UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
ESCOLA DE ENGENHARIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

Lucas Henz Garcia

Detecção de Emoções utilizando Redes Neurais Convolucionais em Sistemas com Recursos Limitados de *Hardware*

Porto Alegre

2021

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
ESCOLA DE ENGENHARIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

Lucas Henz Garcia

Detecção de Emoções utilizando Redes Neurais Convolucionais em Sistemas com Recursos Limitados de *Hardware*

Projeto de Diplomação II, apresentado ao Departamento de Engenharia Elétrica da Escola de Engenharia da Universidade Federal do Rio Grande do Sul, como requisito parcial para a obtenção do grau de Engenheiro Eletricista

UFRGS

Orientador: Prof. Dr. Tiago Oliveira Weber

Porto Alegre
2021

Lucas Henz Garcia

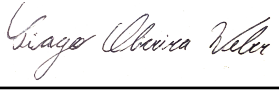
Detecção de Emoções utilizando Redes Neurais Convolucionais em Sistemas com Recursos Limitados de *Hardware*

Projeto de Diplomação II, apresentado ao Departamento de Engenharia Elétrica da Escola de Engenharia da Universidade Federal do Rio Grande do Sul, como requisito parcial para a obtenção do grau de Engenheiro Eletricista

BANCA EXAMINADORA

Prof. Dr. Alexandre Balbinot
UFRGS

Prof. Dr. Raphael Martins Brum
UFRGS



Prof. Dr. Tiago Oliveira Weber
Orientador - UFRGS

Aprovado em 25 de Novembro de 2021.

Resumo

O uso de redes neurais convolucionais na detecção de emoções através da expressão facial (FER, do inglês *Facial Emotion Recognizer*) em sistemas com recursos limitados de *hardware* é uma opção viável por aliar assertividade e sistemas finais com baixa complexidade de uso. A presente pesquisa detalha a análise estatística de seis topologias de código aberto com métricas de assertividade e de número de parâmetros utilizados, com e sem estratégia de *Data Augmentation*, e descreve o projeto de implementação de um detector de emoções baseado na arquitetura com melhor desempenho em um *Raspberry Pi* com câmera. Em adição, a rede foi simplificada e comprimida, através de poda computacional, utilizando esparsidade constante, e de quantização com alcance dinâmico. A estrutura apresentou como resultado uma assertividade média de 68,5% para a base de dados FER-2013, utilizando 398.000 parâmetros (70% de esparsidade), com latência média de 168 ms e tamanho médio de 663.255 *bytes*, sendo capaz de classificar todas as emoções do espectro de Ekman em tempo real.

Palavras-chave: Redes Convolucionais, FER, Poda Computacional, Quantização, *Raspberry Pi*.

Abstract

The use of convolutional neural networks to detect emotions through facial expression (FER) in systems with limited resources is a viable option for combining assertiveness and final systems with low complexity of use. This research details the statistical analysis of six open source topologies with metrics of assertiveness and number of parameters used, with and without Data Augmentation strategy, and describes the implementation project of an emotion detector based on the architecture with the best performance in a Raspberry Pi with camera. In addition, the network was simplified and compressed, through pruning, using constant sparsity, and quantization with dynamic range. The structure presented as a result an average assertiveness of 68.5% for the FER-2013 database, using 398,000 parameters (70% sparsity), with an average latency of 168 ms and an average size of 663,255 bytes, being able to classify all emotions in real-time of the Ekman spectrum.

Keywords: Convolutional Networks, FER, Pruning, Quantization, Raspberry Pi.

Lista de Figuras

Lista de Tabelas

Sumário

1 Introdução

As expressões faciais fornecem informações sobre a resposta emocional inerente ao ser humano, de forma que exercem papel fundamental na interação humana como forma de comunicação não verbal. A expressão facial reproduz uma emoção que pode transmitir uma mensagem única e, em muitas circunstâncias, mais completa que a própria linguagem verbal, pois independe da cultura e da região que o transmissor da emoção vive (EKMAN, 2007).

As diversas aplicações da interpretação da emoção humana, tais como detecção de patologias (depressão), ensino e satisfação por parte de um cliente usando um determinado produto, trouxeram a atenção de diversos pesquisadores ao campo de estudos. A partir dos anos 2000, diversas técnicas que auxiliam em tarefas relacionadas a esse fim foram propostos, como o reconhecimento do discurso do emissor ou a detecção de enrijecimento dos músculos e batimentos cardíacos do indivíduo para detectar uma emoção determinada. Contudo, essas propostas carecem de robustez e de desempenho, já que necessitam de um circuito complexo para interpretar diversos dados de entrada (SCHWENKER; SCHERER, 2019).

Com base nesse fato e nos avanços dos estudos de inteligência artificial e de visão computacional nas últimas décadas, modelos que tomam em consideração somente a imagem do rosto do emissor para detecção de emoções faciais humanas, também conhecidos como FER (do inglês, *Facial Emotion Recognizer*), se tornaram os mais difundidos na comunidade acadêmica. Os melhores modelos FER utilizam técnicas de inteligência artificial que optimizam seu desempenho, como as redes convolucionais - que interpretam padrões locais da imagem através de operações de convolução -, e sua carga computacional exigida (GIANNOPOULOS ISIDOROS PERIKOS, 2017).

Com a maior difusão do uso de aparelhos móveis e dispositivos de *hardware* limitado, muitas pesquisas se direcionaram para a utilização de modelos de inteligência artificial nesses dispositivos. Porém, a maior dificuldade nessa implementação é a diminuição de parâmetros utilizado pela rede, já que é necessário um balanceamento de custo de energia e de processamento com o desempenho do modelo (WARDEN; SITUNAYAKE, 2021). Com isso, pesquisas envolvendo modelos FER em *hardware* limitados estão sendo propostas recentemente, como fez (SHAO; QIAN, 2019) ao propor uma rede convolucional com menor número de parâmetros para esse fim.

Com base nas considerações descritas, a presente monografia propõe a implementação de um sistema FER, com base em redes convolucionais superficializadas inspiradas nos modelos de (RIAZ YAO SHEN, 2020) e (SHAO; QIAN, 2019), em um *hardware* limitado.

Assim, o modelo visa aliar o desempenho e a otimização para que seja factível a detecção de emoção através do dispositivo embarcado.

1.1 Justificativa

A justificativa para o desenvolvimento de trabalhos na área de reconhecimento de emoções se dá pelo amplo espectro de aplicação dos sistemas FER, principalmente utilizando métodos de aprendizado de máquina com redes convolucionais. Os seguintes campos de aplicação desse tipo de sistema exemplifica tal amplitude desse espectro (LI *et al.*, 2020):

- Monitoramento de motoristas: o número de acidentes fatais pode ser diminuído baseado no reconhecimento da emoção do condutor - identificando níveis de irritabilidade, fadiga e estresse - por uma câmera no veículo;
- Serviços médicos: identificação de patologias, como a depressão, e a inserção de pessoas com níveis de autismo na sociedade - pessoas com no espectro autista possuem dificuldade de reconhecer emoções;
- Marketing e Publicidade: a reação mais intuitiva a uma propaganda é sua emoção através da expressão facial. Se aplicada a uma loja, por exemplo, o proprietário pode analisar a reação dos clientes a um certo produto ou anúncio;
- Ensino: um professor ou palestrante pode verificar a reação de seu público a sua exposição através das emoções aferidas pela sua audiência, remotamente ou presencialmente;
- Interatividade com jogos digitais: os jogos digitais já são intrínsecos a nossa cultura, mas ainda trabalhar com a interatividade do usuário com o jogo ainda é uma dificuldade. A retroalimentação da emoção do jogador pode auxiliar no desenvolvimento e mudança do jogo, aumentando o entretenimento do público;

Ainda, há outros campos sendo explorados por tais sistemas, como o da psicologia (identificação de padrões emocionais entre humanos e animais) e da segurança pública (análise de emoções de pessoas em certos ambientes), que dão dimensão da relevância dos sistemas FER.

1.2 Objetivo Geral

O objetivo geral é o desenvolvimento de um detector de emoções humanas com taxa de acerto similar a de sistemas usando rede neurais convolucionais e capaz de rodar

em computador de placa única com restrição de recursos para o problema de FER através de poda computacional e quantização.

1.3 Objetivos Específicos

- Realizar treinamento para base de dados na literatura sem considerar recursos limitados inicialmente.
- Realizar treinamento para base de dados na literatura considerando a redução de parâmetros e taxa de acerto.
- Implementar e testar a arquitetura em um computador de placa única usando câmera.

2 Fundamentação Teórica e Revisão Bibliográfica

Neste capítulo são discutidos e revisados temas e trabalhos pertinentes e necessários para o entendimento da pesquisa. Primeiro é revisado o contexto das emoções na cultura e no âmbito escolar e, após, o aprendizado de máquina.

2.1 Expressões Faciais e Emoções

Todos os seres humanos estão diariamente expostos a estímulos do ambiente em que vivem e a primeira resposta que damos a esses estímulos são nossas emoções através de nossas expressões faciais (EKMAN, 2007). Tal reatividade independe do ambiente - seja no lar, no ambiente laboral ou de lazer - e do grau de afetividade com outras pessoas do espaço compartilhado. Ela é inerente à natureza humana.

Darwin foi o primeiro pesquisador a constatar que as emoções são heranças adaptativas da nossa espécie, independentes da cultura que estamos inseridos ou lugar geográfico que estamos localizados. A confirmação de tais estudos surgiu mais tarde com os estudos do psicólogo e antropólogo Ekman, que realizou estudos de campo em diversos países como Japão, Brasil e Estados Unidos. Porém, foi numa tribo sem comunicação externa que vivia sem nenhum tipo de tecnologia, em Nova Guiné, que o pesquisador fez sua pesquisa mais relevante que confirmou que existe emoções primárias básicas em qualquer ser humano (TENHOUTEN, 2018). A Figura 1 ilustra duas poses capturadas por Ekman na Nova Guiné, sendo de felicidade e de tristeza da mesma pessoa.

Figura 1 – Exemplo de Capturas de Poses na Nova Guiné.

Fonte: Eckman (2007).

O espectro de Ekman de emoções básicas universais contempla seis emoções: raiva, medo, nojo, tristeza, surpresa e felicidade - à parte dessas situa-se a neutralidade. Mais tarde, o próprio pesquisador incluiu a emoção de desprezo ao espectro - emoção dissidente da emoção nojo -, mas seu modelo mais utilizado ainda é o de seis emoções (TENHOUTEN, 2018).

Eckman (2007) ainda diz que algumas emoções podem ser negativas, como o nojo, e positivas, como a felicidade. Porém, tal simplificação precisa ter o contexto, que serve de gatilho para provocar a emoção, analisado. Para algumas pessoas, chorar vendo um filme triste não necessariamente é uma situação de desprazer, por exemplo. Tendo o devido cuidado, as emoções nojo, tristeza, raiva e medo podem ser chamadas de negativas, enquanto a felicidade é claramente positiva. Por sua vez, a surpresa não possui nenhuma categoria, pois além de ser a mais breve das emoções - mais difícil de ser capturada também - ela é tida como uma emoção que precede uma emoção positiva ou negativa, não se enquadrando em nenhuma das duas categorias.

Além do espectro de Ekman, há ainda modelos que levam em conta aspectos psicossociais que apontam ainda para camadas secundárias das emoções básicas, como o modelo de Plutchik e o de MacLean, mas que não serão abordados abordados na monografia.

2.2 Aprendizado de Máquina

Aprendizado de máquina está inserido dentro do contexto de inteligência artificial que surgiu como um novo paradigma de programação (CHOLLET, 2017) e que engloba o aprendizado profundo de máquina - detalhado posteriormente.

Antes dessa abordagem, todas as regras para automatizar uma máquina que pudesse suprir o esforço humano eram explicitamente escritas e com um conjunto de dados a resposta era adquirida. O aprendizado de máquina leva em conta o conjunto de dados, assim como a abordagem clássica para processar informações, e a resposta para, assim, calcular as regras, sendo feito o aprendizado.

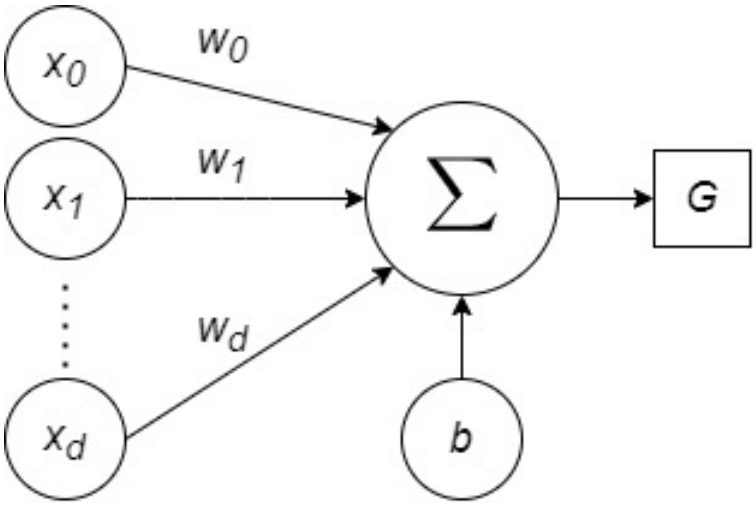
2.2.1 Redes Neurais e Aprendizado Profundo

Há infinitas formas de criar um modelo treinável que aprenda por si só, mas o mais eficiente de todos são as redes neurais artificiais, também chamadas de RNAs (AGHDAM, 2017). RNAs são grupos interconectados por pequenas unidades computacionais chamadas de neurônios que, por usa vez, imitam os neurônios biológicos do cérebro humano de forma simplificada para obtenção de um elemento de processamento.

O neurônio artificial recebe a entrada com um peso atribuída a cada uma, realizando o somatório dessas, posteriormente, com um valor de *bias*. Por fim, é feita ativação do neurônio através de uma função não linear.

A Figura 2 mostra o diagrama de um neurônio artificial, onde x_0, x_1 e x_d são as entradas, w_0, w_0 e w_d são os pesos, b é o valor de bias e G é a função de ativação.

Figura 2 – Neurônio Artificial.



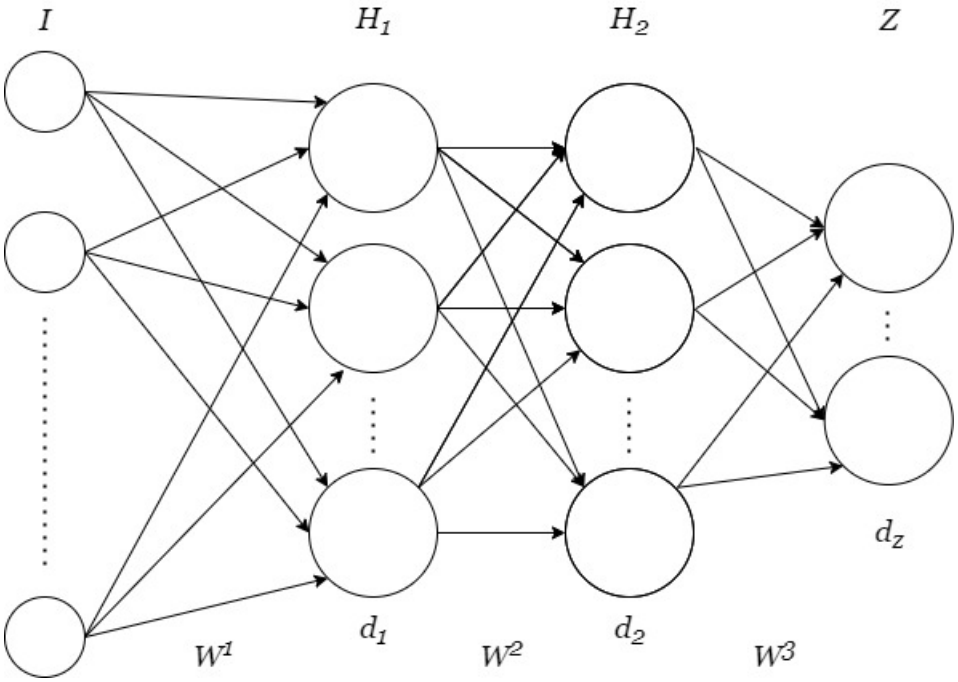
Fonte: Adaptado de Aghdam (2017).

A estrutura básica de uma RNA se divide em três principais elementos:

- Camada de entrada: camada onde é recebido a entrada de dados externos.
- Camada(s) oculta(s): são todas as camadas que se localizam entre a camada de entrada e a de saída, onde é feito o processamento e propagação de valores de entrada em saída. Possui pesos associados, um valor de bias intrínseco e uma função de ativação neuronal - responsável por decidir qual informação será passada ou não adiante.
- Camada de saída: camada que fornece a saída do modelo, podendo ter valores contínuos ou discretos, dependendo da aplicação.

A Figura 3 ilustra tal estrutura, onde I é a camada de entrada, H_1 e H_2 são as camadas ocultas e Z é a camada de saída. W^1, W^2 e W^3 são os pesos e d é número de neurônios de cada camada.

Figura 3 – Estrutura de uma Rede Neural Artificial.



Fonte: O autor.

Nesse contexto, o aprendizado de máquina profundo refere-se a redes que possuem mais que uma camada oculta (CHOLLET, 2017). Enquanto as primeiras estruturas de RNAs possuíam uma ou duas camadas ocultas - chamadas agora de aprendizado superficial de máquina - as de aprendizado profunda possuem dezenas, centenas ou até mesmo milhares, a depender da aplicação.

Uma vez definida a estrutura da rede, deve-se selecionar mais dois parâmetros: a função de perda e a o optimizador. A função de perda mensura a diferença entre o alvo da resposta e a resposta predita - podendo ser chamada, também, de função objetivo -, medindo a qualidade da rede. Por outro lado, o optimizador leva em conta a quantidade mensurada pela função de perda para redefinir os pesos das camadas ocultas. Esse processo de realimentação é chamado de retropropagação (*backpropagation*).

2.2.1.1 Redes Neurais Convolucionais

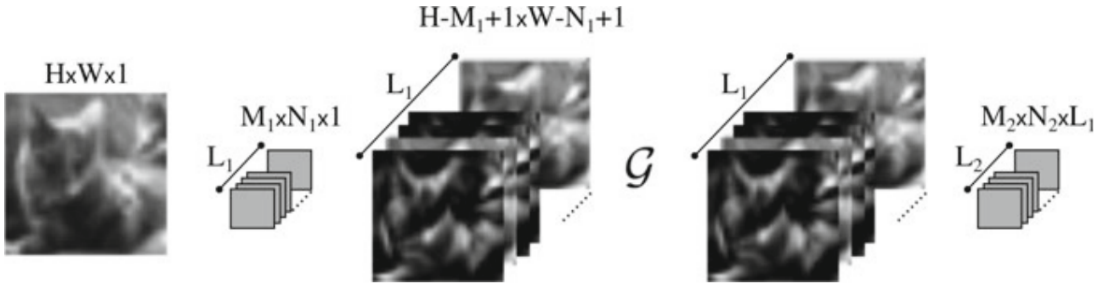
A principal diferença entre redes neurais comuns e redes neurais convolucionais (CNNs) é que a primeira interpreta um padrão global enquanto a rede convolucional aprende por padrões locais, sendo amplamente utilizadas no reconhecimento de imagens (CHOLLET, 2017).

O reconhecimento dos padrões locais é feita por janelas de *pixels* da imagem convolucionadas com uma matriz, geralmente quadrada, que também é chamada de filtro ou neurônio deslizante. O resultado dessa operação, que é feita justamente na camada de convolução, é o mapa de características (AGHDAM, 2017).

Normalmente esses filtros convolucionais são *arrays* de três dimensões: os dois primeiros são valores arbitrários e o terceiro é sempre igual ao número de canais recebido na primeira camada.

Por exemplo, a Figura 4 mostra a convolução de uma imagem preto e branco (escala de cinza) de dimensão H x W x 1 por um filtro de dimensões M_1 x N_1 x 1, produzindo um mapa de características de dimensões $H - M_1$ x $W - N_1$ x 1. O mesmo ocorre para a camada posterior com o filtro de dimensões arbitrária de M_2 por N_2. Vale ressaltar que se a imagem fosse colorida, a terceira dimensão seria do valor 3.

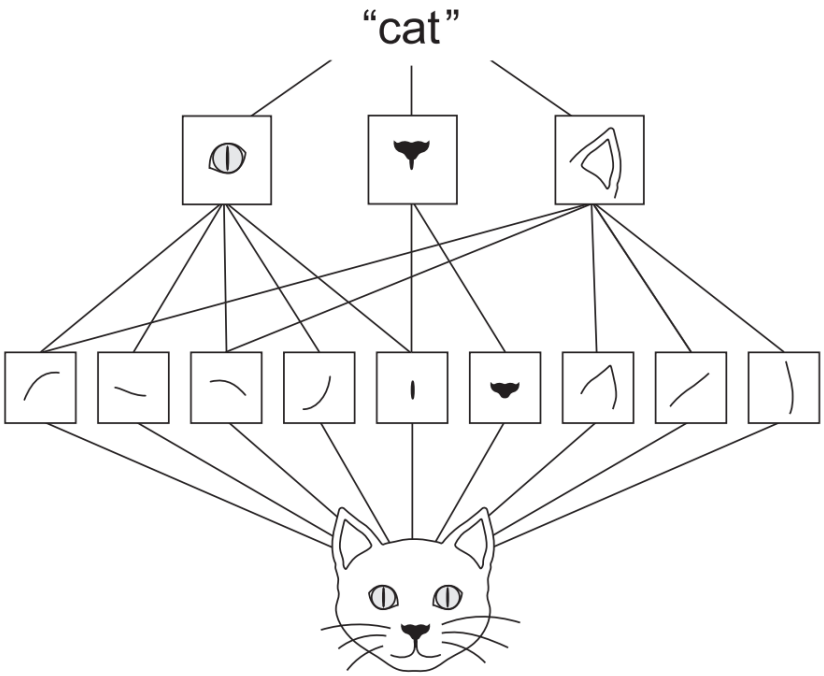
Figura 4 – Filtros convolucionais aplicados numa imagem em escala de cinza.



Fonte: Aghdam (2017).

Uma importante característica das CNNs é que elas hierarquizam padrões espaciais. Por exemplo, quando uma CNN tem como tarefa detectar que a imagem de um animal é um gato, nas primeiras camadas é feita a detecção mais simples da imagem, como as bordas da imagem. Nas camadas posteriores é feita, então, a detecção de orelhas, olhos e focinho para, no final, saber que a imagem de entrada é de fato um gato. Essa hierarquia é ilustrada na Figura 5.

Figura 5 – Representação visual da hierarquia das camadas de uma CNN para a detecção da imagem de um gato.

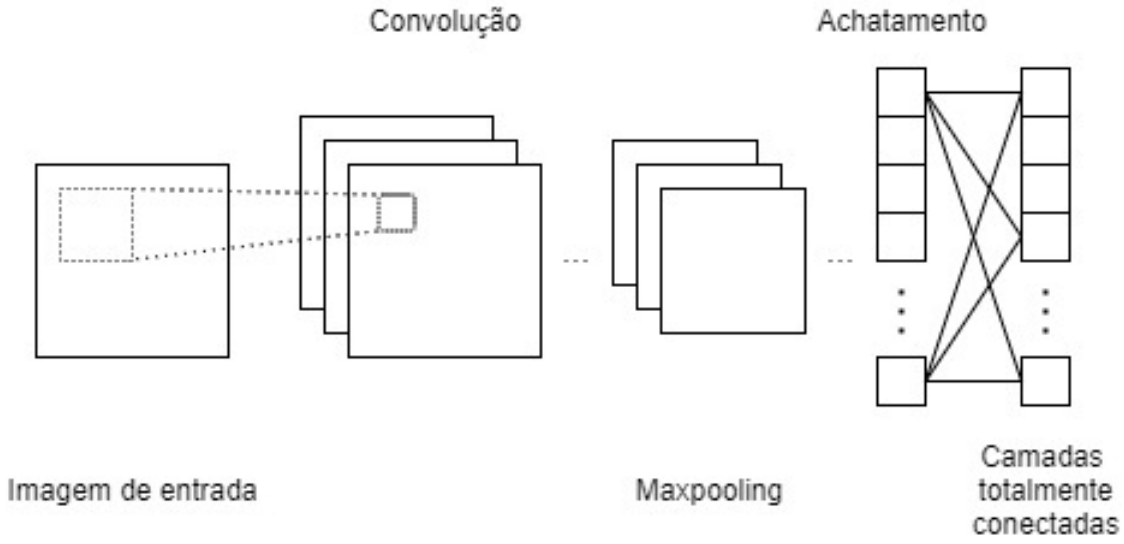


Fonte: Chollet (2017).

A outra camada importante nas redes convolucionais é a camada de *Max-Pool* (podendo ser mais de uma camada). Essa camada auxilia na formação da estrutura hierárquica na fase de treinamento. Além disso, ela reduz os parâmetros e torna as transições mais abruptas. O processo é feito pelo janelamento do mapa de características de um tamanho de p x p, chamado de *stride*. Nesse janelamento é tomado em conta somente o maior valor adquirido na janela. Outro conceito associado a esse processo é o *padding* - preenchimento (geralmente com zeros) do tamanho do perímetro onde o filtro convolucional opera para que não haja perda de informação em regiões periféricas da camada.

Importante salientar que as camadas convolucionais e de Max-pool não estão totalmente conectadas, sendo feito o processo de forma unidirecional - ainda pode haver processos de normalização entre elas para melhor convergência. Ao final dessas, é feita o achatamento (do inglês, *Flatten*) do valor para um vetor unidimensional. Por fim, é transportado esse vetor final para as camadas finais que estão totalmente conectadas, dispostas na forma padrão de RNA (VENKATESAN; LI, 2017). A Figura 6 mostra a estrutura de uma CNN completa.

Figura 6 – Representação de uma rede convolucional completa.



Fonte: O autor.

2.2.1.2 Estruturas de Redes Neurais Convolucionais

Nos últimos anos muitas estruturas de redes neurais convolucionais foram criadas, não somente na academia mas também no âmbito empresarial (KARIM, 2021). A arquitetura LeNET-5, considerada a pioneira no campo de estudo, foi criada em 1998, por Yann Lecun - muito utilizada para reconhecimento de caracteres, com precisão média de 99,2%.

A partir de 2012, com a necessidade de se resolver problemas mais complexos de identificação por imagem, muitas arquiteturas foram propostas, com destaque para a rede proposta pela universidade de Oxford, a VGG-16, de 2014, que superou em taxa de acerto da *AlexNet* (estrutura proposta por Alex Krizhevsky, Ilya Sutskever e Geoff Hinton, em 2012) no concurso de identificação de imagens do banco *ImageNet* (base de dados que contém mais de 14 milhões de imagens com mais de 20.000 classes) - 92,7% e 84,7%, respectivamente (DANG, 2021).

Posteriormente foram propostas mais arquiteturas, com destaque para a *Inception*, proposta pela *Google* (2014) e *ResNet*, proposta pela *Microsoft* (2014), com taxa de acertos de 93,3% e 96,4%, respectivamente, para o banco de dados da *ImageNet*. Esses modelos foram propostos para, além de superar a taxa de acerto da VGG-16, diminuir a carga computacional utilizada para processamento, pois a VGG-16 conta com mais de 130 milhões de parâmetros e as duas últimas com aproximadamente 25 milhões (KARIM, 2021). Apesar desses modelos apresentarem resultados satisfatórios, muitos autores, a depender do problema da pesquisa, propõem o seu próprio modelo para fins de balanceamento de otimização e taxa de acerto, como o fez Riaz (2020) para detectar emoções de rostos humanos.

2.2.1.2.1 Estruturas de Redes Neurais Convolucionais com Código Aberto

Há autores que não somente propõe a sua topologia de rede neural convolucional, mas também disponibilizam o código desenvolvido com o objetivo de implementações idênticas por outros pesquisadores e aperfeiçoamento da arquitetura. Assim fizeram os autores Ratan (2019), Rai (2020), Sharma (2018), Fabien (2019), Kinli (2018) e Zawieska (2018) que propuseram modelos para o desafio da base de dados FER-2013.

A estrutura desses autores são descritas de foram simplificada na Tabela 1, onde as letras 'C', 'P', 'F' e 'D' representam camadas de convolução, de *Max-Pool*, de achatamento e densas, respectivamente. É também descrito o número de parâmetros totais utilizados e a taxa de acerto de cada arquitetura proposta.

Tabela 1 – Estruturas das Topologias com Código Aberto.

Pesquisa	Estrutura	Parâmetros (M)	Taxa de Acerto (%)
Kinli (2018)	*CCPCCPCCPCCPFDDDD*	5,90	64,1
Zawieska (2018)	*CCPCCPCCPCCCCPFDD*	3,33	67,6
Sharma (2018)	*CCPCCPCCPCCPFDDDD*	5,90	66,4
Ratan (2019)	*CCPCCPCCPCCPFDDD*	1,32	61,0
Fabien (2019)	*CCPCCPCCPCCFDDD*	3,97	61,7
Rai (2020)	*CCPCCPCCPFDD*	1,81	64,7

Fonte: O autor.

2.2.2 Aprendizado de Máquinas em Sistemas Embarcados

Na era de revolução digital que a humanidade está presente, o uso de dispositivos do tipo móvel e do tipo embarcado se tornaram cada vez mais corriqueiros no cotiano. Esse cenário somado à difusão dos estudos das RNAs levou os pesquisadores a estudarem modos de implementar os modelos inteligentes para hardwares limitados (ALIPPI SIMONE DISABATO, 2018). Os estudos nessa área foram intensificados na última década, levando o aprendizado de máquina até mesmo a chips de baixo consumo, gerando um novo campo de estudos, para dispositivos que consomem até 1mW: o Aprendizado de Máquina Minúsculo (do inglês *Tiny Machine Learning*) (WARDEN; SITUNAYAKE, 2021).

Contudo, muitos desafios são enfrentados ao se implementar uma RNA em um sistema embarcado. Devido a questões de consumo de energia e privacidade de dados, executar RNAs da nuvem (em serviços computacionais disponíveis através da internet) são impeditivos de adotar tal técnica para modelos de *hardware* limitado (MOONS; BANKMAN; VERHELST, 2018). Logo, uma solução frequentemente buscada em aplicações em *hardware* limitado é a simplificação do modelo (redução do número de parâmetros) pelos seguintes fatos: menor armazenamento do modelo; independência de realizar processamento na nuvem; rapidez de treinamento; rapidez de execução e menor consumo de energia.

2.2.2.1 Poda de Redes Neurais Artificais

Uma das estratégias mais difundidas e estudadas para superficializar as RNAs é a poda (do inglês *pruning*), que visa reduzir o o tempo de processamento, o espaço exigido para armazenamento do modelo e a energia que deve ser investida para processar a rede (HAN; MAO; DALLY, 2016).

O método de poda consiste em cortar certas ligações de neurônios de camadas específicas e anular pesos de neurônios de pouca relevância que possam não impactar, ou pelo menos pouco impactar, no desempenho do modelo. Dessa forma, o modelo conquista a condição de esparsidade (grande quantidade de zeros) e, por conseguinte, fica mais fácil de ser processado (as condições nulas não são processadas) e de ser comprimido (VENKATESAN; LI, 2017).

A processo de poda pode ser feita para camadas específicas, mas com risco de perda de taxa de acerto ou mesmo condição não ideal de esparsidade para o modelo. Com isso, o processo mais robusto de *pruning* é o de esparsidade constante, durante o treinamento, que poda a rede por inteiro nas ligações mais superficiais (ALVAREZ PULKIT BHUWALKA, 2019).

2.2.2.2 Quantização

Os modelos de aprendizado profundo são capazes de lidar com perdas em precisão numérica durante cálculos intermediários e ainda produzir resultados finais precisos em geral - um subproduto de seu processo de treinamento, no qual as entradas são grandes e com ruído, de modo que os modelos aprendem concentrar nos padrões importantes. Isso significa que, na prática, operar com representações de ponto flutuante de 32 bits é quase sempre mais preciso do que o necessário para inferência. Com isso, estudos demonstram que pode se simplificar a representação para até 8 bits sem perda de desempenho na inferência. Esse processo é chamado de Quantização (WARDEN; SITUNAYAKE, 2021).

A quantização pode ser feita durante o treinamento, mas o tipo de quantização mais robusto é conhecido como quantização de peso pós-treinamento com alcance dinâmico. É quando os pesos são quantizados até 8 bits, mas as camadas de ativação permanecem em ponto flutuante. Esse processo é vantajoso porque reduz o tamanho do arquivo do modelo em até 75% e, em até quatro vezes, a latência da inferência, sem a necessidade de retreinamento da rede e de se definir um alcance de quantização, que pode impactar na taxa de acerto e no tamanho do modelo (SIVAKUMAR JIAN LI, 2019).

2.3 Computadores de Placa Única

Os computadores de placa única são uma tecnologia disruptiva que são suficientemente capazes de executar sistemas operacionais e cargas de trabalhos de computadores convencionais (JOHNSTON *et al.*, 2018). Esse tipo de computador possui um *hardware* de dimensões compactas que permite aplicações de baixo custo e de baixo consumo de energia, permitindo diferentes modelos de implementação fora dos ambientes tradicionais de computadores.

O produto líder de mercado em todos os segmentos de computadores de placa única é o *Raspberry Pi*, possuindo o melhor desempenho computacional em relação aos seus concorrentes e menor preço (JOHNSTON *et al.*, 2018).

2.4 Bases de Dados

Um dos pontos críticos da implementação de uma RNA é a escolha de base de dados que o modelo será treinado, testado e validado. Na área de detecção de imagens há diversas bases de dados, mas a mais difundida é a FER-2013 - detalhes nas próximas subseções.

2.4.1 FER-2013

A FER-2013 foi disponibilizada originalmente em um desafio de detecção de reconhecimento de expressões em um workshop da ICML em 2013 (GOODFELLOW *et al.*, 2013). É a mais difundida base de reconhecimentos de emoções por ser totalmente aberta à comunidade e ser a maior de todas, contando com 35887 imagens faciais (registradas automaticamente já centralizadas) rotuladas em seis emoções e mais uma classe neutra, em preto e branco de tamanho 48 x 48.

A base é considerada desafiadora, pois conta com imagens de pessoas de um grande gama de idades, gêneros e raças em ambientes não controlados - a Figura 7 ilustra as duas primeiras imagem das emoções raiva, nojo, medo, felicidade, neutra, tristeza e surpresa, respectivamente. Além disso, ela não é balanceada - explicitado no estudo estatístico feito na seção 4.1 - e foi constituída sem seguir nenhum critério de projetos de experimentos.

Figura 7 – Amostras da FER-2013.

Fonte: O autor.

2.5 Trabalhos Relacionados

A presente seção é dividida na análise de estudos relacionados à presente pesquisa em duas subseções - uma que aborda a detecção de emoções sem limitação de *hardware* e outra em sistemas embarcados.

Vale ressaltar que foi dado ênfase nos trabalhos que utilizaram a base descrita na seção 2.4 por ser a mais difundida na comunidade, mas há ainda outros trabalhos que utilizaram outras bases menores, como a JAFFE (expressões faciais femininas japonesas) e a Cohn-Kanade (CK) e a RAF-DB (*Real-world Affective Faces Database*). Os autores mencionados nas subseções 2.5.1 e 2.5.2 utilizaram todas as classes das bases de dados, sem mistura de dados entre as bases, variando a topologia de autor para autor - alguns inclusive propuseram novas topologias.

Ainda, há trabalhos que utilizam mais sinais biomédicos (YANG; HAN; MIN, 2019), além da própria imagem do rosto, e técnicas de redes convolucionais em três dimensões para detecção de emoção, mas que não serão abordados por não fazer parte do escopo da pesquisa.

2.5.1 Detecção de Emoções

Os melhores resultados registrados na base de dados FER-2013, utilizando todas as classes do conjunto de dados de teste, na academia variam de 60% a 74% (estado da arte), como detalha a Tabela 2[1]. Vale destacar que os autores Giannopoulos, Isodoros e

[1] Os autores marcados com '*' são estruturas propostas pelos próprios pesquisadores, mas implementadas pelo autor Riaz et al (2020) na base de dados descrita.

Perikos (2017) ainda realizaram teste de sensibilidade de emoções - classificador binário com uma classe neutra e a outra sendo uma das demais emoções com 85,5%.

Tabela 2 – Taxa de acerto em Percentual de Pesquisas Similares com FER-2013.

Pesquisa	Taxa de Acerto (%)
Simonyan e Zisserman (2015)*[1]	71,2
He et al. (2015)*	71,1
Tang (2015)	61,7
Liu et al. (2015)	61,7
Burkert et al. (2016)	68,0
Sang et al. (2017)	71,0
Giannopoulos, Isodoros e Perikos (2017)	63,5
Huang et al. (2018)*	67,5
Kinli (2018)	64,1
Zawieska (2018)	67,6
Sharma (2018)	66,4
Ratan (2019)	61,0
Fabien (2019)	61,7
Shao et al. (2019)	71,1
Meudt, Schwenker e Srinivasan (2019)	69,0
Rai (2020)	64,7
Riaz et al. (2020)	73,5
Jiang et al. (2020)	74,0
Zhou et al. (2020)	67,0

Fonte: O autor.

Os trabalhos que utilizaram a base CK+ tiveram como melhor desempenhos taxa de acertos superiores a 92%, sendo a melhor delas aproximadamente 97%. A Tabela 3 mostra a taxa de acerto por pesquisa para a base CK+, assim como fez a Tabela 2 para a FER-2013.

Tabela 3 – Taxa de acerto em Percentual de Pesquisas Similares com CK+.

Pesquisa	Taxa de Acerto (%)
Simonyan e Zisserman (2015)	94,6
He et al. (2015)	94,0
Burkert et al. (2016)	96,0
Lopes et al. (2017)	92,7
Huang et al. (2018)	92,0
Tautkute et al. (2018)	92,0
Jain et al. (2019)	93,2
Shao et al. (2019)	95,2
Riaz et al. (2020)	96,7

Fonte: O autor.

Já para a base RAF-DB, as melhores taxa de acertos estão entre 74 a 87%, conforme mostra a Tabela 4.

Tabela 4 – Taxa de acerto em Percentual de Pesquisas Similares com RAFDB.

Pesquisa	Taxa de Acerto (%)
Simonyan e Zisserman (2015)	82,3
He et al. (2015)	81,7
Burkert et al. (2016)	76,3
Fan et al. (2018)	76,7
Huang et al. (2018)	76,7
Li et al. (2019)	74,2
Riaz et al. (2020)	86,3
Jiang et al. (2020)	86,7

Fonte: O autor.

2.5.2 Detecção de Emoções em Sistemas Embarcados

Modelos propostos para detecção de emoções projetados para hardware com testes específicos são menos difundidos, pelo fato do método de poda poder impactar na taxa de acerto. Outro fator que dificulta a comparação entre trabalhos são as referências de desempenho adotadas por cada pesquisa (além de utilizarem diferentes hardwares) - algumas pesquisas utilizam tempo de processamento e tamanho da rede, enquanto outras utilizam o hardware necessário para a rede ser implementada, a quantidade de parâmetros da rede ou até mesmo o tempo despendido por época de treinamento.

Sendo assim, foi adotado os seguintes critérios para a comparação de trabalhos na literatura com implementações em sistemas embarcados: quantidade de parâmetros, tempo de classificação de uma emoção desde após uma imagem ser reconhecida pela câmera até a resposta final do sistema e taxa de acerto. O *hardware* utilizado pelos autores nas pesquisas selecionadas foi um *Raspberry Pi* (modelos variados) - quando há o parâmetro de tempo na tabela - e a base de dados utilizada como referência para a taxa de acerto é a FER-2013.

A Tabela 5 explicita essa comparação, sendo as pesquisas no estado da arte a de Riaz et al (2020) e Zhou et al (2020), pelo fato de apresentarem resultados que melhor aliam o menor número de parâmetros e taxa de acerto.

Tabela 5 – Desempenho de Pesquisas Similares com *Hardware* Limitado.

Pesquisa	Parâmetros(M)	Tempo(s)	T. de Acerto(%)
Simonyan e Zisserman (2015)*	14,7	3,92	71,2
He et al. (2015)*	11,1	3,86	71,1
Liu et al. (2015)	84,0	-	61,7
Tang (2015)	7,17	-	61,7
Burkert et al. (2016)	3,54	-	68,0
Sang et al. (2017)	4,92	-	71,0
Huang et al. (2018)*	3,00	2,09	67,5
Kinli (2018)	5,90	-	64,1
Zawieska (2018)	3,33	-	67,6
Sharma (2018)	5,90	-	66,4
Ratan (2019)	1,32	-	61,0
Fabien (2019)	3,97	-	61,7
Meudt, Schwenker e Srinivasan (2019)	4,31	0,222	69,0
Shao et al. (2019)	7,12	-	71,1
Rai (2020)	1,81	-	64,7
Riaz et al. (2020)	4,57	0,998	74,0
Jiang et al. (2020)	17,6	-	74,0
Zhou et al. (2020)	0,0584	-	67,0

Fonte: O autor.

Pode-se perceber que o tempo de latência tende a diminuir com a diminuição dos parâmetros - os autores não utilizaram quantização. Vale ressaltar que a pesquisa de Riaz et al(2020) mensurou o tempo de latência acrescido com o tempo de captura da câmera - isso pode ser um indicativo dos tempos de latência mais elevados dessa pesquisa.

É razoável inferir que aquelas pesquisas que não mediram o tempo de latência, possam atingir resultados com tempo de latência menores.

3 Metodologia

A solução proposta para o problema de detecção e classificação de emoções é o desenvolvimento de uma rede neural convolucional, utilizando as topologias já exploradas na academia, reduzida por poda computacional, implementável em hardware limitado.

3.1 Materiais e Ferramentas

3.1.1 Ferramentas Computacionais

Foi utilizada a plataforma *Google Colaboratory* que permite que seja escrito e executado códigos em *Python* versão 3.7 no navegador de internet. O serviço proposto pela *Google* foi escolhido pois não necessita de configuração, possui acesso irrestrito e gratuito a GPU's da empresa e, ainda, possui esquema de compartilhamento robusto de manipulação de arquivos. Para a manipulação das imagens utilizadas durante a pesquisa foi utilizada a biblioteca *Open CV*. Já para a manipulação, análise e visualização de dados foram utilizadas as bibliotecas *Numpy*, *Pandas* e *Matplotlib*, respectivamente. Por fim, a implementação da rede foi feita utilizando a plataforma *TensorFlow* versão 2.3 com uso das bibliotecas de alto nível *Keras* e *Scikit-learn*.

3.1.2 *Hardware* Utilizado

Durante os testes na plataforma *Google Colaboratory*, o usuário não escolhe qual *hardware* está utilizando, mas pode consultá-lo. Assim sendo, durante os testes, foi utilizado o seguinte *hardware*:

- *CPU Intel Xeon 2.2 GHz*;
- Memória *RAM* de 12 GB;
- *GPU Nvidia K80s* de 24 GB.

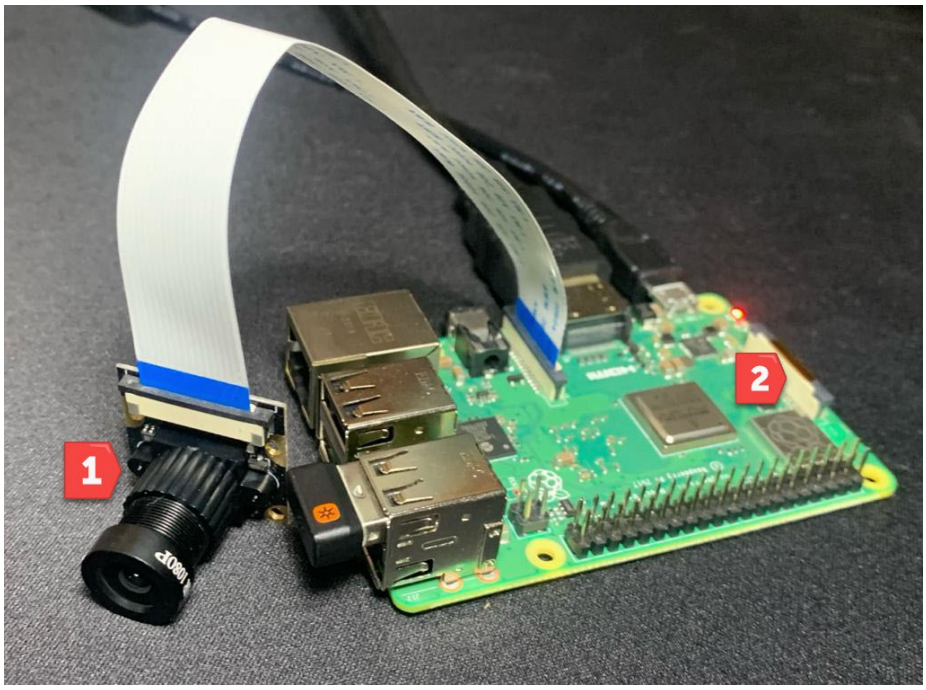
Por sua vez, o computador de placa única utilizado foi o *Raspberry Pi* 3 Modelo B. Os principais atributos que esse modelo conta são:

- *CPU Quad Core* 1.2 GHz *Broadcom BCM2837 64bit*s;
- Memória *RAM* de 1 GB;
- Interface Serial de Câmera (CSI) para conexão de câmera do tipo *Raspberry*.

Já a câmera que foi utilizada na pesquisa é a câmera compatível *Raspberry Pi* de 5 MP. Esse equipamento possui ajuste de foco automático, resolução de 1080 p, abertura focal de 1,8, comprimento focal ajustável de 3,6 mm e dimensões de 25 x 24 x 23,5 mm.

A Figura 8 ilustra destacado em '1' a câmera utilizada e em '2' o computador de placa única, compondo o sistema de *hardware* limitado utilizado na pesquisa.

Figura 8 – Sistema Utilizando Computador de Placa Única e a Câmera.

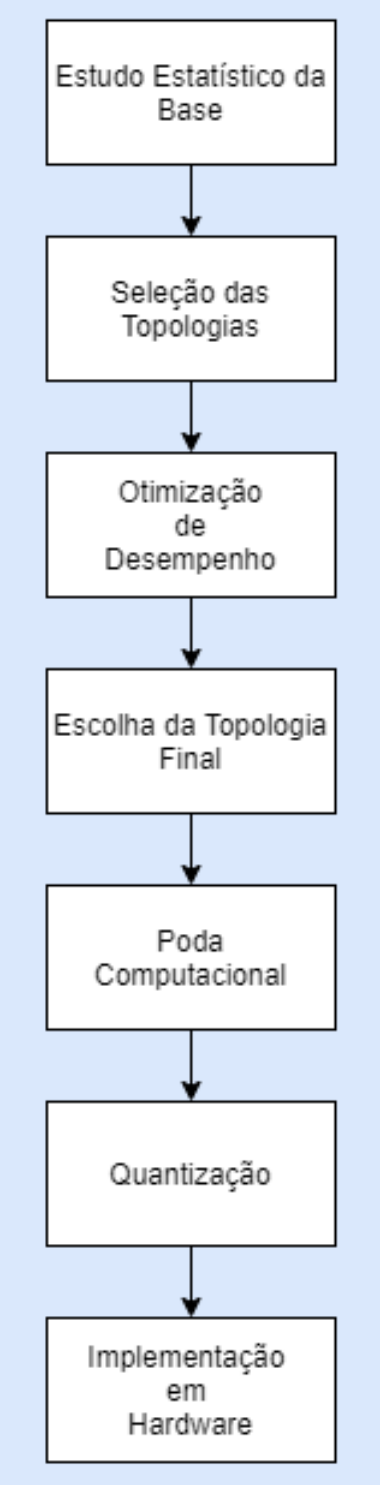


Fonte: O autor.

3.2 Procedimento

O procedimento geral da metodologia da pesquisa pode ser visto na Figura 9 que ilustra-o em um fluxograma.

Figura 9 – Procedimento Geral da Metodologia.



Fonte: O autor.

3.2.1 Estudo e Uso da Base de Dados

O treinamento, validação e teste da rede foram feitos utilizando a base de dados FER-2013. Para isso, foi feito um estudo estatístico, assim como o particionamento da

base em dados de treinamento, validação (utilizando validação cruzada) e teste sendo de 80%, 10% e 10%, respectivamente.

3.2.2 Pré-Processamento da Base de Dados

O pré-processamento da base foi feito nas colunas das imagens e do identificador de emoções. Os seguintes passos foram tomados na coluna das imagens:

- Transformação de todos os *pixels* das imagens em valores inteiros (0 a 255);
- Rearranjo de cada sequência de *pixels* para a altura e largura das imagens indicada pela base de dados - no caso da FER-2013, por exemplo, 48 x 48;
- Transformação da coluna global das imagens e de cada imagem em formato *numpy array*;
- Redimensionamento das imagens adicionado um valor unitário (escala de cinza), ficando assim 48 x 48 x 1, para a FER-2013 por exemplo;
- Normalização das imagens de 0 a 255 *pixels* para 0 a 1, deixando-as em ponto flutuante.

Para a coluna do identificador das emoções foi convertido cada valor categórico em variáveis identificadoras.

3.2.3 Escolha de Hiperparâmetros

Na fase de treinamento, validação e teste de cada topologia, foram utilizados os hiperparâmetros estruturais definidos pelos autores das estruturas. O otimizador e a função de perda, por outro lado, foram escolhidos e utilizados em todos os testes pelo autor da monografia, assim como o número de 500 épocas. A função de perda escolhida foi a entropia cruzada categórica (do inglês *Cross Entropy*), por se tratar de um classificador com mais de duas classes categóricas, e o otimizador escolhido foi o Adam, com valores iniciais recomendados pelos autores do algoritmo, pode ser considerado o otimizador mais rápido e efetivo (KINGMA; BA, 2017).

3.2.4 Projeto de Experimentos

A análise estatística feita nas subseções 3.2.5 a 3.2.10 foi feita através da técnica de projetos de experimentos para analise do impacto dos fatores estudados, com o último passo da técnica sendo o teste de ANOVA. Nas subseções 3.2.5 e 3.2.6 foi utilizado um projeto fatorial completo com dois fatores (modelo e *Data Augumentation*) e os testes das

subseções 3.2.8 a 3.2.10 foi utilizado projeto com um único fator (poda computacional para o teste da subseção 3.2.8 e tipo do modelo para as demais). O número de repetições para cada projeto de experimentos e outras características de cada projeto são detalhados nas referidas subseções.

Vale ressaltar que cada projeto de experimentos foi feito de forma independente dos demais. Assim, não foram testadas combinações dos níveis dos fatores modelo, *data augumentation*, *pruning* e quantização em um mesmo experimento. Nesse sentido, apesar do primeiro projeto de experimentos ser fatorial completo com 2 fatores, o conjunto das decisões subsequentes consideram um fator por vez (*one factor at a time*).

Importante, também, salientar que todos os testes foram feitos sem semente computacional fixa, ou seja, todos os testes computacionais foram randomizados para cumprir a aleatorização na ordem dos ensaios, que é um dos procedimentos importantes para garantir a validade dos cálculos estatísticos de um experimento computacional (SANTNER; WILLIAMS; NOTZ, 2019).

3.2.5 Seleção das Topologias

Foram selecionadas as topologias de pesquisas com código aberto, com taxa de acerto acima de 60% para a base de dados FER-2013 e com número de parâmetros abaixo de dez milhões. Com base nesses critérios, foram selecionadas as estruturas dos seguintes autores: Ratan (2019), Rai (2020), Sharma (2018), Fabien (2019), Kinli (2018) e Zawieska (2018).

Após, as topologias selecionadas foram implementadas e testadas com e sem a técnica de *Data Augmentation*, conforme descrito na subseção 3.2.6, em 20 repetições para mensurar a taxa de acerto - dez com *Data Augmentation* e dez sem *Data Augmentation*.

3.2.6 Técnicas de Otimização de Desempenho

Foram utilizadas as técnicas de *Data Augmentation* da classe *ImageDataGenerator* da biblioteca *Keras* para aumento das bases de dados e maior diversidade das imagens, conforme mostra a Tabela 6:

Essas operações de aumento de dados são feitas na memória, então as imagens geradas são descartadas logo em após sua utilização, sendo o número de imagens que utilizam essas técnicas acordado pelo *batch size*. É retornado apenas os dados transformados aleatoriamente. Ou seja, as imagens originais não serão adicionadas, ou somadas junto às imagens geradas por transformações.

Por fim, foi feito um teste de hipótese ANOVA para verificar se a técnica de *Data Augmentation* impactou significativamente no resultado de taxa de acerto do modelo.

Tabela 6 – Métodos Utilizados da Classe *ImageDataGenerator*

Método e Valor Atribuído	Descrição
rotation_range=30	Rotação aleatório de 0 a 30°
shear_range=0.1	Distorção aleatório por cisalhamento
zoom_range=0.3	*Zoom* aleatório de até 30%
width_shift_range=0.1	Alternância aleatório horizontal até 10%
height_shift_range=0.1	Alternância aleatório vertical até 10%
horizontal_flip=True	Inversão horizontal

Fonte: O autor.

3.2.7 Escolha da Topologia Final

A topologia que apresentou a melhor taxa de acerto nos testes da subseção 3.2.5 (testes realizados para escolha da melhor topologia sem considerar restrições de *hardware*) com técnica de otimização foi a escolhida como a estrutura final da pesquisa. Após essa escolha, foi feita a poda computacional nessa estrutura, a fim de reduzir os parâmetros utilizados - explicitado na subseção 3.2.8. Por fim, foi realizada a quantização do modelo, conforme detalha a subseção 3.2.9, para diminuir o tempo de latência e o tamanho do modelo.

3.2.8 Técnica de Poda Computacional para Redução de Parâmetros

Foi utilizado a técnica de poda da rede através do módulo de esparsidade do *Keras - tfmot.sparsity.keras.ConstantSparsity* que possui uma agenda de poda constante no treinamento da rede em todas as épocas do treino.

Foram feitos os testes com esse módulo com esparsidades de 0%, 40%, 60%, 70%, 80%, 80%, 90% e 95% para escolha da mais alta esparsidade com a mais alta taxa de acerto da rede, sendo realizadas quatro inferências para cada nível de esparsidade. Após, foi feito estudo estatístico dos resultados utilizando o teste ANOVA.

3.2.9 Técnica de Quantização

O modelo foi quantizado através do método de quantização pós-treino com alcance dinâmico (do inglês, *Dynamic range quantization*). É a forma mais robusta de quantização que estaticamente simplifica apenas os pesos do ponto flutuante para inteiro, possuindo 8 bits de precisão. Esse método diminui em até quatro vezes o tamanho do modelo e possui redução de latência de duas a três vezes na inferência final do modelo.

Após feita a quantização do modelo, foi realizado quatro inferências de taxa de acerto e de tamanho no modelo original, no modelo com poda computacional e no modelo quantizado. Vale ressaltar que também foi feito o estudo estatístico dos testes com teste ANOVA para verificar o impacto na taxa de acerto e no tamanho do modelo.

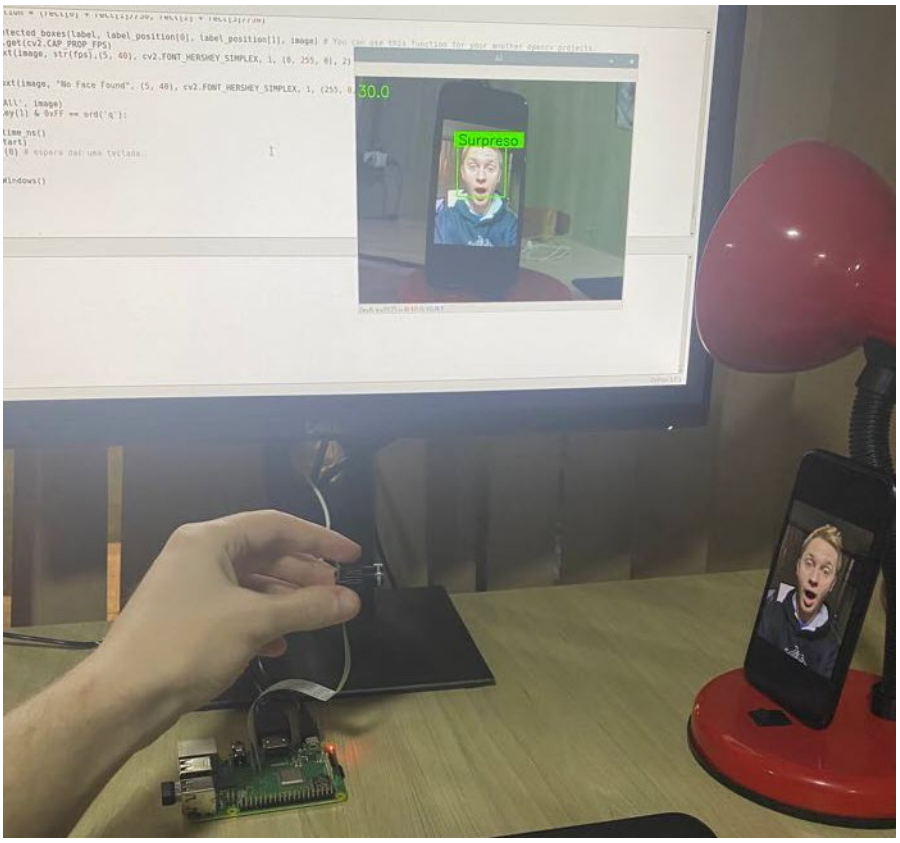
3.2.10 Análise de Desempenho do Modelo

A análise do modelo final foi feito seguindo a métrica da taxa de acerto, número de parâmetros, tamanho em disco e tempo de latência (tempo de processamento para a detecção de uma emoção com e sem *hardware* limitado).

Vale salientar que nos testes sem *hardware* limitado, para o modelo original e podado foi utilizado a função integrada no *Python %timeit*, enquanto para o modelo quantizado foi utilizado a *bechmark tool* disponibilizada pelo *tensorflow* por se tratar de um modelo de extensão *.tflite*. Ambas as ferramentas foram usadas no laço de inferência da base de testes, então é avaliado tempo mais rápido e mais demorado de inferência, assim como a média temporal.

Já para mensurar a latência do modelo desenvolvido para ser utilizado *hardware* limitado, foram feitas dez inferências para cada topologia do modelo para uma mesma imagem mostrada à câmera e feito uma média temporal - Figura 10. Foi utilizado a função *time.time* logo após a captura da imagem.

Figura 10 – Método Utilizado para Mensurar a Latência em *Hardware*.



Fonte: O autor.

Ainda, foi feito uma inferência final do modelo para construção de uma matriz de confusão para verificar a taxa de acerto de cada classe.

3.2.10.1 Comparações com Trabalhos da Literatura

As principais métricas de comparação com pesquisas similares foram a taxa de taxa de acerto e o número de parâmetros utilizados pela rede. Conforme abordado na revisão de literatura, pesquisas envolvendo hardware limitado não possuem métricas de desempenho comum entre elas, como ocorre com as sem restrições de hardware com taxa de acerto, por exemplo. Todavia, foi discutido, quando os autores explicitam em seus trabalhos, o tempo de latência do *hardware*, a fim de manter uma comparação justa entre as pesquisas citadas e a presente monografia.

3.2.10.2 Análise do Modelo Desenvolvido para Uso em *Hardware* Limitado

Foi analisado o desempenho do modelo no hardware limitado em três testes de latência: modelo original, modelo com poda computacional e modelo quantizado. Foi feito uma média temporal com dez inferências em uma aplicação de detecção em tempo real *open source* do Open CV (DUTTA, 2021).

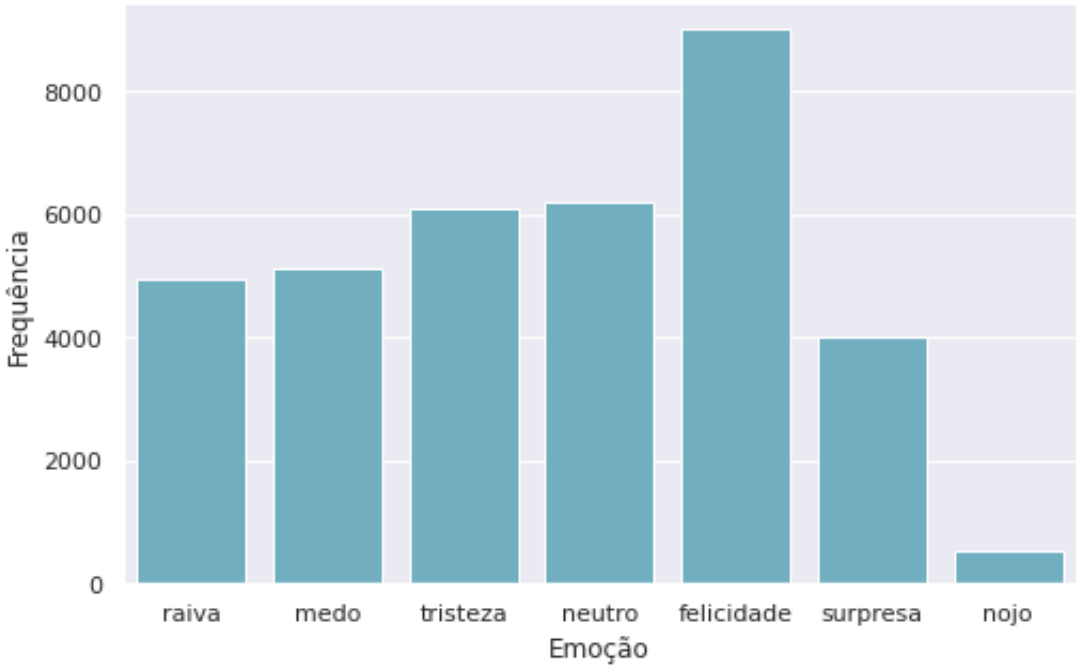
4 Análise de Resultados

Após a coleta de dados e testes discriminados no capítulo três (sobre a metodologia), foi feita a análise dos resultados, conforme segue nas seções.

4.1 Estudo Estatístico da Base de Dados

A base de dados FER-2013 possui 35887 imagens rotuladas em 7 conjuntos de emoções: felicidade (8989); neutro (6198); tristeza (6077); medo (5121); raiva (4953); surpresa (4002) e nojo (547). A frequência de cada classe é explicitada no gráfico da Figura 11 que ilustra-o em um fluxograma.

Figura 11 – Frequência de Classes da FER-2013.



Fonte: O autor.

Pode-se observar que a classe da felicidade é a mais frequente na base de dados e a classe do nojo é a menos frequente.

Vale ressaltar ainda que os autores dividiram a base em três conjuntos: treinamento (80% ou 28709), validação (10% ou 3589) e teste (10% ou 3589) com o intuito de melhor comparar os resultados da comunidade.

4.2 Resultados dos Testes da Seleção de Topologias

As 120 observações - 20 para cada topologia - podem ser visualizadas na Tabela 7 que demonstra a taxa de acerto média de cada modelo no conjunto de teste com seu respectivo desvio padrão. Nota-se que o modelo de Ratan (2019) obteve a melhor performance geral, seguido pelo modelo de Zawieska (2018) - ambos utilizando *data augmentation*.

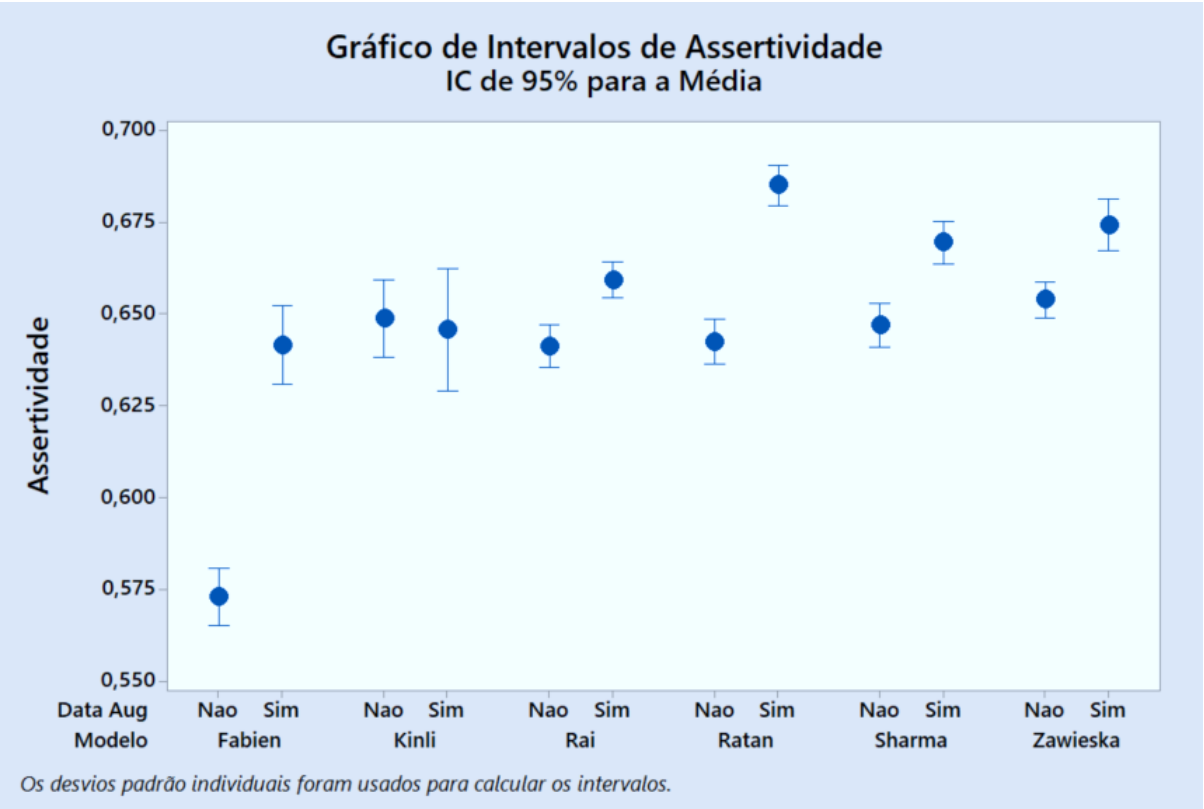
Tabela 7 – Estatística Descritiva dos Resultados dos Modelos.

Pesquisa	*Data Augmentation*	Taxa de Acerto Média (%)	Desvio Padrão
Fabien (2019)	Não	57,3	0,011
	Sim	64,1	0,015
Kinli (2018)	Não	64,9	0,015
	Sim	64,6	0,023
Rai (2020)	Não	64,1	0,0081
	Sim	65,9	0,0069
Ratan (2019)	Não	64,2	0,0085
	Sim	68,5	0,0078
Sharma (2018)	Não	64,7	0,0083
	Sim	66,9	0,0079
Zawieska (2018)	Não	65,4	0,0068
	Sim	67,4	0,0097

Fonte: O autor.

O gráfico de intervalos (utilizando os desvios padrões individuais) da taxa de acerto da Figura 12 evidencia os resultados exposto na Figura 12.

Figura 12 – Gráfico de Intervalos de taxa de acerto para os Modelos.



Fonte: O autor.

Para verificar o impacto de cada modelo na taxa de acerto[1], assim como o impacto do método de *Data Augmentation*, foi realizado uma ANOVA com dois fatores controláveis - modelo (seis níveis) e a presença do método de *Data Augmentation* (dois níveis). Os resultados são evidenciados na Tabela 8.

Tabela 8 – Resultado ANOVA.

Fonte	GL	SQ(Aj.)	QM(Aj.)	Valor F	Valor P
Data Augmentation	1	0,02381	0,023805	174,26	0,000
Modelo	5	0,04551	0,009102	66,63	0,000
*Data Augmentation**Modelo	5	0,01507	0,003015	22,07	0,000
Erro	108	0,01475	0,000137		
Total	119	0,09914			

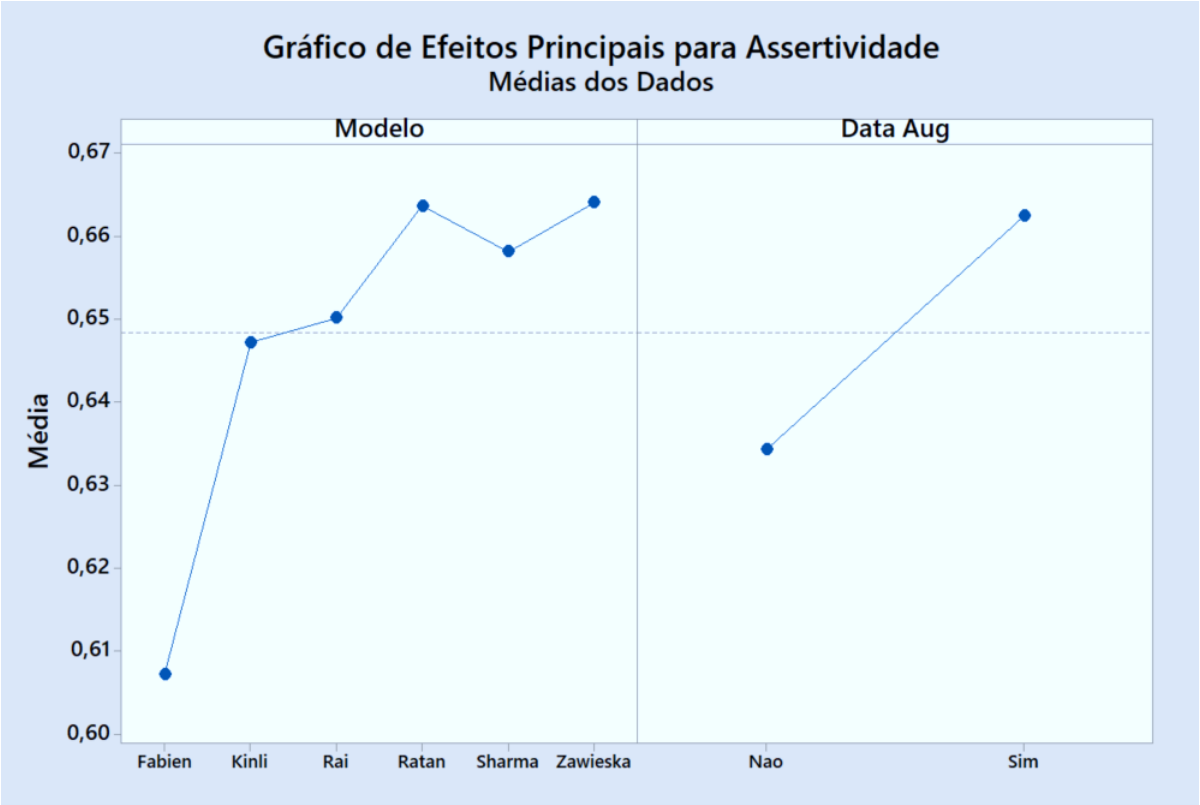
Fonte: O autor.

Pode-se analisar que tanto o modelo (arquitetura proposta por cada pesquisador) quanto o *Data Augmentation* tiveram impacto significativo na taxa de acerto, assim como a interação entre os dois fatores, já que os valores P dos fatores do teste ANOVA tiveram valor menor que 5% - indicando que que a diferença é estatisticamente significativa. Tal resultado fica mais evidente quando é observado o gráfico de efeitos principais para taxa de

[1] A avaliação de taxa de acerto por classe se demonstra relevante, pois a diferença de taxa de acerto média para cada topologia pode ter sido influenciada pelo desbalanceamento da base de dados utilizada.

acerto, Figura 13, principalmente pelo efeito do método de *Data Augmentation* na média da taxa de acerto.

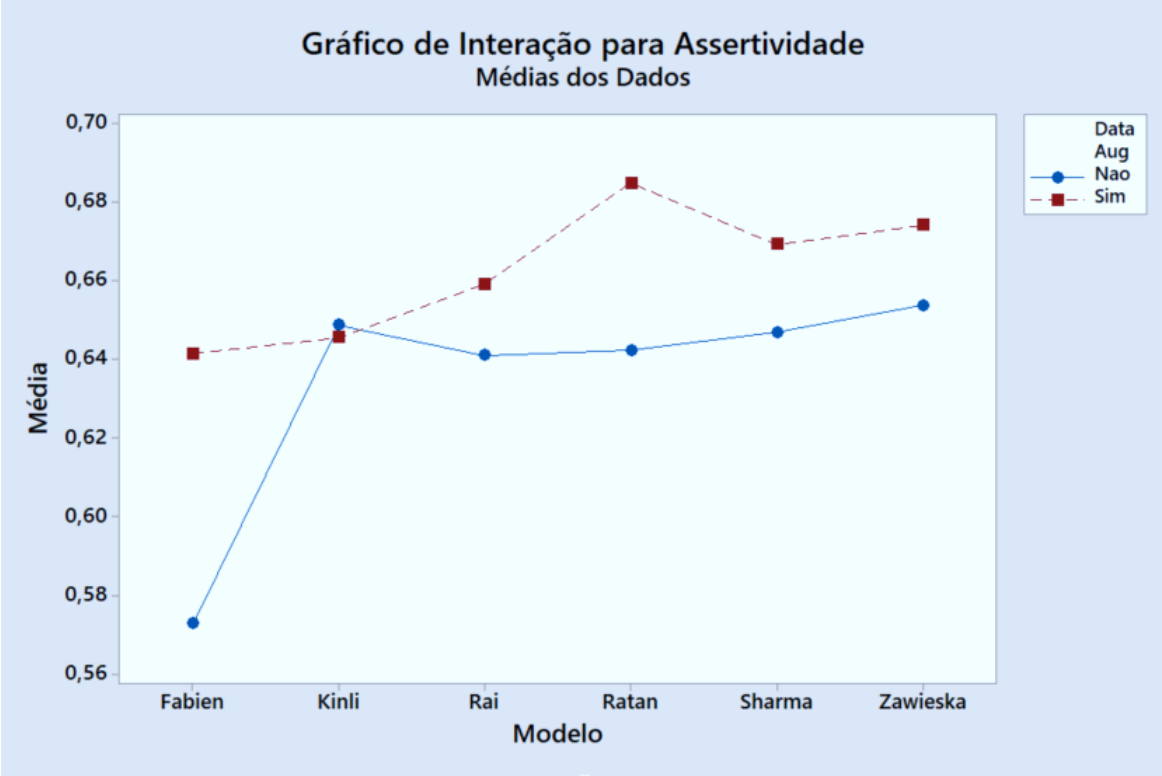
Figura 13 – Gráfico de Efeitos Principais.



Fonte: O autor.

O gráfico de interação entre fatores é mostrado na Figura 14. Analisando os resultados, pode-se inferir que o método de *Data Augmentation* aumenta a média de taxa de acerto, com exceção do modelo de Kinli e que o modelo de Ratan obteve a maior média de taxa de acerto geral, utilizando do método de *Data Augmentation*.

Figura 14 – Interação entre Fatores.



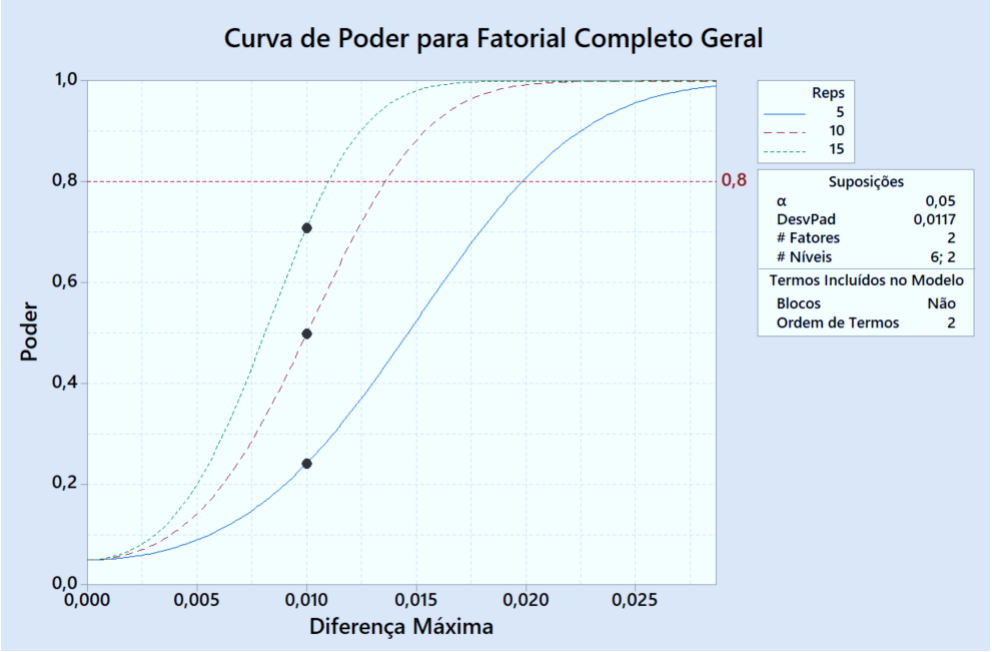
Fonte: O autor.

4.2.1 Potência Estatística do Teste

Valendo-se dos resultados do modelo linear generalizado proposto para o teste de ANOVA com dois fatores, pode-se criar a curva de potência estatística do teste.

Foi comparado as curvas com cinco, dez (utilizada no teste) e quinze repetições para cada modelo com e sem *Data Augmentation*. A Figura 15 mostra que para dez repetições a potência estatística é superior a 88% para uma diferença de 1,5% de taxa de acerto.

Figura 15 – Curva de Potência Estatística do Teste.

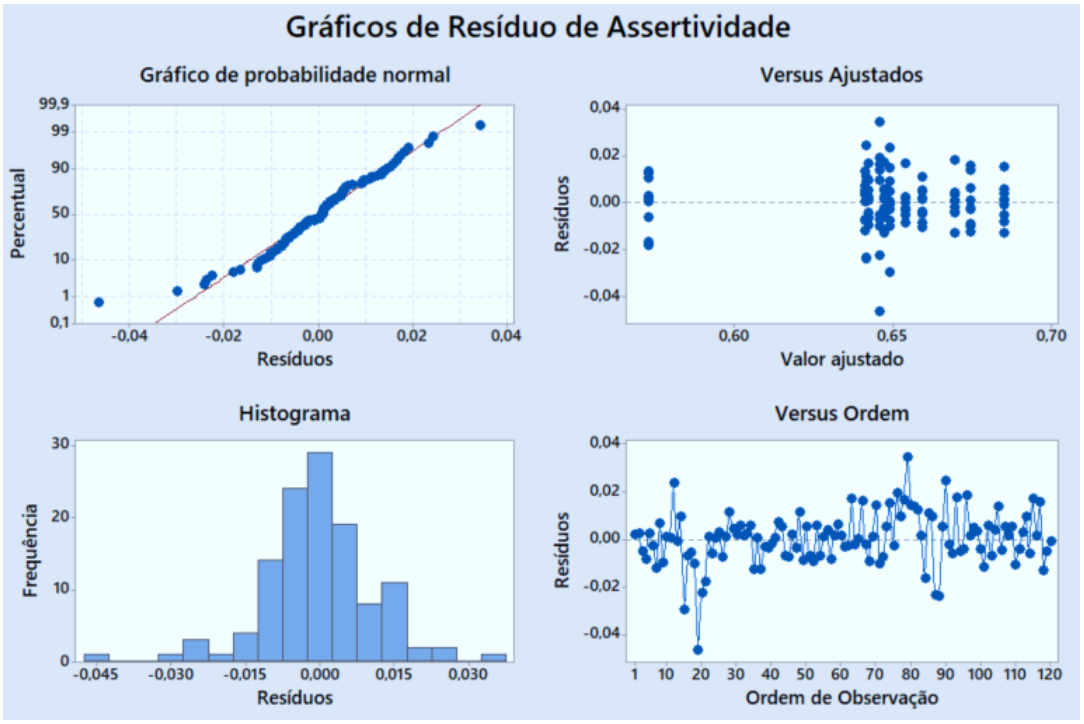


Fonte: O autor.

4.2.2 Análise de Resíduos do Teste de ANOVA e Teste de Normalidade

O gráfico de resíduos de taxa de acerto pode ser visto na Figura 16, onde é apresentado o gráfico de probabilidade normal, de versus ajustados, de histograma e de versus ordem.

Figura 16 – Gráfico de Resíduos.



Fonte: O autor.

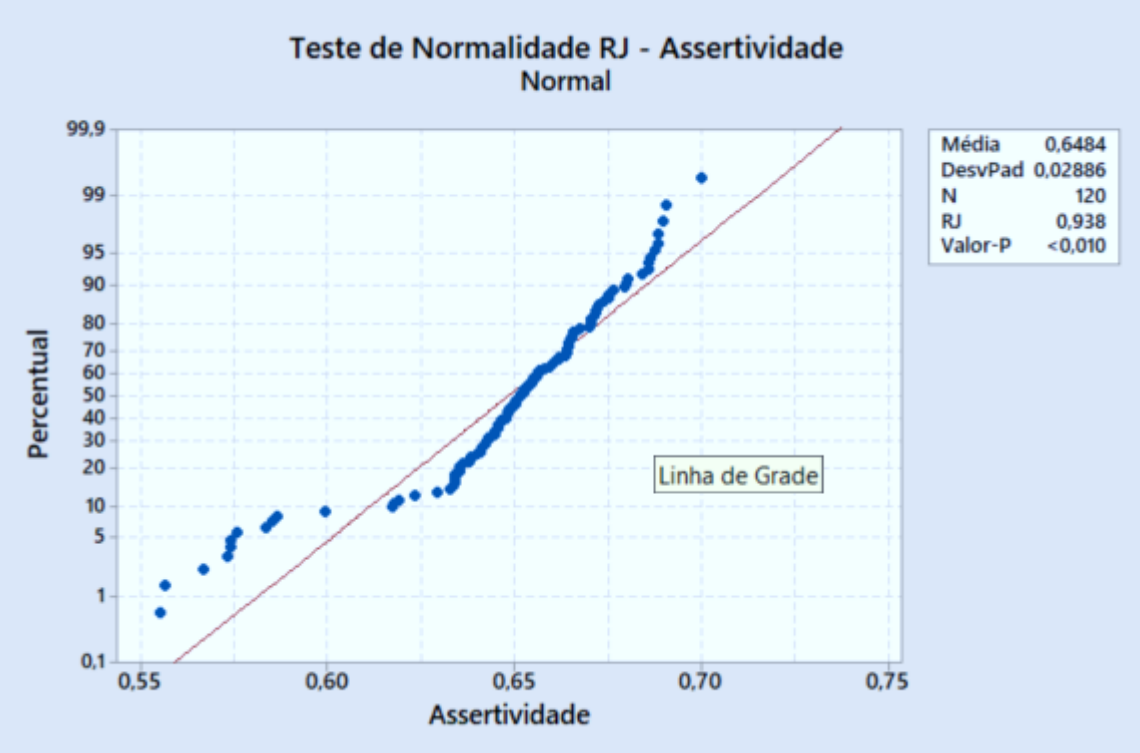
Através do histograma, pode ser visto que a distribuição dos resíduos segue uma forma similar a distribuição normal. O gráfico de probabilidade normal, por sua vez, tende a seguir uma linha reta, no entanto, apresenta indício de violação de pressuposto da tendência normal com tendência de curva "S".

Por sua vez, o gráfico de versus ajustados mostra que os pontos possuem variância constante, com exceção de região próxima do valor ajustado 0,65, onde é possível observar uma dispersão maior das amostras. No entanto, em casos com mesmo número de amostrar por tratamento em um modelo de efeito fixo, a violação na premissa das variâncias iguais tem impacto reduzido no teste F (MONTGOMERY, 2012). Para averiguar em maior detalhe, o caminho ideal seria realizar um teste estatístico de igualdade de variância, o que não foi realizado neste trabalho.

Ainda na Figura 16, o gráfico de versus ordem mostra que que os resíduos são independentes uns dos outros, pois não há nenhuma evidência no gráfico de tendência nem padrão nessa exibição temporal.

Por fim, como o histograma e o gráfico de probabilidade normal não foram conclusivos, foi feito um teste de normalidade de Ryan-Joiner, mostrado na Figura 17, no conjunto de dados da taxa de acerto.

Figura 17 – Teste de Normalidade de Ryan-Joiner.



Fonte: O autor.

Apesar do coeficiente de correlação RJ estar tendendo ao valor unitário, o coeficiente do teste foi menor que o valor crítico do teste, então a hipótese de normalidade da população

foi rejeitada (pressuposto do teste de ANOVA). Entretanto, o teste ANOVA funciona mesmo quando a pressuposição de normalidade é violada, já que a distribuição não é altamente assimétrica e que o tratamento dos testes seguiram o padrão de mesmo número de elementos/rodadas, não se caracterizando como um problema crítico (MINITAB, 2021).

4.2.3 Comparação Múltipla

Foi feito o teste de comparação múltipla para analisar a diferença entre as médias das arquiteturas. Para isso, foi utilizado o método de comparação múltipla com um controle - sendo esse a arquitetura com maior média de taxa de acerto com método de *Data Augmentation* - de Bonferroni com nível de confiança de 95%. O resultado é exibido na Tabela 9.

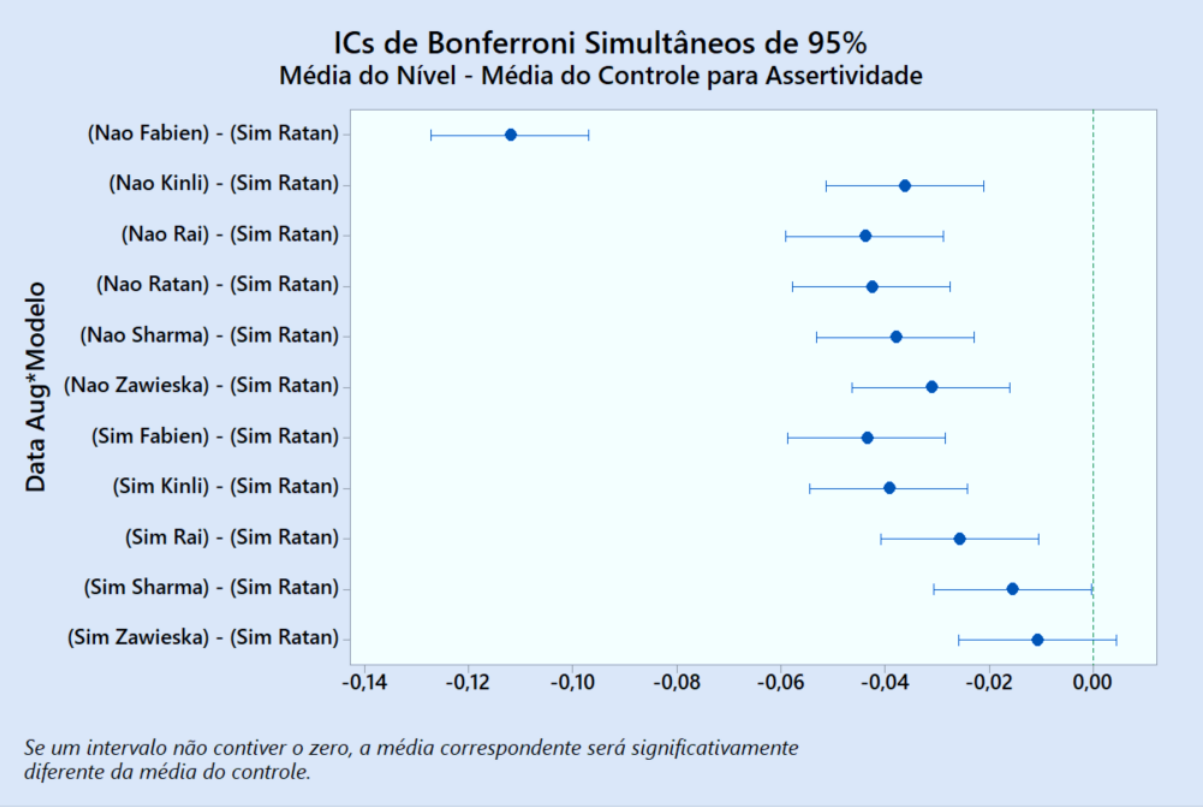
Tabela 9 – Resultado do Teste de Bonferroni com um Controle.

Data Augmentation	Modelo	N	Média (%)	Agrupamento
Sim	Ratan (Controle)	10	68,5	A
Sim	Zawieska	10	67,4	A
Sim	Sharma	10	66,9	
Sim	Rai	10	65,9	
Não	Zawieska	10	65,4	
Não	Kinli	10	64,9	
Não	Sharma	10	64,7	
Sim	Kinli	10	64,6	
Não	Ratan	10	64,2	
Sim	Fabien	10	64,1	
Não	Rai	10	64,1	
Não	Fabien	10	57,3	

Fonte: O autor.

O resultado mostra que há dois modelos com agrupamento A - os dois utilizando *Data Augmentation*. Isso quer dizer que eles não significativamente diferentes entre si é há de fato dois melhores modelos levando em conta a taxa de acerto. A Figura 18 explicita o resultado com uma linha na referência que corta as médias que não são significativamente diferentes.

Figura 18 – Resultado do Teste de Bonferroni com um Controle.



Fonte: O autor.

Com base nesse resultado, foi decidido que o modelo de Ratan e Zawieska seguiram como escolha das melhores arquiteturas - ambas utilizando o método de *Data Augmentation* - para a poda computacional.

4.3 Escolha da Topologia Final

Os modelos escolhidos dos autores Ratan e Zawieska possuem 1,32 e 3,33 milhões de parâmetros respectivamente. Conforme descrito na subseção 3.2.7, foi feito a poda computacional no melhor modelo - no caso nos dois melhores modelos, visto que há dois modelos estaticamente empatados - para redução desses parâmetros, com resultados explicitados na subseção 4.3.1.

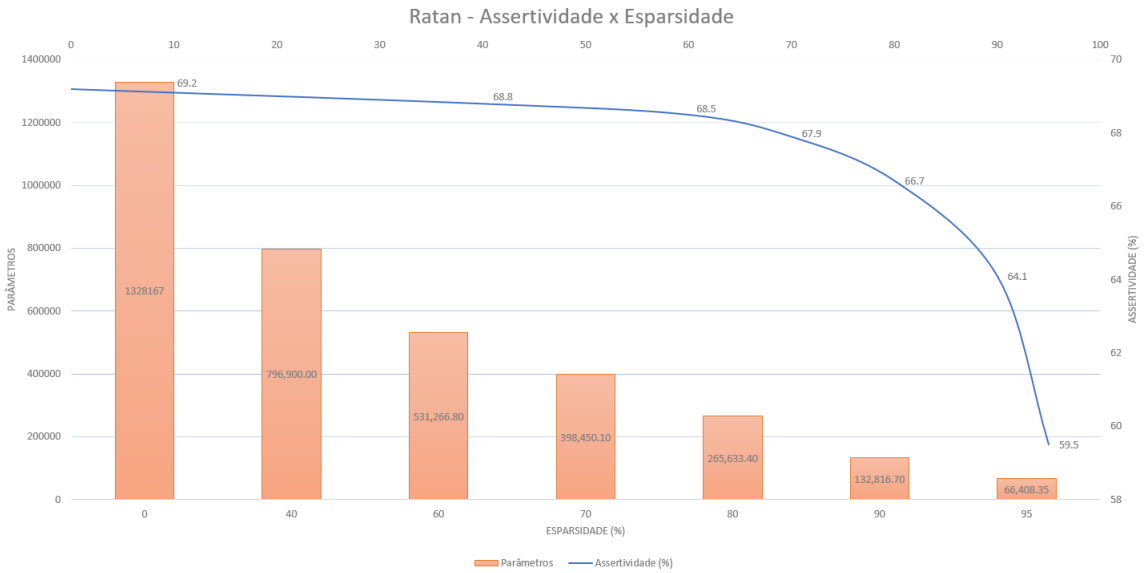
Com base nessa análise, foi decidido que o melhor modelo é aquele que apresenta a maior taxa de acerto com o menor número de parâmetros, dentro da curva de esparsidade escolhida.

4.3.1 Poda Computacional nas Topologias Escolhidas

Os resultados da poda computacional nos modelos de Ratan e Zawieska com 0, 40, 60, 70, 80, 90 e 95% são ilustrados nas Figuras 19 e 20. Vale ressaltar que os resultados

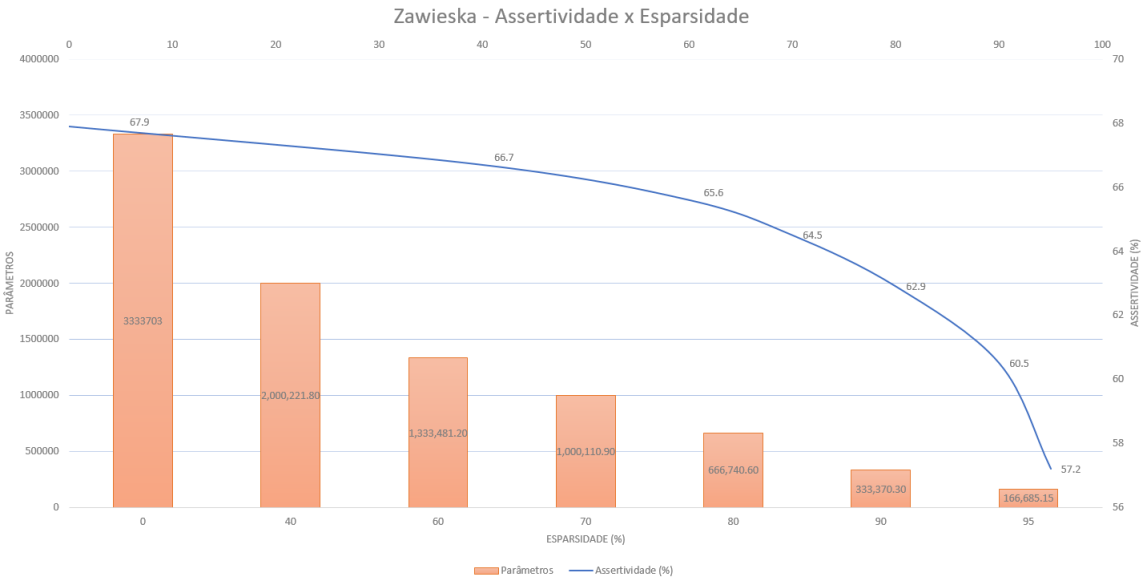
nos gráficos se tratam da mediana das quatro inferências de taxa de acerto a cada nível de esparsidade e que o número de parâmetros foi estimado através de uma proporção simples.

Figura 19 – Ratan: taxa de acerto e Número de Parâmetros x Esparsidade.



Fonte: O autor.

Figura 20 – Zawieska: taxa de acerto e Número de Parâmetros x Esparsidade.



Fonte: O autor.

Pode-se analisar que a curva de taxa de acerto tem um comportamento descendente começando a ter uma queda mais abrupta a partir de valores superiores a 60% de esparsidade.

Comparando as duas curvas pelas referências de taxa de acerto e número de parâmetros da topologia, percebe-se que o modelo de Ratan obteve a melhor resposta nos

dois quesitos, apresentando taxa de acerto superior e menor número de parâmetros em todos os níveis de esparsidade.

Com isso, o modelo de Ratan foi escolhido como a melhor topologia.

4.3.1.1 Nível de Esparsidade Escolhido

Foi realizado um teste de ANOVA com um fator e sete níveis, tendo como preditor a esparsidade e a resposta a taxa de acerto, para analisar o impacto da poda computacional na resposta da topologia. A resposta do teste é ilustrado na Tabela 10.

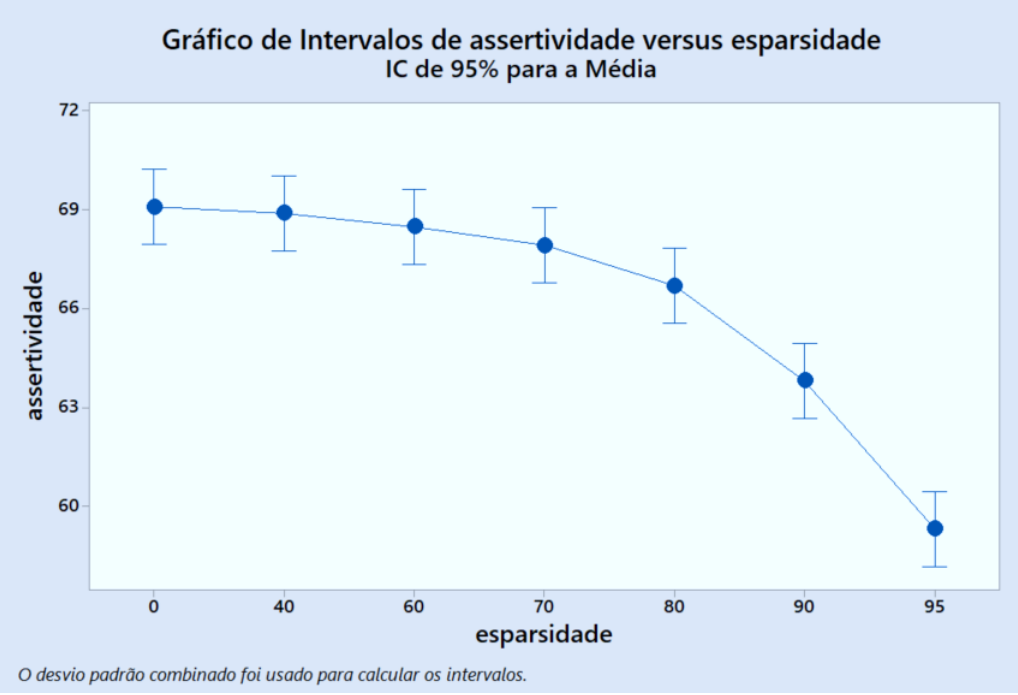
Tabela 10 – ANOVA com Esparsidade como Fator e Taxa de Acerto como Resposta.

Fonte	GL	SQ(Aj.)	QM(Aj.)	Valor F	Valor P
Esparsidade	6	308,05	51,342	42,99	0,000
Erro	21	25,08	1,194		
Total	27	333,13			

Fonte: O autor.

Como esperado, a esparsidade teve um impacto significativo na reposta de taxa de acerto, já que é apresenta uma curva descendente abrupta a partir de esparsidades superiores a 60%. Esse comportamento foi discutido na subseção 4.3.1, sendo reforçado na Figura 21 com resultados levando em conta os intervalos (desvio padrão) de cada nível.

Figura 21 – Ratan: taxa de acerto x Esparsidade.



Fonte: O autor.

Para a determinação do nível de esparsidade ideal, foi realizado o mesmo teste pós hipotese nula da subseção 4.2.3 - de comparação múltipla, utilizando o método de

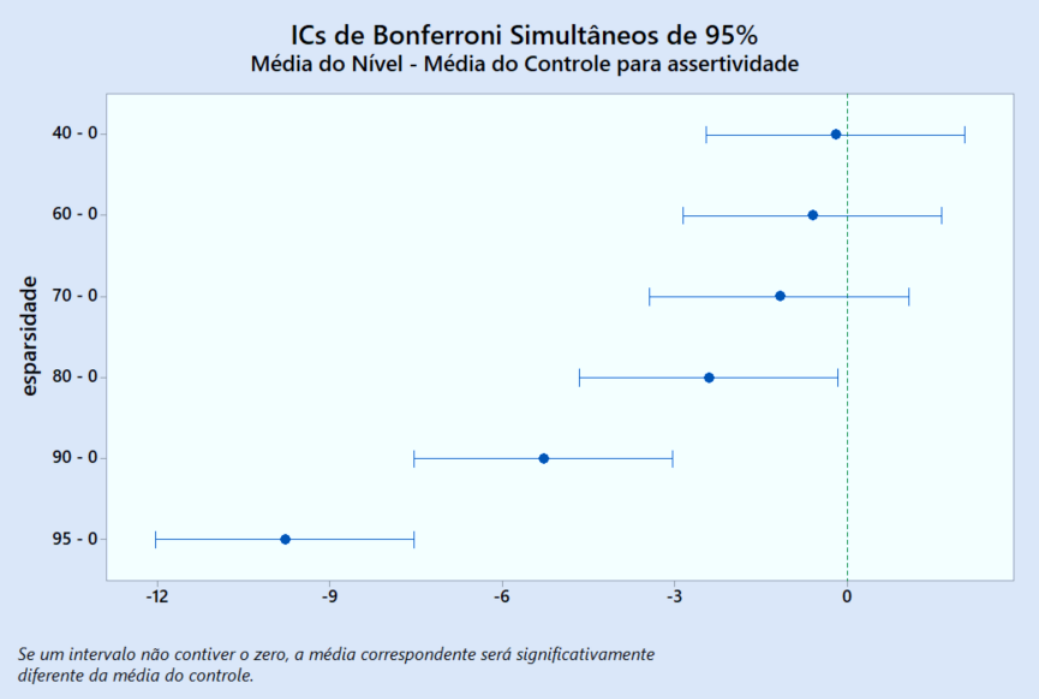
Bonferroni com um fator controlável - o fator escolhido foi a esparsidade nula, já que é buscado a mais alta esparsidade com o menor impacto possível na resposta. A resposta desse teste é descrita na Tabela 11 e ilustrada na Figura 22.

Tabela 11 – Resultado do Teste de Bonferroni de Comparação Múltipla para a Taxa de acerto com um Controle: Esparsidade.

Esparsidade	N	Média (%)	Agrupamento
0 (Controle)	4	69,1	A
40	4	68,9	A
60	4	68,5	A
70	4	67,9	A
80	4	66,7	
90	4	63,8	
95	4	59,3	

Fonte: O autor.

Figura 22 – Intervalo de Confiança de Bonferroni para o Teste de Comparação Múltipla para a taxa de acerto com um Controle: Esparsidade.



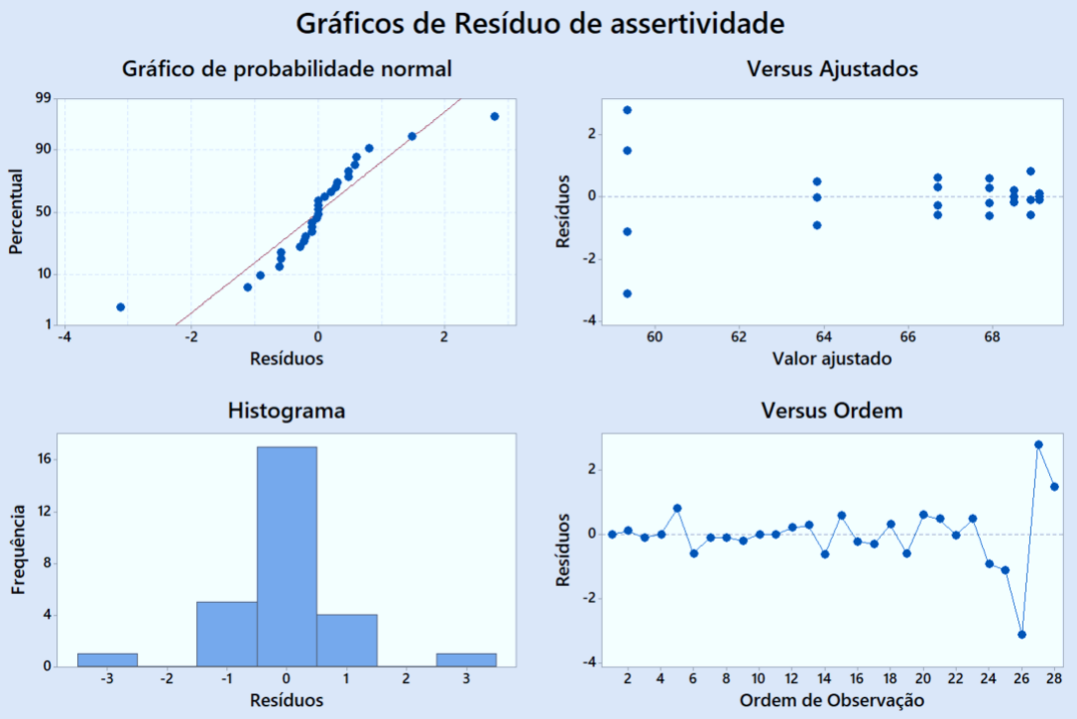
Fonte: O autor.

Pode-se analisar que as esparsidades 0, 40, 60 e 70% fazem parte do grupo 'A', ou seja, suas médias não são significativamente diferentes entre si. Diante disso, foi escolhido como esparsidade ideal a esparsidade de 70% já que é o maior nível que não impacta significativamente a taxa de acerto.

4.3.1.2 Análise de Resíduos do Teste de ANOVA e Teste Normalidade para taxa de acerto e Esparsidade

Assim como na subseção 4.2.2 foi feito a análise dos resíduos do teste de ANOVA - ilustrados na Figura 23.

Figura 23 – Gráfico de Resíduos do teste de ANOVA de taxa de acerto com Fator de Esparsidade.

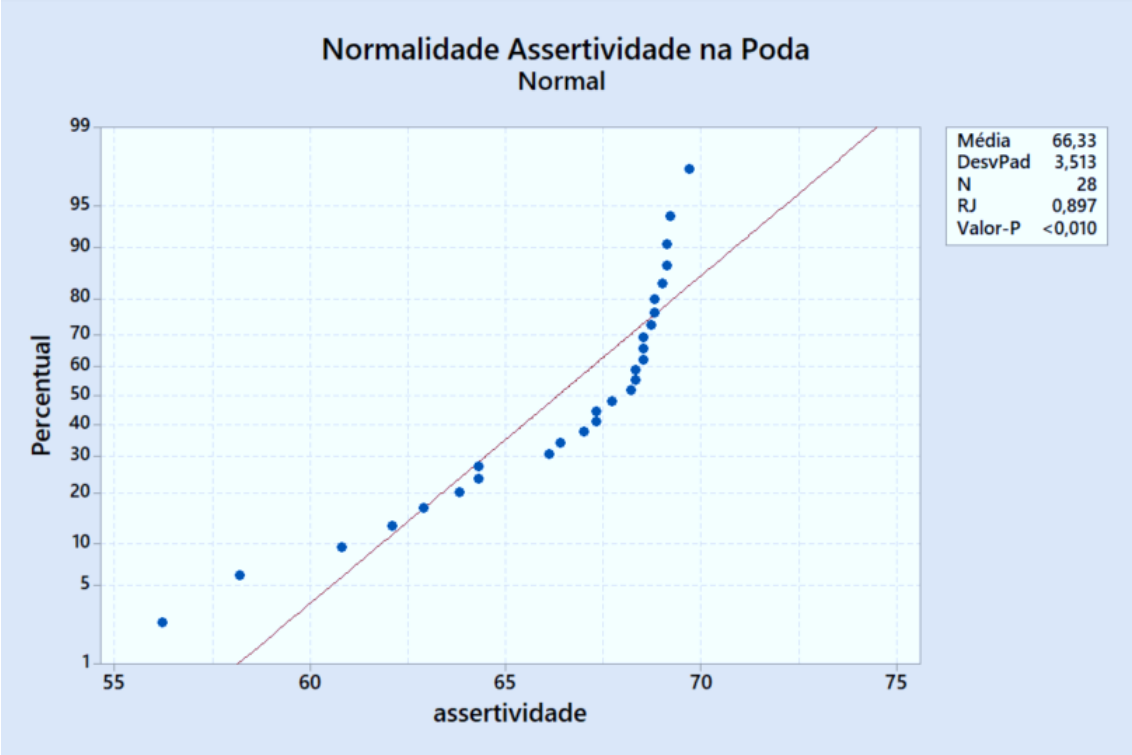


Fonte: O autor.

A análise é análogo ao da subseção 4.2.2 - resíduos independentes, padrão de variância constante com distribuição aleatória com indício de violação de distribuição normal.

Com isso, foi feito, também, o teste de normalidade de Ryan-Joiner para os dados de taxa de acerto, conforme ilustra a Figura 24.

Figura 24 – Teste de Normalidade RJ para Resultados de taxa de acerto nos Testes de Poda Computacional.



Fonte: O autor.

Assim como no teste de normalidade da subseção 4.2.2, a hipótese de normalidade dos dados de taxa de acerto foi rejeitada.

4.3.2 Quantização na Topologia Escolhida

Os testes da quantização do modelo com esparsidade de 70% podem ser analisados nas Tabelas 10 e 12 e ilustrado na Figura 25, que compara a mediana da taxa de acerto (quatro inferências) do modelo quantizado, do modelo podado e do quantizado.

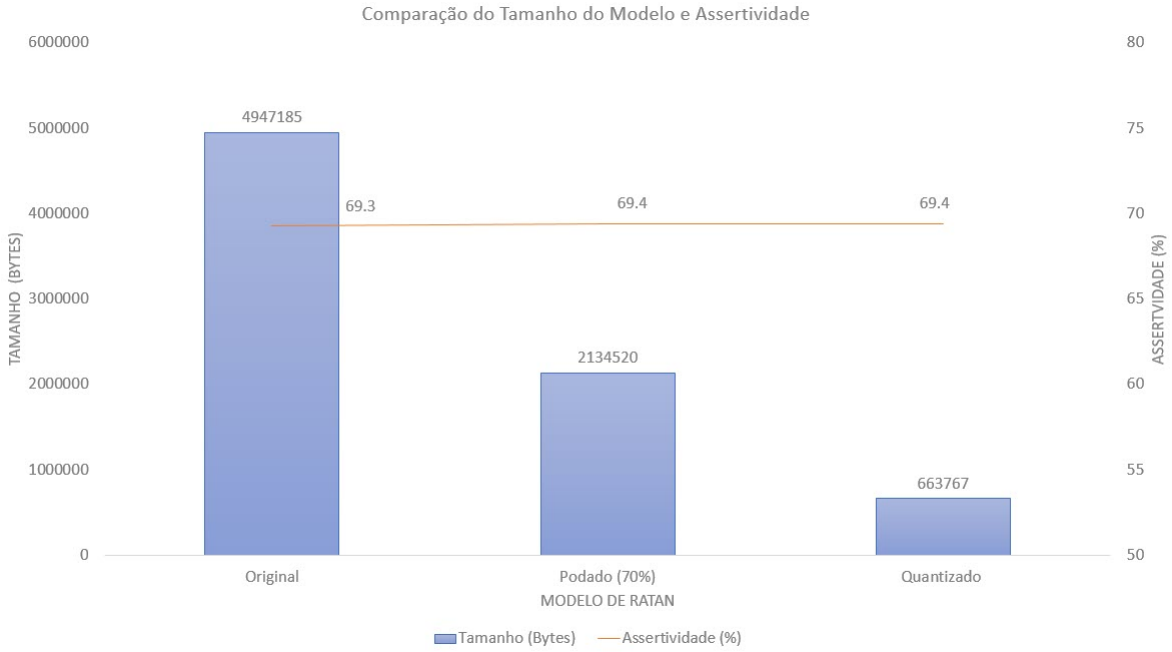
Tabela 12 – Descrição Estatística do Tamanho dos Modelos Original, Podado e Quantizado.

Modelo	N	Tamanho (Bytes)	Desvio Padrão
70% de Esparsidade	4	2134476	673
Original	4	4947139	171
Quantizado	4	663255	1974

Fonte: O autor.

Com base nessas descrições estatísticas, é possível afirmar que inferência dos tamanhos dos modelos denota que os modelos original, podado e quantizado são praticamente constantes, devido aos seus baixos desvios padrão na inferência. Já os dados da taxa de acerto obtiveram maiores desvios.

Figura 25 – Comparação de taxa de acerto e Tamanho do Modelo Original, do Podado e do Quantizado.



Fonte: O autor.

Pode-se afirmar que o tamanho do modelo podado em relação ao original foi diminuído em 2,3 vezes (4,94 M Bytes para 2,13 M Bytes). Já o modelo quantizado em relação ao podado foi comprimido em 3,2 vezes (2,13 M Bytes para 663 K Bytes). Por conseguinte, o modelo quantizado em relação ao original foi reduzido em 7,5 vezes.

A taxa de acerto, representada na linha laranja do gráfico, se manteve praticamente constante, sendo de 69,3% para o modelo original e 69,4% para o modelo podado e para o modelo quantizado.

Com o objetivo de verificar o impacto da poda quantização do modelo na taxa de acerto do modelo, ainda foi feito um teste de ANOVA com um fator - sendo o tipo do modelo - com resposta sendo a taxa de acerto. O resultado é ilustrado na Tabela 13.

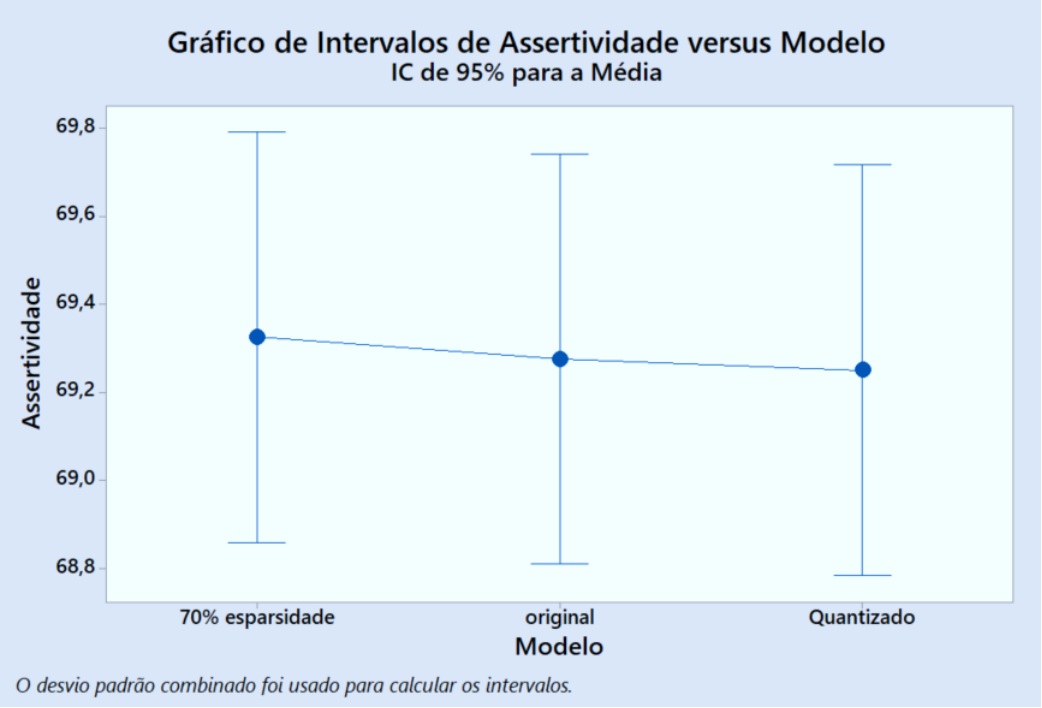
Tabela 13 – ANOVA de um Fator para Verificar o Impacto do tipo do Modelo na Taxa de acerto.

Fonte	GL	SQ(Aj.)	QM(Aj.)	Valor F	Valor P
Esparsidade	2	0,01167	0,005833	0,03	0,966
Erro	9	1,52500	0,169444		
Total	11	1,53667			

Fonte: O autor.

Pode-se verificar, com bases nos resultados expostos, que a quantização não teve impacto significativo na taxa de acerto do modelo original. Essa resposta é ainda mais explicitada quando pelo gráfico de intervalos de desvio padrão da taxa de acerto de cada tipo de modelo, ilustrado na Figura 25.

Figura 26 – Descrição Estatística do Tamanho dos Modelos Original, Podado e Quantizado.

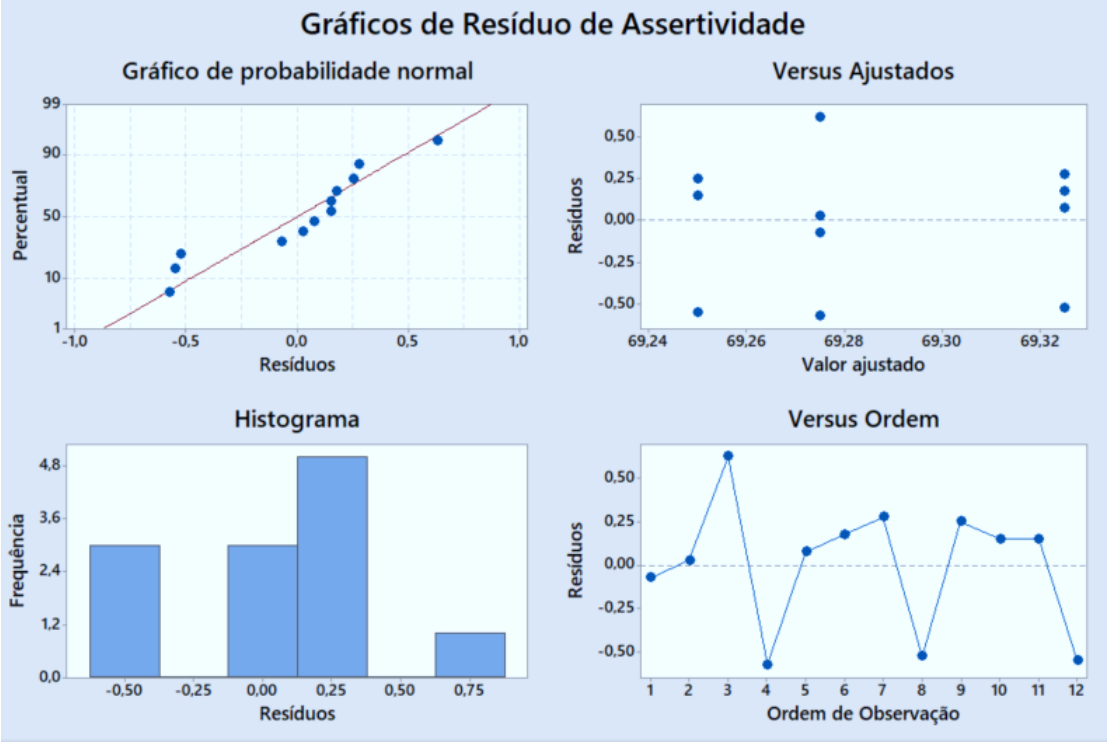


Fonte: O autor.

4.3.2.1 Análise de Resíduos do Teste de ANOVA e Teste de Normalidade para taxa de acerto e Tipo de Modelo

Assim como na subseção 4.2.2 e na subseção 4.3.1.2, foi feito a análise dos resíduos do teste de ANOVA - ilustrados na Figura 27.

Figura 27 – Gráfico de Resíduos do teste de ANOVA de taxa de acerto com Fator de Esparsidade.

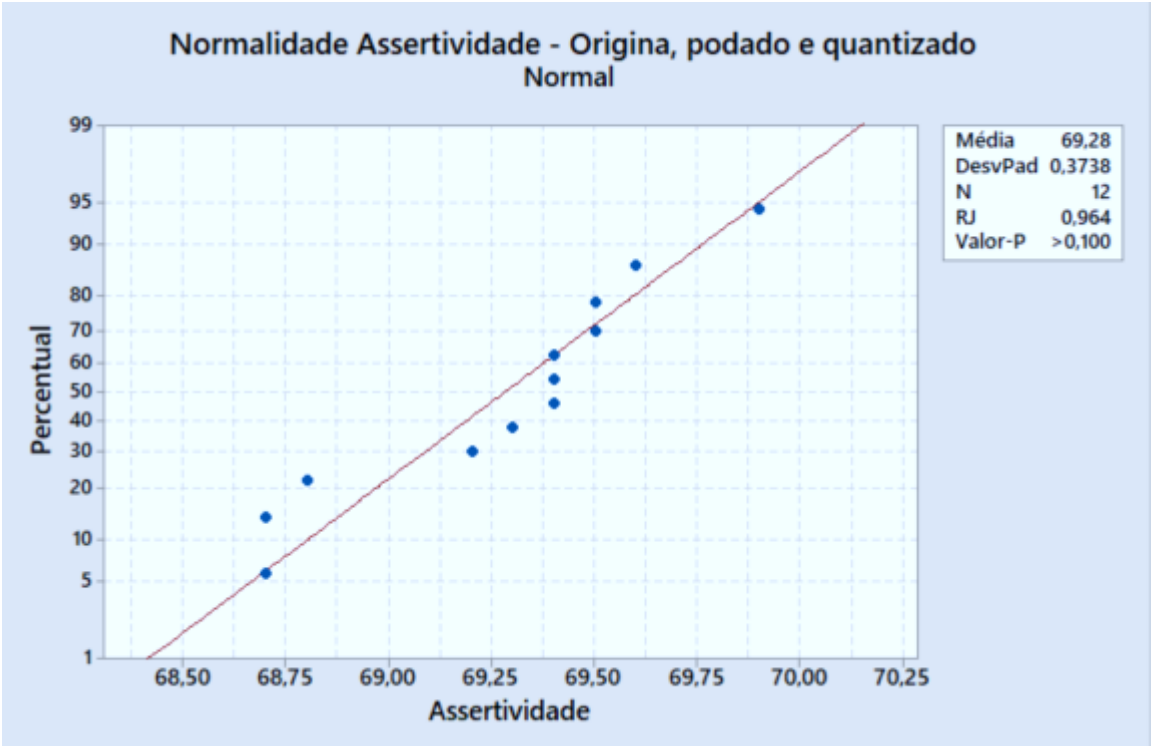


Fonte: O autor.

A análise é análogo ao da subseção 4.2.2 e 4.3.1.2 - independência dos resíduos, variância com padrão constante e com distribuição estocástica com indício de violação de distribuição normal.

O teste de normalidade de Ryan-Joiner para os dados de taxa de acerto também foi realizado, ilustrado na Figura 28.

Figura 28 – Teste de Normalidade RJ para dados de taxa de acerto em Relação ao Tipo de Modelo.



Fonte: O autor.

Pode-se afirmar que esse conjunto de dados de taxa de acerto tem distribuição normal, pelo fato do valor 'P' do teste ser maior que a significância e pelo coeficiente de 'RJ' tende a um valor unitário.

4.4 Resultados de Desempenho do Modelo

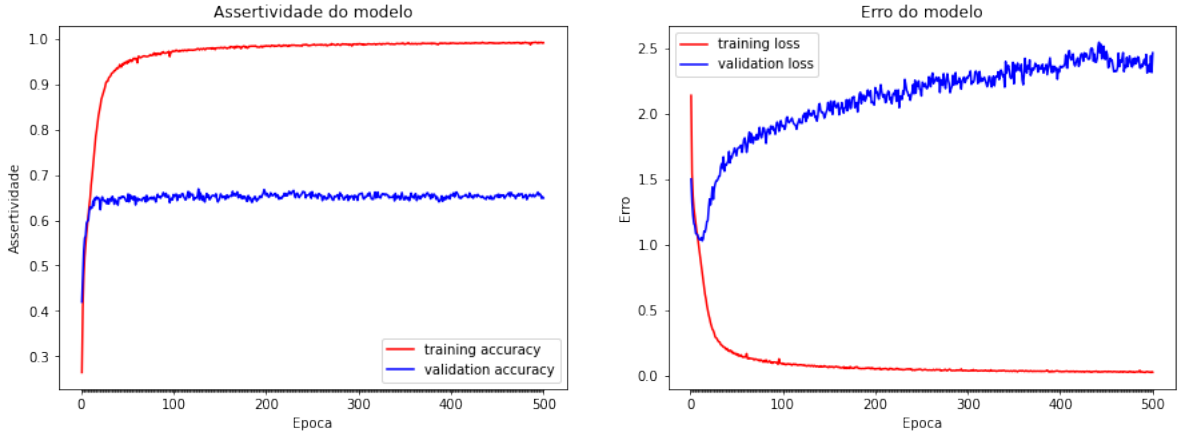
As imagens 29 e 30 ilustram o treinamento do modelo para inferência final da estrutura, sem *Data Augmentation* e com *Data Augmentation*.

A Figura 29 ilustra que há tendência de *overfitting* antes da centésima época ocorrer, pois o erro no conjunto de validação aumenta a medida que o erro do conjunto de treinamento diminui. Esse comportamento é suavizado com a implementação do *Data Augmentation*, mas ainda assim não há decréscimo no erro do conjunto de validação após a centésima época. No retreinamento com a poda computacional (análogo ao treinamento com 1000 épocas do modelo), por sua vez, a cada época de treinamento a mais tendência de *overfitting*.

Vale ressaltar que nos casos ilustrados, o erro e a taxa de acerto não está integralmente inversamente correlacionados, já que o erro mensura a diferença entre a previsão bruta e o rótulo da classe. Esse fato também pode ser amplificado pelo otimizador adotado, pois a entropia cruzada penaliza mais uma previsão ruim do que compensa uma boa

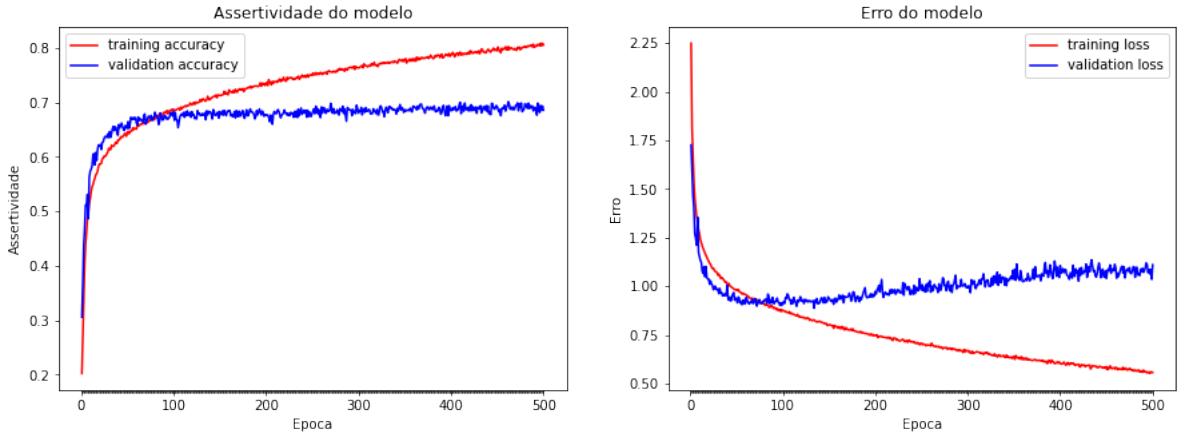
previsão, aumentando o erro, mas sem a diminuição da taxa de acerto. Diante disso, é razoável inferir que a quantidade de épocas utilizada poderia ser reduzido ou mesmo ter sido utilizado condições de parada prematura. Contudo, como foi comparado com outros modelos de outros autores, é também razoável afirmar que condições justas de comparação foram atendidas.

Figura 29 – Treinamento sem Método de *Data Augmentation*.



Fonte: O autor.

Figura 30 – Treinamento com Método de *Data Augmentation*.

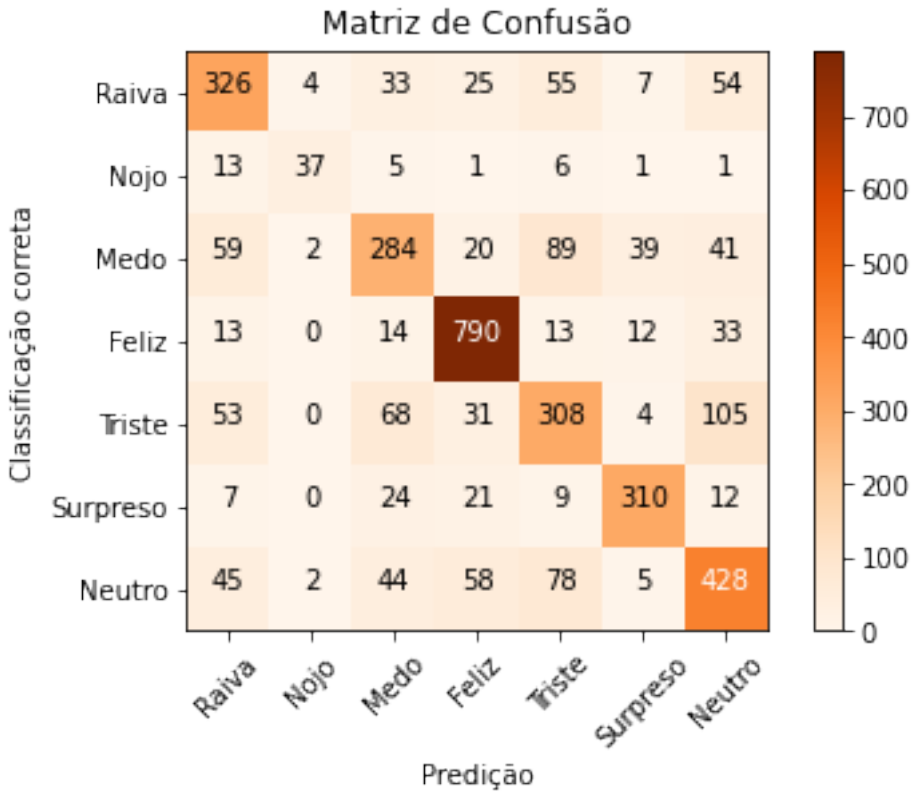


Fonte: O autor.

Em relação à taxa de acerto, o modelo sem método de *Data Augmentation*, referente ao treinamento da Figura 29, obteve uma taxa de acerto de 63,5%. O modelo com método de *Data Augmentation* alcançou 67,5% de taxa de acerto e o modelo com poda computacional de 70% obteve no conjunto de teste uma taxa de acerto de 69,2%.

Por conseguinte, a matriz de confusão do modelo final é ilustrada na Figura 31. Observa-se que a classe que possui maior classificação correta é a felicidade e a que menos apresenta classificação correta é o "nojo".

Figura 31 – Matriz de Confusão do Modelo Final.



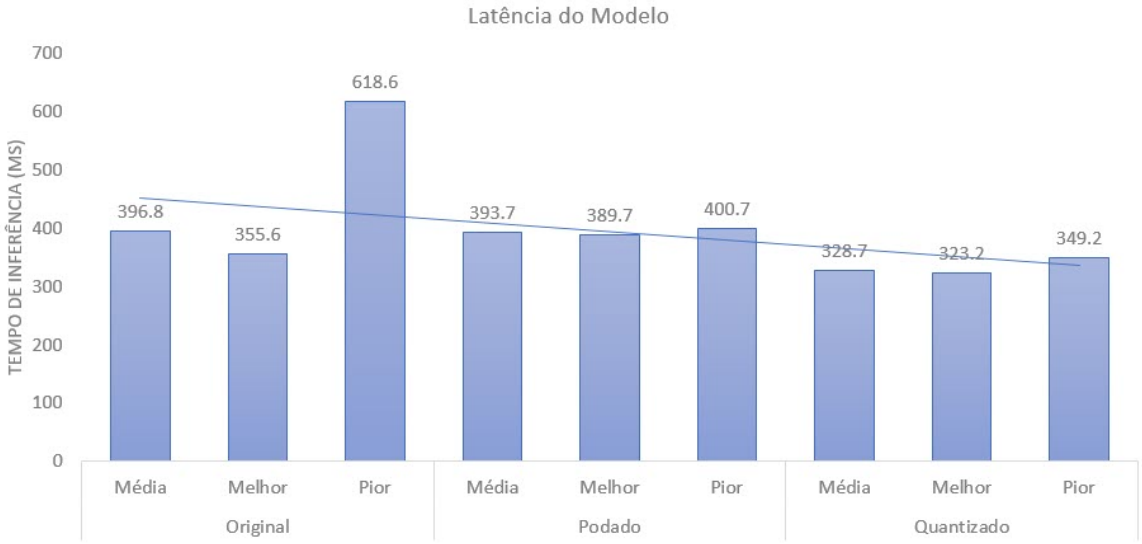
Fonte: O autor.

O modelo quantizado possui 398.451 parâmetros, 660.500 *bytes* de tamanho final e 69,0% de taxa de acerto.

A latência de cada topologia do modelo utilizando a função %*timeit* e a *bechmark tool* para o modelo quantizado é ilustrada no gráfico da Figura 32 que apresenta valores em ms e possui linha de tendência.

Nota-se que há uma tendência de redução de latência do modelo original para o podado e para o quantizado, evidenciado nos piores casos e média temporal de cada tipo do modelo. Contudo, o caso mais rápido do laço da função para mensurar o tempo no modelo original foi mais rápido que o melhor caso que o modelo esparso. Ainda, vale ressaltar que o quantizado apresentou menos tempo de inferência na base de dados de teste.

Figura 32 – Latência do Modelo.

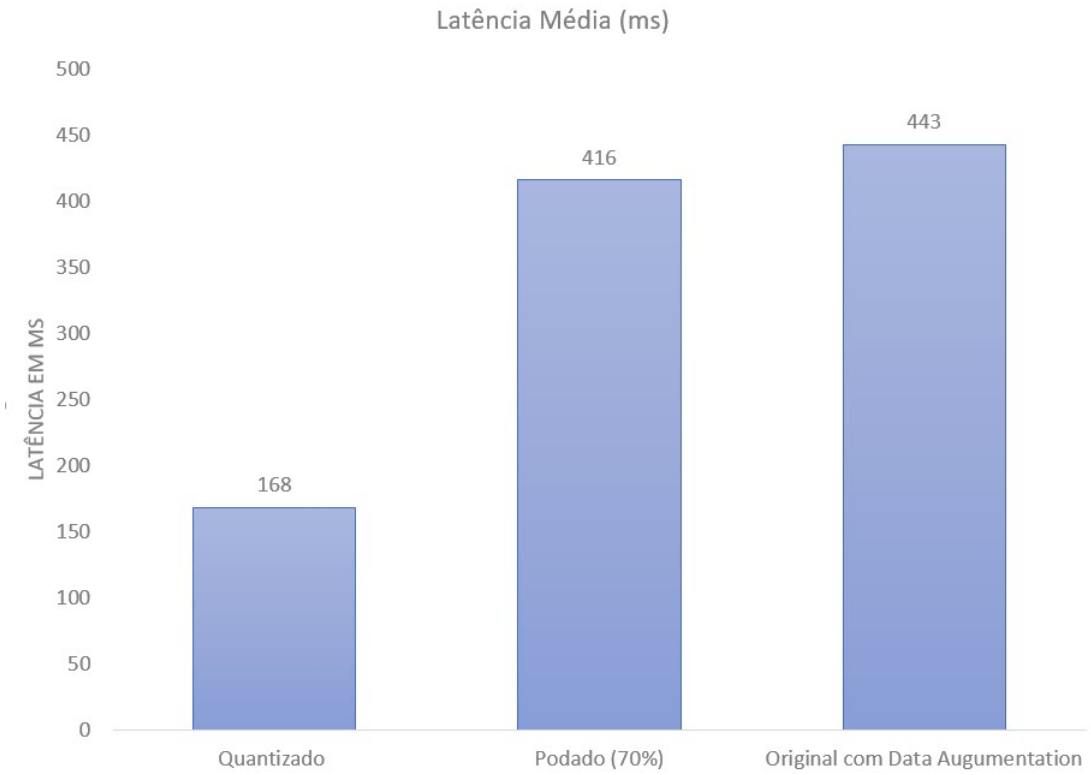


Fonte: O autor.

4.4.1 Resultado do Modelo Utilizado em *Hardware* Limitado

A latência média - resultado das dez inferências por modelo - das topologias em *hardware* é ilustrada na Figura 33.

Figura 33 – Latência Média.



Fonte: O autor.

É possível inferir, com base nos dados expostos, que em *hardware* a latência foi reduzida em 58,9% (ou 2,4 vezes) do modelo original ao quantizado e em 6,1% do original ao podado. Seguindo a mesma lógica, a redução de latência do quantizado ao podado foi de 56,3% (ou 2,3 vezes).

Importante salientar que o modelo original é executado no *hardware*, mas sem o mesmo desempenho do quantizado, já que a latência do modelo quantizado é otimizada. A Figura 34 ilustra sete inferências utilizando a câmera e o modelo final quantizado para todas as classes de detecção.

Figura 34 – Inferências Utilizando o Sistema.

Fonte: O autor.

4.4.1.1 Estudo Estatístico da Latência do Modelo Utilizado em *Hardware* Limitado

A Tabela 14 detalha a descrição estatística dos testes de latência com cada modelo. Com base nessa, é possível perceber que, além de apresentar a menor média, o modelo quantizado obteve o menor desvio padrão.

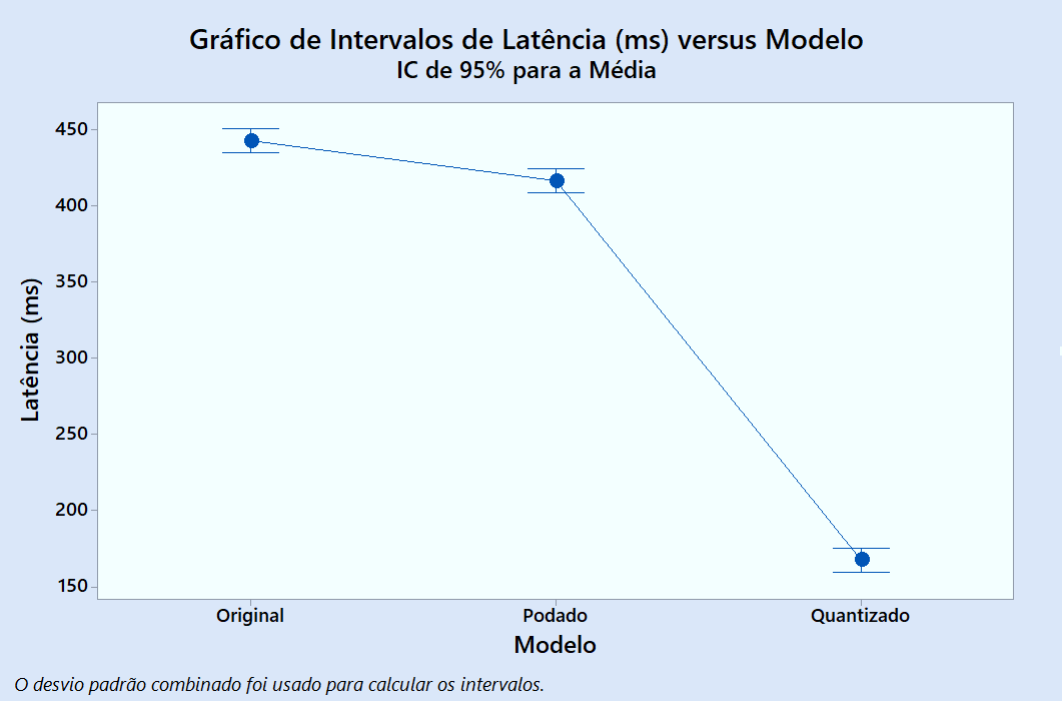
Tabela 14 – Descrição Estatística dos Testes de Latência.

Modelo	N	Latência (ms)	Desvio Padrão
Original	10	442,6	19,2
Podado	10	416,1	9,18
Quantizado	10	167,5	1,17

Fonte: O autor.

O gráfico de intervalos, ilustrado na Figura 35, explicita a discrepância das médias de latência do modelo quantizado e original, com seus respectivos desvios padrão.

Figura 35 – Gráfico de Intervalos de Latência por Modelo.



Fonte: O autor.

Após, pode-se analisar na Tabela 15 no teste de ANOVA com um fator - sendo o tipo do modelo - com resposta sendo a latência que o tipo de modelo tem impacto significativo na latência, já que o valor 'P' foi nulo.

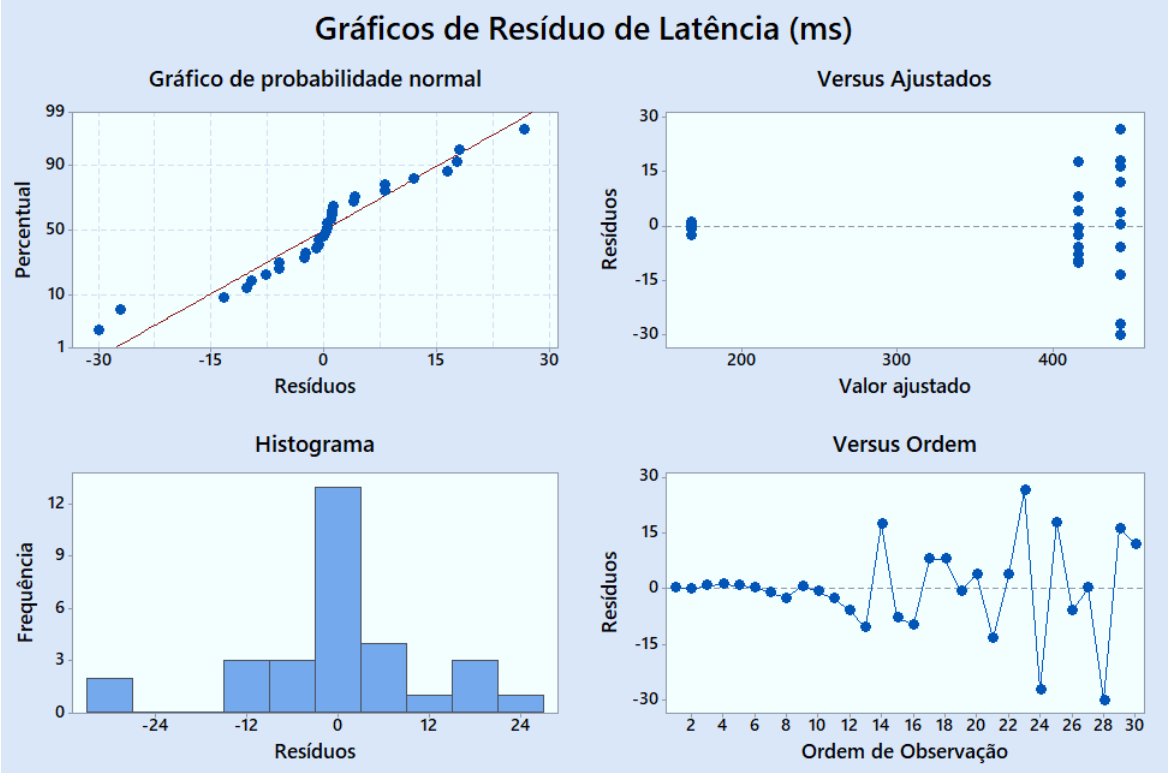
Tabela 15 – ANOVA de um Fator para Verificar o Impacto do tipo do Modelo na Latência.

Fonte	GL	SQ(Aj.)	QM(Aj.)	Valor F	Valor P
Modelo	2	460754	230377	1525,32	0,00
Erro	27	4078	151		
Total	29	464832			

Fonte: O autor.

Os gráficos de resíduos do teste da ANOVA são ilustrados na Figura 38. A análise segue o raciocínio das análise de resíduos feitas na subseções 4.2.2 e 4.3.1.2 - os resíduos possuem independência, padrão constante na variância e tendência de normalidade

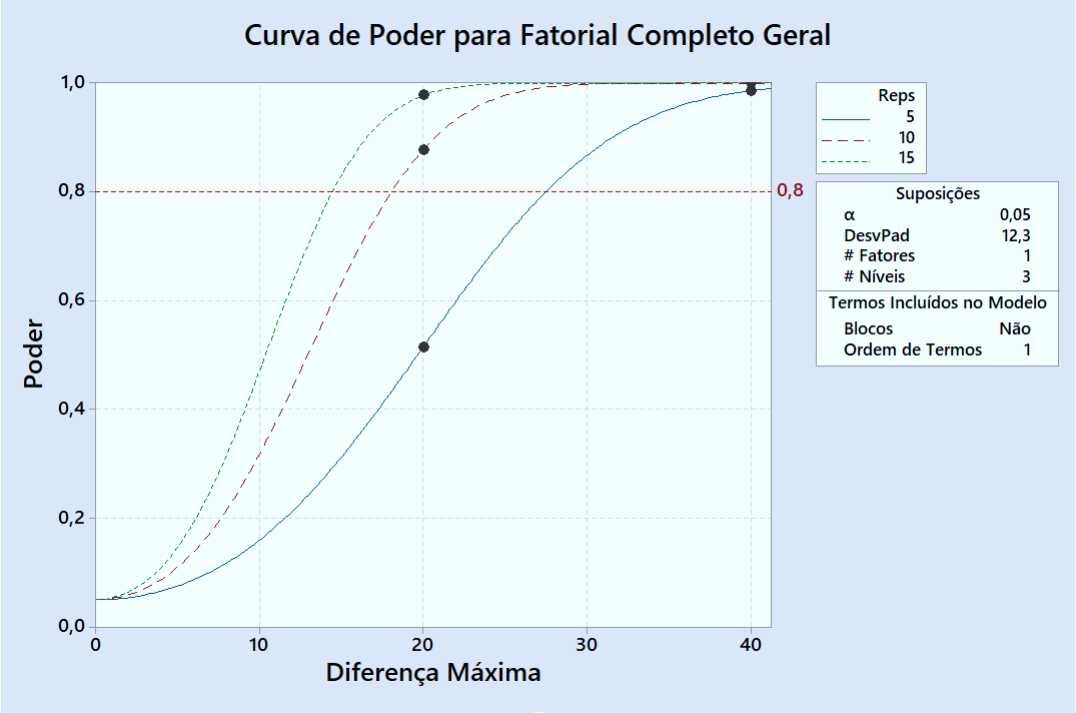
Figura 36 – Resíduos do Teste ANOVA da Latência.



Fonte: O autor.

De posse dos resultados do modelo linear generalizado do teste de ANOVA com um fator e três níveis, foi criar a curva de potência estatística do teste. Foi comparado as curvas com cinco, dez (utilizada no teste) e quinze repetições para cada tipo de modelo - original, podado e quantizado. A Figura 37 ilustra a curva que mostra que para dez repetições possui 87,7% para uma diferença de 20 ms.

Figura 37 – Curva de Potência Estatística Referente aos Testes de Latência.



Fonte: O autor.

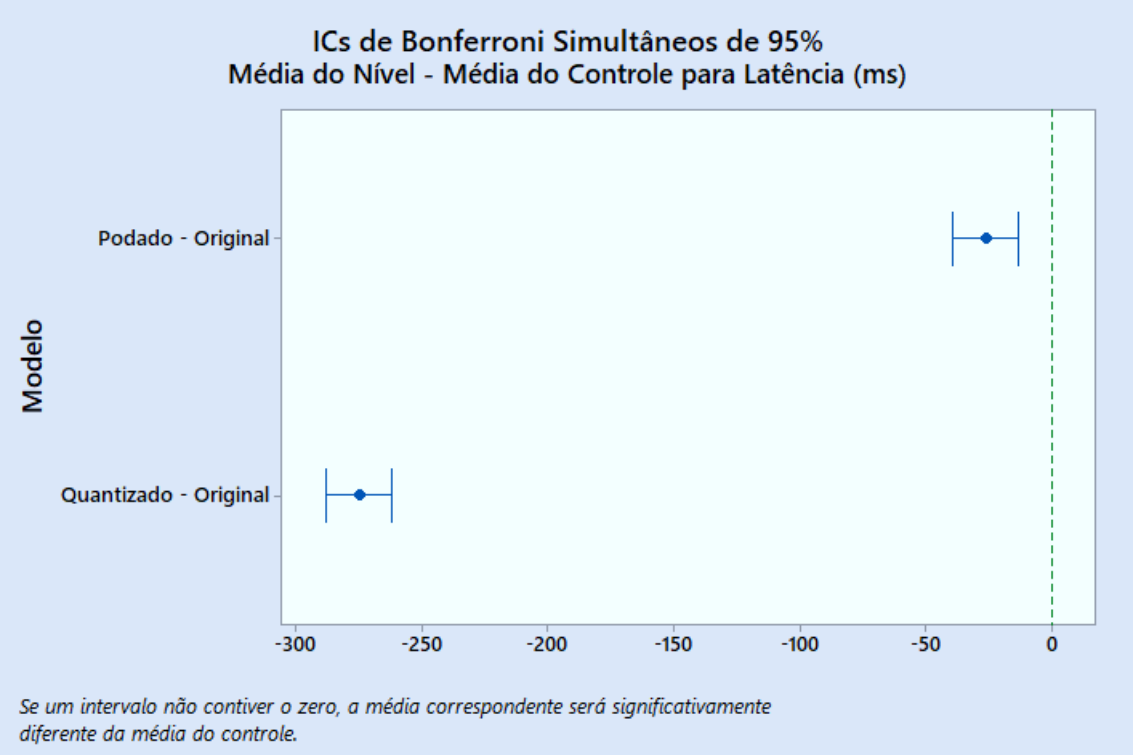
O teste pós hipótese nula de Bonferroni com o controle sendo o modelo original dos resultados do testes de latência mostram que a diferença entre as médias dos modelos original, podado e quantizado são diferentes entre si, já que há apenas um agrupamento - descrito com a letra 'A' na Tabela 16 para o fator de controle. Essa informação é explicitada na Figura 38 que mostra que nenhuma média contém o zero (valor de referência da média de controle).

Tabela 16 – Resultado do Teste de Bonferroni de Comparação Múltipla para a Latência com um Controle: Modelo Original.

Modelo	N	Média (ms)	Agrupamento
Original (Controle)	10	442,6	A
Podado	10	416,2	
Quantizado	10	167,5	

Fonte: O autor.

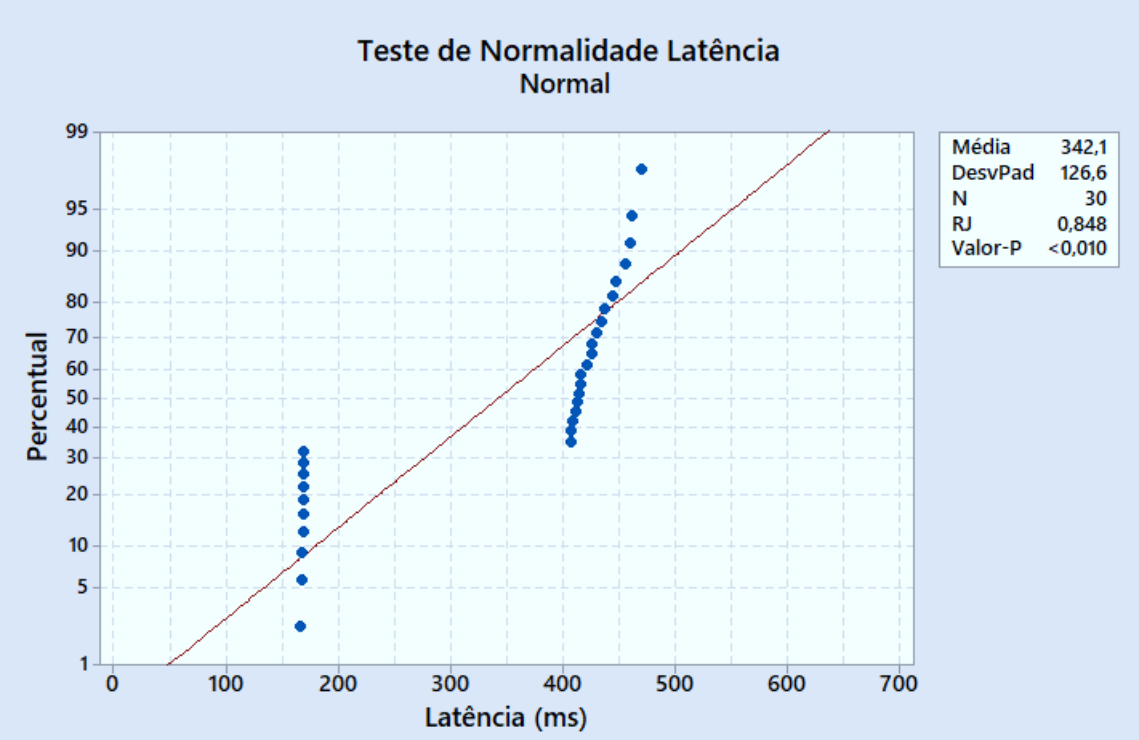
Figura 38 – Intervalo de Confiança de Bonferroni para o Teste de Comparação Múltipla para Latência com um Controle: Modelo Original.



Fonte: O autor.

O teste de normalidade de Ryan-Joiner para os dados de latência é ilustrado na Figura 39, que indica que o conjunto de dados de latência não tem distribuição normal - valor 'P' do teste ser menor que a significância.

Figura 39 – Teste de Normalidade RJ para dados de Latência em Relação ao Tipo de Modelo.

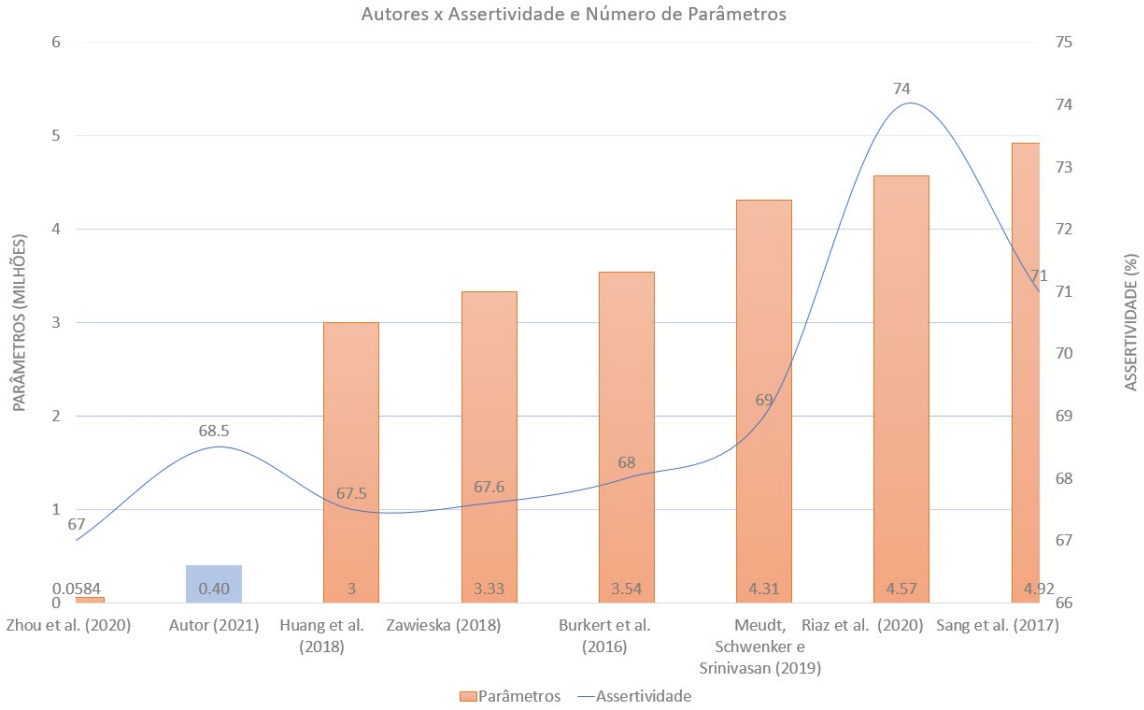


Fonte: O autor.

4.4.2 Resultado da Comparação com Trabalhos da literatura

O gráfico ilustrado na Figura 40 compara os resultados dos principais autores da comunidade com os resultados obtidos na monografia (destacado em azul).

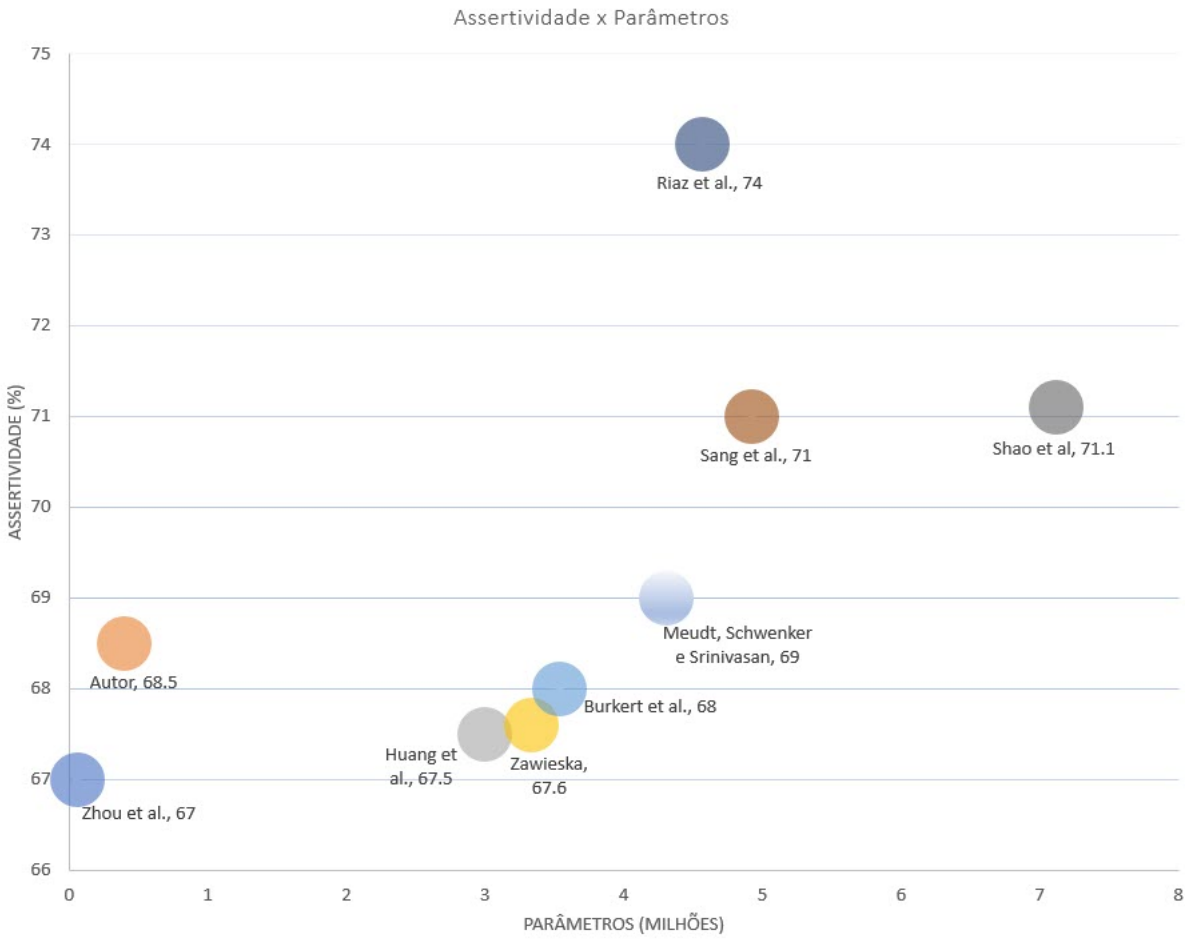
Figura 40 – Comparação de taxa de acerto e Número de Parâmetros entre Autores.



Fonte: O autor.

Pode-se inferir que, com a esparsidade escolhida, o número de parâmetros da estrutura pesquisa se encontra entre as menores, porém, em relação a taxa de acerto há trabalhos que obtiveram um percentual superior a 70%. Esse raciocínio fica mais evidente no gráfico da Figura 41 - onde a presente pesquisa se encontra destacada em laranja - que tem como condição ótima a coordenada mais próxima da origem no eixo das abscissas e tendendo a 100% no eixo das ordenadas.

Figura 41 – Comparação de taxa de acerto e Número de Parâmetros entre Autores.



Fonte: O autor.

A latência final pode ser comparada com os autores Riaz et al (2020), que obteve até 998 ms de latência (não é descrito se é a latência já com a caputura da imagem ou a inferência da emoção) e Meudt, Schwenker e Srinivasan (2019) que obtiveram em sua até 222 ms de latência, já englobando a captura da imagem pela câmera - Tabela 17. Todavia, é evidente que essa métrica de desempenho não é padronizada na literatura como é a taxa de acerto, por exemplo, e comparações envolvendo a latência com outros autores podem não ser conclusivas.

Tabela 17 – Comparativo de Desempenho de Pesquisas Similares com *Hardware* Limitado.

Pesquisa	Parâmetros (M)	Tempo (ms)	Taxa de Acerto (%)
Autor (2021)	0,398	168	68,5
Meudt, Schwenker e Srinivasan (2019)	4,31	222	69,0
Riaz et al. (2020)	4,57	998	74,0

Fonte: O autor.

5 Conclusões

O trabalho apresentou um sistema de detecção de emoções em *hardware* limitado em tempo real, sendo relevante em diversas aplicações. O modelo implementado contou com uma taxa de acerto média de 68,5 % com incerteza de 1%, utilizando uma estrutura de 398.451 parâmetros que possui latência média de 182 ms.

A arquitetura utilizada foi proposta por Ratan (2019) e otimizada para ter a maior taxa de acerto (usando fundamentalmente o método de *Data Augmentation*) utilizando o menor número de parâmetros possível. Apesar de na literatura haver mais pesquisas que obtiveram maior taxa de acerto, essas utilizam uma estrutura com maior número de parâmetros. Ainda, comparando com outras pesquisas que também utilizam *hardware* limitado, o trabalho demonstrou um tempo de latência razoável, sendo inferior às pesquisas comparativas, demonstrando a eficácia da otimização feita pela combinação entre poda computacional e quantização, que reduziu em mais de duas vezes o tempo de latência, sem impactar na taxa de acerto.

O método para medir a latência da rede sem considerar o *hardware* limitado conteve fragilidades, já que duas ferramentas diferentes foram utilizadas para calcular o tempo de processamento entre o modelo original/podado e o quantizado. Por outro lado, o teste de latência utilizando o *hardware* limitado se demonstrou mais eficaz, já que apresentou resultados expostos na literatura sobre métodos de otimização de modelos.

Apesar dos testes utilizados na ANOVA nos testes de escolha da topologia e de escolha da esparsidade da poda computacional violarem a distribuição normal do conjunto de dados da resposta da taxa de acerto, o procedimento está dentro do padrão do tipo de teste, já que a distribuição não se demonstrou altamente assimétrica.

5.1 Trabalhos Futuros

Os resultados apresentados na pesquisa demonstram que há possibilidade de melhoramentos na estrutura da rede, como refinamento dos hiperparâmetros ou até mesmo escolha de outra topologia, e nos métodos empregados de poda computacional - método de poda que não seja constante ou que tenha diferente agenda de corte - e de quantização - podem ser testados métodos mais arrojados de quantização como um método híbrido de quantização durante e pós treinamento.

No segmento de uso de adaptação para uso em *hardware* podem ser realizados mais testes envolvendo o processamento de cada classe - pode ser realizado até mesmo o

agrupamento de classes da base de dados a depender do campo de implementação de um futuro projeto, como detecção binária de emoções.

Por fim, ainda há possibilidade de uso de diferentes tipos bases de dados ou mesmo o aumento do número de registros por meio de suplementação artificial, já que a FER-2013 demonstrou desbalanceada - explicitado pelos número de imagens e taxa de acerto para emoção de nojo e felicidade.

Referências Bibliográficas

AGHDAM, E. J. H. H. H. *Guide to Convolutional Neural Networks: A Practical Application to Traffic-Sign Detection and Classification*. [S.l.: s.n.], 2017.

ALIPPI SIMONE DISABATO, M. R. C. Moving convolutional neural networks to embedded systems: the alexnet and vgg-16 case. *ACM/IEEE International Conference on Information Processing in Sensor Networks*, 2018.

ALVAREZ PULKIT BHUWALKA, L. C. A. C. T. D. S. G. J. L. Y. L. S. S. D. S. S. S. R. *TensorFlow Model Optimization Toolkit — Pruning API — The TensorFlow Blog*. 2019. Disponível em: <https://blog.tensorflow.org/2019/05/tf-model-optimization-toolkit-pruning-API.html>. Acesso em: 16 maio 2021.

BURKERT, P.; TRIER, F.; AFZAL, M. Z.; DENGEL, A.; LIWICKI, M. *DeXpression: Deep Convolutional Neural Network for Expression Recognition*. 2016.

CHOLLET, F. *Deep Learning with Python*. [S.l.: s.n.], 2017. ISBN 9781617294433.

DANG, T. A. *Top 10 CNN Architectures Every Machine Learning Engineer Should Know | by Trung Anh Dang | Towards Data Science*. 2021. Disponível em: <https://towardsdatascience.com/top-10-cnn-architectures-every-machine-learning-engineer-should-know-68e2b0e07201>. Acesso em: 07 junho 2021.

DUTTA, J. *Raspberry Pi Based Emotion Recognition using OpenCV, TensorFlow, and Keras*. 2021. Disponível em: <https://circuitdigest.com/microcontroller-projects/raspberry-pi-based-emotion-recognition-using-opencv-tensorflow-and-keras>. Acesso em: 25 agosto 2021.

EKMAN, P. *Emotions Revealed, Second Edition*. [S.l.: s.n.], 2007. ISBN 9780805083392.

FABIEN, M. *Multimodal Emotion Recognition*. Disponível em: <https://https://github.com/maelfabien/Multimodal-Emotion-Recognition/blob/master/03-Video/Notebooks/00-Fer2013.ipynb>. Acesso em: 20 maio 2021.

FAN, Y.; LAM, J. C. K.; LI, V. O. K. *Multi-Region Ensemble Convolutional Neural Network for Facial Expression Recognition*. 2018.

GIANNOPOULOS ISIDOROS PERIKOS, I. H. P. *Advances in Hybridization of Intelligent Methods*. [S.l.: s.n.], 2017.

GOODFELLOW, I. J.; ERHAN, D.; CARRIER, P. L.; COURVILLE, A.; MIRZA, M.; HAMNER, B.; CUKIERSKI, W.; TANG, Y.; THALER, D.; LEE, D.-H.; ZHOU, Y.; RAMAIAH, C.; FENG, F.; LI, R.; WANG, X.; ATHANASAKIS, D.; SHAWE-TAYLOR, J.; MILAKOV, M.; PARK, J.; IONESCU, R.; POPESCU, M.; GROZEA, C.; BERGSTRA, J.; XIE, J.; ROMASZKO, L.; XU, B.; CHUANG, Z.; BENGIO, Y. *Challenges in Representation Learning: A report on three machine learning contests*. 2013.

HAN, S.; MAO, H.; DALLY, W. J. *Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding*. 2016.

HE, K.; ZHANG, X.; REN, S.; SUN, J. *Deep Residual Learning for Image Recognition*. 2015.

HUANG, G.; LIU, Z.; MAATEN, L. van der; WEINBERGER, K. Q. *Densely Connected Convolutional Networks*. 2018.

JAIN, D. K.; SHAMSOLMOALI, P.; SEHDEV, P. Extended deep neural network for facial emotion recognition. *Pattern Recognit. Lett.*, v. 120, p. 69–74, 2019.

JOHNSTON, S. J.; BASFORD, P. J.; PERKINS, C. S.; HERRY, H.; TSO, F. P.; PEZAROS, D.; MULLINS, R. D.; YONEKI, E.; COX, S. J.; SINGER, J. Commodity single board computer clusters and their applications. *Future Generation Computer Systems*, v. 89, p. 201–212, 2018. ISSN 0167-739X. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0167739X18301833>. Acesso em: 15 maio 2021.

KARIM, R. *Illustrated: 10 CNN Architectures | by Raimi Karim | Towards Data Science*. 2021. Disponível em: <https://towardsdatascience.com/illustrated-10-cnn-architectures-95d78ace614d>. Acesso em: 07 junho 2021.

KINGMA, D. P.; BA, J. *Adam: A Method for Stochastic Optimization*. 2017.

KINLI, F. *Deep Learning Lab: fer2013*. Disponível em: <https://medium.com/birdortyedi_23820/deep-learning-lab-episode-3-fer2013-c38f2e052280>. Acesso em: 20 maio 2021.

LI, K.; JIN, Y.; AKRAM, M.; HAN, R.; CHEN, J. Facial expression recognition with convolutional neural networks via a new face cropping and rotation strategy. *The Visual Computer*, v. 36, 02 2020.

Li, S.; Deng, W. Reliable crowdsourcing and deep locality-preserving learning for unconstrained facial expression recognition. *IEEE Transactions on Image Processing*, v. 28, n. 1, p. 356–370, 2019.

Liu, K.; Zhang, M.; Pan, Z. Facial expression recognition with cnn ensemble. In: *2016 International Conference on Cyberworlds (CW)*. [S.l.: s.n.], 2016. p. 163–166.

LOPES, A. T.; de Aguiar, E.; De Souza, A. F.; OLIVEIRA-SANTOS, T. Facial expression recognition with convolutional neural networks: Coping with few data and the training sample order. *Pattern Recognition*, v. 61, p. 610–628, 2017. ISSN 0031-3203. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0031320316301753>.

MINITAB. *O que é ANOVA?* 2021. Disponível em: <https://support.minitab.com/pt-br/minitab/18/help-and-how-to/modeling-statistics/anova/supporting-topics/basics/what-is-anova/>. Acesso em: 20 agosto 2021.

MONTGOMERY, D. C. *Statistical Quality Control*. [S.l.: s.n.], 2012. ISBN 9781118146811.

MOONS, B.; BANKMAN, D.; VERHELST, M. *Embedded Deep Learning*. [S.l.: s.n.], 2018. ISBN 9783319992228.

RAI, P. *FER2013-Facial-Emotion-Recognition*. Disponível em: <https://github.com/pranjalrai-iitd/FER2013-Facial-Emotion-Recognition->. Acesso em: 20 maio 2021.

RATAN, R. *Using LittleVGG for Emotion Detection*. Disponível em: <https://github.com/rajeevratan84/DeepLearningCV/blob/master/18.2\%20Building\%20an\%20Emotion\%20Detector\%20with\%20LittleVGG.ipynb>. Acesso em: 20 maio 2021.

RIAZ YAO SHEN, M. S. M. G. M. N. exnet: An efficient approach for emotion recognition in the wild. *Sensors*, 2020.

SANG, D.; DAT, V.; THUAN, D. Facial expression recognition using deep convolutional neural networks. In: . [S.l.: s.n.], 2017. p. 130–135.

SANTNER, T. J.; WILLIAMS, B. J.; NOTZ, W. I. *The Design and Analysis of Computer Experiments*. [S.l.: s.n.], 2019. ISBN 9781493988457.

SCHWENKER, F.; SCHERER, S. *Multimodal Pattern Recognition of Social Signals in Human-Computer-Interaction*. [S.l.: s.n.], 2019. ISBN 9783030209841.

SHAO, J.; QIAN, Y. Three convolutional neural network models for facial expression recognition in the wild. *Neurocomputing*, v. 355, p. 82–92, 2019. ISSN 0925-2312. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0925231219306137>.

SHARMA, N. *How to do Facial Emotion Recognition Using A CNN? | by Nishank Sharma | the ML blog | Medium*. Disponível em: <https://medium.com/themlblog/how-to-do-facial-emotion-recognition-using-a-cnn-b7bbae79cd8f>. Acesso em: 20 maio 2021.

SIMONYAN, K.; ZISSERMAN, A. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 2015.

SIVAKUMAR JIAN LI, S. S. Y. L. A. C. R. A. L. C. D. S. T. D. S. S. S. *TensorFlow Model Optimization Toolkit — Post-Training Integer Quantization — The TensorFlow Blog*. 2019. Disponível em: <https://blog.tensorflow.org/2019/06/tensorflow-integer-quantization.html>. Acesso em: 16 maio 2021.

TANG, Y. *Deep Learning using Linear Support Vector Machines*. 2015.

TAUTKUTE, I.; TRZCINSKI, T. *Classifying and Visualizing Emotions with Emotional DAN*. 2018.

TENHOUTEN, W. From primary emotions to the spectrum of affect: An evolutionary neurosociology of the emotions. *Neuroscience and Social Science: The Missing Link*, 2018.

VENKATESAN, R.; LI, B. *Convolutional Neural Networks in Visual Computing*. [S.l.: s.n.], 2017. ISBN 9781498770392.

WARDEN, P.; SITUNAYAKE, D. *TinyML*. [S.l.: s.n.], 2021. ISBN 9781492052043.

YANG, H.; HAN, J.; MIN, K. A multi-column cnn model for emotion recognition from eeg signals. *Sensors*, v. 19, p. 4736, 10 2019.

ZAWIESKA, L. *CNN based on images from Kaggle's FER2013 competition*. Disponível em: <https://github.com/elzawie/FER2013/blob/master/Best_model.ipynb>. Acesso em: 20 maio 2021.