelo. Outras versões incluem 34, 50, 101 e 152 camadas. O modelo ResNet é treinado com o *dataset* ImageNet e saiu vencedor do concurso de classificação de imagens, ILSVRC2015.

A primeira versão do YOLO (v1) foi apresentada em 2015. Originalmente, o YOLO surgiu para resolver problemas de detecção de objetos, sendo treinado com o *dataset* COCO (Microsoft) em mais de 80 objetos. A versão 8 (YOLOv8) utilizada neste trabalho foi criada e distribuída pela Ultralytics no início de 2023. Esta versão permite analisar problemas gerais de detecção, segmentação, classificação e também problemas mais específicos como estimativa de pose e rastreio de objetos. Terven e Cordova-Esparza (2023) apresentam uma revisão detalhada do histórico e das características de cada versão do YOLO. Hussain (2023) desenvolveu uma revisão que analisa as versões do YOLO em contraposição às necessidades da indústria manufatureira.

Ambos os autores reforçam que a versão 8 do YOLO é superior às anteriores, em performance, velocidade e diversidade de tarefas.

Por padrão, o método de treinamento com o YOLOv8 aplica algumas técnicas de aumento de dados como girar a imagem e variações nas cores (*color jitter*).

Foram realizados treinamentos utilizando os conjuntos de dados original e balanceados. Ao longo desses treinamentos surgiram questionamentos a respeito da possibilidade de *overfitting* devido à elevada acurácia obtida inicialmente. Os testes indicaram que o desbalanceamento dos *datasets* poderia estar afetando a capacidade dos modelos em extrair padrões e generalizar. Por exemplo, alguns testes com o modelo ResNet18 mostraram que os modelos gerados, embora com elevada acurácia para a validação, possuíam dificuldade para distinguir as classes “LV” e “OK”, especialmente para as imagens com menor recorte (FS_7). A Tabela 4.4 mostra uma comparação entre as duas classes para os três recortes estudados.

Visualmente, essas imagens são bastante parecidas e podem confundir o olho humano. As linhas verticais são representadas por traços em tons de cinza claro a branco, bastante destacadas para o recorte original e, em escala reduzida, até o recorte FS_5. Para a classe “OK”, a variação entre cinza e branco ocorre de forma descontínua.

As linhas horizontais são representadas por trechos longitudinais em tons cinza escuro e preto. O efeito da escala também ocorre para as linhas horizontais (Figura 4.1.c).

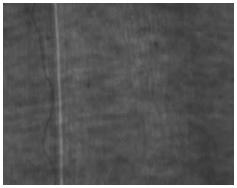
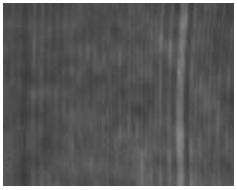
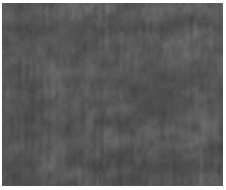
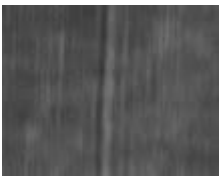
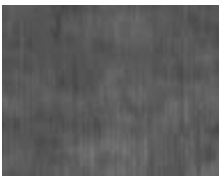
A rede ResNet18 é uma rede pré-treinada já bem conhecida da comunidade de Visão Computacional. Neste trabalho, os novos treinamentos foram conduzidos a partir da API fast.ai e suas bibliotecas em linguagem Python.

A rede YOLO é disponibilizada pela plataforma Ultralytics e está em sua 8ª versão. O modelo de classificação usado é também pré-treinado, neste caso com a base de dados ImageNet.

Os treinamentos foram realizados no ambiente do Google Collaboratory e on premise. O computador utilizado possui processador intel core i7 de 7ª geração, 4 GB de memória RAM e SSD

de 256 GB, sendo prioritariamente destinado ao uso generalista. Para os treinamentos on-line foi utilizado o GPU T4 ofertado gratuitamente e a duração variou entre 2min46seg e 14min. Os treinamentos com computador pessoal duraram entre um quarto de hora e 50 minutos aproximadamente.

TABELA 4.4. COMPARAÇÃO DAS CLASSES “LV” E “OK” PARA RECORTES DIFERENTES

Recorte	“LV”	“OK”
FS_3		
FS_5		
FS_7		

O desempenho dos treinamentos foi avaliado a partir de duas métricas: a perda (loss) e a acurácia. Esta última é calculada para as etapas de validação e teste.

A quantidade de exemplos para testes limitou-se a 10% do conjunto de dados, o que representa um pequeno número para a maioria das classes. Assim, em alguns testes foram utilizadas imagens, para todas as classes e recortes, rotacionadas em 180°.

Os testes iniciais utilizaram o *dataset* de testes original e, quando os resultados preliminares pareciam inconclusivos, foi utilizado um *dataset* de testes maior, incluindo as imagens rotacionadas. Não foram aplicadas técnicas de aumento de dados ao *dataset*.

IV. RESULTADOS E DISCUSSÕES

Neste trabalho, foram empregadas duas redes neurais profundas, ResNet18 e YOLOv8, para a executar a tarefa de classificação de imagens reais da linha de produção de malha tecida, contendo quatro classes de defeitos.

A Tabela 4.5 contém um resumo dos treinamentos realizados corretamente, após definição dos hiperparâmetros e seus principais resultados. São apresentados os valores de acurácia para a validação (acc val) e teste (acc teste). Foram utilizados os *dataset*s original (ResNet18) e balanceado (YOLOv8).

A acurácia do teste 1 (0,220) corrobora o resultado obtido durante a validação (0,213), reforçando que este treinamento não é adequado para a classificação dos defeitos. Os demais treinamentos com o *dataset* original apresentaram resultados muito superiores. Na seara da Inteligência Artificial, a acurácia de 100% costuma despertar questionamentos quanto a possibilidade de *overfitting*, pois muito raramente um modelo atinge o ápice de suas métricas. Apesar disso, nos últimos anos alguns autores têm registrado exemplos de modelos que realizaram o feito (KAUR, 2020; ALZUBAIDI, 2021; PHUNG; RHEE, 2019).

Uma das formas de avaliar a possibilidade de *overfitting* consiste em avaliar a evolução da perda (*loss*) durante o treinamento. A Figura 4.4 mostra os valores de perda para o treinamento e respectiva validação 4 (ID 4, Tabela 4.5).

Os valores de perda para a validação começam abaixo dos valores para o treinamento, o que é curioso, pois, em geral, ocorre o inverso. Ao final de cinco épocas, a perda na validação é menor do que no treinamento, resultado mais coerente e em consonância com o que é esperado.

TABELA 4.5. RESULTADOS DAS SIMULAÇÕES

ID	Modelo	*dataset*	acc treino	acc val	acc teste
001	ResNet18	FS_1*	0,28	0,21	0,22
002	ResNet18	FS_1*	1,00	1,00	1,00
003	ResNet18	FS_3*	1,00	1,00	1,00
004	ResNet18	FS_5*	1,00	0,99	1,00
005	ResNet18	FS_7*	1,00	0,99	1,00
006	ResNet18	FS_3**	0,97	0,99	1,00
007	ResNet18	FS_5**	0,98	0,99	1,00
008	ResNet18	FS_7**	0,97	0,99	1,00
009	YOLO	FS_1*	0,98	0,95	1,00
010	YOLO	FS_3*	0,99	0,88	0,35
011	YOLO	FS_5*	0,93	0,84	0,45
012	YOLO	FS_7*	0,94	0,86	0,40
013	YOLO	FS_1**	0,99	0,97	0,75
014	YOLO	FS_3**	0,99	0,98	1,00
015	YOLO	FS_5**	0,99	0,99	1,00
016	YOLO	FS_7**	0,97	0,98	1,00

* original; ** balanceado; acc val = acurácia validação; acc teste = acurácia no teste.

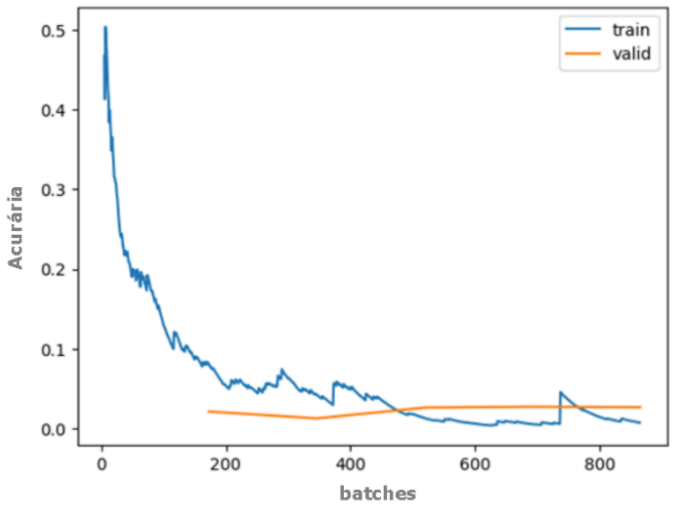


FIGURE 4.4. Evolução da perda para o treinamento ID 4.

A Figura 4.5 representa a matriz de confusão para o treinamento 4. Considerando 1.389 imagens usadas para a validação e o desbalanceamento do *dataset*, em que o número de instâncias da classe “OK” é até duas ordens de grandeza maior do que as demais classes, obteve-se um resultado bastante significativo. A acurácia do teste reforça a validação, sugerindo ser este um modelo robusto para a classificação dos defeitos. Na Figura 4.6 é apresentada a matriz de confusão referente ao treinamento 15 (YOLO, FS_5 balanceado). A partir do *dataset* de validação balanceado o modelo confundiu uma instância rotulada com a classe “LV” como “RC” e sete instâncias do tipo “OK” como “LV”. Esses erros podem estar associados ao fator de recorte das imagens e à capacidade do próprio YOLO em identificar os padrões contidos nas imagens analisadas. Apesar disso, a acurácia é bastante elevada. Na Figura 4.7 é exibida a matriz de confusão para o treinamento 16, em que se utilizou o modelo YOLOv8 com o *dataset* FS_7 (balanceado). A rede neural confundiu 9 exemplos da classe “OK” com “LV”, entretanto nenhuma instância da classe “LV” foi confundida com a classe “OK”, mas com a classe mais distinta, “RC”. Considerando que as matrizes de confusão possuem origem em treinamentos com estratégias e *dataset*s diferentes, pode-se conjecturar que os resultados são convergentes, ao mostrarem-se bastante próximos, de forma que os treinamentos realizados com o YOLO reforçam os resultados obtidos com o ResNet18.

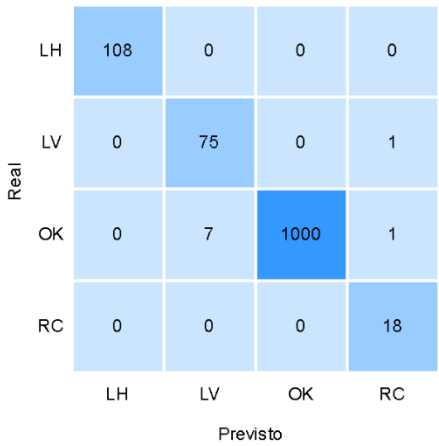


FIGURE 4.5. Matriz de confusão da validação para o treinamento 4.

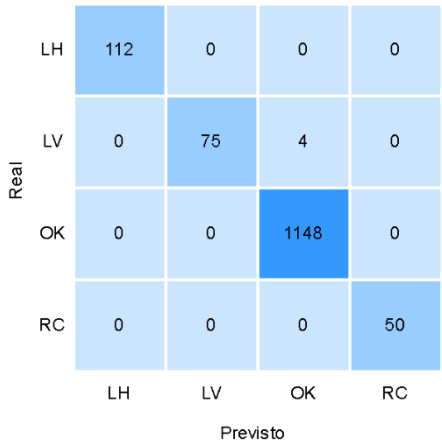


FIGURE 4.6. Matriz de confusão para a validação do treinamento 15.

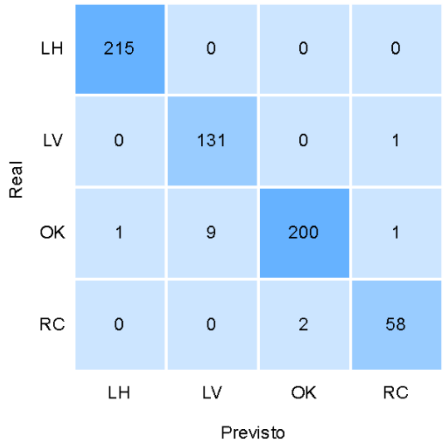


FIGURE 4.7. Matriz de confusão para a validação do treinamento 16.

O treinamento 09 foi baseado no mesmo *dataset* que o treinamento 1 e gerou um resultado muito superior. O ocorrido possivelmente se deve ao fato de que o YOLOv8 é pré-treinado com o *dataset* ImageNet, o que facilitou a identificação de padrões. Todavia, a acurácia do teste foi de 75% e o modelo confundiu a classe “LH” com “OK” em todas as tentativas, acertando, contudo, as outras classes.

Do ponto de vista do problema de visão de máquina, as imagens são "vistas" como matrizes (arrays) cujas células possuem valores que variam desde 0 (preto) a 255 (branco). Conforme mencionado anteriormente, os padrões das linhas verticais e linhas horizontais são bastante distintivos pelas cores e padrões característicos. Os autores acreditam que os modelos podem estar atrelando a ocorrência de "linhas" contendo valores muito próximos de 255 à classe "LV" – as instâncias dessa classe possuem linhas bastante claras e destacadas na imagem. Sua contraparte também pode ser verdadeira, com os modelos reconhecendo "linhas" da matriz com valores muito próximos de 0 como a classe "LH" – cujas linhas escuras ocorrem horizontalmente.

A certa altura, algumas imagens foram rotacionadas em 90º, numa operação para tentar transformar uma imagem da classe "LV" em "LH" e vice-versa. Os testes, contudo, mostraram que os modelos com maior acurácia "enxergam" a classe de rotulação de cada imagem, resultado este que subsidia a hipótese anterior.

As altas acurácias verificadas nos testes 6 a 8 reforçam a necessidade de realizar mais testes para confirmar ou refutar os resultados. Neste ponto a pesquisa possui a limitação do número de imagens disponíveis e, portanto, será necessário criar *dataset*s secundários aplicando técnicas de aumento de dados como alteração do brilho, zoom e até novos recortes (possivelmente, FS_2, FS_4 e FS_6), além do método de rotação em 180º já empregado.

Os testes 9 a 12 foram conduzidos após treinamento com o modelo YOLOv8 e *dataset*s não balanceados. Os treinamentos e validação apresentaram bons resultados e consistentes entre si. Porém, os testes revelaram que os modelos obtidos falham diante de novos exemplos. Para a maioria dos testes, a classe identificada é o "RC".

A falta de equilíbrio entre o número de exemplos entre as classes e as características pronunciadas da classe "RC" podem estar contribuindo para gerar esse resultado. Em futuros trabalhos poderão ser aplicadas técnicas de redução de dimensionalidade como PCA e t-SNE para analisar os dados e avaliar sua capacidade de separação em um *dataset*.

V. CONSIDERAÇÕES FINAIS

Neste trabalho, foram usadas duas redes neurais profundas (ResNet18 e YOLOv8) para a classificação de defeitos na fabricação de malhas tecidas, considerando as imagens originais e seus recortes nas proporções de FS_3 a FS_7.

Os resultados são promissores, apresentando alta acurácia na validação e nos testes. Os testes foram conduzidos, também, com um *dataset* criado a partir da rotação em 180o das imagens originais. A questão do índice de confusão entre "LV" e "OK" pode ser considerada um viés dos modelos para o defeito "LV", porém, os resultados se replicam para as duas redes neurais tanto com o *dataset* original como para o balanceado. Assim, novos testes serão necessários para aumentar a segurança nos resultados.

As redes neurais profundas surgem como uma alternativa às técnicas clássicas de visão computacional, porém, mesmo com o potencial demonstrado pelos métodos de *deep learning,* nota-se pela literatura que o uso conjunto com técnicas clássicas ainda é bastante utilizado e pertinente. Logo, uma estratégia futura de validação dos modelos pode utilizar-se da aplicação conjunta destes com técnicas tradicionais.

Para ambos os *dataset*s – balanceado e não balanceado – a rede ResNet18 apresentou bons resultados, enquanto a rede YOLOv8 não produziu resultados satisfatórios para os *dataset*s desbalanceados FS_3 a FS_7 e, portanto, a ResNet18 possui o melhor desempenho global.

REFERÊNCIAS

(ALZUBAIDI, 2021) ALZUBAIDI, L.; ZHANG, J.; HUMAIDI, A. J.; AL-DUJAILI, A.; DUAN, Y.; AL-SHAMMA, O.; SANTAMARÍA, J.; FADHEL, M. A.; AL-AMIDIE, M.; FARHAN, L. Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. J. Big Data, v. 8, n. 53, 2021.

(ÇIKLAÇANDIR, 2023) ÇIKLAÇANDIR, F. G. Y.; UTUK, S.; ÖZDEMIR, H. Determination of various fabric defects using different machine learning techniques. The J. Text. Inst., 2023. doi: 10.1080/00405000.2023.220197.

(DAS, 2022) DAS, S.; WAHIB, A.; KUMAR, S. M.; MISHRA, R. S.; SUNDARAMURTHY, S. Moment-Based Features of Knitted Cotton Fabric Defect Classification by Artificial Neural Networks. J. Nat. Fib., v. 19, n. 4, p. 1498-1506, 2022.

(GUOSHENG, 2022) GUOSHENG, X.; YANG, X.; ZHIQI, Y.; YIZE, S. An intelligent defect detection system for warp-knitted fabric. Tex. Res. J., v. 92, n. 9-10, p. 1394-1404, 2022.

(HE, 2023) HE, K.; ZHANG, X.; REN, S.; SUN, J. Deep Residual Learning for Image Recognition. arXiv:1512.03385v1. 2023.

(HU, 2023) HU, Y.; JIANG, G. Weft-knitted fabric defect classification based on a Swin transformer deformable convolutional network. Tex. Res. J., v. 93, n. 9-10, p. 2409-2420, 2023.

(HUSSAIN, 2023) HUSSAIN, M. YOLO-v1 to YOLO-v8, the Rise of YOLO and Its Complementary Nature toward Digital Manufacturing and Industrial Defect Detection. Machines, v. 11, p. 677, 2023.

(JOHNSON , 2019) JOHNSON, J. M.; KHOSHGOFTAAR, T. M. Survey on deep learning with class imbalance. Journal of Big Data, v. 6, n. 1, 2019. doi: 10.1186/s40537-019-0192-5.

(KAHRAMAN, 2022) KAHRAMAN, Y.; DURMUSOGLU, A. Deep learning-based fabric defect detection: A review. Tex. Res. J., v. 93, n. 5-6, p. 1485-1503, 2022.

(KAUR, 2020) KAUR, T.; GANDHI, T. K. Deep convolutional neural networks with transfer learning for automated brain image classification. Mach. Vis. Appl., v. 31, n. 20, 2020.

(PHUNG, 2019) PHUNG, V. H.; RHEE, E. J. A High-Accuracy Model Average Ensemble of Convolutional Neural Networks for Classification of Cloud Image Patches on Small *Dataset*s. Appl. Sci., v. 9, n. 21, 2019.

(RASHEED, 2020) RASHEED, A.; ZAFAR, B.; RASHEED, A.; ALI, N.; SAJID, M.; DAR, S. H.; HABIB, U.; SHEHRYAR, T.; MAHMOOD, M. T. Fabric defect detection using Computer Vision Techniques: A comprehensive review. Mathematical Problems in Engineering, p. 1–24, 2020. doi: 10.1155/2020/8189403.

(STIVANELLO, 2016) STIVANELLO, M. E.; VARGAS, S.; ROLOFF, M. L.; STEMMER, M. R. Automatic Detection and Classification of Defects in Knitted Fabrics. IEEE Lat. Am. Trans., v. 14, n. 7, p. 3065-3073, 2016.

(TALU, 2022) TALU, M. F.; HANBAY, K.; VARJOV, M. H. CNN-Based Fabric Defect Detection System on Loom Fabric Inspection. Textile and Apparel, v. 32, n. 3, p. 208-219, 2022.

(TERVEN, 2023) TERVEN, J. R.; CORDOVA-ESPARZA, D. M. A Comprehensive Review of YOLO: from YOLOv1 and beyond. arXiv:2304.00501v5, 2023.

(TEXTILE, 2023) TEXTILE MARKET SIZE, SHARE, TRENDS & GROWTH REPORT, 2030. Textile Market Size, Share, Trends & Growth Report, 2030. Disponível em: https://tinyurl.com/yc5hkr3s. Acesso em: 10/04/2024.

(TOKEL, 2022) TOKEL, D.; DOGAN, I.; HOCAOGLU-OZYIGIT, A.; OZYIGIT, I. I. Cotton agriculture in Turkey and worldwide economic impacts of Turkish Cotton. Journal of Natural Fibers, v. 19, n. 15, p. 10648–10667, 2022.

(STIVANELLO, 2020) VARGAS, S.; STIVANELLO, M. E.; ROLOFF, M. L.; STIEGELMAIER, E.; STEMMER, M. R. Development of an Online Automated Fabric Inspection System. J. Cont. Aut. Elect. Sys., v. 31, p. 73-83, 2020.

5.

Desvio de obstáculos por agentes autônomos em drones: uma abordagem com aprendizado por imitação

Discente: Fernando Vasata

Discente: Thiago Haigerti Bertoldi

Prof. Orientador: Sérgio Luciano Avila

RESUMO Missões com drones contam com características que podem transformá-las em altamente críticas. O controle automático que usualmente está disponível para esse tipo de equipamento é capaz de reagir a mudanças no ambiente, mas pode não ser capaz de tomar decisões no contexto de uma missão a fim de alcançar um resultado. Para que o drone tenha esse tipo de autonomia, é necessário que seu software contenha um agente autônomo que não seja meramente reativo, mas também proativo, ou seja, tenha a capacidade não apenas de responder a estímulos externos, mas também de tomar iniciativas com base em seu entendimento da situação. A implementação de uma política ótima de uma missão no agente precisa de uma referência de identificação de uma política ótima para dada situação. Para encontrar esse comportamento base para comparativos, uma estratégia plausível é assumir que as políticas de um operador humano são políticas ótimas. Nesse contexto, surge o conceito de aprendizado por imitação: o treinamento de agentes autônomos através da modelagem de

comportamentos de operadores especialistas (humanos ou não). Neste trabalho teórico, são discutidas técnicas de aprendizado por imitação para um problema de desvio de obstáculos com drones. O problema é compatível com técnicas de aprendizado por imitação, pois os sistemas de voo envolvem várias variáveis em um sistema complexo, difícil de ser modelado e resolvido através de alguma programação explícita. O debate proposto aqui é encorajador na busca por maior autonomia para agentes ditos autônomos.

PALAVRAS-CHAVE: Aprendizado por Imitação, Agentes Autônomos, Drones, Simulação.

I. INTRODUÇÃO

A vida real é complexa, sendo que várias tarefas não podem ser automatizadas por meio de métodos analíticos. Existe, sem dúvida, uma necessidade da análise humana (inclusive em processos de decisão) nos mais diferentes cenários. Um sistema de manobra de um drone é constituído por incontáveis variáveis que descrevem a dinâmica de voo. Apesar de que existem sistemas de controle robustos no mercado, nem sempre é possível descrever de forma algorítmica uma tarefa para que um computador de bordo possa executá-la.

Além das variáveis de baixo nível (como pressão do ar, umidade, força do vento etc.), existe uma camada de decisões de alto nível que aparecem até nas manobras e/ou tarefas mais simples. Um exemplo de tal decisão de alto nível pode ser a estratégia de como contornar um obstáculo. Contornar por cima, por baixo ou lateral, pode ter consequências difíceis de serem mapeadas a priori. Entretanto, um operador humano experiente é capaz de tomar uma decisão satisfatória nesses cenários de incerteza. Dessa forma, pode-se tomar o operador humano como uma boa métrica de desempenho de um sistema.

Um controle automático que possua um desempenho pior que um operador humano claramente não é desejável, tendo em vista que em sistemas críticos, a decisão mais adequada, é minimizar as chances de erro, evitando possíveis acidentes. Pode-se aproveitar essa informação, o comportamento humano, para treinarmos agentes autônomos capazes de imitar os comportamentos humanos.

Muitas vezes, a linha de raciocínio de um ser humano não pode ser mapeada completamente, mas observando suas ações em diferentes estados, é possível encontrar uma função custo sendo otimizada. Essa função custo consegue descrever, dentro das capacidades do operador humano, o comportamento ideal do sistema. Desta forma, pode-se iniciar uma análise de técnicas de aprendizado por imitação, que permitem a um agente autônomo tomar decisões similares às decisões de um operador humano especialista.

Explorando as várias técnicas de aprendizado por imitação, pode-se elencar pelo menos duas categorias de técnicas (Ross, 2011; Kumar et al, 2022): técnicas online, que demandam uma interação contínua entre o agente e o operador, e técnicas a posteriori, que usam dados de missões realizadas por operadores humanos para treinarem, de forma offline, agentes autônomos capazes de replicar o comportamento humano. Dentro dessas categorias, pode-se considerar que, para boa parte dos cenários, uma interação com um operador humano é mais difícil e ou custosa de ser realizada e, portanto, um enfoque em técnicas de aprendizado offline é razoável. Das técnicas de aprendizado offline, destacam-se duas: a clonagem comportamental (Zheng, 2022) e o *Inverse Reinforcement Learning* (Choi et al, 2017). As diferenças e características de cada abordagem serão amplamente discutidas em sequência.

II. O PROBLEMA, OS SIMULADORES E A EXECUÇÃO

Foi escolhido um problema de desvio de obstáculos como ponto de partida para a análise. A Figura 5.1 ilustra a missão a ser executada, onde os cones são os obstáculos (em vermelho os sensibilizados pela proximidade) e circuito percorrido está identificado pelas bolas verdes (tendo como alvo a inspeção de um poste de energia elétrica, por exemplo). Nas mais variadas missões, o contorno de obstáculos se faz presente porque, salvo cenários simulados, os cenários da vida real estão em constante mudança, seja por conta de intervenção humana, seja pela própria passagem do tempo. O problema de contorno de obstáculos é particularmente interessante para dados cenários e pode ser considerado como um problema em duas dimensões e não um problema em três dimensões.

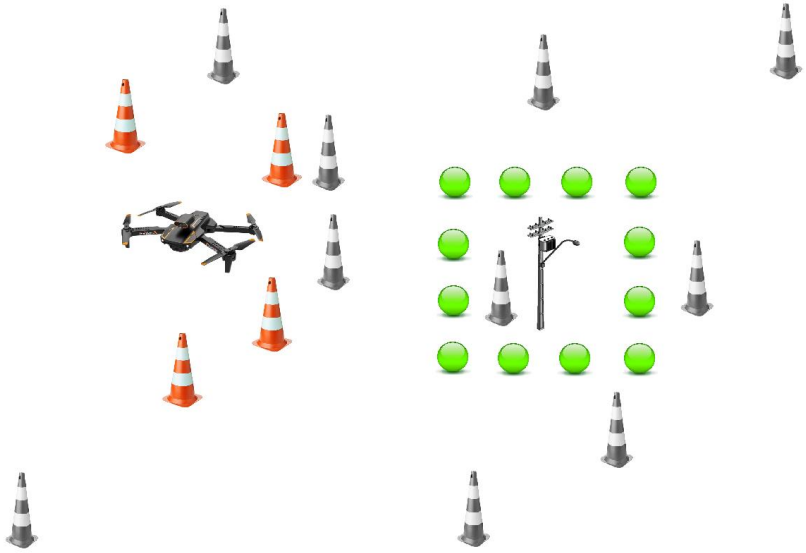


Figura 5.1. Simulação do drone voando entre os obstáculos.

Nesse contexto, há uma possível simplificação na análise de alguns problemas, e com isso, a quantidade de simuladores compatíveis com o problema aumenta significativamente. O uso de simuladores se justifica pela natureza do problema sendo resolvido: é o projeto de um agente autônomo que deve ser executado em um drone. Em sua maioria, drones são equipamentos com um custo alto, difíceis de serem substituídos devido a danos causados por falhas durante os testes. Entretanto, são equipamentos com vários dados de telemetria que podem ser registrados durante o voo (Vasylenko, 2018), possibilitando o uso posterior desses dados em simuladores. Sendo assim, é possível definir uma simulação com valores coletados no mundo real, e essa simulação pode ser uma primeira etapa de validação para a implementação do agente autônomo.

São vários os simuladores disponíveis para problemas dessa natureza (Mairaj; Baba; Javaid, 2019). Dentre eles, pode-se citar o RealFlight, Liftoff, DRL Simulator etc. Por simplicidade, optou-se pelo uso de um simulador em duas dimensões. O simulador escolhido foi o Gymnasyum. Esse simulador é escrito em Python e contém uma interface gráfica simples e intuitiva, ideal para a análise de problemas como o do cenário aqui descrito. Nesse simulador, pode-se aplicar alguma técnica de aprendizado por reforço ou ainda pode-se modificar, via programação, o comportamento padrão para que um operador humano possa utilizar comandos de teclado para controlar um veículo aéreo simulado.

Outras customizações relevantes para o problema têm a ver com o feedback necessário para a interação de um operador humano com o ambiente (virtual ou não). Um dos sensores que poderiam ser implementados com certa facilidade é o LiDAR (Tsai; Nisar; Hu, 2022). O LiDAR é um sensor em duas dimensões com uma rotação num eixo perpendicular ao plano 2D, capaz de gerar uma nuvem de pontos

destacando obstáculos próximos. Dessa forma, é possível utilizar sensores desse tipo para mapeamento em 3D. Como no cenário simulado o problema foi reduzido para duas dimensões, foi implementado um LiDAR virtual em 2D para o simulador.

A missão usada para a simulação, é constituída pelo deslocamento do drone, usando o GPS para mapear o caminho. O drone identifica a presença dos obstáculos, quando estão dentro do raio de alcance do LiDAR, conforme apresentado na Figura 5.1. Os obstáculos de cor vermelha foram identificados, os demais estão fora do alcance, o que dificulta criar uma trajetória perfeita até os objetivos que são identificados pelas coordenadas representadas em verde.

Um dos problemas mais significativos que surgem devido a essa limitação na detecção de obstáculos é o "*deadlock*" ou impasse. Isso ocorre quando o drone tenta alcançar um alvo pré-estabelecido, mas encontra um obstáculo inesperado no caminho. Como o drone não consegue antecipar a presença do obstáculo com antecedência suficiente, ele pode ficar preso em um ciclo de tentativas repetidas de alcançar o destino, e ser impedido pelo mesmo obstáculo repetidamente.

A utilização de *Long Short-Term Memory* (LSTM) (Li; Cai; Li, 2021) em combinação com algoritmos de *Reinforcement Learning* (RL) (Wijaya, 2023) pode tornar o algoritmo mais eficiente e evitar *deadlocks*. LSTM é um tipo de rede neural recorrente projetada para lidar com sequências de dados, o que pode ser útil para modelar o histórico de ações do drone, permitindo que ele tome decisões mais assertivas.

Antes de detalhar cada método, é necessário estabelecer um glossário básico. O controle de um drone por um operador humano, de um ponto de partida até a realização de um objetivo é o que chamaremos de missão. Em cada missão, os sinais dos sensores do

drone, do começo ao fim da missão, são as informações que caracterizam os estados. O espaço de estados é, portanto, contínuo. Para posterior análise, pode ser feita uma redução de dimensionalidade nas informações de estado. Certas variáveis de pouca influência podem ser removidas, o sinal contínuo pode ser amostrado etc. Outro elemento que aparece ao longo de uma missão são as ações. As ações são comandos enviados pelo operador para o drone. Dependendo da natureza dos comandos, as ações podem ser sinais contínuos ou conjuntos discretos. O operador realiza ações de acordo com uma política. Uma política é uma decisão, em determinado estado, por uma ação ou outra. Diferentes operadores implementam diferentes políticas para atingir um objetivo. O objetivo de um ou mais operadores, no contexto de uma missão, é também chamado de uma intenção. As técnicas de aprendizado por imitação elencadas anteriormente se relacionam com esses conceitos através da estratégia utilizada para definir o quanto o operador deve ser imitado.

III. TÉCNICAS DE APRENDIZADO POR IMITAÇÃO

Com base na discussão apresentada anteriormente, é proposto o uso de técnicas de aprendizado por imitação para o problema de manobras de desvio do drone. Serão utilizadas duas técnicas distintas: a clonagem comportamental (Codevilla et al, 2019) e o *Inverse Reinforcement Learning* (Arora; Doshi, 2021). A clonagem comportamental se preocupa em obter um agente autônomo que implementa as mesmas políticas do operador humano. O *Inverse Reinforcement Learning* conta com dois estágios: no primeiro, tenta, através da política (que pode ser inferida relacionando estados e ações) encontrar uma possível intenção do operador. Essa intenção toma a forma de uma função custo, que é utilizada em um ambiente simulado

em um treinamento de *Reinforcement Learning* tradicional, na qual o agente é livre para explorar um ambiente (real ou simulado) e, através de suas ações, alcançar um objetivo minimizando a função custo. Minimizar a função custo corresponde a encontrar um conjunto de políticas ótimas para um determinado cenário. No segundo estágio, após observar as demonstrações do especialista, o objetivo é inferir a função de recompensa que motivou o especialista a seguir as ações tomadas, onde o agente tenta reconstruir a função de recompensa que melhor explique o comportamento observado. A Figura 5.2 ilustra o processo do aprendizado por imitação, onde o especialista toma ações em um ambiente, e o agente tenta reproduzir essas ações.

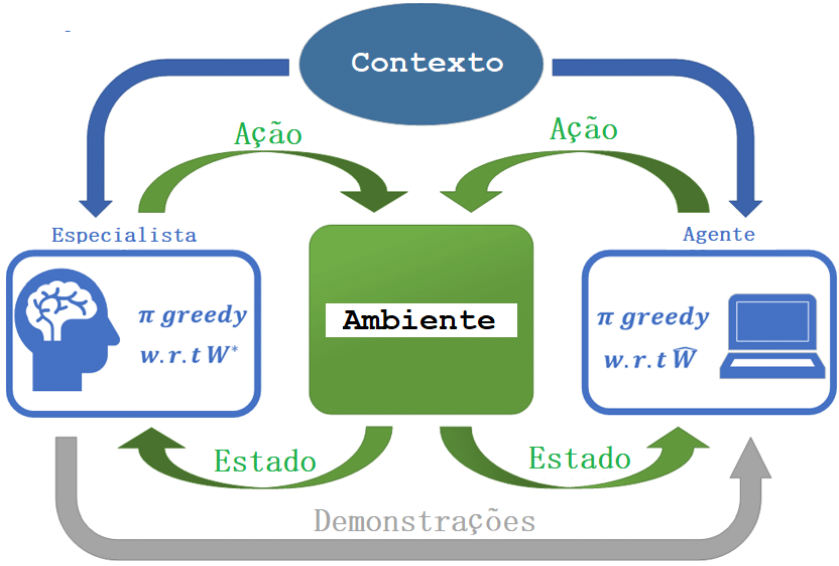


Figura 5.2. Representação da interação entre as partes durante o treinamento de um agente autônomo via aprendizado por imitação. Fonte: Adaptado (Belogolovsky, 2021).

Cada método terá seus prós e contras. A implementação base de uma clonagem comportamental tende a ser mais simples, porém, os resultados do agente autônomo serão, no máximo, tão bons quanto os do operador humano. Sob essa perspectiva, podem surgir vários problemas, como a absorção do viés do operador pelo agente. Além disso, um problema comum da clonagem comportamental é não reagir bem a estados diferentes dos encontrados durante o treinamento, já que a clonagem comportamental não modela uma intenção, e sim uma política.

IV. MISSÃO DO DRONE COM OPERADOR

Para compreender completamente o conceito da missão do drone com o operador, é essencial realizar uma análise do processo e dos objetivos envolvidos. Um framework que pode ser utilizado para guiar essa análise é o *Cross-Industry Standard Process for Data Mining* (CRISP-DM), que fornece uma estrutura sólida para a condução de projetos de mineração de dados e aprendizado de máquina. Vamos explorar como o CRISP-DM se aplica ao contexto da missão do drone com operador:

Entendimento do Negócio (*Business Understanding*): nesta fase inicial, estabelece-se uma compreensão clara dos objetivos gerais da missão do drone e como ela se relaciona com as necessidades do negócio ou da aplicação. Isso pode envolver a definição de metas de voo, critérios de sucesso e a identificação das principais questões a serem abordadas;

Entendimento dos Dados (*Data Understanding*): em seguida, é necessário explorar e compreender os dados relevantes para a missão. Isso inclui os dados provenientes dos sensores do drone, como o LiDAR, bem como quaisquer outros dados relacionados à missão, como coordenadas GPS, informações sobre obstáculos, entre outros;

Preparação dos Dados (*Data Preparation*): uma vez que os dados tenham sido coletados, é necessário prepará-los para análise. Isso pode incluir a limpeza de dados, tratamento de valores ausentes, normalização e outras etapas de pré-processamento para garantir que os dados estejam prontos para serem usados nos algoritmos de aprendizado de máquina;

Modelagem (*Modeling*): nesta etapa, são usadas as técnicas de aprendizado por imitação, como a clonagem comportamental e o *Inverse Reinforcement Learning*. Modelos de aprendizado de máquina são treinados com base nos dados coletados e preparados para imitar o comportamento do operador humano. Isso envolve a criação de políticas de voo que permitam ao drone tomar decisões semelhantes às tomadas pelo operador;

Avaliação (*Evaluation*): os modelos treinados são avaliados quanto à sua capacidade de desempenhar as tarefas da missão de forma eficaz e segura. Isso pode ser feito por meio de simulações e testes em ambientes controlados. A avaliação também pode incluir a comparação do desempenho do agente autônomo com o do operador humano;

Implantação (*Deployment*): uma vez que o agente autônomo tenha demonstrado um desempenho satisfatório e atenda aos requisitos da missão, ele pode ser implantado em um drone real para execução em cenários do mundo real. É importante garantir que todas as considerações de segurança e regulatórias sejam atendidas durante esse processo;

Monitoramento (*Monitoring*) e Manutenção (*Maintenance*): após a implantação, é essencial monitorar continuamente o desempenho do agente autônomo em situações reais e realizar manutenções conforme necessário. Isso pode envolver a reavaliação das políticas de voo,

atualizações de software e ajustes para lidar com mudanças no ambiente ou nas condições da missão.

A aplicação do CRISP-DM permite uma abordagem sistemática e iterativa para o desenvolvimento de soluções de aprendizado por imitação para missões de drones autônomos. Isso ajuda a garantir que as metas de desempenho sejam atingidas e que o sistema opere de forma confiável em situações reais. A Figura 5.3 ilustra o CRISP-DM.

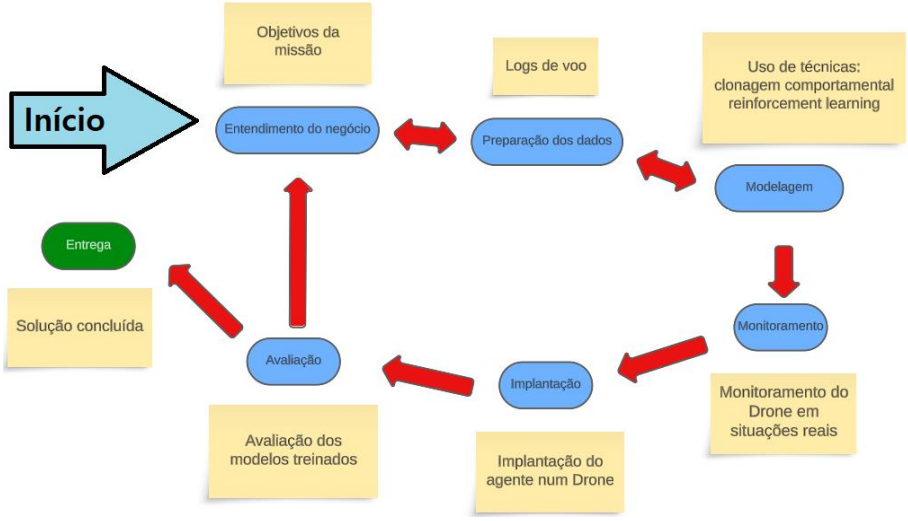


Figura 5.3. Representação do processo CRISP-DM. Fonte: Adaptado de (IBM, 2023)**.**

Em resumo, o processo de missão do drone com operador envolve a análise cuidadosa dos objetivos e dos dados, seguido pela modelagem de políticas de voo autônomo usando técnicas de aprendizado por imitação. A implementação desse processo pode melhorar

significativamente a capacidade do drone de realizar tarefas complexas de forma autônoma, aproveitando o conhecimento e o comportamento de operadores humanos especializados.

Para o cenário desta pesquisa, além das etapas descritas no processo acima, haverá uma etapa realizada a priori: a coleta de dados do especialista em uma missão de desvio de obstáculos com drones. A instrumentação necessária para coleta de dados, seja de sistemas reais ou simulados, é descrita a seguir.

V. COLETA DE DADOS

Para a coleta dos dados, foi utilizado nos experimentos a placa controladora PX4, onde os sinais de sensores, atuadores e comandos ficam registrados num arquivo de log, que pode ser ingerido por um processo de ETL (que consome, transforma e carrega os dados) resultando em dados puros prontos para serem consumidos por analistas (Drescek et al., 2020). A partir do dado puro, isolado das demais informações de logs, o processo CRISP-DM pode refinar a informação e transformá-la em um objeto de valor para o sistema sendo projetado. A Figura 5.4 ilustra o processo de coleta de dados.

Figura 5.4. Extração, transformação e carga. Adaptado de (SUDAN, 2023).

No caso dos simuladores, há um esforço adicional para instrumentação e posterior coleta de dados. Se faz necessário escrever códigos para oferecer ao especialista uma interface de comandos, e códigos para armazenar dados da simulação em algum arquivo que também possa ser consumido por um processo de ETL (Kimball, 2002).

VI. CLONAGEM COMPORTAMENTAL

A clonagem comportamental é uma técnica de aprendizado supervisionado na qual é gerado um agente autônomo que implementa uma política similar a política inferida a partir dos dados de demonstração de um especialista (Souza, 2023).

A demonstração do especialista gera um conjunto de dados com informações de sensores (que definem o estado) e ações do especialista em cada estado. A partir dessas informações, é possível de se treinar uma rede neural com base nos dados, relacionando estados e ações e gerando assim um conjunto de pesos para a rede neural (Hussein et al, 2017). A rede neural pode então ser salva, de forma que possa ser utilizada posteriormente em um simulador para averiguar seu desempenho.

É de se esperar que essa técnica possa encontrar algumas limitações, sendo a mais evidente delas o fato de que o treinamento do agente é condicionado à qualidade das demonstrações do sistema especialista. A Figura 5.5 apresenta um exemplo de falha de controle autônomo, onde a linha pontilhada azul representa a trajetória do especialista, a linha pontilhada vermelha representa a política aprendida, neste exemplo ocorreu algo inesperado, e o sistema não soube retornar, pois esta condição não foi prevista.

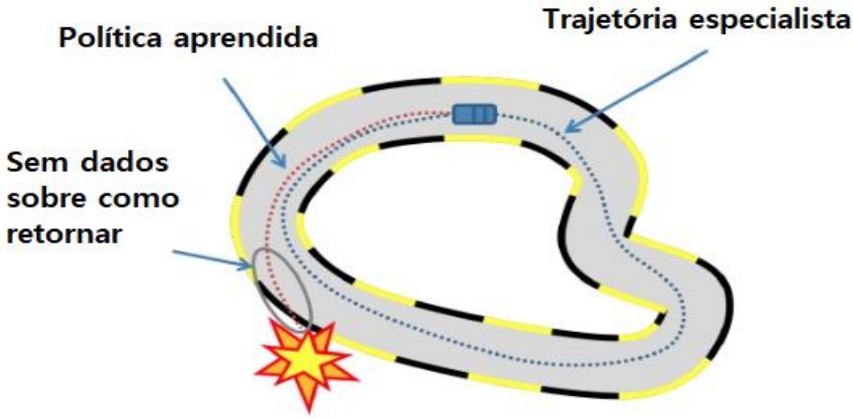


Figura 5.5. Exemplo de falha de controle autônomo ocasionada pelo treinamento de um agente com base numa demonstração incompleta. Fonte: Adaptado de (Brunskill, 2019).

Demonstrações incompletas ou com forte viés podem ter efeitos catastróficos na operação do agente autônomo. Para mitigar esses problemas, foram consideradas algumas técnicas, das quais cabe destacar o aprendizado federado (Yang; Shi; Xia, 2023). O aprendizado federado é geralmente utilizado quando se deseja treinar modelos genéricos para um problema de aprendizado de máquina, porém sem compartilhar informações sensíveis entre organizações e/ou partes interessadas. Usando o aprendizado federado, é possível combinar diferentes modelos de diferentes organizações, a fim de se obter um modelo mais genérico. Ao optar por essa técnica, são abertos caminhos interessantes de arquitetura de aprendizado de máquina distribuído, no qual diferentes especialistas em diferentes organizações poderiam colaborar na construção de modelos cada vez mais genéricos e

abrangentes, no que tange o caso de uso de uma missão de contorno de obstáculos. Algoritmos de combinação de modelos são amplamente difundidos dentro da literatura técnica, então caberia uma análise das opções disponíveis para averiguar o quão realista é a possibilidade do sucesso dessa técnica para o problema de desvio de obstáculos com drones. A Figura 5.6 mostra a representação de uma arquitetura de aprendizado federado.

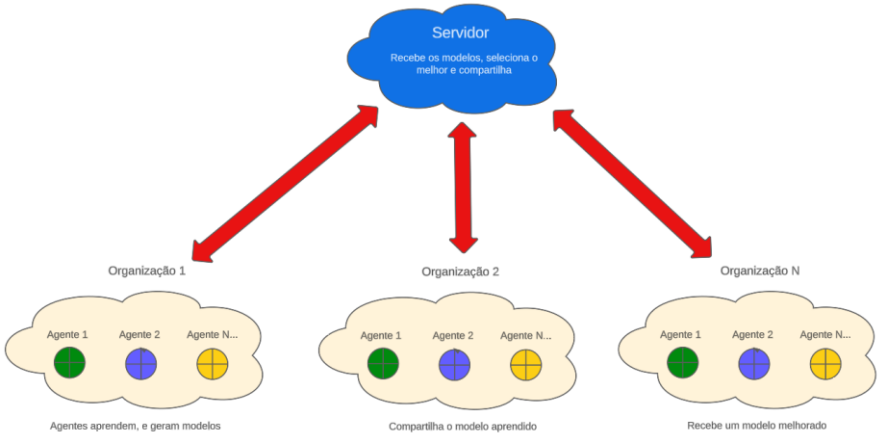


Figura 5.6. Representação de uma arquitetura de aprendizado federado.

VII. REINFORCEMENT LEARNING

O uso de algoritmos de *Reinforcement Learning* (aprendizado por reforço) para controlar o voo autônomo de drones representa um avanço significativo na tecnologia de drones (Wu et al., 2019). No entanto, a criação de uma função de recompensa adequada para guiar o comportamento do drone desviando de obstáculos com o uso de um

LiDAR, e navegação baseada em coordenadas GPS, é uma tarefa complexa e repleta de desafios. A função de recompensa deve equilibrar os objetivos, pois o drone deve desviar de obstáculos identificados pelo LiDAR para garantir segurança, enquanto também busca atingir as coordenadas GPS programadas.

Funções de recompensa frequentemente dependem de vários parâmetros que determinam como as recompensas são calculadas e combinadas. Encontrar o peso apropriado para cada objetivo é uma tarefa complicada, e muitas vezes requer ajustes e experimentação (empirismo) para encontrar uma configuração que funcione bem em diferentes situações. O mapeamento preciso do ambiente é fundamental para avaliar riscos e planejar trajetórias seguras. Erros na detecção de obstáculos ou na representação do ambiente podem levar a decisões não determinísticas, onde o drone fica se movimentando entre pontos sem conseguir chegar a uma determinada coordenada, como se estivesse preso em um labirinto. O LiDAR identifica a proximidade dos objetos quando está a uma determinada distância deles, é como se estivesse dentro de um labirinto, sabe a direção que deve ir, mas não sabe qual o caminho pegar. Quando se olha um labirinto por cima é fácil identificar a saída, pois se pode antever as possibilidades dos caminhos, o que não é possível com o LiDar. O espaço de ações para o drone pode ser vasto e contínuo, tornando a definição de uma política de ação eficaz uma tarefa desafiadora. Determinar como o drone deve manobrar para evitar obstáculos e alcançar as coordenadas GPS é uma questão complexa de controle. Ambientes reais são afetados por incertezas, como a precisão limitada do LiDAR, as variações climáticas e obstáculos móveis. A função de recompensa deve ser robusta para lidar com essas incertezas e evitar tomadas de decisões excessivamente conservadoras ou arriscadas.

À medida que o drone interage com o ambiente, ele pode encontrar situações não vistas durante o treinamento. Na prática, a função de

recompensa deve permitir que o drone aprenda com essas situações novas e inesperadas, mantendo a segurança e eficácia. Neste trabalho são considerados apenas os obstáculos fixos, os quais podem ser alterados de posição em missões diferentes, mas dentro de uma missão eles são estáticos. Na Tabela 5.1 é apresentado um exemplo de código em Python para calcular a função de recompensa para ser utilizada com *Reinforcement Learning.*

Tabela 5.1 – Código para cálculo da função de recompensa.

```
def calculate_reward(current_position,
                     target_position,
                     obstacle_distances,
                     prev_distance_to_target):
    weight_distance = 0.5
    weight_safety = 0.3
    weight_progress = 0.2
    weight_target_reached = 10
    distance_to_target = math.sqrt((current_position[0] -
                     target_position[0])**2 +
                     (current_position[1] -
                    target_position[1])**2)
    reward = -weight_distance * distance_to_target
    if distance_to_target <= 2:
        reward += weight_target_reached
    if distance_to_target < prev_distance_to_target:
        reward += weight_progress
    for distance in obstacle_distances:
        reward -= weight_safety * (12 - distance) / 12
return reward
```

No exemplo da Tabela 5.1, cada variável tem um peso diferente, na faixa de zero até dez, sendo atribuído o peso máximo de 10 quando o drone alcança o objetivo. O alcance máximo do LiDAR neste exemplo é de 12 metros, portanto, no cálculo da distância dos objetos, é utilizado este valor. Essa função é aplicada a cada movimento do drone,

retornando um valor calculado usando suas coordenadas, a distância do alvo, e a distância dos objetos.

VIII. INVERSE REINFORCEMENT LEARNING

Inverse Reinforcement Learning, ou IRL, é o procedimento de encontrar uma função de recompensa que descreve o comportamento observado na demonstração de um especialista, na qual se considera que a política adotada pelo operador é uma política ótima (Arora; Doshi, 2021). A técnica é um meio termo entre aprendizado supervisionado, e aprendizado por reforço, pois usa algoritmos enquadrados em ambas as definições. O IRL é particularmente útil em situações em que uma função de recompensa para uma determinada tarefa é muito complexa para ser definida explicitamente, já que a função de recompensa é inferida a partir da demonstração de um especialista. Essa técnica, por capturar a essência da intenção do especialista, permite ao algoritmo de treinamento do agente autônomo usar a função de recompensa como função a ser otimizada num treinamento de *Reinforcement Learning* clássico (Gugan; Haque, 2023). Dessa forma, o agente autônomo é treinado não para imitar uma política, e sim para gerar uma nova política ótima com base na intenção da tarefa, representada pela função de recompensa. Por conta dessas características, as políticas implementadas por agentes que foram criados através de IRL podem acabar sendo até mais eficientes do que as políticas de operação do sistema por parte especialista nas demonstrações utilizadas para treinamento. A Figura 5.7 apresenta um comparativo entre RL e IRL.

Figura 5.7. Comparativo entre *Reinforcement Learning* (RL) e *Inverse Reinforcement Learning* (IRL). Fonte: Adaptado (Vereshchaka, 2019).

No contexto do aprendizado por imitação, a função de recompensa resultante é utilizada no treinamento de uma nova política ótima. O especialista, portanto, provê uma política através de uma demonstração. O ambiente onde é feita a demonstração também é registrado. Com base nesses dois componentes externos, pode-se aplicar o IRL para gerar uma função de recompensa. A partir dessa função de recompensa, pode-se chegar até uma nova política otimizada (possivelmente livre de viés) através da aplicação do RL. Figura 5.8 ilustra o que é observado e o que é computado no IRL. Essa figura também representa explicitamente a etapa de RL inerente ao treinamento do agente com políticas otimizadas.

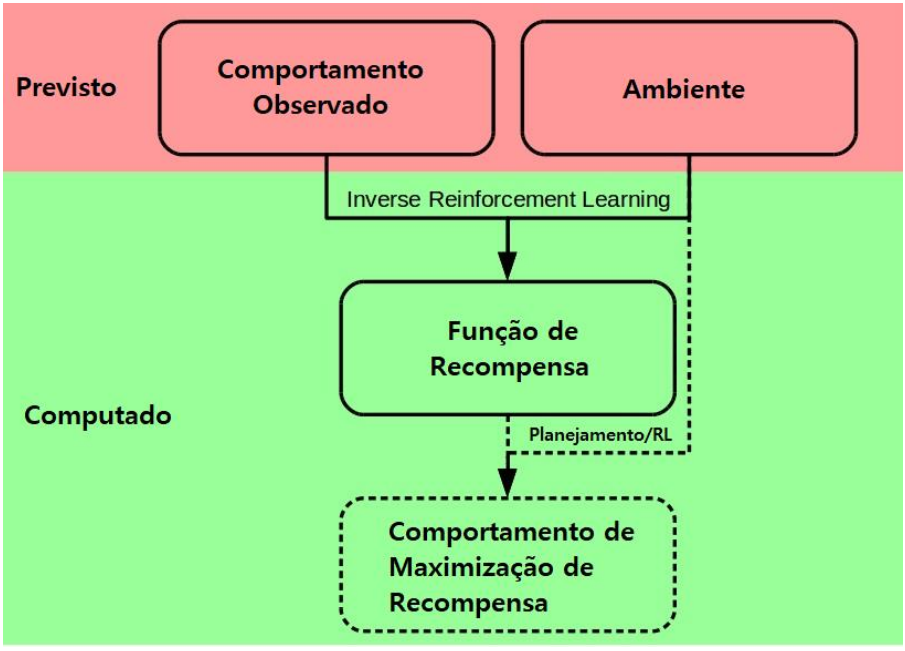


Figura 5.8. Processo do *Inverse Reinforcement Learning* (IRL). Fonte: Adaptado (Kasenberg, 2017).

IX. COMPARATIVO ENTRE OS MÉTODOS APRESENTADOS

Os métodos apresentados pertencem ao conjunto maior de técnicas de aprendizado por imitação. A existência dessas estratégias distintas é justificada pelo fato de que cada estratégia conta com prós e contras para diferentes cenários. As duas abordagens são diferentes no campo do aprendizado por imitação, ambas têm o objetivo de ensinar agentes autônomos a imitar o comportamento humano, mas utilizam métodos distintos para alcançar esse objetivo.

A. CLONAGEM COMPORTAMENTAL

Como vantagem dessa técnica, pode-se citar o fato de que é simples de ser implementada, quando comparada às demais aqui apresentadas, a ponto de servir como benchmark para outras abordagens (Sun, 2022). Além disso, também é muito efetiva em alguns cenários (principalmente quando a operação real do sistema não difere muito do universo de demonstrações do especialista).

Como pontos negativos, pode-se citar o fato de que é suscetível a falhas catastróficas, pois até mesmo pequenos desvios em estados conhecidos podem levar a estados desconhecidos não contemplados nas demonstrações. Outro fato relevante, ela não leva explicitamente em consideração a incerteza associada às ações do especialista humano, assumindo que as ações observadas são as ações certas a serem imitadas.

O uso dessa técnica em aplicações de pouca criticidade pode ser adequado, tendo em vista a rapidez com que se pode chegar a um resultado satisfatório.

B. REINFORCEMENT LEARNING

Pode ser aplicado em uma variedade de cenários sem a necessidade de conhecimento prévio do comportamento humano, o que torna mais adequado para tarefas em que o comportamento de um agente não precisa ser modelado explicitamente (Sutton, 2020). É flexível, permite que os agentes aprendam a partir de suas próprias experiências, se destaca também pela sua eficiência em aprender políticas de ação ótima quando a exploração é necessária.

Tem a desvantagem de depender de recompensas bem projetadas e, muitas vezes, pode ser difícil definir recompensas que representam com precisão os objetivos desejados. Em algumas situações pode

requerer muitas interações com o ambiente para aprender políticas eficazes, aumentando os custos e/ou até inviabilizando o projeto.

C. INVERSE REINFORCEMENT LEARNING

Como principal vantagem, pode-se citar o fato de que ao fim do treinamento, o agente resultante poderá ser tão eficiente ou até mais eficiente que o especialista que realizou a demonstração utilizada no treinamento.

Como principal desvantagem, pode-se citar o fato de que a implementação desse método tem muita complexidade, por conta da inferência de função de recompensa e posterior treinamento do agente via RL.

Por fim, pode-se afirmar que esse método é adequado para aplicações de alta complexidade, nas quais a função de recompensa (ou intenção) é difícil de ser representada analiticamente.

X. CONCLUSÃO

Ao considerar o uso de *Reinforcement Learning* (RL), *Inverse Reinforcement Learning* (IRL) e aprendizado por imitação no contexto do objetivo do presente trabalho, a escolha do IRL se justifica em fatores específicos da tarefa e das necessidades do projeto, sendo os seguintes motivos os principais:

No contexto, é essencial considerar a segurança e a eficiência. O IRL permite que seja incorporado diretamente o conhecimento de especialistas humanos ou a demonstração de comportamento humano seguro e eficaz. Isso é particularmente relevante quando se trata de manobras de desvio de obstáculos e contorno de objetos, onde o conhecimento especializado é essencial.

O voo autônomo em ambientes dinâmicos requer a tomada de decisões complexas em tempo real. O IRL é capaz de capturar essas decisões complexas e adaptar o comportamento do drone com base em cenários reais observados, fornecendo um grau mais alto de confiabilidade em comparação com RL puro.

A segurança deve ser considerada crítica em projetos de drones, especialmente em situações em que a falha pode ter sérias consequências. O uso de IRL permite garantir que o comportamento do drone seja alinhado com padrões de segurança estabelecidos por especialistas, minimizando riscos.

O IRL pode reduzir a necessidade de experimentação real arriscada ao permitir que o drone aprenda a partir de demonstrações pré-existentes, evitando acidentes potenciais durante o treinamento.

É importante destacar que a escolha da abordagem foi decidida pelos requisitos específicos do projeto e das tarefas. Em certos cenários, uma combinação de RL, IRL e aprendizado por imitação pode ser a abordagem mais adequada para alcançar um equilíbrio entre autonomia, segurança e eficácia. Portanto, ao optar pelo IRL, a decisão é fundamentada em aproveitar o conhecimento humano e especializado para criar sistemas de voo autônomo mais confiáveis e seguros.

REFERÊNCIAS

(ARORA, 2021) ARORA, S.; DOSHI, P. “A survey of inverse reinforcement learning: Challenges, methods, and progress. Artificial Intelligence,” v. 297, p. 103500, 2021.

(BELOGOLOVSKY,2021) BELOGOLOVSKY, STAV, et al. “Inverse Reinforcement Learning in Contextual MDPs,” Machine Learning, vol. 110, no. 9, 2021, pp. 2295–2334, doi:10.1007/s10994-021-05984-x.

(BRUNSKILL, 2019) BRUNSKILL, E. “Lecture 7: Imitation Learning in Large State Spaces,” 2019. <https://web.stanford.edu/class/cs234/CS234Win2019/slides/lecture7.pdf>

(CHOI, 2017) CHOI, S. et al. “Inverse reinforcement learning control for trajectory tracking of a multirotor UAV. International Journal of Control, Automation and Systems,” 2017. doi:10.1007/s12555-015-0483-3

(CODEVILLA, 2019) CODEVILLA, FELIPE et al. “Exploring the limitations of behavior cloning for autonomous driving. In Proceedings of the IEEE/CVF International Conference on Computer Vision,” 2019.

(WU, 2019) WU C. et al., "UAV Autonomous Target Search Based on Deep Reinforcement Learning in Complex Disaster Scene," EEE Access, 2019, doi: 10.1109/ACCESS.2019.2933002.

(DRESCEK, 2020) DRESCEK, U. et al. “Spatial ETL for 3D Building Modelling Based on Unmanned Aerial Vehicle Data in Semi-Urban Areas. Remote Sensing,” 2020. doi: 10.3390/rs12121972.

(GUGAN, 2023) GUGAN, G.; HAQUE, A. “Path Planning for Autonomous Drones: Challenges and Future Directions,” Drones, 2023, doi: 10.3390/drones7030169.

(FARAMA, 2023) Gymnasium Documentation. Disponível em: <https://gymnasium.farama.org/index.html>. Acesso em: 13 out. 2023.

(HUSSEIN, 2017) HUSSEIN, A. et al. “Imitation Learning. ACM Computing Surveys,” 2017, doi:10.1145/3054912.

(IBM, 2023) IBM SPSS Modeler CRISP-DM Guide. Disponível em: <https://www.ibm.com/docs/it/SS3RA7_18.3.0/pdf/ModelerCRISPDM.pdf>.

(KASENBERG,2017) KASENBERG, D. “Inverse Reinforcement Learning to the Rescue”. 2017. Disponível em: <https://dkasenberg.github.io/inverse-reinforcement-learning>

(KIMBALL, 2002) KIMBALL, R. et al. “The data warehouse toolkit: the complete guide to dimensional modeling”. 2nd Wiley Computer Publishing. 2002. ISBN 0-471-20024-7.

(KUMAR, 2022) KUMAR, A. et al. “When Should we Prefer Offline Reinforcement-Learning over Behavioral Cloning?”. Department of EECS, UC Berkeley, 2022. doi: 10.48550/arXiv.2204.05618.

(LI, 2021) LI, S.; CAI, T.; LI, J. Trajectory Prediction using Generative Adversarial Network in Multi-Class Scenarios, 2021. doi:10.48550/arXiv.2110.11401.

(MAIRAJ, 2019) MAIRAJ, A.; BABA, A. I.; JAVAID, A. Y. “Application specific drone simulators: Recent advances and challenges. Simulation Modelling Practice and Theory,” v. 94, jul. 2019. <doi.org/S1569190X19300048>

(ROSS, 2011) ROSS, S. et al. “A reduction of imitation learning and structured prediction to no-regret online learning,” In AISTATS, 2011. 1, 2 doi: 10.48550/arXiv.1806.06877.

(SOUZA, 2023) SOUZA, F. R. “Aprendizado por reforço assistido por imitação para jogos digitais,” 2023. <https://repositorio.ufjf.br/jspui/handle/ufjf/15482>

(SUTTON, 2020) SUTTON, R. et al., “Reinforcement learning: an introduction.” ISBN 9780262039246. The MIT Press. 2020.

(SUDAN, 2023) SUDAN A. What Is ETL And How the ETL process works. <https://www.datachannel.co/blogs/what-is-etl-and-how-the-etl-process-works>

(SUN, 2022) SUN, X. et al., “A Benchmark Comparison of Imitation Learning-based Control Policies for Autonomous Racing,” 2022. doi:10.48550/arXiv.2209.15073.

(TSAI, 2021) TSAI, C.-Y.; NISAR H.; HU Y. C. “Mapless LiDAR Navigation Control of Wheeled Mobile Robots Based on Deep Imitation Learning,” IEEE Access, 2021, doi: 10.1109/ACCESS.2021.3107041.

(VASYLENKO, 2018) VASYLENKO, M. P. et al. "Telemetry System of Unmanned Aerial Vehicles,” 2018. DOI: 10.18372/1990-5548.57.13244.

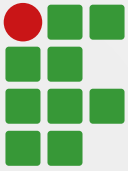
(VERESHCHAKA, 2019) VERESHCHAKA, A. “Lecture 15: Imitation Learning: Behavior Cloning”, 2019. <https://cse.buffalo.edu/~avereshc/rl_fall19/lecture_15_Imitation_Learning_Behavior_Cloning_IRL.pdf>

(WIJAYA, 2023) WIJAYA, S. et al. “Long short-term memory (LSTM) model-based reinforcement learning for nonlinear mass spring damper system control, Procedia Computer Science,” v. 216, p. 213–220, 2023.

(YANG, 2022) YANG B.; SHI H.; XIA X. “Federated Imitation Learning for UAV Swarm Coordination in Urban Traffic Monitoring,” IEEE Transactions on Industrial Informatics, vol. 19, no. 4, pp. 6037-6046, 2022, doi: 10.1109/TII.2022.3192675.

(ZHENG, 2022) ZHENG, Boyuan, et al. “Imitation Learning: Progress, Taxonomies and Challenges,” 2022. doi: 10.48550/arXiv.2106.12177.

Pg.



INSTITUTO FEDERAL
Santa Catarina

Câmpus
Florianópolis