UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Felipe Dennis de Resende Oliveira

Compressão de Redes Neurais Profundas Explorando Estratégias de Regularização Baseadas em Norma ℓ_0

Florianópolis
2026

Felipe Dennis de Resende Oliveira

Compressão de Redes Neurais Profundas Explorando Estratégias de Regularização Baseadas em Norma ℓ_0

Tese de Doutorado submetida ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Santa Catarina.
Orientador: Prof. Eduardo Luiz Ortiz Batista, Dr.
Coorientador: Prof. Rui Seara, Dr.

Florianópolis
2026

Ficha catalográfica gerada por meio de sistema automatizado gerenciado pela BU/UFSC.
Dados inseridos pelo próprio autor.

Oliveira, Felipe Dennis de Resende
Compressão de Redes Neurais Profundas Explorando Estratégias de Regularização Baseadas em Norma l0 / Felipe Dennis de Resende Oliveira ; orientador, Eduardo Luiz Ortiz Batista, coorientador, Rui Seara, 2026.
87 p.

Tese (doutorado) - Universidade Federal de Santa Catarina, Centro Tecnológico, Programa de Pós-Graduação em Engenharia Elétrica, Florianópolis, 2026.

Inclui referências.

1. Engenharia Elétrica. 2. Redes Neurais Profundas. 3. Redução de Complexidade. 4. Regularização por norma l0. I. Batista, Eduardo Luiz Ortiz. II. Seara, Rui. III. Universidade Federal de Santa Catarina. Programa de Pós Graduação em Engenharia Elétrica. IV. Título.

Felipe Dennis de Resende Oliveira

Compressão de Redes Neurais Profundas Explorando Estratégias de Regularização Baseadas em Norma ℓ_0

O presente trabalho em nível Doutorado foi avaliado e aprovado, em 2 de dezembro de 2025, pela banca examinadora composta pelos seguintes membros:

Prof. Leonardo Tomazeli Duarte, Dr.
UNICAMP

Prof. Ciro André Pitz, Dr.
UFSC

Prof. Richard Demo Souza, Dr.
UFSC

Certificamos que esta é a **versão original e final** do trabalho que foi julgado adequado.

Documento assinado digitalmente
Carlos Renato Rambo
Data: 27/02/2026 09:59:36-0300
CPF: ***.748.198-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Carlos Renato Rambo, Dr
Coordenação do Programa de
Pós-Graduação em Engenharia Elétrica

Documento assinado digitalmente
Eduardo Luiz Ortiz Batista
Data: 26/02/2026 19:11:41-0300
CPF: ***.521.889-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Eduardo Luiz Ortiz Batista, Dr.
Orientador

Documento assinado digitalmente
Rui Seara
Data: 27/02/2026 09:04:15-0300
CPF: ***.107.869-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Rui Seara, Dr.
Coorientador

Florianópolis, 2026.

A Deus, à minha esposa, à minha filha e aos meus pais.

AGRADECIMENTOS

Primeiramente, agradeço a Deus, cujos dons permitiram-me trilhar o caminho que me trouxe até este momento.

À minha esposa, Maiara, agradeço pelo amor, cuidado e pela motivação constante, especialmente nos momentos de dificuldade. Seu apoio e companheirismo são fundamentais para todas as minhas realizações. Obrigado pelo amor incondicional que, tantas vezes, refletiu-se nos sacrifícios necessários para a concretização deste trabalho.

À minha filha, Alice, agradeço por ser luz e alegria em nossa casa.

Agradeço aos meus pais, pelo exemplo e por serem a base de toda a minha formação.

Ao meu orientador, professor Eduardo Luiz Ortiz Batista, expresso minha gratidão pela excelente orientação acadêmica. Além disso, também agradeço pelo apoio e pela compreensão demonstrados nos momentos de maior dificuldade. Sua paciência, incentivo e confiança foram essenciais para que eu superasse os desafios desta jornada e chegasse à conclusão desta tese.

Ao meu coorientador, professor Rui Seara, pelos ensinamentos e pelas valiosas discussões ao longo de todo este percurso.

Aos membros da banca examinadora, pela dedicação na análise deste trabalho e pelas relevantes sugestões que permitiram o seu aprimoramento.

A todos aqueles que, direta ou indiretamente, contribuíram para a realização deste trabalho, o meu sincero muito obrigado.

RESUMO

Apesar da crescente disponibilidade de plataformas computacionais de alta capacidade, a complexidade ainda impõe grandes desafios para a implantação de redes neurais modernas em aplicações práticas. Essa preocupação não se deve exclusivamente aos altos custos computacionais das arquiteturas de redes de última geração, mas também ao recente impulso em direção à computação de borda e ao uso de redes neurais em aplicações embarcadas. Nesse contexto, as técnicas de compressão de redes têm ganhado interesse devido à sua capacidade de reduzir os custos de implantação mantendo a acurácia de inferência em níveis satisfatórios. A presente tese é dedicada ao desenvolvimento de novos esquemas de compressão para redes neurais. Para isso, estratégias de aproximação da norma ℓ_0 são inicialmente exploradas para desenvolvimento novas formas de regularização. Como resultado, regularizações baseadas na aproximação via exponencial da norma ℓ_0 e aproximação linearizada são obtidas. Em ambos os casos, as regularizações obtidas demonstram importante capacidade de induzir esparsidade em redes durante o treinamento. Os pesos da rede atraídos à faixa de valores próximos de zero durante treinamento são podados e redes menores, mas altamente eficazes, são obtidas. O uso da aproximação linearizada da norma ℓ_0 permite alcançar resultados de indução de esparsidade semelhantes à aproximação exponencial, mas com reduções de complexidade matemática e, consequentemente, computacional da regularização durante o treinamento. Os esquemas de compressão propostos também envolvem o uso de regularização de norma ℓ_2 para evitar o *overfitting*, bem como o *fine-tuning* para melhorar o desempenho da rede podada. Resultados experimentais são apresentados com o objetivo de demonstrar a eficácia dos esquemas propostos, bem como para realizar comparações com abordagens concorrentes.

Palavras-chave: Redes Neurais Profundas. Redução de Complexidade. Regularização por norma ℓ_0.

ABSTRACT

Despite the growing availability of high-capacity computational platforms, implementation complexity still has been a great concern for the real-world deployment of neural networks. This concern is not exclusively due to the huge costs of state-of-the-art network architectures, but also due to the recent push towards edge intelligence and the use of neural networks in embedded applications. In this context, network compression techniques have been gaining interest due to their ability for reducing deployment costs while keeping inference accuracy at satisfactory levels. The present thesis is dedicated to the development of novel compression schemes for neural networks. To this end, the ℓ_0-norm aproximation strategies are initially explored to develop new forms of regularization. As a result, two new regularizations are obtained based on the exponential approximation and the linear approximation of the ℓ_0-norm. In both cases, the resulting regularizations demostrated the ability to induce sparsity in networks during training. The network weights attracted to the near-zero range during training are pruned, and smaller yet highly effective networks are obtained. The use of the linear approximation of ℓ_0-norm allows the achievement of sparsity induction results similar to the exponential approximation, but with reductions in the mathematical and computational complexity of the regularization during training. The proposed compression schemes also involve the use of ℓ_2-norm regularization to prevent overfitting, as well as fine tuning to improve the performance of the pruned network. Experimental results are presented aiming to demonstrate the effectiveness of the proposed schemes, as well as to make comparisons with competing approaches.

Keywords: Deep Neural Networks. Complexity Reduction. ℓ_0-norm Regularization.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

A importância das redes neurais em aprendizagem de máquina e aplicações de engenharia têm crescido substancialmente nas últimas décadas [1–3]. Alguns dos fatores que contribuem para esse crescimento incluem os avanços significativos em algoritmos de treinamento e a disponibilidade crescente de plataformas computacionais de alta capacidade, as quais têm permitido a implementação de redes neurais com elevado número de camadas internas. Tais redes, conhecidas como redes neurais profundas (DNN - *Deep Neural Networks*), têm obtido êxito ao tratar um elevado número de problemas complexos, tais como predições para mercados de ações [4–6], análise de sentimentos [7, 8], processamento de linguagem natural (NLP - *Natural Language Processing*) [9, 10], reconhecimento de fala [11–13], e processamento de imagem [14–16].

Embora sistemas com elevada capacidade computacional estejam amplamente disponíveis atualmente, os custos computacionais ainda impõe grandes desafios para o uso de redes neurais profundas em aplicações práticas. Um dos principais motivos para tais desafios é a elevada quantidade de parâmetros (pesos ou coeficientes) necessários às redes com desempenho satisfatório. Um exemplo em particular e nem tão recente de rede neural que enfrenta esse tipo de problema é a AlexNet (vencedora do *2012 Imagenet Large Scale Visual Recognition Challenge*) [16], a qual possui mais de 60 milhões de pesos treináveis. A aplicação prática de redes neurais dessa ordem de complexidade exige, tanto em seu treinamento quanto para inferência em si, o uso de plataformas com elevada capacidade de processamento e alta disponibilidade de energia [17]. Como resultado, tem-se grandes dificuldades para a sua utilização em diversos cenários práticos, sobretudo em ambientes com recursos limitados, como aplicações que envolvem processamento em tempo real e/ou em sistemas embarcados. Particularmente nestes últimos, o uso de redes neurais profundas frequentemente resulta em tempos de inferência elevados, esgotamento rápido de bateria e consumo de memória excessivo [17, 18].

Além de produzir um esgotamento de recursos computacionais, a elevada complexidade de redes neurais atuais traz implicações ambientais significativas. Como mencionado anteriormente, o treinamento dessas redes e o seu posterior uso para fins de inferência exigem alto consumo de energia, o que traz impactos negativos ao meio ambiente [19–21]. Esse custo ambiental é particularmente crítico em grandes *data centers*, onde as demandas de refrigeração e a operação contínua aumentam ainda mais o consumo total de energia elétrica [22].

Com o objetivo de contornar tais problemas, consideráveis esforços de pesquisa têm sido dedicados ao desenvolvimento de técnicas de compressão de redes neurais. Ao reduzir o número de pesos e também as operações matemáticas envolvidas, as

técnicas de compressão diminuem diretamente a carga computacional, viabilizando o uso das redes neurais em uma gama maior de aplicações. Além disso, a consequente redução no consumo de energia resulta na redução de emissões associadas de gases de efeito estufa, além de prolongar a vida útil do *hardware* ao mitigar o estresse térmico e a utilização de recursos, o que ajuda também a reduzir a geração de lixo eletrônico [23].

Diferentes abordagens vêm sendo utilizadas para a compressão de redes neurais profundas. Algumas dessas abordagens baseiam-se no uso de decomposições matriciais para encontrar aproximações adequadas da rede com complexidade computacional reduzida [24–27]. Outras abordagens exploram características de implementação de *hardware* para obter reduções no custo computacional [28–30]. Além disso, abordagens baseadas em poda de neurônios ou pesos também vem sendo consideradas para o desenvolvimento de técnicas de compressão de redes [31–36]. Especificamente no contexto de compressão por meio de poda, estratégias que utilizam regularização baseada em penalização por normas vetoriais vêm ganhando atenção significativa [37–40]. Isso se deve ao fato de que tais técnicas de regularização tendem a induzir esparsidade na rede, o que potencialmente aumenta a eficácia de uma poda subsequente de pesos ou neurônios.

1.1 OBJETIVOS

A presente tese de doutorado tem como foco central o desenvolvimento de novas estratégias para compressão de redes neurais, técnicas essas baseadas na regularização por norma seguida pela poda de pesos da rede. Mais especificamente, a ideia é empregar a norma ℓ_0 em substituição às normas ℓ_1 ou ℓ_2 mais comumente utilizadas, uma vez que a penalização imposta pela norma ℓ_0 tende a induzir esparsidade de forma mais eficaz, conforme discutido em [41]. Como consequência, os pesos irrelevantes da rede tendem a ser mais facilmente identificados e posteriormente eliminados, resultando em modelos esparsos que demandam significativamente menos recursos computacionais durante a etapa de inferência.

Diferentes formas de regularização com norma ℓ_0 já foram aplicadas à compressão de redes neurais [39, 42–45], cada uma adotando abordagens distintas para contornar as dificuldades inerentes à não diferenciabilidade da função de norma ℓ_0. No presente trabalho, adota-se inicialmente uma aproximação exponencial, discutida em [46], para viabilizar a utilização da norma ℓ_0. Tal aproximação torna a função custo diferenciável, possibilitando o uso de métodos clássicos baseados no método do gradiente descendente para o treinamento da rede. Adicionalmente, propõe-se uma regularização combinada ℓ_2-ℓ_0, na qual a penalização pela norma ℓ_2 atua na prevenção do *overfitting*, enquanto a penalização pela norma ℓ_0 é responsável por induzir esparsidade. Vale ressaltar que a proposta difere de abordagens anteriores, como a

de [45], ao atribuir papéis distintos às penalizações envolvidas, buscando uma separação distinta entre os mecanismos de controle de complexidade e de compressão estrutural.

Assim, em resumo, o objetivo central do presente trabalho é estabelecer um novo esquema de treinamento e compressão de redes neurais que una simplicidade computacional, via aproximação diferenciável da norma ℓ_0, e eficácia na indução de esparsidade. Dessa forma, busca-se otimizar o desempenho de estratégias de poda de pesos para gerar arquiteturas compactas que apresentem baixo custo operacional e alta velocidade de resposta no momento da inferência, sem comprometer a acurácia do sistema.

Cabe destacar que a área de compressão de redes neurais profundas caracteriza-se por uma dinâmica de pesquisa acelerada, com elevada quantidade de novas publicações e técnicas sendo introduzidas continuamente. Diante dessa dinâmica, o acompanhamento exaustivo de cada nova variante de algoritmo torna-se um importante desafio. Para contornar essa dificuldade e garantir uma fundamentação teórica sólida, esta tese apoia-se em trabalhos que consolidam o estado da arte e oferecem visões estruturadas do campo [47–50]. Tais referências permitem situar as contribuições originais deste trabalho dentro de um referencial consolidado.

1.2 ORGANIZAÇÃO DO DOCUMENTO

Este trabalho de tese está organizado conforme descrito a seguir. O Capítulo 2 estabelece a fundamentação teórica necessária para o desenvolvimento do trabalho. Nesse capítulo, são abordadas as arquiteturas de redes neurais e o processo de aprendizagem, destacando o papel do algoritmo do gradiente descendente, do algoritmo *backpropagation*, da generalização e das estratégias de regularização. O Capítulo 3 discute as técnicas de compressão de modelos neurais complexos. Nessa discussão, destaca-se o uso de regularizações baseadas em normas vetoriais para induzir esparsidade nas redes, definindo o contexto para a pesquisa realizada. A primeira contribuição original desta tese é apresentada no Capítulo 4. Esse capítulo aborda então o uso da norma ℓ_0 para a regularização do treinamento de redes neurais, explorando as características de indução de esparsidade proporcionadas por tal norma. A partir disso, é desenvolvido um esquema de regularização por normas ℓ_2 e ℓ_0 capaz de atuar simultaneamente na redução do *overfitting* e na redução da complexidade da rede. Resultados experimentais comparam a complexidade e o desempenho das redes treinadas com o esquema proposto em relação a outras abordagens da literatura. O Capítulo 5 propõe o uso de uma forma alternativa de aproximação para a norma ℓ_0 no contexto de regularização do treinamento de redes neurais. Essa aproximação resulta em uma formulação linearizada, com consequente redução da complexidade computacional e matemática durante o treinamento. Ao final, resultados experimentais

comparam a complexidade e a acurácia obtidas utilizando as aproximações exponencial e linearizada da norma ℓ_0. Finalmente, o Capítulo 6 apresenta o sumário e a discussão dos resultados obtidos, além de considerações finais e propostas para trabalhos futuros.

1.3 PUBLICAÇÃO DA TESE

Como resultado direto das investigações realizadas nesta tese, o seguinte artigo científico foi publicado em periódico internacional:

1. F. D. de Resende Oliveira, E. L. O. Batista, and R. Seara, “On the compression of neural networks using ℓ_0-norm regularization and weight pruning,” Neural Networks, vol. 171, pp. 343–352, 2024

Cabe ressaltar que a estratégia proposta neste trabalho tem despertado algum interesse da comunidade científica, servindo de base para estudos independentes. Recentemente, em [51], a metodologia aqui desenvolvida (referenciada pelos autores como *Differentiable Relaxation of* ℓ_0 *Regularization*) foi utilizada como referência em uma análise comparativa com demais trabalhos da literatura. Naquele estudo, o excelente desempenho de tal metodologia foi verificado em novos experimentos independentes. Além disso, os autores destacaram a eficácia da estratégia proposta nesta tese, classificando-a como uma das abordagens de melhor desempenho dentre as diversas técnicas de compressão avaliadas.

2 FUNDAMENTAÇÃO TEÓRICA

Com suas diversificadas arquiteturas, as redes neurais artificiais formam a base de muitos sistemas modernos de aprendizagem de máquina. Este capítulo apresenta os fundamentos teóricos para a compreensão de tais redes, os quais são organizados em três eixos principais: (i) as arquiteturas básicas de redes completamente conectadas (Seção 2.2.1) e convolucionais (Seção 2.2.2), que formam a base para a construção de sistemas mais complexos; (ii) os princípios do processo de aprendizagem, incluindo o algoritmo de gradiente descendente (Seção 2.3.1) e o algoritmo *backpropagation* (Seção 2.3.2); e (iii) os conceitos fundamentais de generalização (Seção 2.4) e as principais técnicas de regularização (Seção 2.5), que garantem o bom desempenho desses modelos em problemas reais.

2.1 HISTÓRIA E CONTEXTUALIZAÇÃO

Embora os fundamentos teóricos de redes neurais artificiais remontem ao modelo de McCulloch e Pitts (1943) [52], foi com o Perceptron de Rosenblatt (1958) que surgiu o primeiro modelo treinável de neurônio artificial [53], base para muitas das arquiteturas utilizadas até os dias atuais [14, 54, 55]. A Figura 1 apresenta uma linha do tempo com os principais marcos históricos que delinearam a evolução dessas redes, com ênfase nas arquiteturas profundas do tipo perceptron multicamadas (MLP, *multilayer perceptron*) MLP e redes neurais convolucionais (*convolutional neural networks* – CNN).

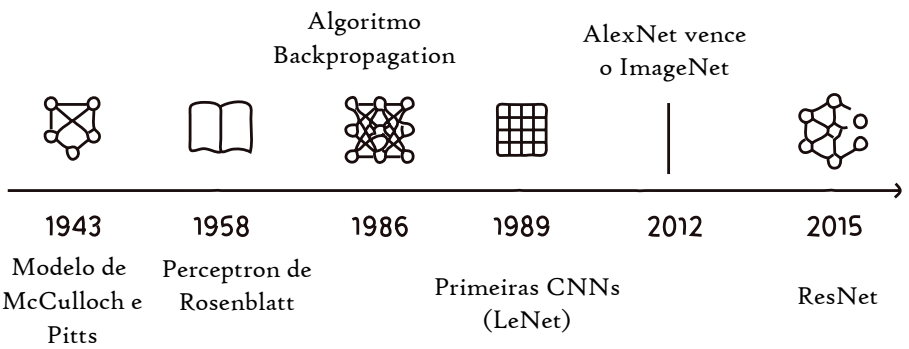


Figura 1 – Linha do tempo: Redes Neurais Artificiais MLPs e CNNs.

O perceptron, introduzido por Rosenblatt [53], constitui um modelo de processamento que recebe múltiplos sinais de entrada e os combina por meio de uma soma ponderada. O resultado dessa combinação é submetido a uma função de ativação não linear, usualmente uma função limiar (*threshold*), que é ativada somente quando sua entrada excede um valor predefinido. Dessa forma, o perceptron é parametrizado pelos pesos da soma ponderada, que controlam a força das conexões, e pelo *bias*, que estabelece o ponto de ativação da função limiar. Originalmente, a função dos pesos é

análoga à das sinapses do modelo do neurônio biológico, controlando a importância das conexões entre os neurônios artificiais. Atualmente, os neurônios artificiais em redes neurais são baseados no perceptron, sendo a principal diferença a adoção de outras funções de ativação (e.g., sigmoide, tangente hiperbólica, ou unidade linear retificada – ReLU) que possuem uma derivada bem-definida. Essa propriedade, fundamental para o cálculo do gradiente em todas as regiões, viabiliza o treinamento eficiente das redes neurais profundas via o algoritmo *backpropagation* [1].

O diagrama representativo de um neurônio artificial é apresentado na Figura 2. A partir dessa figura, tem-se que a relação de entrada e saída desse neurônio artificial pode ser definida por

$$y(n) = f_\sigma \left[\sum_{i=1}^{N} w_i x_i(n) - \theta \right] \tag{1}$$

onde w_i é o peso associado a entrada $x_i(n)$, $f_\sigma[\cdot]$ é a função de ativação do neurônio, que está relacionada com a não linearidade do neurônio artificial, e θ é o *bias*.

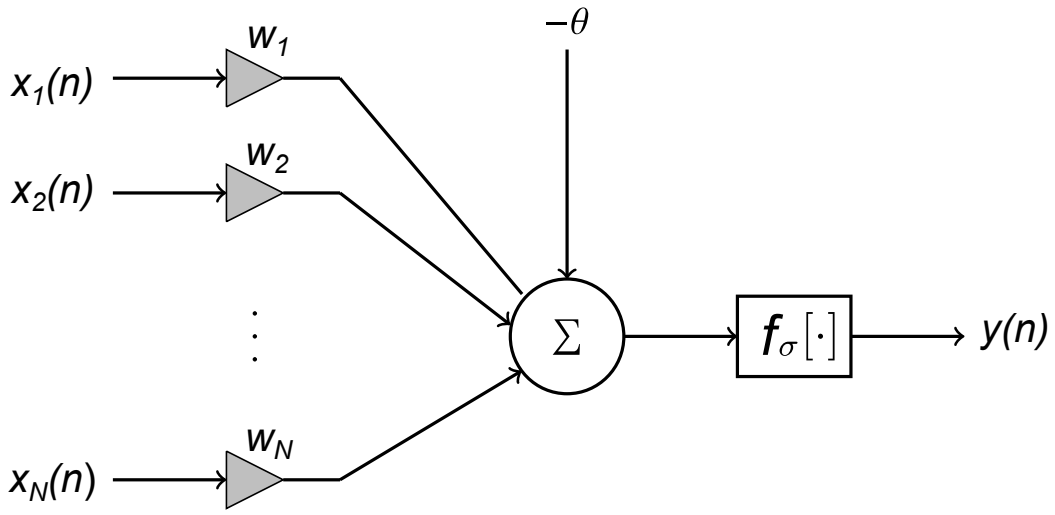


Figura 2 – Modelo de um neurônio artificial.

Ainda que o trabalho de Rosenblatt tenha servido de base para diversas pesquisas na época de sua publicação [56–59], a demonstração de que um *perceptron* de camada única era incapaz de modelar tarefas de separação não linear impôs grandes limitantes à sua aplicação. A prova dessa limitação, formalizada em [60], desencadeou um período de estagnação na pesquisa de redes neurais [1,61]. Posteriormente, com a comprovação da viabilidade prática do uso do algoritmo *backpropagation* para o treinamento de redes com múltiplas camadas compostas por neurônios artificais [54], houve um ressurgimento do interesse da comunidade acadêmica no estudo de redes neurais artificiais. Assim, emergiram as aplicações envolvendo funções mais complexas, tais como reconhecimento de padrões simples, regressão estatística e classificação de dígitos manuscritos [31]. No entanto, persistiam restrições conceituais quanto à profundidade ideal dessas redes, sendo amplamente aceito na época que redes neurais

com apenas duas camadas ocultas eram suficientes para capturar qualquer relação funcional relevante nos dados [62,63]. Como consequência, para problemas mais complexos, como visão computacional avançada e modelagem de linguagem natural, as redes neurais da época ainda apresentavam desempenho inferior a métodos estatísticos tradicionais. Além disso, a complexidade computacional exigida para treinamento de redes neurais profundas impôs desafios significativos ao uso dessas redes em problemas mais complexos. Como resultado, até o início dos anos 2000, a maioria das aplicações práticas ainda utilizava redes relativamente rasas.

Foi apenas com o advento de novas técnicas de treinamento, como inicialização pré-treinada (*pretraining*) e funções de ativação como ReLU, além do aumento do poder computacional com unidades gráficas de processamento (GPUs $-$ *graphical processing unit*), é que as redes neurais profundas começaram a demonstrar seu verdadeiro potencial. Isso levou ao desenvolvimento do aprendizado profundo (*deep learning*) a partir da década de 2010, com arquiteturas como as CNNs [16] e redes recorrentes aprimoradas [64].

2.2 REDES NEURAIS ARTIFICIAIS: DEFINIÇÕES GERAIS

As redes neurais artificiais podem ser classificadas como redes conectadas diretamente (*feedfoward connected networks*) ou redes recorrentes. A informação nas redes conectadas diretamente segue um fluxo contínuo da entrada para a saída, enquanto as redes recorrentes possuem laços de realimentação [1–3]. As discussões neste trabalho de pesquisa são focadas nas redes conectadas diretamente, as quais ocupam um espaço de destaque em aplicações de aprendizagem de máquina, tendo sua utilização consagrada em uma ampla variedade de tarefas.

De maneira geral, as redes neurais conectadas diretamente são algoritmos de aprendizagem de máquina modelados na forma de uma função $f(\mathbf{X}, \mathbf{\Theta})$, a qual recebe um tensor $\mathbf{X}$ como entrada e têm como objetivo estimar um certo valor ou um conjunto de valores, representados aqui por em um tensor de saída $\mathbf{Y}$ correspondente. Nesse contexto, os pesos ou coeficientes da rede, representados pelo tensor $\mathbf{\Theta}$, são ajustados de forma que o modelo da rede definido por $f(\cdot, \mathbf{\Theta})$ alcance a melhor aproximação possível, considerando os dados de treinamento disponíveis e um certo algoritmo de aprendizagem.

A organização das redes diretamente conectadas se dá tipicamente na forma de sucessivas camadas e, portanto, a função $f(\mathbf{X}, \mathbf{\Theta})$ pode ser tratada como um conjunto de funções aplicadas em sequência. Dessa forma, é possível modelar uma rede neural de três camadas como

$$f(\mathbf{X}, \mathbf{\Theta}) = f_3\left\{f_2\left[f_1(\mathbf{X}, \mathbf{\Theta_1}), \mathbf{\Theta_2}\right], \mathbf{\Theta_3}\right\} \tag{2}$$

onde $f_1(\cdot, \mathbf{\Theta_1})$ representa a função correspondente à primeira camada da rede (com

tensor de coeficientes dado por $\mathbf{\Theta_1}$), $f_2(\cdot, \mathbf{\Theta_2})$ corresponde à segunda camada (tensor de coeficientes dado por $\mathbf{\Theta_2}$) e $f_3(\cdot, \mathbf{\Theta_3})$ correspondente à terceira camada (com tensor de coeficientes dado por $\mathbf{\Theta_3}$). Cada uma dessas funções opera sobre um tensor de entrada da camada, calculando uma resposta ou tensor de saída. Assim, considerando que $\mathbf{X}$ é o tensor de entrada da primeira camada, temos sucessivamente $\mathbf{Y}_1 = \mathbf{X}_2 = f_1(\mathbf{X}, \mathbf{\Theta_1})$ como o tensor de saída da primeira camada (de entrada da segunda camada), $\mathbf{Y}_2 = \mathbf{X}_3 = f_2[\mathbf{X}_2, \mathbf{\Theta_2}]$ como o tensor de saída da segunda camada (de entrada para a terceira camada), e finalmente $\mathbf{Y} = f(\mathbf{X}, \mathbf{\Theta}) = \mathbf{Y}_3 = f_3[\mathbf{X}_3, \mathbf{\Theta_3}]$ como o tensor de saída da rede.

Dependendo da sua posição, as camadas de uma rede neural são classificadas como de entrada, ocultas ou de saída. Usualmente temos uma única camada de entrada e uma única camada de saída, além de múltiplas camadas ocultas conforme os requisitos da aplicação. O número total de camadas da rede define sua profundidade, sendo que as redes com elevado número de camadas são conhecidas como DNNs.

As camadas de uma rede neural podem realizar diferentes operações, dependendo de sua função específica. Exemplos de tipos de camadas incluem: camadas densas (totalmente conectadas), camadas convolucionais, camadas de *pooling* e camadas de *flattening* [1,61]. Por simplicidade, as próximas seções vão discutir dois tipos de redes, que recebem seus nomes em função do tipo predominante de camada que as compõem. Nessas discussões, os diferentes tipos de camadas serão discutidos conforme necessário.

2.2.1 Redes Neurais Completamente Conectadas

O modelo clássico das redes neurais é o das redes completamente conectadas, também conhecidas como MLPs. Nessas redes, a relação entre entrada e saída da k-ésima camada é tipicamente dada por

$$\mathbf{x}_{k+1} = f_k(\mathbf{x}_k, \mathbf{W}_k, \mathbf{b}_k) = f_\sigma(\mathbf{W}_k\mathbf{x}_k - \mathbf{b}_k) \tag{3}$$

onde $\mathbf{x}_k$ e $\mathbf{x}_{k+1}$ representam, respectivamente, os vetores de entrada ($m_{k-1} \times 1$) e saída ($m_k \times 1$) da camada k. Os parâmetros treináveis da camada compõem a matriz de pesos $\mathbf{W}_k$ (com dimensões $m_k \times m_{k-1}$) e o vetor de *bias* $\mathbf{b}_k$ ($m_k \times 1$). A função $f_\sigma(\cdot)$ representa a função de ativação não linear, sendo exemplos comuns a ReLU, sigmoide ou tangente hiperbólica. Nessa notação, o número de neurônios da k-ésima camada é dado por m_k, enquanto o número de neurônios da camada anterior é dado por m_{k-1}. A função de ativação não linear é aplicada elemento a elemento, ou seja, $f_\sigma(\cdot)$ é aplicada a cada componente do vetor resultante de $\mathbf{W}_k\mathbf{x}_k - \mathbf{b}_k$. Para fins de visualização, uma rede neural completamente conectada com três camadas, com 4, 4 e 2 neurônios, respectivamente, é ilustrada na Figura 3.

Note que a operação realizada pela camada totalmente conectada exige que a entrada seja organizada na forma de um vetor. Caso a entrada possua múltiplas dimensões (e.g., uma matriz representando uma imagem), aplica-se previamente a operação de *flattening* para reorganizá-la de forma vetorial.

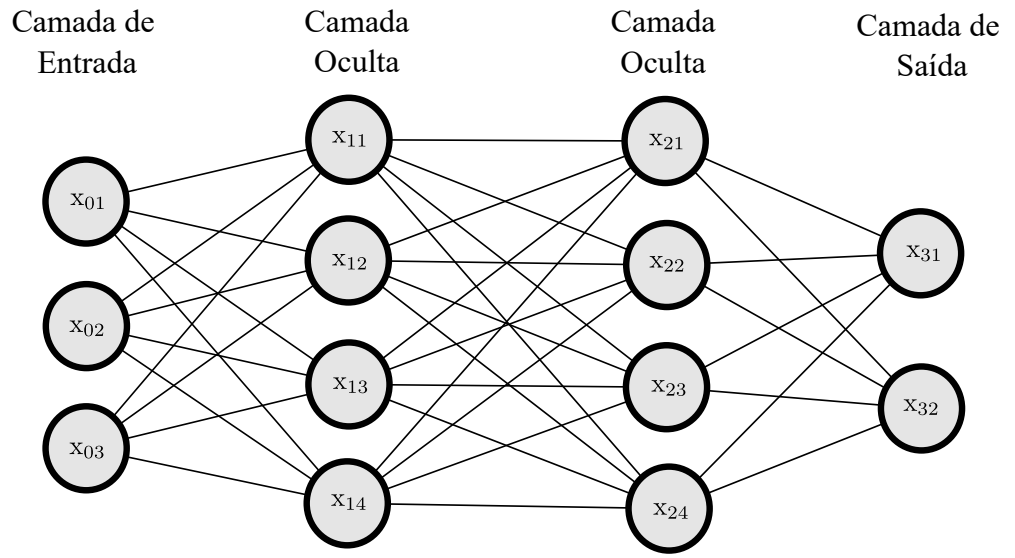


Figura 3 – Rede Neural Totalmente Conectada.

2.2.2 Redes Convolucionais

As CNNs consolidaram-se como arquitetura dominante em visão computacional a partir de 2012, graças ao desempenho alcançado em desafios como o ImageNet [16]. Essas arquiteturas são caracterizadas por camadas convolucionais especializadas no processamento de dados com estrutura topológica (e.g., imagens), onde operações de convolução discreta extraem características hierárquicas e localmente invariantes [1]. Cada camada convolucional consiste em *k kernels* (filtros) aprendíveis, cujas dimensões espaciais são inferiores às do tensor de entrada. Cada um desses *kernels* varre o tensor de entrada de forma iterativa, deslocando-se com passos (*strides*) fixos e computando produtos internos locais que geram ativações no mapa de características de saída. A Figura 4 ilustra esse processo para uma matriz de entrada de dimensões 3×4 e um *kernel* 2×2, destacando como a dimensionalidade da saída é determinada pelos *hiperparâmetros* de convolução (*stride* e tamanho do *kernel*).

As CNNs exigem uma quantidade significativamente menor de parâmetros em comparação a camadas totalmente conectadas equivalentes. Essa eficiência paramétrica decorre diretamente de duas propriedades fundamentais: conectividade esparsa e compartilhamento de pesos [1]. Enquanto uma camada totalmente conectada com entrada m-dimensional e saída n-dimensional requer $n \times m$ pesos (além de n *biases*), uma camada convolucional opera através de *kernels* de tamanho fixo (tipicamente $r \times r$ com $r \ll m$) que compartilham os mesmos parâmetros ao longo de toda a extensão da

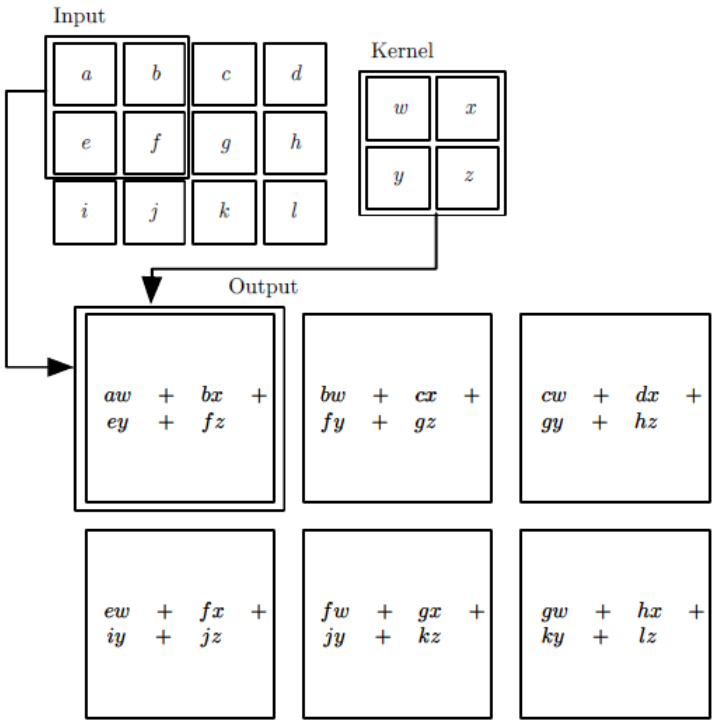


Figura 4 – Exemplo de convolução 2D sem o uso de *kernel clipping*.

entrada (compartilhamento de pesos). Essa arquitetura, combinada com a varredura local (*sliding window*), reduz a complexidade paramétrica para apenas $k \cdot r^2$ pesos (onde k é o número de filtros), ao mesmo tempo que preserva a capacidade de extrair características hierárquicas: desde bordas e texturas básicas (nas primeiras camadas) até padrões semânticos complexos (em camadas profundas) [1].

A topologia de implementação de redes convolucionais ainda permite o processamento eficiente de entradas com dimensões variáveis, pois os *kernels* operam localmente sem depender do tamanho global da entrada. Dessa forma, a convolução via *sliding window* viabiliza o processamento direto de tensores com variações dimensionais (e.g., imagens 256×256 ou 512×512), dispensando operações custosas de ajuste de tamanho. Para reforçar a invariância a pequenas perturbações na entrada, camadas de *pooling* são frequentemente inseridas após as convoluções [1]. A operação de *max pooling*, por exemplo, resume uma região retangular ao seu valor máximo, descartando informações redundantes e reduzindo dimensionalidade espacial. Outro fator que potencializa a redução de consumo de memória para a implementação de redes neurais convolucionais é o compartilhamento de parâmetros. Mais especificamente, a operação de convolução de um mesmo *kernel* é percorrida para cada posição do tensor de entrada [1], de modo que os mesmos pesos são reutilizados para calcular a saída de diferentes neurônios da camada seguinte. Essas propriedades fundamentais (conectividade esparsa e compartilhamento de parâmetros) são integradas em uma arquitetura híbrida característica das CNNs modernas. Enquanto as camadas convolucionais iniciais processam eficientemente os dados de entrada através de operações

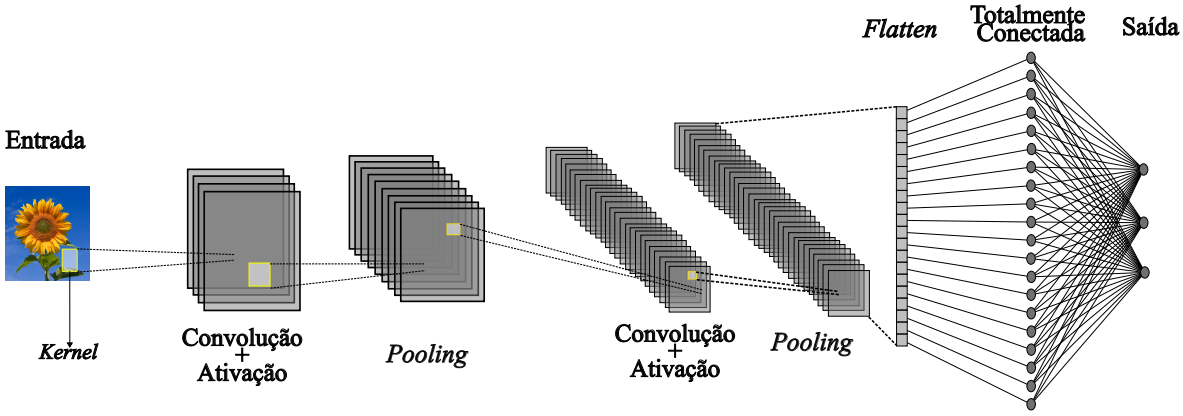


Figura 5 – Estrutura típica de uma CNN, destacando a extração hierárquica de características e a classificação por camadas totalmente conectadas.

locais e compartilhadas, a transição para camadas totalmente conectadas na fase final da rede é realizada por meio de uma operação de *flattenning* (em uma camada do tipo *flatten*). Tal transformação reorganiza os mapas de características multidimensionais, resultantes das sucessivas camadas convolucionais e de *pooling*, em um vetor unidimensional, permitindo que as camadas densas posteriores sintetizem as características hierárquicas extraídas para a tarefa de classificação ou regressão. Como ilustrado na Figura 5, a estrutura combinando diferentes tipos de operações das redes convolucionais permite aproveitar as vantagens computacionais das convoluções (eficiência espacial e redução paramétrica) com a capacidade discriminativa das redes totalmente conectadas, estabelecendo o padrão arquitetural dominante em aplicações de visão computacional [16].

2.3 TREINAMENTO DE REDES NEURAIS

O treinamento de redes neurais se divide em duas abordagens principais: a supervisionada e a não supervisionada. Na aprendizagem supervisionada, os pesos da rede são ajustados através da minimização do erro em relação a saídas desejáveis ou conhecidas, enquanto na não supervisionada a rede deve extrair características e padrões diretamente da estrutura dos dados, sem referência a rótulos externos. Esses dois paradigmas exploram capacidades complementares das arquiteturas de redes neurais, desde o ajuste preciso de mapeamentos entrada-saída até a descoberta autônoma de representações úteis dos dados [2].

Embora a aprendizagem não supervisionada em redes neurais seja fundamental para explorar estruturas intrínsecas dos dados, seu êxito varia significativamente entre aplicações. Essa abordagem mostra-se particularmente eficaz em tarefas de agrupamento automático (*clustering*) [65] e em tarefas de redução de dimensionalidade, com arquiteturas como *autoencoders* variacionais (VAEs) [66]. Por outro lado, a aprendizagem não supervisionada tem limitações em tarefas que demandam ma-

peamentos entrada-saída precisos, como tradução automática ou classificação de imagens, em que a ausência de rótulos dificulta a correlação entre características abstratas e objetivos específicos [67].

As redes neurais baseadas em aprendizagem supervisionada em geral obtêm êxito em aplicações onde a relação entrada-saída é bem definida e onde há disponibilidade de quantidade significativa de dados rotulados para treinamento. Por exemplo, no campo de visão computacional, conjuntos de dados amplos e com dados rotulados têm sido construídos, facilitando o desenvolvimento de soluções para classificação de imagens baseadas em apredizagem supervisionada [16]. O mesmo acontece em outras áreas de aplicação do aprendizado de máquina, e assim as soluções baseadas em aprendizagem supervisionada têm predominado em diversas aplicações práticas [1, 61].

2.3.1 Aprendizagem Baseada no Gradiente Descendente

O foco do presente trabalho está na aprendizagem supervisionada, em função da sua predominância em aplicações envolvendo redes neurais [68]. Nesse contexto, o treinamento de redes neurais, e em especial o das profundas, tipicamente envolve problemas de otimização não-convexos decorrentes da alta dimensionalidade e não-linearidade das funções envolvidas. Tal complexidade costuma resultar em múltiplos mínimos locais e pontos de sela na função custo, exigindo métodos iterativos como o do gradiente descendente (e suas variantes) para obtenção de uma solução adequada. Assim, o processo de treinamento tipicamente envolve a atualização iterativa dos parâmetros da rede na direção oposta à do gradiente da função custo $\nabla_\theta J(\theta)$ em relação aos pesos ou parâmetros da rede. Essa abordagem, embora eficiente, enfrenta desafios como convergência para mínimos subótimos ou oscilações em regiões de curvatura irregular [1, 2].

Outro desafio para o uso do algoritmo do gradiente descendente em sua forma convencional para o treinamento de redes neurais está ligado à complexidade computacional envolvida. Mais especificamente, o cálculo do gradiente considerando *datasets* amplos e complexos resulta em alto custo em termos de operações matemáticas e também de uso de memória, comprometendo a escalabilidade do treinamento [1]. Assim, variações do algoritmo de gradiente descendente vêm sendo preferidas, como o gradiente descendente estocástico (SGD), o qual tem se tornado método padrão em aplicações práticas. Em tal método, os parâmetros da rede são atualizados usando subconjuntos aleatórios (*mini-batches*) dos dados. Como consequência, o SGD aproxima o gradiente verdadeiro por uma estimativa ruidosa, reduzindo o custo por iteração [1]. Essa abordagem não apenas viabiliza o treinamento de grandes redes em conjuntos massivos de dados, mas também introduz ruído estocástico que ajuda a escapar de mínimos locais rasos [69]. Cada atualização do vetor de coeficientes da rede neural

utilizando a SGD pode ser representada da seguinte maneira:

$$\mathbf{w}^{+} \leftarrow \mathbf{w} - \eta \tilde{\nabla}_{\mathbf{w}} J(\mathbf{w}; \mathbf{x}, \mathbf{y}), \tag{4}$$

com $J(\mathbf{w}; \mathbf{x}, \mathbf{y})$ representando a função custo utilizada, a qual depende dos parâmetros da rede, e dos tensores de entrada e de saída, $\tilde{\nabla}_{\mathbf{w}}$ é a estimativa do gradiente em função dos parâmetros da rede, e η é a taxa de aprendizagem do algoritmo.

Diversos fatores influenciam a eficácia do treinamento de redes neurais profundas utilizando variações do gradiente descendente. Por exemplo, uma inicialização apropriada dos parâmetros da rede é bastante importante. Conforme mencionado em [1], para redes MLP, uma regra geral é inicializar os pesos da rede com valores aleatórios e de baixa magnitude, enquanto os *bias* devem ser inicializados com zero ou com valores positivos próximos de zero. Além disso, a escolha dos hiperparâmetros usados no treinamento, como o fator ou passo de aprendizagem η, influencia tanto a acurácia final obtida pela rede quanto a velocidade de convergência no treinamento. Nesse contexto, diversos outros métodos adaptativos têm ganhado importância, como por exemplo o Adam [70], o qual envolve o ajuste da taxa de aprendizado com base em estimativas de momentos de primeira e segunda ordem do gradiente.

2.3.2 Algoritmo *Backpropagation*

Um dos desafios para aplicação do algoritmo SGD no treinamento de redes neurais profundas está relacionado com a composição em múltiplas camadas não lineares das redes neurais modernas, composição essa que dificulta o cálculo do gradiente da função custo em relação aos pesos da rede. Para viabilizar esse cálculo de forma eficiente, o treinamento de redes neurais profundas é realizado com o auxílio do algoritmo *backpropagation*. Nesse algoritmo, o gradiente da função custo com respeito ao pesos da rede é computado de maneira estruturada, utilizando a regra da cadeia da derivada em duas fases: (i) a fase direta, na qual os dados de entrada percorrem a rede no sentido entrada-saída, armazenando-se os valores intermediários de ativação; e (ii) a fase de retropropagação (*backpropagation*), na qual os gradientes são propagados no sentido reverso, da saída para a entrada da rede, aplicando-se a regra da cadeia de forma eficiente e reutilizando derivadas parciais previamente calculadas [1]. Essa abordagem permite que o SGD, e outros métodos derivados, seja aplicado de forma viável ao treinamento de arquiteturas profundas, reduzindo tanto a complexidade computacional quanto o uso de memória ao longo das iterações de otimização.

2.4 GENERALIZAÇÃO E REPRESENTAÇÃO EM APRENDIZAGEM DE MÁQUINA

A elevada taxa de acertos de uma rede neural no processamento de dados vistos pelo algoritmo durante o processo de treinamento não é o principal parâmetro

adotado para avaliar seu desempenho. De fato, o grande desafio é garantir desempenho adequado considerando dados de interesse que ainda não tenham sido vistos. Essa característica de algoritmos de aprendizagem de máquina é usualmente denotada na literatura como capacidade de generalização [1]. Uma prática comum que permite avaliar o comportamento do modelo treinado quando submetido a dados ainda não vistos é separar os dados disponíveis para o treinamento em dois grupos, um para o treinamento em si e outro para teste ou avaliação de desempenho. É relevante comentar que, nos casos em que a quantidade de dados disponível para o treinamento da rede não é suficientemente grande, dividir os dados disponíveis em grupos de treinamento e de teste pode ser inviável. Para esses casos, existem na literatura algumas estratégias para avaliar o desempenho dos algoritmos de aprendizagem de máquina quando os dados disponíveis são insuficientes, como é o caso das técnicas de validação cruzada [71].

Sabendo que o desempenho do algoritmo para dados ainda não vistos (conjunto de teste) é usualmente inferior ao desempenho obtido no conjunto de treinamento, garantir um desempenho satisfatório na fase de treinamento é essencial para que a rede neural opere adequadamente com dados novos. Nos casos em que a rede não atingiu valores de acurácia pretendidos nem mesmo para os dados de treinamento, diz-se que ocorreu o *underfitting*. Além do *underfitting*, outro desafio encontrado em aprendizagem de máquina é o *overfitting*, que se relaciona com a diferença entre o erro de treinamento e o erro de teste. Mais especificamente, diz-se que o modelo sofreu *overfitting* quando há uma diferença significativa na acurácia obtida com esses dois tipos de conjuntos de dados. É possível controlar se o modelo estará favorável a sofrer *underfitting* ou *overfiting* a partir de sua capacidade de representação [1].

2.4.1 Capacidade de Representação em Aprendizagem de Máquina

A capacidade de representação de um modelo de aprendizagem de máquina relaciona-se com os tipos de funções às quais ele é capaz de se adaptar. Modelos de baixa capacidade de representação são os incapazes de representar as funções necessárias para o bom desempenho do algoritmo no conjunto de treinamento, resultando em baixa acurácia. Em contraste, modelos de alta capacidade de representação tendem a aprender propriedades muito específicas do conjunto de treinamento que não são encontradas no conjunto de teste. Como consequência, o modelo torna-se especialista em processar os dados do conjunto de treinamento mas não é capaz de ter um bom desempenho no conjunto de teste, perdendo sua capacidade de generalização.

Existem diversas maneiras de se controlar a capacidade de representação de um modelo de aprendizagem de máquina. Uma alternativa para tal é escolher diretamente o espaço de hipóteses que o modelo é capaz de operar, isto é, especificar o conjunto de funções que o algoritmo de aprendizagem pode escolher como solução.

Por exemplo, em um problema de regressão linear, onde a predição do algoritmo é definida matematicamente por

$$\hat{y} = \mathbf{w}^T \mathbf{x}, \tag{5}$$

com $\mathbf{w}$ representando o vetor de parâmetros *n* dimensional e $\mathbf{x}$ o vetor de amostras, o único tipo de função que o modelo descrito por (5) é capaz de escolher como solução é a linear. Esse tipo de comportamento pode ser bastante restritivo em algumas situações, resultando em predições com baixa acurácia. Uma alternativa para incrementar a capacidade do modelo em questão é aumentar o espaço de hipóteses do modelo para um espaço de hipóteses com uma não linearidade (quadrática, por exemplo) em função da entrada, isto é, considerar que a predição do modelo pode ser definida por

$$\hat{y} = \mathbf{w}_1^T \mathbf{x} + \mathbf{x}^T \mathbf{W}_2^T \mathbf{x} \tag{6}$$

onde $\mathbf{w}_1^T$ e $\mathbf{W}_2^T$ são, respectivamente, os parâmetros relacionados aos componentes linear e de ordem quadrática em função dos dados.

Aumentar o espaço de hipóteses do modelo não significa necessariamente que o algoritmo terá melhor desempenho. De modo geral, quanto mais a capacidade de representação do modelo se aproximar da complexidade da tarefa que será realizada, melhor o desempenho do algoritmo [1]. O campo da teoria do aprendizado estatístico traz algumas métricas para verificar a capacidade dos modelos de aprendizagem, o que permite estabelecer importantes predições quantitativas sobre o modelo [1]. A mais importante dessas predições mostra que a diferença entre os erros de treinamento e de teste é limitado por um valor que cresce em função da capacidade do modelo e diminui com o aumento da quantidade de amostras usadas no treinamento. Esse resultado justifica a questão de que o aumento de capacidade tende a resultar em modelos com baixa generalização. Contudo, vale comentar que as predições de capacidade do modelo definida pelo campo do aprendizado estatístico são raramente utilizadas no contexto de aprendizagem de máquina profunda. Nesse caso, o problema de otimização que busca a solução ótima do modelo não é convexo e, como não há um amplo entendimento sobre esse tipo de problema, as predições realizadas não são amplamente aceitas dentre os estudiosos da área.

2.5 TÉCNICAS DE REGULARIZAÇÃO

No problema de regressão utilizado no exemplo da Seção 2.4.1, a especificação do espaço de hipóteses do modelo de aprendizagem de máquina se deu a partir do grau polinomial em que a saída depende da entrada. Esse tipo de especificação não é a única possibilidade de definir a capacidade representacional do modelo e, em alguns casos, pode ser bastante restritiva com respeito à quantidade de funções representáveis. Nesse contexto e considerando também que é interessante limitar um pouco

a capacidade de representação do modelo visando previnir a ocorrência de *overfitting*. É comum adicionar ao processo de aprendizagem métodos informativos sobre quais tipos de funções são preferidas. Definir preferências sobre determinados tipos de funções é um método mais geral e menos restritivo para controlar a capacidade do modelo do que definir diretamente seu espaço de hipóteses [1]. Essa estratégia viabiliza o aprendizado das funções pretendidas durante o processo de treinamento e, além disso, não exclui a possibilidade de o algoritmo convergir para as funções menos pretendidas quando os resultados com o conjunto de treinamento forem significativamente superiores do que se consideradas as funções pretendidas. Há na literatura diversos métodos de expressar preferências sobre os tipos de funções escolhidas para o modelo, sendo que tais métodos são comumente conhecidos como métodos de regularização. Mais especificamente, "*regularização é qualquer modificação feita em um algoritmo de aprendizado que se destina a reduzir seu erro de generalização, mas não seu erro de treinamento*" [1].

Atualmente, encontra-se disponível um vasto conjunto de técnicas de regularização que vem sendo utilizado com sucesso no contexto de DNN, como é o caso da técnica de *dropout*, da *early-stopping* e das baseadas em penalizações do vetor de parâmetros. A seguir, as principais estratégias de regularização utilizadas no contexto de redes neurais são brevemente descritas.

2.5.1 *Dataset Augmentation*

Os problemas de *overfitting* envolvidos no treinamento de redes neurais artificiais geralmente são relacionados ao fato de que o tamanho do *dataset* de treinamento não é suficientemente grande para atender à complexidade do modelo. Uma alternativa para solucionar tal problema é diminuir a complexidade da rede neural buscando a redução no *overfitting*. Contudo, tal estratégia pode limitar o desempenho da rede a valores de acurácia inferiores do que o desejado tanto no contexto dos dados de treinamento quanto dos de teste, resultando em *underfitting*. Para contornar tal problema pode-se utilizar uma quantidade superior de dados no treinamento da rede. Como a maioria das aplicações têm seus correspondentes *datasets* de treinamento limitados, uma possível alternativa para potencializar a capacidade de generalização da rede é o uso de *dataset augmentation*, que consiste em gerar novos dados para o treinamento da rede utilizando o conjunto de dados original [1]. Um exemplo claro em que tal estratégia é eficaz é o reconhecimento de objetos em imagem, onde uma possível imagem pode facilmente ser utilizada para gerar diversas réplicas a serem utilizadas no treinamento. Técnicas de processamento de imagem comumente utilizadas na geração de tais réplicas são a operação de rotacionamento e escalamento de imagens.

2.5.2 *Early Stopping*

No treinamento de redes neurais profundas, um importante hiperparâmetro que deve ser configurado é a quantidade de "épocas" de um determinado *dataset* que será utilizada para o treinamento da rede. Pode-se imaginar que quanto maior for o número de épocas utilizado, melhor será a acurácia da rede. Contudo, é comum que tal comportamento seja válido apenas no contexto dos dados de treinamento, uma vez que com o passar das épocas a rede começa a tornar-se especialista para tratar esse conjunto de dados, isto é, a rede perde a capacidade de generalização e sua acurácia em aplicações envolvendo dados ainda não vistos diminui. O comportamento usual de redes neurais é o de alcançar a máxima acurácia possível para o conjunto de teste da rede – de acordo com o modelo escolhido, que envolve a quantidade de camadas, técnicas de otimização, função custo, funções de ativação, dentre outras características da rede – e, então, diminuir sucessivamente o desempenho conforme mais épocas são usadas no treinamento. Nesse contexto, uma estratégia que pode ser utilizada como forma de potencializar a generalização é parar o treinamento no ponto em que o erro para um conjunto separado de validação começa a aumentar. Tal estratégia é conhecida na literatura como *early stopping* e sua utilização é bastante difundida devido à sua eficácia e facilidade de implementação [1].

2.5.3 *Dropout*

Devido à sua eficácia em prevenir o *overfitting* de redes neurais profundas, associada a um baixo custo de implementação, a regularização por *dropout* é utilizada com grande frequência. Em linhas gerais, o objetivo dessa técnica é combinar a predição de diferentes modelos de redes neurais para um dado *dataset* de testes [72]. Para tal, neurônios de um modelo são aleatoriamente desativados (juntamente com as suas conexões) durante o treinamento criando uma série de sub-redes com capacidade de generalização inferior à rede original. Na etapa de inferência, todos os neurônios da rede neural são ativados para calcular a predição da rede.

2.5.4 Regularização via Penalização dos Parâmetros da Rede

Dentre as regularizações que utilizam a norma dos parâmetros como forma de penalização, as baseadas em norma ℓ_2 e ℓ_1 são as mais comuns. De modo geral, tais regularizações têm como objetivo limitar a capacidade de representação da rede para prevenir a ocorrência de *overfitting* e consistem em acrescentar um termo de penalização $\Omega(w)$ à função custo $J(w; X, y)$, a qual é minimizada pelo algoritmo de otimização ou treinamento. Mais especificamente, a função custo original $J(w; X, y)$ é redefinida de acordo com

$$\tilde{J}(\mathbf{w}; \mathbf{X}, \mathbf{y}) = J(\mathbf{w}; \mathbf{X}, \mathbf{y}) + \alpha\Omega(\mathbf{w}). \tag{7}$$

onde, $\mathbf{w}$ representa um vetor de parâmetros da rede, $\mathbf{X}$ e $\mathbf{y}$ são dados utilizados no treinamento, e α é um hiperparâmetro que indica a importância do termo de penalização $\Omega(.)$ no processo de atualização dos parâmetros. No contexto de redes neurais profundas, a minimização da função custo é frequentemente realizada por algoritmos como o SGD, em que a atualização dos parâmetros é feita dando passos no sentido oposto ao do gradiente da função custo em relação aos parâmetros do problema. Assim, a equação de atualização dos pesos da rede, considerando a função custo dada em (7), é definida como

$$\mathbf{w}^{+} \leftarrow \mathbf{w} - \eta\nabla_{\mathbf{w}} \left[J(\mathbf{w}; \mathbf{X}, \mathbf{y}) + \alpha\Omega(\mathbf{w}) \right]. \tag{8}$$

onde η representa a taxa de aprendizagem do algoritmo e ∇ o operador gradiente. Note em (8) que o acréscimo do termo de penalização pode ser avaliado à parte no processo de atualização dos parâmetros, uma vez que o gradiente é uma função linear.

3 COMPRESSÃO DE REDES PROFUNDAS COM AUXÍLIO DE TÉCNICAS DE REGULARIZAÇÃO POR NORMAS VETORIAIS

Apesar do notável sucesso alcançado pelas DNNs modernas, o seu elevado número de pesos ou parâmetros aprendíveis impõe demandas significativas em termos de plataformas computacionais necessárias para treinamento e inferência. Essas demandas são ainda mais significativas em aplicações envolvendo processamento em tempo real e/ou em sistemas embarcados, nas quais existem restrições de tempo de processamento e/ou de recursos computacionais. Para enfrentar tais desafios, diversos estudos vêm se dedicando ao desenvolvimento de técnicas de compressão de DNNs.

Uma das formas de se buscar a compressão de DNNs é por meio de decomposições matriciais ou tensorais, que geram aproximações da rede com complexidade computacional reduzida [24–27]. Uma técnica interessante, nesse contexto, é a decomposição em posto reduzido, que combina fatoração e esparsidade estruturada para diminuir o número de pesos, especialmente em camadas totalmente conectadas [73]. O processo consiste em aplicar a decomposição em balores singulares (SVD — *singular value decomposition*) ou métodos similares para fatorar a matriz de pesos. Subsequentemente, as linhas e colunas correspondentes aos neurônios menos relevantes são eliminadas. Preservando-se os componentes mais significativos, alcança-se uma compressão substancial, muitas vezes sem exigir um retreinamento extensivo do modelo [74–76]. Abordagens modernas integram essa otimização diretamente na fase de treinamento. Nelas, aplicam-se penalizações aos tensores e matrizes decompostas, forçando sua convergência gradual para uma estrutura de posto reduzido e, assim, determinando os parâmetros de posto ótimos para cada camada [77].

Outras técnicas de compressão de DNNs aproveitam as especificidades do *hardware* para obter reduções no custo computacional, visando otimizar a eficiência em dispositivos específicos [28–30, 78, 79]. A quantização dos parâmetros da rede, por exemplo, é uma estratégia que proporciona ganhos significativos na inferência de modelos complexos em sistemas com recursos computacionais e de memória limitados [49, 78]. Um desafio inerente a essa abordagem é a perda de precisão que a rede sofre após a quantização. Para mitigar esse problema, propõem-se tanto o retreinamento dos modelos já quantizados [78] quanto estratégias que, durante o treinamento, simulam a aritmética de ponto fixo do *hardware* para reduzir a degradação da acurácia [79]. Adicionalmente, a quantização pode ser combinada com outras estratégias de compressão, alcançando níveis ainda mais elevados de otimização para a inferência [49].

Neste cenário de busca por maior eficiência e menor custo computacional para DNNs modernas, estratégias de poda destacam-se como proeminentes e amplamente

investigadas para a compressão de redes [37, 80, 81]. Em contraste com a decomposição de matrizes, que envolve reestruturação dos pesos, e da quantização, que envolve redução de precisão, a poda concentra-se na eliminação de conexões e/ou neurônios considerados de menor relevância para o desempenho do modelo, o que resulta em uma arquitetura mais esparsa e eficiente [31–36]. Essa abordagem proporciona uma redução substancial do número de parâmetros e operações (FLOPS - *floating point operations per second*), fator crucial para a implementação de DNNs em ambientes com restrições computacionais e energéticas.

3.1 COMPRESSÃO DE REDES NEURAIS VIA ESTRATÉGIAS DE PODA

Os primeiros trabalhos que exploraram a poda para reduzir a complexidade de redes neurais com impacto mínimo em seu desempenho remontam à década de 1990 [32,82]. O critério mais comum para essa tarefa é de reduzir a magnitude absoluta do peso, segundo o qual conexões com valores menores são consideradas menos importantes e, consequentemente, são removidas [17, 80]. Por sua simplicidade e eficácia, essa abordagem é a base de muitos métodos de poda, especialmente os não estruturados. Alternativamente, os critérios podem se basear na sensibilidade do peso à função de erro [82] ou em métricas que avaliam a contribuição das conexões para a ativação de neurônios [83, 84]. Quanto à granularidade, a poda pode ser global, onde um único limiar de esparsidade ou magnitude é definido para toda a rede [17]. Por outro lado, a abordagem por camada (*per-layer*) estabelece um limiar de poda específico para cada camada, permitindo um controle mais fino da compressão [17]. Outras variações incluem a poda aleatória [85, 86], que remove pesos de forma estocástica e serve principalmente como linha de base comparativa, sendo raramente usada na prática devido à sua imprevisibilidade.

Outra forma de classificar as técnicas de poda é em termos da abordagem de remoção utilizada, resultando em técnicas não estruturadas e estruturadas [49, 81, 87]. A poda não estruturada envolve a remoção de pesos individuais, independentemente de sua localização ou de como afetam a estrutura geral da rede [32, 82, 88–90]. A rede neural resultante da aplicação desse método é, em geral, esparsa, conforme ilustrado na Figura 6(a). Quando todas as conexões de entrada e saída de um neurônio são eliminadas, o próprio neurônio é efetivamente podado, como observado na Figura 6(b). Por outro lado, a poda estruturada remove grupos inteiros de pesos, como filtros, canais ou neurônios completos, o que resulta em uma rede mais regular e compatível com arquiteturas de *hardware* otimizadas, facilitando a inferência acelerada [91–93]. Adicionalmente, abordagens de poda estruturada avaliam a importância de neurônios ou canais inteiros utilizando métricas de relevância derivadas de suas ativações ou gradientes, visando remover componentes completos com baixo impacto no desempenho [84]. Existe também a poda semi-estruturada, que busca um equilí-

brio entre as duas abordagens anteriores, removendo pesos em padrões específicos, como blocos esparsos, para otimizar tanto a taxa de compressão quanto a eficiência de hardware [48, 94].

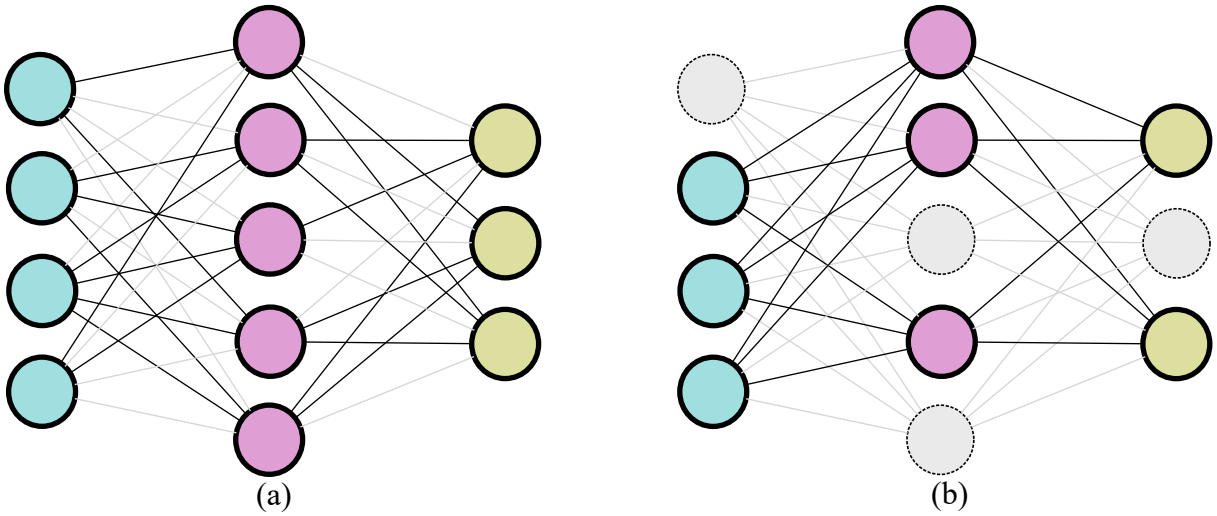


Figura 6 – Rede neural após o processo de poda: (a) pesos removidos e (b) neurônios desativados.

Embora a compatibilidade da poda estruturada com as arquiteturas de *hardware* convencionais, como as GPUs, seja mais evidente devido à sua otimização para operações densas, a poda não estruturada apresenta uma vantagem significativa: a capacidade de atingir níveis de esparsidade substancialmente mais altos com mínima degradação de desempenho [17]. Essa granularidade fina na remoção de pesos permite explorar redundâncias intrínsecas que as abordagens estruturadas não conseguem capturar, resultando em modelos de menor tamanho e menor consumo de energia. Contudo, o processamento eficiente de tais estruturas altamente esparsas e irregulares é um desafio para GPUs tradicionais, que enfrentam gargalos devido aos acessos à memória dispersos e à subutilização de seus núcleos densos. Nesse contexto, plataformas de hardware com características que permitem explorar eficientemente a esparsidade irregular emergem como alternativas promissoras. Tais plataformas, como as FPGAs (*Field-Programmable Gate Arrays*) e aceleradores de hardware customizados, destacam-se pela sua capacidade de reconfiguração e paralelismo. Isso possibilita o desenvolvimento de arquiteturas de processamento personalizadas, capazes de lidar com a irregularidade das redes esparsas não estruturadas de forma eficiente. Ao projetar uma arquitetura dedicada à rede esparsa, desconsiderando computações com zeros e armazenando apenas elementos não nulos, essas plataformas podem explorar integralmente a alta esparsidade alcançada pela poda não estruturada. Por exemplo, o trabalho de Wang et al. [95] demonstra a eficácia dessa abordagem, onde um acelerador baseado em FPGA obteve ganhos significativos em desempenho e eficiência energética através de um motor de computação esparsa otimizado. Tal abordagem

viabiliza a implantação desses modelos em ambientes com recursos limitados ou que demandam alta eficiência energética.

3.2 REGULARIZAÇÃO BASEADA EM PENALIZAÇÃO DE NORMAS VETORIAIS E SUA APLICAÇÃO À PODA DE PESOS

Conforme discutido na Seção 2.5.4, as estratégias de regularização que utilizam a penalização dos parâmetros da rede, como as baseadas em normas ℓ_1 e ℓ_2, buscam mitigar o *overfitting* ao acrescentar um termo de penalização à função custo. Essa abordagem, que consiste em redefinir a função custo $J(\mathbf{w};\mathbf{X},\mathbf{y})$ para $\tilde{J}(\mathbf{w};\mathbf{X},\mathbf{y}) = J(\mathbf{w};\mathbf{X},\mathbf{y}) + \alpha\Omega(\mathbf{w})$, permite que uma importante influência seja exercida sobre o treinamento do modelo, de forma a controlar a sua complexidade.

Como mencionado no capítulo anterior, a minimização da função custo envolvida no treinamento de uma rede neural é tipicamente realizada usando métodos baseados no gradiente descendente. Assim, a atualização dos pesos de uma rede neural via minimização da função custo regularizada $\tilde{J}(\mathbf{w};\mathbf{X},\mathbf{y})$ assume a forma de (8). Individualizando essa expressão para o *j*-ésimo peso da rede, tem-se

$$w_j^+ \leftarrow w_j - \eta\frac{\partial\left[J(\mathbf{w},\mathbf{x},\mathbf{y}) + \alpha\Omega(\mathbf{w})\right]}{\partial w_j} \tag{9}$$

onde η é a taxa de aprendizagem. Note que, para $\alpha = 0$, (9) torna-se a equação de atualização de pesos padrão (não-regularizada), a qual é dada por

$$w_j^+ \leftarrow w_j - \eta\frac{\partial J(\mathbf{w},\mathbf{x},\mathbf{y})}{\partial w_j}. \tag{10}$$

3.2.1 Regularização baseada na norma ℓ_2

Uma das técnicas de regularização baseada em norma vetorial mais populares baseia-se no uso da norma ℓ_2 dos parâmetros da rede como função de penalização. Tal técnica é desenvolvida fazendo

$$\Omega(\mathbf{w}) = ||\mathbf{w}||_2^2 = \sum_{j\in\phi} w_j^2 \tag{11}$$

em (7) [1], com $||\mathbf{w}||_2$ representando a norma ℓ_2 do vetor de pesos $\mathbf{w}$, e ϕ representando o conjunto de todos os parâmetros penalizados da rede. Substituindo (11) em (8), considerando que

$$\frac{\partial||\mathbf{w}||_2^2}{\partial w_j} = 2w_j \tag{12}$$

e manipulando a expressão resultante, obtemos a seguinte equação de atualização:

$$w_j^+ \leftarrow w_j - 2\eta\alpha w_j - \eta\frac{\partial J(\mathbf{w},\mathbf{x},\mathbf{y})}{\partial w_j}. \tag{13}$$

Comparando (13) com (10), pode-se observar que a regularização baseada na norma ℓ_2 introduz um termo adicional dado por $-2\eta\alpha w_j$ na equação de atualização dos parâmetros. Uma vez que $2\eta\alpha$ é sempre positivo e usualmente muito menor do que um (a fim de garantir a convergência do algoritmo), esse termo extra, usualmente denominado termo de penalização, produz uma redução na magnitude dos pesos da rede a cada iteração, o que contribui para aumentar a capacidade de generalização do modelo.

3.2.2 Regularização baseada em norma ℓ_1

No caso da regularização baseada em norma ℓ_1, a função de penalização é dada por [1]

$$\Omega(\mathbf{w}) = ||\mathbf{w}||_1 = \sum_{i \in \phi} |w|_i \tag{14}$$

com $||\mathbf{w}||_1$ representando a norma ℓ_1 do vetor de pesos $\mathbf{w}$. Assim, considerando (8), (14) e também que

$$\frac{\partial ||\mathbf{w}||_1}{\partial w_j} = \text{sign}(w_j) \tag{15}$$

a seguinte equação de atualização dos pesos é obtida:

$$w_j^+ \leftarrow w_j - \eta\alpha\,\text{sign}(w_j) - \eta\frac{\partial J(\mathbf{w}, \mathbf{x}, \mathbf{y})}{\partial w_j}. \tag{16}$$

Note que (16) corresponde a (10) com a inclusão do termo de penalização $-\eta\alpha\,\text{sign}(w_j)$. Uma vez que $0 < \eta\alpha \ll 1$, esse termo também tende a produzir uma redução na magnitude dos parâmetros da rede de um modo similar ao termo de penalização acrescentado à equação de atualização dos parâmetros com regularização de norma ℓ_2.

3.2.3 Compressão de Redes Neurais via Regularização Baseada em Normas Vetoriais e Poda de Pesos

Como mostrado na seção anterior, as regularizações de norma ℓ_2 e ℓ_1 inserem termos de penalização na equação de atualização dos pesos que acabam por induzir reduções de magnitude durante o processo de treinamento. Tal efeito, além de prevenir o *overfitting*, produz uma atração para zero sobre os pesos, sendo que os pesos com menor magnitude tendem a ser os mais afetados. Assim, tem-se uma rede final com características esparsas, ou seja, com um maior número de coeficientes com valores tendendo a zero. Nesse contexto, estratégias que combinam regularizações baseadas em penalização de norma vetorial e poda de redes neurais têm ganhado atenção significativa quando deseja-se comprimir redes neurais complexas [37–40]. Em [37], por exemplo, a associação de regularização baseada na norma ℓ_2 e a poda de pesos levou

a reduções de até 12 vezes no número de pesos da rede sem afetar significativamente seu desempenho. Além disso, em [40], reduções de até 38 vezes foram obtidas ao custo de pequenas perdas de desempenho usando uma combinação de regularização com norma ℓ_1 e poda de pesos. Quando o objetivo é eliminar neurônios, técnicas de regularização em grupo têm levado a resultados promissores [39, 40]. Ainda, como mostrado em [38], é possível combinar penalização de neurônios e de pesos para aumentar a compressão da rede.

3.2.4 Poda de Pesos e Regularização por Norma ℓ_0

Como mencionado anteriormente, as normas ℓ_2 ou ℓ_1 são as mais comumente usadas para fins de regularização no contexto do treinamento das redes neurais, tanto com a finalidade de mitigar o *overfitting* quanto para induzir esparsidade e facilitar a compressão. Especialmente com respeito a indução de esparsidade, uma outra pseudonorma vetorial, a norma ℓ_0[1], também vem obtendo um crescente interesse. Isso se deve ao conhecido fato que a penalização com norma ℓ_0 tipicamente apresenta um efeito mais forte de indução de esparsidade em problemas de otimização, conforme discutido em [41]. É essa a motivação para o uso da norma ℓ_0 no presente trabalho de pesquisa.

Outros trabalhos de pesquisa da literatura têm explorado o uso da norma ℓ_0 para compressão de redes neurais [39, 42–45]. Cada um desses trabalhos adota uma abordagem diferente para lidar com as dificuldades decorrentes do uso da função não diferenciável que caracteriza a norma ℓ_0 no contexto de problemas de otimização via algoritmos baseados no gradiente descendente. Em [39], por exemplo, é apresentado um arcabouço geral para usar funções *surrogate* da norma ℓ_0 na regularização, juntamente com uma aproximação dessa norma chamada *hard-concrete*. A estratégia apresentada em [42] é baseada em [39], substituindo o estimador de gradiente baseado em *hard-concrete* pelo estimador chamado *augment-reinforce-merge* (ARM), o que permite obter resultados superiores. O foco tanto em [39] quanto em [42] é principalmente explorar esparsidade em grupos, visando realizar poda de neurônios. Em [43], a poda de pesos é formulada como um problema de otimização não convexa com restrição, onde a restrição está na cardinalidade dos pesos (ou seja, na norma ℓ_0 do vetor de pesos). O método dos multiplicadores de direção alternada (ADMM $-$ *alternating direction method of multipliers*) é utilizado para resolver sistematicamente esse problema, e a poda é então aplicada removendo os pesos próximos de zero [43]. Em [44], o problema de poda é formulado de duas maneiras: como um problema de otimização com restrição de norma ℓ_0, similar a [43], e também como um problema de otimização da função custo penalizada. Abordagens de otimização alternada são

[1] É importante comentar que por definição a norma ℓ_0 é, na verdade, uma *pseudonorma*. Contudo, para fins de simplificação, ela é denotada neste trabalho também como uma norma específica.

usadas para resolver os problemas obtidos nessas duas formas, resultando em algoritmos de duas etapas para o treinamento de redes com poda de parâmetros [44]. Além disso, em [45], a abordagem introduzida em [44] é utilizada para combinar as normas ℓ_0 e ℓ_2, tratando-as como penalidades ou restrições na formulação da otimização, o que resultou em melhores resultados.

4 COMPRESSÃO DE REDES NEURAIS USANDO REGULARIZAÇÃO BASEADA NA NORMA ℓ_0

Este capítulo é dedicado à contribuição principal do presente trabalho: uma nova abordagem de regularização para o treinamento de redes neurais baseada em regularização por norma ℓ_0. Tal abordagem tem por objetivo explorar as características de indução de esparsidade da norma ℓ_0 para penalizar os parâmetros da rede durante treinamento e assim reduzir a magnitude dos pesos de menor relevância. Em seguida, é apresentada uma discussão sobre as características de indução à esparsidade da abordagem proposta baseada na norma ℓ_0 em comparação com outras abordagens de regularização baseadas em normas. Dessa discussão, surge um esquema que combina regularizações de norma ℓ_2 e ℓ_0, o qual é associado à poda de pesos para dar origem ao esquema de compressão de rede proposto.

4.1 ABORDAGEM DE REGULARIZAÇÃO BASEADA NA NORMA ℓ_0 PROPOSTA

A função conhecida como norma ℓ_0 é uma medida estrita de esparsidade vetorial, uma vez que ela é formalmente definida como o número de coeficientes não nulos em um vetor [39, 46, 96]. Devido a esse fato, quando considerada como termo de penalização em aplicações envolvendo otimização, a norma ℓ_0 apresenta a interessante capacidade de induzir esparsidade no vetor de parâmetros do problema em questão [39, 42, 46, 96, 97]. Essas aplicações compartilham importantes similaridades com as estratégias de compressão de redes neurais previamente discutidas no presente trabalho.

Com o objetivo de desenvolver uma nova abordagem de regularização que resulte em penalização dos pesos durante o treinamento de redes neurais, vamos começar com a definição formal da norma ℓ_0 de um vetor de pesos $\mathbf{w}$, a qual é dada por

$$||\mathbf{w}||_0 = \sum_{j \in \Phi} f(w_j) \tag{17}$$

com

$$f(w_j) = \begin{cases} 0 \text{ se } w_j = 0 \\ 1 \text{ para outros valores.} \end{cases} \tag{18}$$

É importante notar que (17) é uma função não-convexa e não-diferenciável. Essas características impõem dificuldades importantes para o desenvolvimento de estratégias de regularização como as descritas na Seção 2.5.4, especialmente devido ao papel central dos métodos de otimização por gradiente descendente [veja (8)]. Para contornar tal problema, considera-se aqui uma versão aproximada diferenciável da norma ℓ_0, a

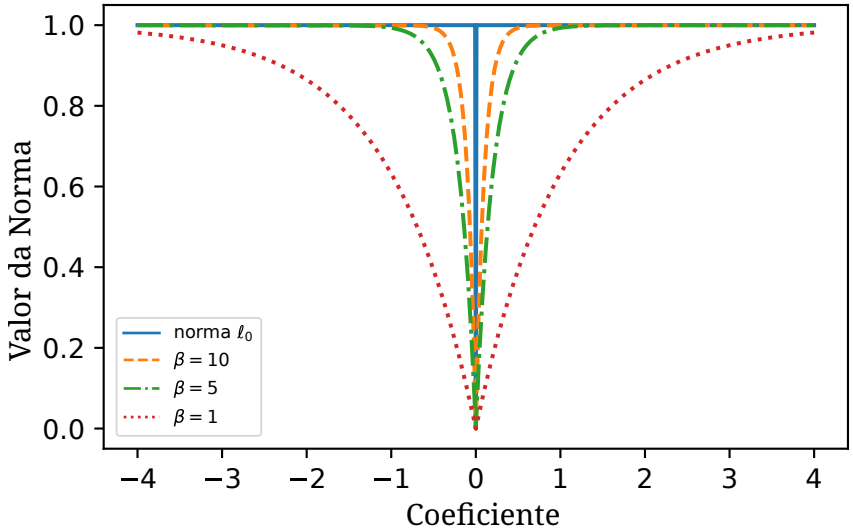


Figura 7 – Comparação entre norma ℓ_0 e versão da norma ℓ_0 aproximada com exponencial para diferentes valores de β.

qual está descrita em [46]. Com isso, (18) é aproximada por

$$\tilde{f}(w_j) = 1 - e^{-\beta|w_j|}, \tag{19}$$

com $\beta \geq 1$. Utilizando agora (19) em substituição a (18) no cálculo de (17), obtém-se a seguinte aproximação da norma ℓ_0:

$$||\mathbf{w}||_0 \cong \sum_{j \in \Phi} \tilde{f}(w_j) = \sum_{j \in \Phi} (1 - e^{-\beta|w_j|}). \tag{20}$$

A Figura 7 mostra uma compoaração da norma ℓ_0 com a sua aproximação dada por (20) para diferentes valores de β. Em tal figura, observa-se que, à medida que β aumenta, a versão aproximada da norma ℓ_0 aproxima-se mais da definição original de (17). Mais precisamente, a faixa de valores do coeficiente em que o valor da norma tende a zero torna-se cada vez mais estreita com aumento de β. Nesse caso, conclui-se que, à medida que $\beta \to \infty$, $||\mathbf{w}||_0^* \to ||\mathbf{w}||_0$.

Agora, visando desenvolver o esquema de regularização baseado na norma ℓ_0 proposto, considera-se a equação de atualização dos pesos da rede dada em (8) e substitui-se $\Omega(\mathbf{w})$ pela aproximação da função de norma ℓ_0 de (20) em tal expressão. Assim, a regra de atualização dos pesos torna-se

$$w_j^+ \leftarrow w_j - \eta\alpha \frac{\partial \tilde{f}(w_j)}{\partial w_j} - \eta \frac{\partial J(\mathbf{w}, \mathbf{X}, \mathbf{y})}{\partial w_j}. \tag{21}$$

Então, tomando o gradiente de (20) com respeito a w_j, obtém-se

$$\frac{\partial \tilde{f}(w_j)}{\partial w_j} = \beta \, \text{sign}(w_j) \, e^{-\beta|w_j|}. \tag{22}$$

Finalmente, substituindo (22) em (21), a seguinte equação de atualização dos pesos baseados em penalização de norma ℓ_0 é obtida:

$$w_j^+ \leftarrow w_j - \eta\alpha\beta\text{sign}(w_j)e^{-\beta|w_j|} - \eta\frac{\partial J(\mathbf{w},\mathbf{X},\mathbf{y})}{\partial w_j}. \tag{23}$$

Observe que (23) corresponde a (10) com a inclusão do termo de penalização dado por $-\eta\alpha\beta\text{sign}(w_j)e^{-\beta|w_j|}$. Como na prática $0 \leq \eta\alpha\beta e^{-\beta|w_j|} \ll 1$, conclui-se que tal termo exerce uma atração para zero sobre o peso, de forma similar aos casos das regularizações por norma ℓ_2 e ℓ_1 discutidas anteriormente neste trabalho.

4.2 ANÁLISE DO EFEITO DE ATRAÇÃO PARA ZERO DAS DIFERENTES ESTRATÉGIAS DE REGULARIZAÇÃO BASEADAS EM NORMAS VETORIAIS

Conforme mencionado na Seção 3.2.3, os termos de penalização presentes em (13) (atualização dos parâmetros com regularização por norma ℓ_2) e (16) (atualização dos parâmetros com regularização por norma ℓ_1) exercem um efeito de atração em direção a zero sobre os pesos da rede durante o treinamento. O mesmo tipo de efeito também é observado para o termo de penalização considerando a regularização com penalização de norma ℓ_0 descrita por (23). As características da indução a zero produzidas por tais termos de penalização são, no entanto, significativamente diferentes entre si. Ao compará-las, pode-se notar que:

i) a penalização por norma ℓ_2 (via o termo de penalização $-2\eta\alpha w_j$) depende de w_j e, portanto, da magnitude do peso, o que resulta em uma atração a zero decrescente à medida que o peso se aproxima de zero;

ii) a penalização por norma ℓ_1 (via o termo de penalização $-\eta\alpha\,\text{sign}(w_j)$) depende do sinal do peso (não de sua magnitude), resultando em um efeito constante de atração a zero;

iii) a penalização baseada na norma ℓ_0 (via o termo $-\eta\alpha\beta\text{sign}(w_j)e^{-\beta|w_j|}$) depende do sinal do peso e também de sua magnitude via $e^{-\beta|w_j|}$, resultando em uma atração a zero que aumenta à medida que o peso se aproxima de zero.

Essas características são evidenciadas pelas curvas mostradas na Figura 8, que descrevem o comportamento de um peso sujeito às distintas penalizações de norma. Essas curvas foram obtidas definindo o valor inicial do peso como $1,0$ e ajustando os parâmetros das diferentes regularizações para obter aproximadamente a mesma taxa de penalização inicial.

Para obter uma compreensão mais aprofundada das diferenças entre as estratégias de regularização baseadas em normas, um aspecto interessante a ser analisado é o escalamento experimentado por um peso após sua atualização devido à atração

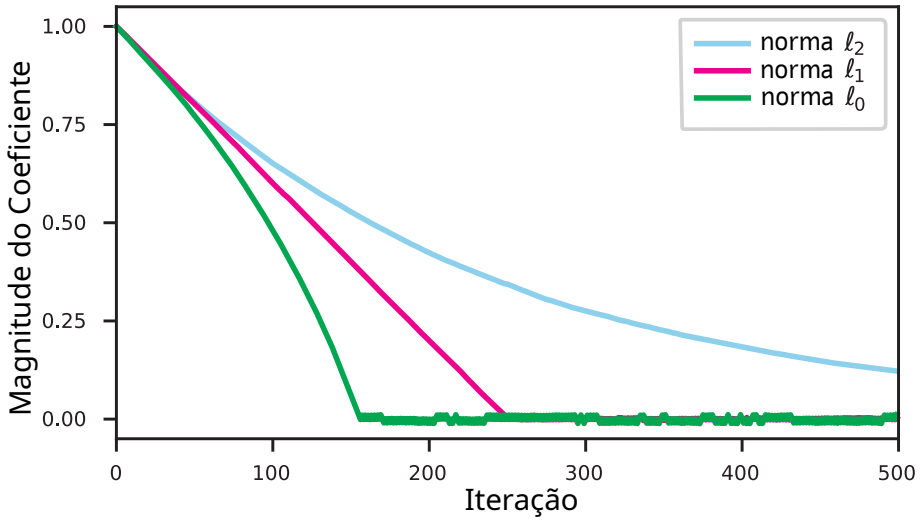


Figura 8 – Comportamento obtido para um peso inicialmente definido como 1, 0 e submetido às diferentes penalizações com taxa de penalização inicial similar.

a zero. Para tanto, definimos um fator de escala de atração a zero de atualização de peso λ_j, que é dado pela razão entre w_j^+ e w_j quando apenas o termo de penalização é considerado para sua atualização [ou seja, ignorando $J(\mathbf{w}, \mathbf{x}, \mathbf{y})$ em (13), (16), ou (23)]. Assim, removendo o último termo do lado direito de (13) e manipulando a expressão resultante, obtém-se o seguinte fator de escala de atração a zero para a regularização baseada na norma ℓ_2:

$$\lambda_j = \frac{w_j^+}{w_j} = 1 - 2\eta\alpha. \tag{24}$$

Agora, para a regularização baseada na norma ℓ_1, removendo o último termo do lado direito de (16), considerando que $\text{sign}(w_j) = \frac{w_j}{|w_j|}$, e manipulando a expressão resultante, obtém-se o seguinte fator de escala de atração a zero:

$$\lambda_j = \frac{w_j^+}{w_j} = 1 - \eta\alpha\frac{1}{|w_j|}. \tag{25}$$

Finalmente, para o caso da regularização proposta baseada na norma ℓ_0 proposta, removendo o último termo do lado direito de (23), considerando que $\text{sign}(w_j) = \frac{w_j}{|w_j|}$, e manipulando a expressão resultante, obtém-se

$$\lambda_j = \frac{w_j^+}{w_j} = 1 - \eta\alpha\beta\frac{e^{-\beta|w_j|}}{|w_j|}. \tag{26}$$

O intervalo de valores de interesse para o fator de escalonamento λ_j depende do propósito da regularização. No caso de mitigação de *overfitting*, λ_j deve ser menor que e próximo de um, pois isso significa que o peso é apenas levemente afetado a cada iteração. Por outro lado, se o objetivo é a poda de pesos, o fator de escalamento ideal deve ser próximo de zero (redução mais forte da magnitude do peso), contudo apenas para os pesos a serem podados. Em termos gerais, se $-1 < \lambda_j < 1$, a magnitude do

peso é reduzida. Tal intervalo aceitável para λ_j pode ser usado para estabelecer limites para a escolha do termo de penalização α. Por exemplo, substituindo em $-1 < \lambda_j < 1$ e manipulando a expressão resultante, obtém-se os seguintes limites de α para a regularização baseada na norma ℓ_2:

$$0 < \alpha < \frac{1}{\eta}. \tag{27}$$

Fazendo o mesmo procedimento para a regularização baseada na norma ℓ_1, utilizando (25) em $-1 < \lambda_j < 1$ e manipulando a expressão resultante, obtêm-se os seguintes limites de α:

$$0 < \alpha < \frac{2|w_j|}{\eta}. \tag{28}$$

Finalmente, para o caso da regularização proposta baseada na norma ℓ_0, utilizando (26) em $-1 < \lambda_j < 1$ e manipulando a expressão resultante, obtêm-se

$$0 < \alpha < \frac{2|w_j|}{\eta\beta e^{-\beta|w_j|}}. \tag{29}$$

Ao analisar os limites superiores em (27), (28), e (29), pode-se notar que, tanto para as regularizações baseadas na norma ℓ_1 quanto na norma ℓ_0, o limite superior para α que garante a redução da magnitude do peso é diretamente dependente da magnitude do peso $|w_j|$. Consequentemente, à medida que o peso é atraído para zero, o limite superior diminui progressivamente até ser violado. Neste ponto, teremos $|\lambda_j| > 1$ e um crescimento resultante da magnitude do peso, correspondendo a uma repulsão a zero em vez de atração. O crescimento do peso resultará em aumentar o limite superior de α, revertendo a repulsão a zero em atração, iniciando o processo novamente em direção à repulsão. Tal alternância entre atração e repulsão resultará em oscilações no valor do peso no entorno de zero, que serão adicionadas às oscilações ruidosas causadas pelo próprio algoritmo de treinamento após a convergência dos pesos (ruído do gradiente). Essas oscilações podem, eventualmente, impedir que a magnitude do peso seja mantida abaixo do limiar de poda, resultando na poda de um número menor de pesos. Para mitigar esse problema, α deve ser definido para valores relativamente pequenos, levando em conta também o limiar de poda adotado.

Agora, considerando a discussão acima e o resumo, apresentado na Tabela 1, das expressões discutidas nesta seção, algumas conclusões importantes podem ser tiradas em relação às diferentes estratégias de regularização baseadas em normas, bem como às suas capacidades de atração a zero. Considerando primeiro a regularização da norma ℓ_2, da Tabela 1 podemos notar que seu limite de α para redução da magnitude do peso é independente da magnitude do peso. Consequentemente, o peso será sempre atraído para zero sem apresentar as oscilações mencionadas causadas pela atração a zero. No entanto, uma vez que a força de tal atração se torna mais fraca à medida que o peso se torna pequeno, uma evolução exponencial suave do peso em

Tabela 1 – Características de Atração a Zero para as Diferentes Regularizações Baseadas em Normas

	reg. norma ℓ_2	**reg. norma ℓ_1**	**reg. norma ℓ_0 (proposta)**
Termo de penalização	$-2\eta\alpha w_j$	$-\eta\alpha\,\text{sign}(w_j)$	$-\eta\alpha\beta\text{sign}(w_j)e^{-\beta\lvert w_j\rvert}$
Fator de Escala ($\lambda_j = w_j^+/w_j$)	$1-2\eta\alpha$	$1-\frac{\eta\alpha}{\lvert w_j\rvert}$	$1-\frac{\eta\alpha\beta e^{-\beta\lvert w_j\rvert}}{\lvert w_j\rvert}$
Limites de α para decaimento de pesos	$\alpha<\frac{1}{\eta}$	$\alpha<\frac{2\lvert w_j\rvert}{\eta}$	$\alpha<\frac{2\lvert w_j\rvert}{\eta\beta e^{-\beta\lvert w_j\rvert}}$

direção a zero será observada, conforme ilustrado na Figura 8. Como consequência, são necessárias muitas épocas de treinamento para que o peso seja reduzido abaixo do limiar de poda do correspondente peso. Em resumo, a atração a zero fornecida pela regularização da norma ℓ_2 será muito interessante para evitar o *overfitting*, mas não tão relevante para a poda de pesos.

A regularização por norma ℓ_1, por sua vez, não apresenta o problema de atração a zero evanescente da regularização por norma ℓ_2, uma vez que seu termo de penalização não depende da magnitude do peso (veja Tabela 1). No entanto, a regularização por norma ℓ_1 apresenta as oscilações mencionadas causadas pela violação do limite de α, o que exigirá que α seja mantido com valores relativamente pequenos. Esse fato prejudicará inevitavelmente a capacidade da regularização por norma ℓ_1 de evitar o *overfitting*, uma vez que a penalização aplicada a pesos maiores é significativamente limitada por um α pequeno. Em outras palavras, a penalização por norma ℓ_1 tem um problema de objetivos conflitantes, pois α pode ser definido tanto para evitar o *overfitting* quanto para favorecer a atração a zero visando a poda de pesos.

O primeiro aspecto a se destacar em relação à regularização baseada na norma ℓ_0 proposta está relacionado à dependência direta de seu termo de penalização em $e^{-\beta|w_j|}$ (veja Tabela 1). Visto que quando $|w_j| \to \infty$ e $e^{-\beta|w_j|} \to 0$, constatando-se que a penalização tende a se tornar muito pequena para pesos maiores. Assim, a regularização por norma ℓ_0 proposta tende a ser menos eficaz na prevenção do *overfitting*. Por outro lado, em termos de atração a zero para os coeficientes menores, a regularização por norma ℓ_0 tem um efeito mais forte e que aumenta à medida que o peso se torna menor. Essa característica é ilustrada na Figura 9 para $\eta = 0,04$ e $\alpha = 0,01$, bem como para diferentes valores de β. Como resultado, durante o treinamento da rede, um peso que se torne pequeno é cada vez mais acelerado em direção a zero, levando um número menor de épocas para atingir o limiar de poda de peso. Esse comportamento pode ser claramente observado na curva correspondente à penalização da norma ℓ_0

mostrada na Figura 8. Em termos de oscilações devido à atração a zero, a regularização baseada na norma ℓ_0 também apresentará tais oscilações, que se devem à dependência direta em $|w_j|$ de seu limite de α para redução da magnitude do peso. Esse aspecto exigirá que α seja mantido com um valor relativamente pequeno, sem grande impacto sobre os pesos maiores, uma vez que estes não são afetados pela regularização baseada na norma ℓ_0.

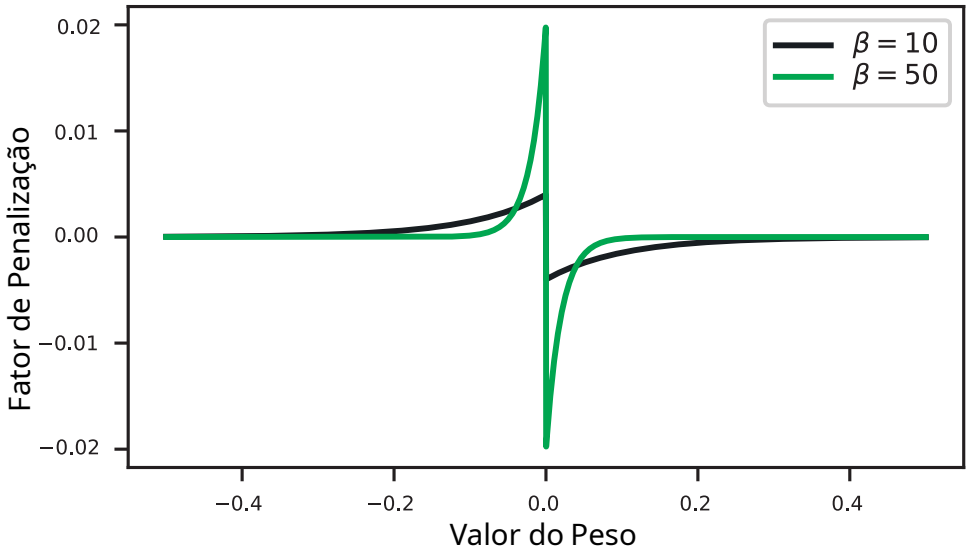


Figura 9 – Penalização obtida com regularização ℓ_0 proposta para dois valores de β.

4.3 ESQUEMA DE COMPRESSÃO DE REDES BASEADO EM REGULARIZAÇÃO COMBINADA POR NORMAS ℓ_2 E ℓ_0

Da discussão apresentada na seção anterior, fica claro que a abordagem de regularização baseada na norma ℓ_0 proposta é bastante apropriada para a obtenção de redes com maior número de pesos com baixa magnitude, embora apresente uma capacidade limitada de evitar o *overfitting*. Em contraste, a regularização da norma ℓ_2 é eficaz na prevenção do *overfitting*, mas não tanto eficiente para a compressão da rede. Assim, propõe-se aqui a exploração de tal complementaridade entre as regularizações baseadas nas normas ℓ_2 e ℓ_0 para obter redes com alta capacidade de compressão que também sejam eficazes em termos de desempenho de inferência. Nesse contexto, considera-se então a seguinte função de penalização combinada:

$$\Omega(\mathbf{w}) = ||\mathbf{w}||_2^2 + ||\mathbf{w}||_{\tilde{0}}. \tag{30}$$

Seguindo passos similares aos descritos na Seção 4.1 para derivar a estratégia de regularização proposta da norma ℓ_0, obtém-se a seguinte regra de atualização de pesos para a regularização combinada:

$$w_j \leftarrow w_j - 2\eta\alpha_{\ell_2} w_j - \eta\alpha_{\ell_0}\beta\text{sign}(w_j)e^{-\beta|w_j|} - \eta\frac{\partial J(\mathbf{w};\mathbf{X},\mathbf{y})}{\partial w_j}. \tag{31}$$

Note que dois parâmetros diferentes de penalização de norma (α_{ℓ_2} e α_{ℓ_0}) são utilizados em (31) para permitir o controle independente do nível de penalização para cada norma. Este aspecto é de grande interesse, pois, como mencionado anteriormente, cada penalização de norma tem uma tarefa diferente no algoritmo de treinamento proposto. Assim, ajustando α_{ℓ_2}, pode-se controlar a força da prevenção do *overfitting*, enquanto ajustando α_{ℓ_0}, a força da atração a zero é controlada. Especificamente em relação a α_{ℓ_0}, tal parâmetro de fato permite controlar o compromisso entre a perda de acurácia e o nível de compressão, o que é típico em técnicas de compressão de rede. Neste contexto, utilizando valores maiores de α_{ℓ_0}, a compressão é favorecida em detrimento da manutenção do nível de acurácia da rede não comprimida, enquanto valores menores de α_{ℓ_0} levam a níveis de compressão menores, mantendo a acurácia original.

A partir de (31), um esquema para compressão de redes neurais é então proposto, o qual é baseado no treinamento de rede utilizando a regularização das normas ℓ_2-ℓ_0 combinadas seguida pela poda dos pesos menos significativos da rede resultante. O esquema proposto é concebido tendo em mente a poda não estruturada (de pesos), uma vez que a penalização produz uma atração a zero para cada peso independentemente. Apesar disso, a poda estruturada (de neurônios) também pode ser alcançada através de um *design* inteligente da estratégia de poda proposta.

4.4 RESULTADOS EXPERIMENTAIS

Esta seção é dedicada à avaliação, de forma experimental, das estratégias propostas no presente trabalho. Nesse contexto, inicialmente é feita uma avaliação experimental da capacidade de indução de esparsidade da regularização baseada na norma ℓ_0 proposta. Em seguida, avalia-se o desempenho do esquema de compressão de rede proposto baseado na regularização das normas ℓ_2-ℓ_0 combinadas. Em todos os experimentos conduzidos nesta tese, a poda é realizada em uma etapa subsequente ao treinamento, após a convergência dos parâmetros sob a influência do esquema de regularização proposto.

De forma geral, seis experimentos diferentes são considerados. No primeiro, a ideia é comparar a capacidade de indução de esparsidade da regularização proposta baseada na norma ℓ_0 com as regularizações mais comuns baseadas nas normas ℓ_2 e ℓ_1, bem como avaliar o seu impacto quando distintas estratégias de poda são utilizadas. No segundo experimento, o esquema de compressão de rede proposto é usado para obter versões comprimidas das redes Lenet-300-100 e Lenet-5-Caffe aplicadas ao conjunto de dados MNIST. O terceiro experimento envolve a aplicação do esquema de compressão de rede proposto para a tarefa de comprimir a rede VGG-16 e também uma rede residual chamada ResNet20, ambas aplicadas ao conjunto de dados CIFAR-10. No quarto experimento, o foco é na compressão da rede VGG-16 aplicada à

tarefa mais desafiadora do CIFAR-100. O quinto experimento envolve também a tarefa desafiadora do CIFAR-100, mas agora utilizando a rede ResNet-50. Finalmente, no sexto experimento, o uso da estratégia proposta em um contexto de poda estrutural (remoção de neurônios completos) é avaliado. Em todos os experimentos, as redes são avaliadas em termos do compromisso entre acurácia e taxa de compressão, esta última definida como a razão entre o número total de parâmetros da rede original e o número de parâmetros remanescentes após a poda. Para a seleção dos pesos a serem removidos, adotam-se dois critérios conforme o objetivo de cada teste: i) a poda por limiar de magnitude; e ii) a poda por taxa de compressão alvo. No critério por limiar de magnitude, estabelece-se um limiar mínimo para a permanência do peso (ou o limiar de poda), o que permite identificar o potencial máximo de compressão para fins de comparação com a literatura. Já no critério por taxa de compressão alvo, a remoção é guiada por uma meta de compressão preestabelecida, permitindo o mapeamento progressivo da curva de desempenho da rede e a visualização da característica de acurácia *versus* taxa de compressão.

4.4.1 Experimento 1: Comparação de Indução de Esparsidade entre Diferentes Estratégias de Regularização Baseadas em Normas

O objetivo deste experimento é avaliar a capacidade de indução de esparsidade da regularização proposta baseada na norma ℓ_0 em comparação com as regularizações baseadas nas normas ℓ_2 e ℓ_1. Assim, redes neurais inspiradas na Lenet-5-Caffe[1] são treinadas para a tarefa de classificação de imagens do conjunto de dados MNIST. Para avaliar o compromisso entre e taxa de compressão da rede, o critério de poda por taxa de compressão alvo é adotado, iniciando com uma taxa alvo de 2 que é progressivamente aumentada, sendo a acurácia resultante avaliada para cada uma das taxas de compressão consideradas. A compressão da rede é realizada utilizando três diferentes estratégias de poda de pesos [80]: i) poda global por magnitude (*global pruning* - GP), em que os pesos menos significativos da rede como um todo são alvos para poda; ii) poda por magnitude por camada (*layer pruning* - LP), em que os pesos menos significativos em cada camada são alvos (resultando na mesma taxa de compressão para cada camada); e iii) poda aleatória (*random pruning* - RP), em que os pesos são selecionados aleatoriamente de toda a rede, independentemente de sua magnitude. O ajuste fino (*fine tuning* - FT) [14] (ou seja, o retreinamento da rede após a poda) também é considerado neste experimento.

As Figuras 10 e 11 apresentam as curvas de acurácia *versus* taxa de compressão obtidas sem e com FT, respectivamente. O primeiro aspecto a se destacar, a partir dessas figuras, é a superioridade tanto da poda de pesos global quanto da poda por camada em relação à poda aleatória. Esse fato demonstra a eficácia da compressão

[1] A Lenet-5-Caffe é uma versão modificada da Lenet-5 [14] que é discutida em [98].

da rede via poda de pesos baseada em magnitude. Agora, em relação à capacidade das técnicas diferentes de regularização baseadas em normas para induzir esparsidade e, assim, proporcionar maiores taxas de compressão, as curvas das Figuras 10 e 11 mostram que a regularização proposta baseada na norma ℓ_0 (tanto sozinha quanto em combinação com a norma ℓ_2) supera significativamente as regularizações baseadas nas normas ℓ_1 e ℓ_2. Essa conclusão é direta dos resultados superiores obtidos utilizando a regularização proposta com poda de pesos global ou por camada. Além disso, o desempenho inferior obtido pela regularização proposta com poda aleatória também se deve à sua superior capacidade de indução de esparsidade, uma vez que um nível de esparsidade mais elevado implica maior concentração de energia em poucos pesos, com alguns desses pesos mais significativos sendo inevitavelmente alvos de uma estratégia de poda aleatória. Outro aspecto importante a ser destacado é que o uso de FT leva a resultados significativamente melhores em todos os casos, embora a regularização proposta baseada na norma ℓ_0 tenha obtido resultados muito bons mesmo sem o uso de FT (veja Figura 10).

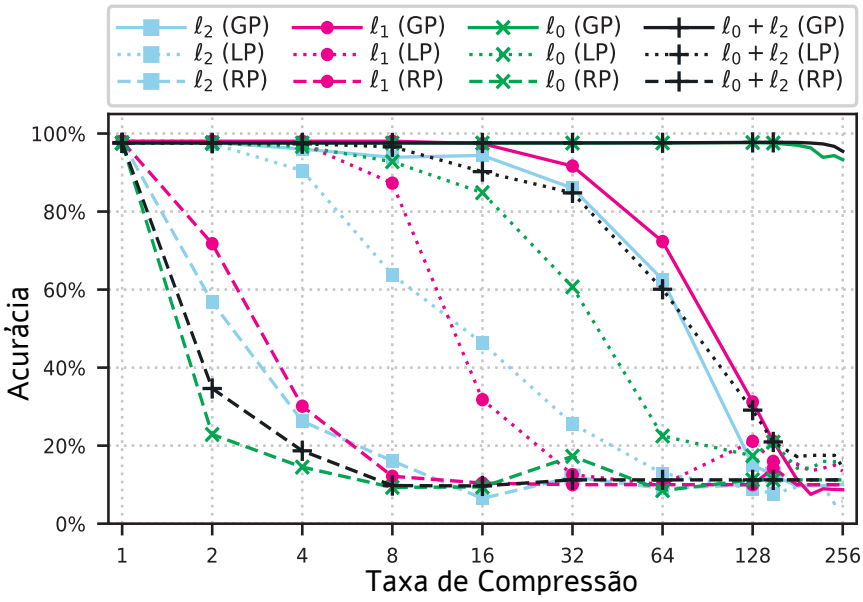


Figura 10 – Experimento 1. Acurácia *versus* taxa de compressão sem FT.

4.4.1.1 Análise do Perfil de Esparsidade e Distribuição de Parâmetros por Regularização

Além das métricas de acurácia *versus* taxa de compressão já analisadas neste experimento, a comparação entre as distribuições dos valores absolutos (magnitudes) dos parâmetros de modelos treinados com as regularizações de normas ℓ_2, ℓ_1 e com a aproximação ℓ_0 proposta evidencia características de indução a esparsidade durante treinamento, sendo tal indução mais proeminente para as normas ℓ_1 e ℓ_0. Com o objetivo de demonstrar tais características, os histogramas de distribuição dos parâmetros

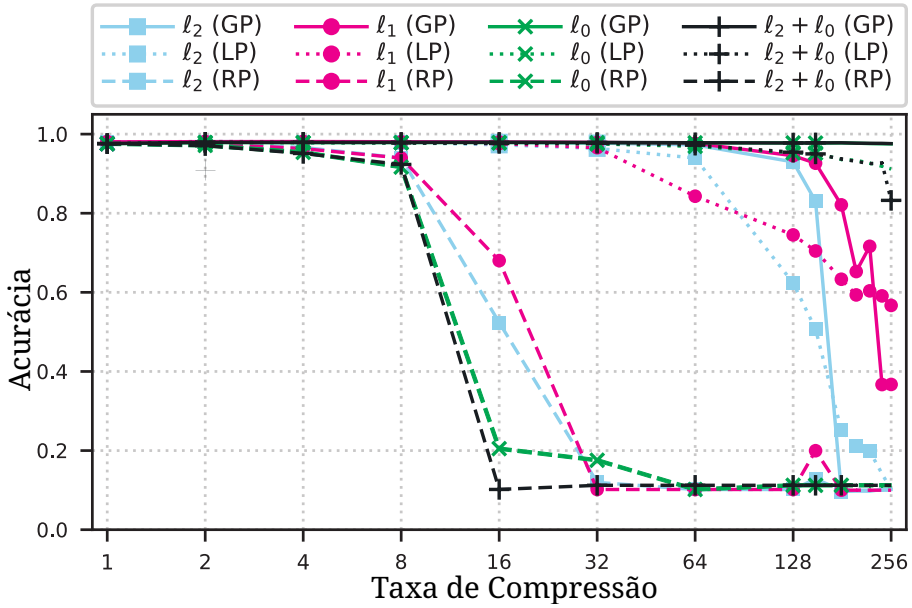
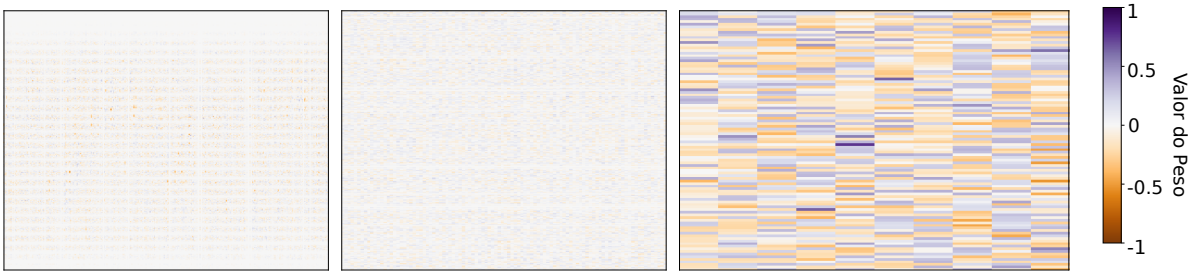


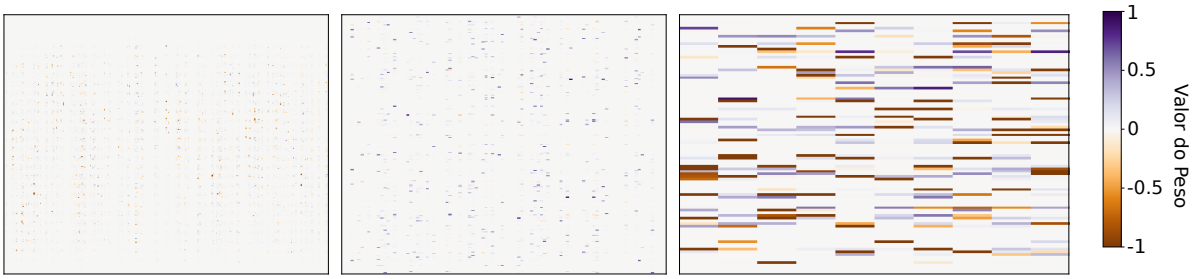
Figura 11 – Experimento 1. Acurácia contra taxa de compressão com FT.

(unidimensionais e bidimensionais) para as diferentes camadas da rede LeNet-300-100 são aqui analisados. Essa análise permite identificar a natureza da esparsidade induzida e como a rede preserva a informação relevante frente a diferentes penalizações. Nesse contexto, os histogramas apresentados na Figura 12 revelam comportamentos distintos entre as normas ℓ_2, ℓ_1 e a aproximação ℓ_0 proposta, evidenciando como a penalização induz a esparsidade durante o treinamento, concentrando a energia em pesos específicos da rede, para os casos que a maior taxa de compressão é alcançada. Observa-se ainda na Figura 12 que a aplicação da norma ℓ_2 resulta em uma distribuição mais uniforme e suave dos parâmetros em todas as camadas. Nas duas primeiras camadas, que concentram o maior volume de parâmetros e o maior potencial de compressão, os pesos permanecem concentrados em uma faixa intermediária da escala, sem atingir as extremidades (± 1). Do ponto de vista de compressão, essa característica é desfavorável, pois ao atribuir energias similares a praticamente todos os parâmetros, não houve indução de esparsidade da rede durante treinamento e não é possível identificar, através da magnitude dos pesos, quais conexões são desprezadas para a tarefa de classificação e podem ser podadas.

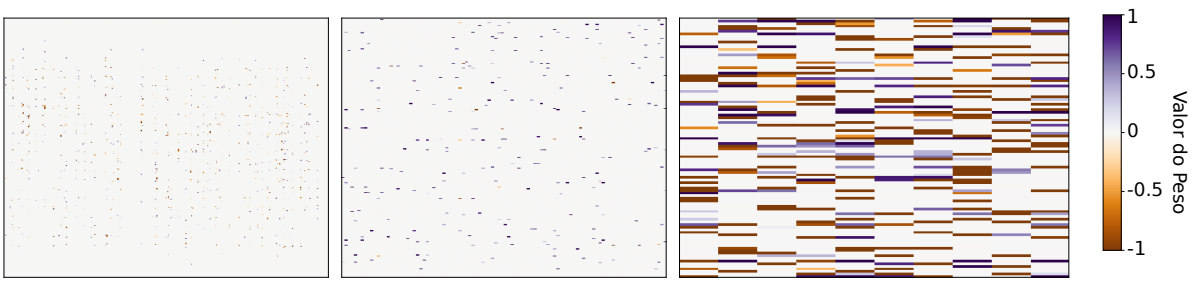
Em contraste, os histogramas referentes às regularizações ℓ_1 e ℓ_0 revelam uma mudança importante na arquitetura resultante. Nota-se que o modelo treinado possui estrutura mais esparsa, com a energia dos pesos da rede sendo redistribuída e acumulada em um conjunto menor de pesos de maior magnitude. Assim, verifica-se que, durante o processo de treinamento, as normas ℓ_1 e ℓ_0 atuam como mecanismos de seleção de parâmetros ou pesos. Mais precisamente, enquanto pesos irrelevantes são induzidos a zero, os parâmetros com maior contribuição para saída da rede têm suas magnitudes preservadas ou realçadas. Essa segregação fica evidente pela presença de pesos com magnitudes superiores às observadas na norma ℓ_2. Tal efeito indica



(a) Regularização ℓ_2



(b) Regularização ℓ_1

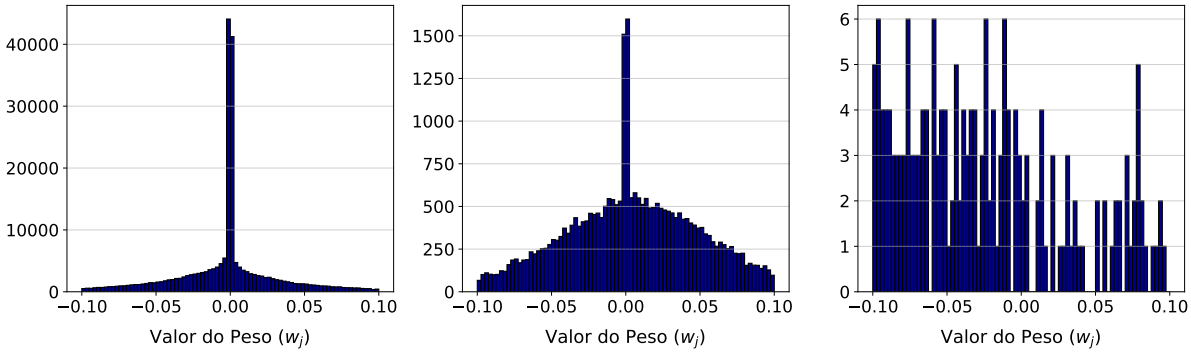


(c) Regularização ℓ_0 (Proposta)

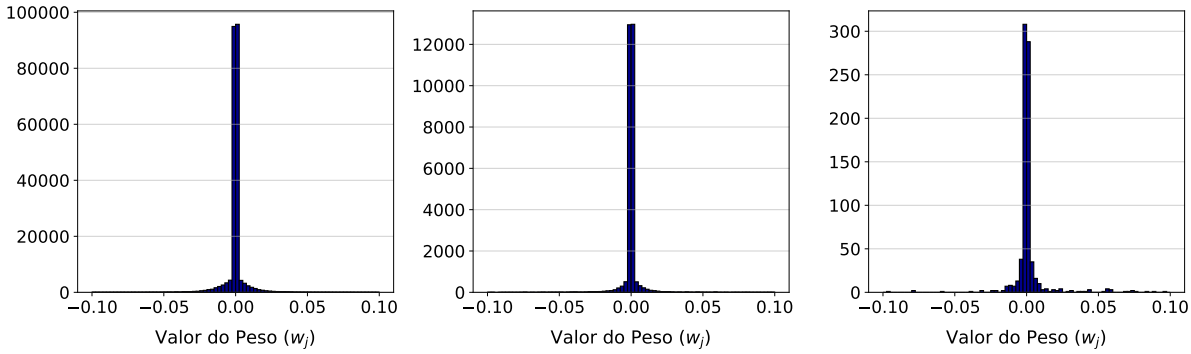
Figura 12 – Distribuição de pesos por camada para a rede LeNet-300-100. Comparativo detalhado da dispersão e da energia entre (a) ℓ_2, (b) ℓ_1 e (c) a proposta baseada em ℓ_0.

que a rede aprendeu a concentrar a informação necessária em poucas conexões mais robustas, facilitando a identificação de elementos passíveis de remoção sem prejuízo à acurácia.

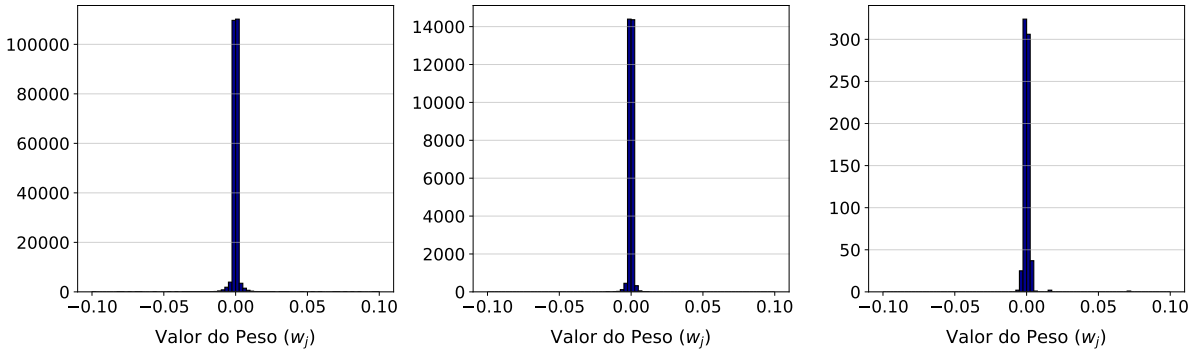
Com respeito às diferentes capacidades de compressão das estratégias baseadas na norma ℓ_1 e na norma ℓ_0 proposta, a superioridade desta última torna-se evidente ao comparar os histogramas unidimensionais, mostrados na Figura 13. Embora ambas as normas busquem a esparsidade, a baseada na ℓ_0 apresenta camadas com maior número de parâmetros próximos a zero e com um número ainda menor de parâmetros com magnitude significativa. A análise da distribuição de pesos é confirmada quantitativamente pela Tabela 2, que detalha a taxa de compressão obtida em cada camada da rede LeNet-300-100 sob um limiar de poda fixo. Observa-se que a estratégia ℓ_0 proposta, para as camadas iniciais, alcança uma taxa de compressão



(a) Distribuição unidimensional com regularização ℓ_2



(b) Distribuição unidimensional com regularização ℓ_1



(c) Distribuição unidimensional com regularização ℓ_0 (Proposta)

Figura 13 – Histogramas unidimensionais da densidade de pesos para a rede LeNet-300-100. A escala revela a magnitude da concentração de parâmetros em torno de zero e a preservação da energia nos pesos remanescentes.

significativamente superior às demais estratégias. Enquanto a estratégia baseada na norma ℓ_2 mantém quase a totalidade dos pesos ativos (mesmo que com magnitudes reduzidas), a norma ℓ_0 identifica e elimina as conexões desnecessárias de forma efetiva no processo de treinamento.

Tabela 2 – Taxa de compressão por camada para a LeNet-300-100 com limiar de poda fixo.

Regularização	**Pesos Remanescentes Pós-Poda (Threshold** = 0.05**)**			**Acurácia**
	Entrada	**Oculta**	**Saída**	
Sem Poda	235.200	30.000	1.000	98.36%
ℓ_2	49.826	9.713	913	98.23%
ℓ_1	7.643	823	235	98.36%
ℓ_0	1.864	246	303	98.04%

4.4.2 Experimento 2: Compressão de Rede para o Conjunto de Dados MNIST

Neste experimento, o conjunto de dados MNIST é considerado e os resultados obtidos usando o esquema de compressão de rede proposto são comparados com outras abordagens de compressão de rede baseadas em poda da literatura. Nesse contexto, duas topologias de rede são consideradas, a saber: Lenet-300-100 e Lenet-5-Caffe. A estratégia de poda escolhida é a poda global devido aos resultados superiores observados no primeiro experimento. Para a configuração dos hiperparâmetros de treinamento, duas estratégias diferentes são adotadas. A primeira, denominada NORM, fundamenta-se na observação de que a aplicação de uma penalização global e uniforme para todas as camadas resulte em uma capacidade de compressão subótima. Camadas com diferentes dimensões respondem de forma heterogênea para um mesmo termo de penalização. Nesse sentido, para maximizar a indução de esparsidade, a estratégia NORM estabelece valores de referência para os hiperparâmetros α_{ℓ_2} e α_{ℓ_0}, que são então escalados de acordo com o número de parâmetros de cada camada. Esse escalamento ajusta a força de regularização à capacidade de armazenamento de informação de cada camada. Uma vez que camadas mais profundas ou densas possuem maior liberdade paramétrica, a intensificação do termo de penalização permite induzir uma esparsidade mais agressiva. Esse processo, em camadas com maior número de parâmetros, identifica os parâmetros de maior relevância garantindo que o sacrifício de precisão seja minimizado em favor de um incremento substancial na taxa de compressão. Na segunda estratégia, chamada SEP, valores individuais de α_{ℓ_2}, α_{ℓ_0} e β são usados de forma independente para cada camada da rede. Nesse caso, começamos com os valores de hiperparâmetros da estratégia NORM e ajusta-

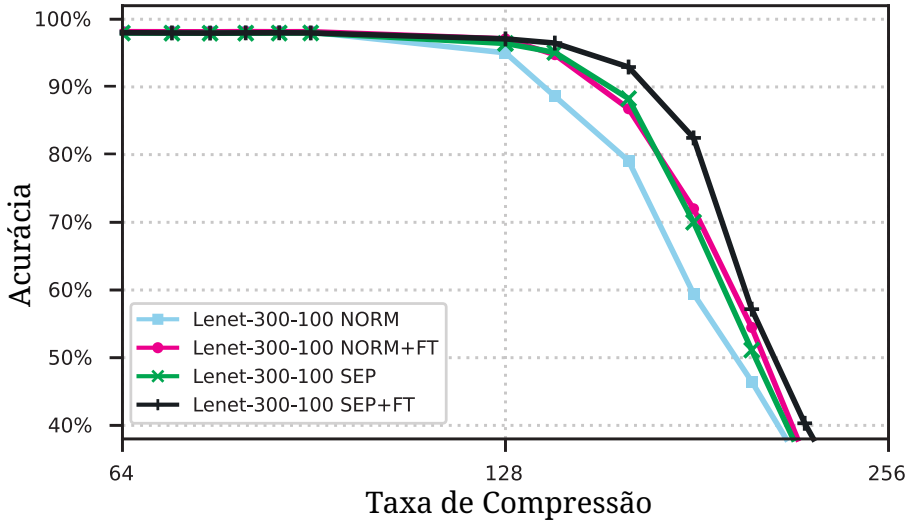


Figura 14 – Experimento 2. Acurácia *versus* taxa de compressão obtida aplicando o esquema proposto à rede LeNet-300-100.

mos individualmente os hiperparâmetros para cada camada experimentalmente. Esse procedimento permite aumentar a penalização em camadas que possuem um número maior de pesos com magnitudes maiores sem comprometer a acurácia da rede. Em termos gerais, NORM é mais simples de usar, enquanto SEP tende a produzir melhores resultados devido ao aumento dos graus de liberdade.

O primeiro aspecto avaliado neste experimento é o compromisso entre acurácia e taxa de compressão obtido usando a abordagem proposta. Para isso, o critério de poda por taxa de compressão alvo é considerado, começando com uma taxa alvo de 64 que é progressivamente aumentada e a acurácia resultante é mostrada para cada taxa de compressão. As Figuras 14 e 15 ilustram tal compromisso para as redes Lenet-300-100 e Lenet-5-Caffe, respectivamente, considerando as estratégias NORM e SEP, bem como FT. A partir dessas figuras, é possível notar que taxas de compressão muito altas podem ser obtidas usando a abordagem proposta com perda de desempenho desprezível para os casos aqui considerados. Como esperado, os melhores resultados são obtidos usando a combinação da estratégia SEP com FT (cerca de 100$\times$ de taxa de compressão para Lenet-300-100 e mais de 300$\times$ para Lenet-5-Caffe). Ainda assim, até mesmo a estratégia NORM (mais simples) mostrou-se eficaz, alcançando taxas de compressão de cerca de 90$\times$ para Lenet-300-100 e 200$\times$ para Lenet-5-Caffe.

Os resultados obtidos utilizando a abordagem proposta são agora confrontados com aqueles obtidos utilizando os seguintes métodos de compressão baseados em poda da literatura: poda por limiar (THP) [37], cirurgia dinâmica de rede (DNS) [33], *group horseshoe* (BC-GHS) [99], *sparse variational dropout* (SparseVD) [100], algoritmo de plano de corte (CPA) [101], momento esparso global (GSM) [102], e poda automática de rede (Autoprune) [103]. Além disso, também consideramos os seguintes métodos de poda baseados no uso da norma ℓ_0: ℓ_0 com aproximação *hard-concrete*

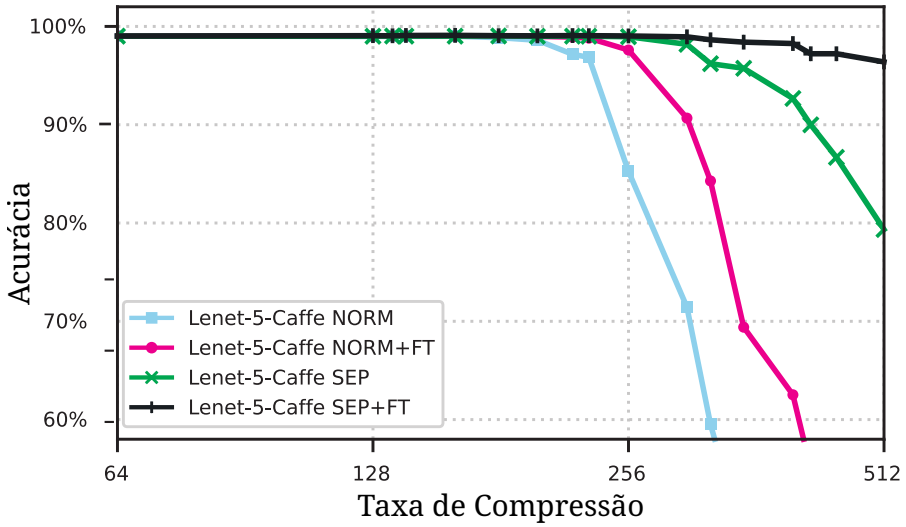


Figura 15 – Experimento 2. Acurácia *versus* taxa de compressão obtida aplicando o esquema proposto à rede LeNet-5-Caffe.

(L0-HC) [39], ℓ_0 *augment-reinforce-merge* (L0-ARM) [42], ℓ_0 baseado em ADMM (L0-ADMM) [43], e aprendizado de compressão usando normas ℓ_2 e ℓ_0 (L0-L2-LC) [45] (os resultados de [44] não estão incluídos, uma vez que a abordagem em [45] é uma evolução direta daquela em [44] e apresenta resultados superiores). As comparações entre todas as abordagens consideradas são apresentadas nas Tabelas 3 e 4 para LeNet-300-100 e LeNet-5-Caffe, respectivamente. Ressalta-se que, para a obtenção dos resultados da abordagem proposta, utilizou-se o critério de poda por limiar de magnitude. A partir das tabelas, pode-se inferir que a abordagem proposta é muito competitiva, resultando geralmente em *trade-offs* superiores entre acurácia e taxa de compressão em comparação com as outras abordagens da literatura.

4.4.3 Experimento 3: CIFAR-10

Para este terceiro experimento, considerou-se a tarefa de classificação de imagens do conjunto de dados CIFAR-10 utilizando as redes VGG-16 [104] e ResNet20 [55]. A VGG-16 possui cerca de 15 milhões de parâmetros e tipicamente obtém uma taxa de erro de $6,75\%$ para o CIFAR-10. A ResNet20 pertence à família de redes residuais (ResNets), nas quais conexões residuais são exploradas para obter custos computacionais menores. Assim, essa rede é implementada com 22 camadas e cerca de 274 mil parâmetros, obtendo uma taxa de erro de $7,8\%$ para o conjunto de dados CIFAR-10. Nesse experimento, essas duas arquiteturas são consideradas com o objetivo de avaliar a capacidade do esquema de compressão proposto para lidar com redes maiores (como a VGG-16), bem como com uma rede residual (ResNet20) para a qual a compressão de rede tende a ser uma tarefa mais desafiadora devido ao número já reduzido de parâmetros.

Para os casos das redes VGG-16 e ResNet-20, as estratégias NORM e SEP são utilizadas para o ajuste de hiperparâmetros, seguidas pela aplicação do FT. Além

Tabela 3 – Resultados obtidos com a arquitetura LeNet-300-100 no conjunto de dados MNIST

Abordagem de Compressão	**Erro Top-1**	**Quantidade de Parâmetros**	**Taxa de Compresão**
Referência (Sem Poda)	1.64%	267k	-
THP [37]	1.59%	22k	12×
DNS [33]	1.99%	4.8k	56×
BC-GHS [99]	1.8%	29.6k	9×
SparseVD [100]	1.92%	3.92k	68×
CPA [101]	1.77%	3.14k	85×
GSM [102]	1.82%	4.45k	60×
Autoprune [103]	1.78%	3.34k	80×
L_0-HC [39]	1.8%	26.72k	9.99×
L_0-ARM [42]	1.9%	18.8k	14.2×
L_0-ADMM [43]	*	11.6k	23×
L_0-L_2-LC [45]	1.6%	5.34k	50×
L_0-L_2-LC [45]	1.99%	4k	66.8×
Proposta (NORM)	1.85%	2.96k	90×
Proposta (SEP)	1.99%	2.7k	96×

* Igual ao de referência.

disso, o critério de poda por limiar de magnitude é considerado para a obtenção dos dados com a abordagem proposta. As Tabelas 5 e 6 apresentam comparações com os resultados da literatura para a VGG-16 e ResNet-20, respectivamente. A partir desses resultados, observa-se que o esquema proposto supera as abordagens concorrentes, demonstrando a eficácia da regularização ℓ_2-ℓ_0 combinada mesmo em arquiteturas mais profundas e residuais.

Adicionalmente, para a rede ResNet-20, realizou-se uma análise detalhada utilizando o critério de poda por taxa de compressão alvo, com valores variando entre 1,5 e 5,5 vezes a taxa original. As curvas de acurácia *versus* taxa de compressão resultantes são apresentadas na Figura 16. Nesses testes, observa-se que a discrepância entre as estratégias NORM e SEP foi reduzida em comparação ao observado no Experimento 2, ao passo que o FT desempenhou um papel mais crítico no ganho de desempenho. No limite de compressão de 5 vezes, a perda de acurácia manteve-se em patamares aceitáveis, reforçando a robustez da técnica de poda de pesos aplicada a blocos residuais.

Tabela 4 – Resultados obtidos com arquitetura LeNet-5-Caffe no conjunto de dados MNIST

Abordagem de Compressão	**Erro Top-1**	**Quantidade de Parâmetros**	**Taxa de Compressão**
Referência (Sem Poda)	0.8%	431k	-
THP [37]	0.7%	36k	12×
DNS [33]	0.91%	4.0k	108×
BC-GHS [99]	1.0%	2.76k	156×
SparseVD [100]	0.75%	1.54k	280×
CPA [101]	0.79%	1.35k	317×
GSM [102]	0.94%	1.44k	300×
Autoprune [103]	0.91%	1.39k	310×
L_0-HC [39]	1.0%	4.66k	92.6×
L_0-ARM [42]	1.3%	2.2k	196×
L_0-ADMM [43]	*	6.05k	71.2×
L_0-L_2-LC [45]	0.89%	4.3k	100×
Proposta (NORM)	0.99%	2.15k	200×
Proposta (SEP)	0.97%	1.2k	359×

* Igual a de referência.

Tabela 5 – Resultados obtidos com arquitetura VGG-16 aplicada ao conjunto de dados CIFAR-10

Abordagem de Compressão	**Perda de Acurácia**	**Parâmetros Podados**	**Taxa de Compressão**
SSS [105]	*	30%	1.67×
EConvNets [106]	−0.15%	64%	2.78×
VAR-CONV [107]	0.07%	73.34%	3.75×
PP [108]	0.03%	82.8%	5.8×
PP [108]	0.14%	84.4%	6.43×
RFP [109]	0.13%	78.1%	4.6×
RFP [109]	0.72%	89.5%	9.52×
Proposta (NORM)	0.41%	85.5%	6.92×
Proposta (SEP)	0.7%	93.2%	14.76×

* Igual a de referência.

4.4.4 Experimento 4: VGG-16 no CIFAR-100

Neste quarto experimento, a rede VGG-16 é aplicada à tarefa de classificação do conjunto de dados CIFAR-100, que compreende 100 classes diferentes em contraste com as 10 classes diferentes do CIFAR-10. Assim, pode-se notar que a VGG-16

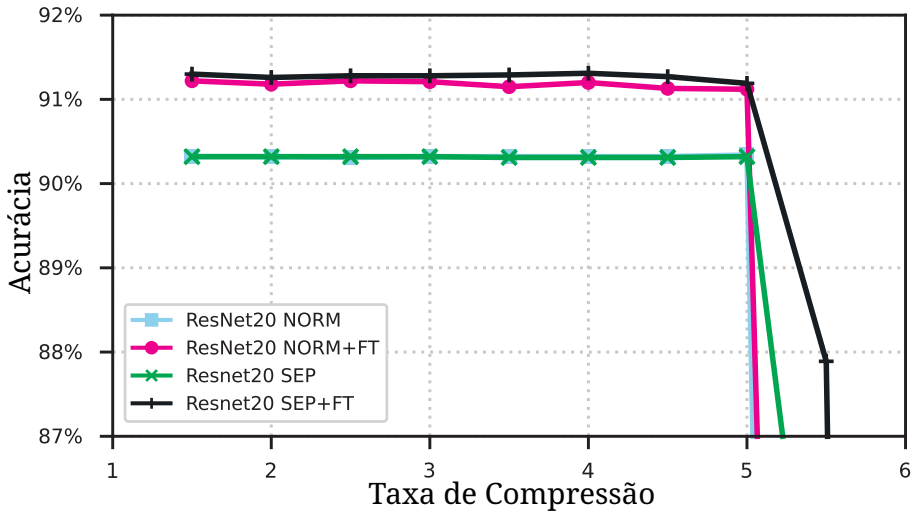


Figura 16 – Experimento 3. Acurácia *versus* taxa de compressão obtida aplicando o esquema proposto à rede ResNet20.

Tabela 6 – Resultados obtidos com arquitetura ResNet-20 aplicada ao conjunto de dados CIFAR-10

Abordagem de Compressão	**Perda de Acurácia**	**Parâmetros Podados**	**Taxa de Compressão**
VAR-CONV [107]	0.54%	20.41%	1.26×
CNN-FCF [110]	1.07%	42.75%	1.75×
CNN-FCF [110]	2.67%	68.44%	3.17×
SNLI [111]	1.1%	37.22%	1.59×
SNLI [111]	3.2%	67.83%	3.11×
DHP [112]	1.0%	55.95%	2.27×
Proposta (NORM)	1.1%	80%	5×
Proposta (SEP)	1%	80%	5×

está agora sujeita a uma tarefa mais desafiadora, resultando em níveis de acurácia na ordem de 70% (contra mais de 90% para o CIFAR-10). Tal aumento na dificuldade também resulta em uma tarefa de compressão de rede mais desafiadora. Para simplificar, o FT não é utilizado nesse caso e apenas a estratégia NORM é usada para ajustar os parâmetros do esquema proposto. Além disso, quatro valores diferentes de α_{ℓ_0} são considerados: 0 (ou seja, sem penalização da norma ℓ_0), 1×10^{-7}, $2,5 \times 10^{-7}$ e 5×10^{-7}. A ideia é demonstrar como o compromisso entre acurácia e taxa de compressão pode ser controlado através do ajuste de α_{ℓ_0}.

A Figura 17 mostra as curvas de acurácia *versus* taxa de compressão obtidas por meio do critério de poda por taxa de compressão alvo. A partir dessas curvas, pode-se notar que, à medida que o valor de α_{ℓ_0} aumenta (ou seja, o efeito da regularização é fortalecido), a taxa de compressão que pode ser alcançada sem perda significativa de acurácia é aumentada. Isso ocorre ao custo de uma leve redução no nível de acurácia

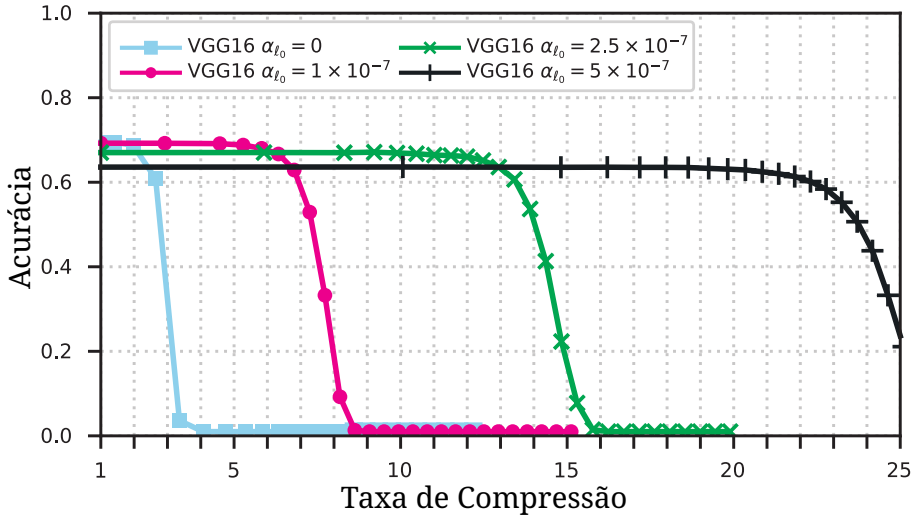


Figura 17 – Experimento 4. Acurácia *versus* taxa de compressão obtida aplicando o esquema proposto à rede VGG-16 com o conjunto de dados CIFAR-100.

inicial (ou seja, aquele obtido antes da poda). Mais especificamente, a acurácia inicial obtida para $\alpha_{\ell_0} = 0$ é reduzida em $0,19\%$, $2,42\%$ e $5,85\%$ para os valores de α_{ℓ_0} de 1×10^{-7}, $2,5 \times 10^{-7}$ e 5×10^{-7}, respectivamente.

A Tabela 7, por sua vez, detalha os resultados obtidos via poda por limiar de magnitude para cada um dos valores considerados de α_{ℓ_0}, juntamente com os resultados de [107]. Observa-se que a abordagem proposta alcança taxas de compressão significativamente superiores àquelas da literatura (4,67 vezes contra 1,61 vezes), mantendo a acurácia na mesma ordem de magnitude ao considerar $\alpha_{\ell_0} = 1 \times 10^{-7}$. Outro resultado de uma configuração um tanto similar da literatura pode ser encontrado em [113]. O método proposto lá é capaz de reduzir o número de parâmetros em $81,1\%$ para a VGG-16 no CIFAR-100, correspondendo a uma taxa de compressão de $5,29\times$, ao custo de um aumento na taxa de erro de $0,17\%$. Esse resultado é comparável com aqueles obtidos utilizando o esquema proposto aqui. No entanto, a versão da VGG-16 considerada lá tem mais do que o dobro do tamanho da VGG-16 considerada aqui e, portanto, após a compressão, ainda possui 6,43 milhões de parâmetros, enquanto em nosso caso a rede comprimida com $\alpha_{\ell_0} = 1 \times 10^{-7}$ tem cerca de 3,2 milhões de parâmetros. A partir de todos esses resultados, pode-se notar novamente a eficácia do esquema proposto.

4.4.5 Experimento 5: ResNet-50 no CIFAR-100 com Número Limitado de Épocas

O quinto experimento deste trabalho é dedicado a avaliar o desempenho da abordagem proposta para a compressão de um modelo ResNet-50 [55] no CIFAR-100, considerando um número limitado de 100 épocas de treinamento. A ideia é avaliar a capacidade da abordagem proposta para produzir uma rede esparsa e compressível sem muito treinamento. Como referência, a acurácia atingida pelo modelo considerado

Tabela 7 – Resultados obtidos com rede VGG-16 aplicada ao conjunto de dados CIFAR-100

Abordagem de Compressão	**Perda de Acurácia**	**Parâmetros Podados**	**Taxa de Compressão**
VAR-CONV [107]	-0.07%	37.87%	$1.61\times$
Proposta com $\alpha_{\ell_0} = 1 \times 10^{-7}$	0.3%	78.6%	$4.67\times$
Proposta com $\alpha_{\ell_0} = 2.5 \times 10^{-7}$	2.67%	90.4%	$10.47\times$
Proposta com $\alpha_{\ell_0} = 5 \times 10^{-7}$	5.95%	94.6%	$18.63\times$

após 100 épocas de treinamento, sem usar a regularização proposta, é de cerca de $71,5\%$. Novamente, para simplificar, o FT não é utilizado e a estratégia NORM é aplicada para ajustar os hiperparâmetros do esquema proposto. Além disso, o critério de poda por taxa de compressão alvo é adotado e os hiperparâmetros de treinamento são definidos da seguinte forma: dois valores diferentes de β (5 e 10) e três valores diferentes de α_{ℓ_0} (1×10^{-7}, 5×10^{-7} e 1×10^{-6}) são considerados.

As curvas de acurácia obtidas *versus* taxa de compressão são apresentadas na Figura 18. A partir dessas curvas, pode-se notar novamente a eficácia da abordagem proposta, produzindo taxas de compressão variando de $13\times$ (com $\beta = 5$ e $\alpha_{\ell_0} = 1\times10^{-7}$) a $25\times$ (com $\beta = 10$ e $\alpha_{\ell_0} = 1\times10^{-6}$) sem redução perceptível no desempenho. Além disso, pode-se observar que a mudança no valor de β de 5 para 10 não resulta em uma alteração severa no desempenho, mostrando uma certa robustez da estratégia proposta em relação à escolha de tal parâmetro.

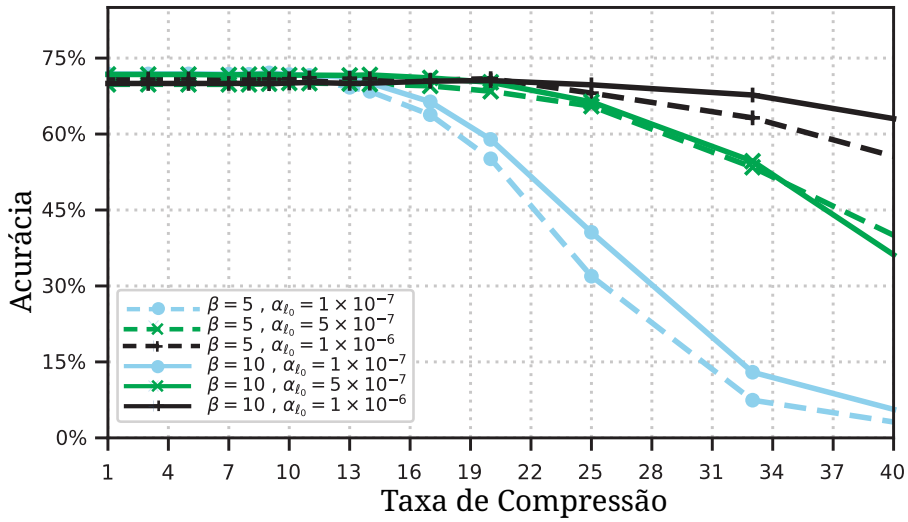


Figura 18 – Experimento 5. Acurácia *versus* taxa de compressão obtida aplicando o esquema proposto à rede ResNet-50 com o conjunto de dados CIFAR-100.

4.4.6 Experimento 6: Poda Estrutural

Neste sexto experimento, a eficácia da abordagem de regularização proposta aplicada à poda estrutural é avaliada. Assim como no Experimento 3, a arquitetura considerada é a VGG-16 e o conjunto de dados é o CIFAR-10. Duas estratégias de poda de rede são consideradas: i) poda de neurônios (*neuron pruning* - NP), na qual podamos os neurônios (neurônios de camada densa ou filtros de camada convolucional) cujos vetores de peso apresentam as menores normas ℓ_2; e ii) NP combinada com poda global de pesos (NP+GP). Essas estratégias são comparadas em termos de taxa de compressão e redução no número de operações de ponto flutuante (FLOPs) exigidas para inferência. Em ambos casos, o critério de poda por limiar de magnitude é adotado. No caso da comparação de FLOPs, consideramos a redução exclusivamente devido à poda de neurônios (implicando que, no caso NP+GP, consideramos apenas a redução de FLOPs devido à NP). A Tabela 8 mostra os resultados obtidos utilizando o esquema de compressão proposto, juntamente com os achados de [106]. Tais resultados sugerem que a estratégia de compressão proposta tem o potencial de ser utilizada também para poda de neurônios, pois os níveis de compressão obtidos são competitivos com aqueles alcançados por uma abordagem centrada em poda estrutural [106]. Além disso, quando a NP é combinada com a GP, a redução no número de FLOPs é novamente muito grande, enquanto os níveis de compressão de rede atingidos (que também consideram a poda de pesos individuais) são realmente muito altos.

Tabela 8 – Resultados obtidos com poda estrutural aplicada em VGG-16 com o conjunto de dados CIFAR-10

Método de Poda	**Aumento do Erro**	**Redução em FLOPs***	**Razão de Compressão**
EConvNets [106]	-0.15%	34.2%	$2.78\times$
Proposta (NP)	0.31%	27.75%	$1.46\times$
Proposta (NP)	0.49%	42.7%	$1.7\times$
Proposta (NP+GP)	0.27%	27.75%	$9.21\times$
Proposta (NP+GP)	0.72%	42.7%	$18.38\times$

* Exclusivamente devido à poda de neurônios.

4.5 CONCLUSÕES

Este capítulo foi dedicado à apresentação das contribuições principais do presente trabalho. A primeira delas é a regularização por norma ℓ_0 (baseada em uma aproximação exponencial), a qual foi descrita em detalhes, destacando seu elevado efeito de atração dos parâmetros para zero durante o treinamento. A segunda contribuição é um esquema combinado com as regularizações ℓ_2 e ℓ_0, o qual define uma

regularização com forte efeito de atração para zero durante o treinamento e que também é eficaz para o tratamento do *overfitting*.

Resultados experimentais foram apresentados comprovando a eficácia tanto da regularização quanto do esquema de compressão propostos. A análise dos resultados experimentais permite concluir que a eficácia da estratégia de compressão proposta é dependente da arquitetura da rede e da natureza do conjunto de dados utilizado.

Observou-se que a capacidade de indução de esparsidade é favorecida em arquiteturas que apresentam maior redundância estrutural. Esse aspecto é evidenciado ao comparar os resultados da Seção 4.4.3 (Experimento 3), onde, para o mesmo conjunto de dados (CIFAR-10), a rede VGG-16 alcançou uma taxa de compressão de 6,92$\times$ contra 5$\times$ da ResNet-20, mantendo níveis de perda de acurácia similares. Tal comportamento sugere que modelos com maior contagem absoluta de parâmetros oferecem uma margem de simplificação mais ampla antes que a capacidade de representação do modelo seja comprometida.

Adicionalmente, a complexidade do conjunto de dados impõe limites diretos ao potencial de compressão. Como mostrado na Seção 4.4.4 (Experimento 4), ao elevar a complexidade do problema do CIFAR-10 para o CIFAR-100 utilizando a mesma arquitetura VGG-16, a taxa de compressão reduziu de 6,92$\times$ para 4,67$\times$ para uma mesma faixa de acurácia. Esse resultado indica que a maior diversidade de classes e características do CIFAR-100 exige a preservação de uma maior densidade de parâmetros para a correta extração de atributos hierárquicos. Portanto, os experimentos validam a robustez da regularização baseada na norma ℓ_0 proposta, evidenciando sua capacidade de adaptar-se dinamicamente à capacidade da rede e à exigência da tarefa, otimizando o compromisso entre eficiência computacional e precisão de inferência.

5 COMPRESSÃO DE REDES NEURAIS USANDO REGULARIZAÇÃO COM APROXIMAÇÃO LINEARIZADA DA NORMA ℓ_0

O Capítulo 4 do presente trabalho foi dedicado à apresentação de um esquema de compressão de redes baseado em uma nova estratégia de regularização com aproximação exponencial da norma ℓ_0. Conforme demonstrado via extensivos resultados de simulação, essa abordagem tem se mostrado eficaz na indução de esparsidade em modelos de redes neurais profundas, permitindo a obtenção de modelos compactos sem comprometer significativamente a acurácia. O presente capítulo, é dedicado ao desenvolvimento de uma outra estratégia de compressão baseada em uma segunda forma de regularização com aproximação da norma ℓ_0. A ideia aqui é explorar uma aproximação linearizada da norma ℓ_0, em contraste com a aproximação exponencial discutida no Capítulo 4. Assim, busca-se manter as propriedades de indução de esparsidade, mas agora com reduções de complexidade matemática e computacional da regularização durante o treinamento.

5.1 APROXIMAÇÃO LINEARIZADA PARA O CÁLCULO DA NORMA ℓ_0

Analisando a função de aproximação exponencial para a norma ℓ_0 descrita em (20), observa-se que tal função depende da aplicação da função exponencial a cada um dos pesos da rede, o que precisa ser feito a cada iteração do processo de treinamento. É bem conhecido o fato de que o cálculo da função exponencial é computacionalmente custoso [114], especialmente em plataformas de *hardware* que não possuem unidades dedicadas para o cálculo de funções matemáticas complexas. Visando então reduzir o impacto computacional do cálculo da norma ℓ_0 aproximada, propõe-se aqui o uso de uma outra aproximação para a norma ℓ_0 que não dependa do cálculo da função exponencial.

Como descrito em [115], a função exponencial centrada em zero pode ser aproximada por meio da série de Taylor truncada centrada em zero. Assim, o termo exponencial presente em (20) pode ser escrito como

$$e^{-\beta|w_j|} \approx \sum_{i=0}^{N} \frac{(-\beta)^i}{i!} |w_j|^i. \tag{32}$$

Quanto maior o valor de N em (32), melhor é a representatividade da aproximação exponencial obtida para uma maior faixa de valores em torno de zero conforme ilustrado na Figura 19. De maneira similar, nota-se, comparando a Figura 19(a) com a Figura 19(b), que β controla a faixa de valores de w_j para os quais a aproximação de $e^{-\beta|w_j|}$ pela série de Taylor converge na direção da função original. Observe que, na Figura 19(a) onde $\beta = 10$ e $N = 1$, essa faixa é definida aproximadamente por

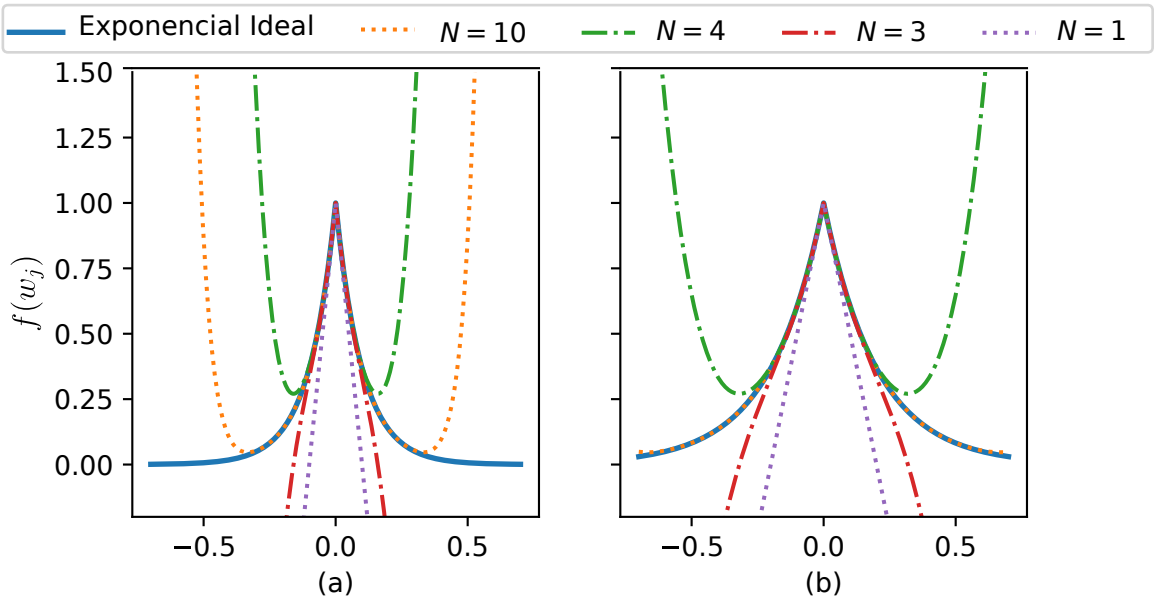


Figura 19 – Aproximação de exponencial utilizando (33) para diferentes valores de N e (a) $\beta = 10$ (b) $\beta = 5$.

$|w_j| \leq \frac{1}{10} = 0,1$, enquanto na Figura 19(b), onde $\beta = 5$, a faixa de convergência é definida por $|w_j| \leq \frac{1}{5} = 0,2$.

Conforme comentado anteriormente, o interesse aqui é em redução de complexidade computacional. Assim, a primeira escolha direta visando obter uma aproximação adequada e simples é considerar o menor valor possível para N em (32), isto é, $N = 1$. Isso resulta em uma aproximação linear em relação a $|w_j|$ para $e^{-\beta|w_j|}$. Além disso, como a aproximação descrita por (32) para $N = 1$ é satisfatória para valores de w_j próximos de zero, considera-se tal aproximação apenas para $|w_j| \leq \frac{1}{\beta}$ (ponto de intersecção em que (32) com $N = 1$ é igual a zero). Dessa forma, a seguinte aproximação para $e^{-\beta|w_j|}$ é aqui considerada:

$$e^{-\beta|w_j|} = \begin{cases} 1 - \beta|w_j| & \text{se } |w_j| \leq \frac{1}{\beta}, \\ 0 & \text{para outros valores} \end{cases}. \tag{33}$$

Finalmente, substituindo (33) em (20), obtém-se a seguinte aproximação para a norma ℓ_0:

$$||\mathbf{w}||_{\tilde{0}} = \sum_{j \in \phi} \bar{z}_j, \tag{34}$$

onde

$$\bar{z}_j = \begin{cases} \beta|w_j| & \text{se } \beta|w_j| \leq 1 \\ 1 & \text{para outros valores.} \end{cases} \tag{35}$$

Por simplicidade, essa aproximação será aqui denominada aproximação linearizada da norma ℓ_0.

Uma comparação entre a função norma ℓ_0 e a sua versão aproximada de (34) é apresentada na Figura 20, considerando o caso simples de um vetor com um único

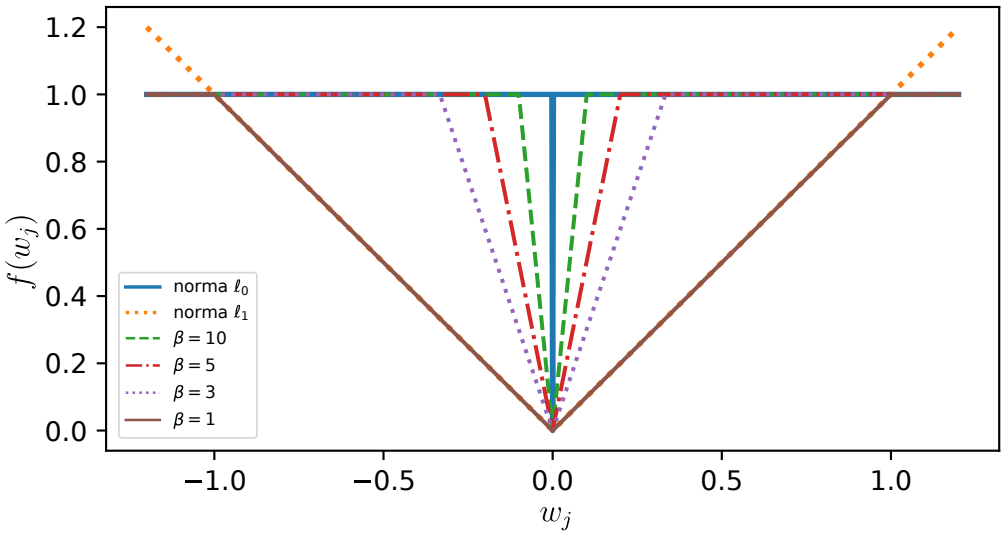


Figura 20 – Comparação entre norma ℓ_0 e aproximação dada por (34) para diferentes valores de β.

peso. Nota-se que, similarmente a (20), o parâmetro β de (35) controla o grau de similaridade entre a norma ℓ_0 e a função aproximada, sendo que tais funções tornam-se idênticas para $\beta \to \infty$. A Figura 20 também evidencia a característica similar entre (34) e a norma ℓ_1 para $|w_j| \leq \frac{1}{\beta}$, o que fica ainda mais claro quando $\beta = 1$. Assim, a aproximação resultante pode ser vista como uma combinação das normas ℓ_1 e ℓ_0: comporta-se como a norma ℓ_1 quando $|w_j| \leq \frac{1}{\beta}$ e como a norma ℓ_0 para os demais valores de $|w_j|$.

5.2 REGULARIZAÇÃO BASEADA NA APROXIMAÇÃO LINEARIZADA DA NORMA ℓ_0

Para obter uma estratégia de regularização baseada na aproximação linearizada da forma ℓ_0, considera-se o termo de penalização definido pela expressão apresentada em (34), resultando em

$$\Omega(\mathbf{w}) = ||\mathbf{w}||_{\tilde{0}} = \sum_{j \in \phi} \bar{z}_j. \tag{36}$$

Substituindo (36) em (8) e individualizando a equação obtida para o j-ésimo peso da rede, obtém-se a seguinte regra de atualização:

$$w_j^+ \leftarrow w_j - \eta\alpha \frac{\partial \bar{z}_j}{\partial w_j} - \eta \frac{\partial J(\mathbf{w}, \mathbf{X}, \mathbf{y})}{\partial w_j}. \tag{37}$$

Note que a atualização de pesos descrita em (37) depende da derivada da aproximação linearizada da norma ℓ_0 com respeito a w_j. No entanto, essa função possui três pontos de inflexão em seu domínio (especificamente em $w_j = 0$, $-\frac{1}{\beta}$ e $\frac{1}{\beta}$, veja a Figura 20), para os quais a derivada é indefinida. Para contornar esse problema,

explora-se aqui o conceito de sub-derivada [116] e assume-se que a derivada em tais pontos é dada pelo limite da derivada à esquerda ou à direita. Assim, é possível escrever a derivada de $\sum_{j\in\phi}\bar{z}_j$ com respeito a w_j simplesmente como

$$\frac{\partial \bar{z}_j}{\partial w_j} = \begin{cases} \beta\,\text{sign}(w_j) & \text{se } \beta|w_j| \leq 1 \\ 0 & \text{para outros valores.} \end{cases} \tag{38}$$

Substituindo agora (38) em (37), a seguinte regra de atualização de pesos é obtida:

$$w_j \leftarrow \begin{cases} w_j - \eta\alpha\beta\text{sign}(w_j) - \eta\frac{\partial J(\mathbf{w};\mathbf{X},\mathbf{y})}{\partial w_j} & \text{se } |w_j| \leq \frac{1}{\beta} \\ w_j - \eta\frac{\partial J(\mathbf{w};\mathbf{X},\mathbf{y})}{\partial w_j} & \text{para outros valores.} \end{cases} \tag{39}$$

5.3 ANÁLISE DA REGULARIZAÇÃO COM APROXIMAÇÃO LINEARIZADA DA NORMA ℓ_0

A partir de (39), é possível notar que a atualização dos pesos na regularização com aproximação linearizada da norma ℓ_0 é dividida em duas regiões distintas, delimitadas por $|w_j| = 1/\beta$. Mais especificamente, na região onde $|w_j| \leq 1/\beta$ (pesos de menor magnitude), a atualização de pesos inclui o termo de penalização $-\eta\alpha\beta\text{sign}(w_j)$, enquanto, para $|w_j| > 1/\beta$ (pesos maiores), a atualização é equivalente à não aplicação de qualquer regularização. O termo de penalização aplicado aos pesos menores é igual ao termo da regularização por norma ℓ_1 (veja Tabela 1) multiplicado por β e, portanto, conclui-se que os pesos menores são submetidos a uma regularização equivalente à obtida com a norma ℓ_1. Por outro lado, o fato de não haver qualquer penalização para pesos maiores é uma característica típica da norma ℓ_0. Assim, conclui-se que a regularização com aproximação linearizada da norma ℓ_0 exibe um comportamento híbrido, combinando as características da regularização por norma ℓ_1 (para pesos de pequena magnitude) e ℓ_0 (não afeta pesos de grande magnitude).

Uma das principais vantagens do comportamento híbrido da regularização com aproximação linearizada da norma ℓ_0 é a mitigação do problema de objetivos conflitantes da regularização por norma ℓ_1 mencionado na Seção 4.2. Em resumo, esse problema decorre do fato de que a regularização por norma ℓ_1 envolve a penalização de todos os pesos uniformemente e, assim, quando seus hiperparâmetros são ajustados para evitar *overfitting* (o que requer uma penalização maior), a penalização aplicada a pesos pequenos é excessiva e causa maiores oscilações de peso próximas a zero. Por outro lado, quando os hiperparâmetros são ajustados para favorecer a esparsidade (o que requer uma penalização geral mais leve), a penalização aplicada a pesos maiores torna-se insuficiente para evitar o *overfitting*. A regularização com aproximação linearizada da norma ℓ_0, ao não penalizar pesos de grande magnitude, acaba por desacoplar esses dois objetivos, permitindo que a penalização aplicada a pesos pequenos seja ajustada de forma à obtenção do nível de esparsidade desejado.

Com respeito ao fator de escala descrito na Seção 4.2, para a regularização com aproximação linearizada da norma ℓ_0 tem-se $\lambda_j = 1$ para os pesos com magnitude maior do que $1/\beta$ (o que implica nenhuma atração para a zero) e $\lambda_j = 1 - \frac{\eta\alpha\beta}{|w_j|}$ para a faixa de pesos menores (similarmente à regularização por noma ℓ_1). Nota-se então que, para a faixa de pesos menores, esse fator irá variar monotonicamente de valores negativos de grande magnitude (quando $|w_j| \to 0$) até $1 - \eta\alpha\beta^2$ (quando $|w_j| = 1/\beta$). Assim, haverão de fato oscilações em torno de zero (quando $|w_j|$ se tornar pequeno o suficiente para que $|\lambda_j| > 1$), o que é um comportamento típico da regularização por norma ℓ_1. Por outro lado, para evitar oscilações quando $|w_j| \to 1/\beta$, é necessário que

$$0 < \alpha < \frac{2}{\eta\beta^2}. \tag{40}$$

A partir de (40) e ajustando também β de forma apropriada, é possível controlar a faixa de pesos que serão penalizados, bem como a intensidade dessa penalização.

Em síntese, a regularização com aproximação linear da norma ℓ_0 é uma solução que combina as vantagens da norma ℓ_1 para a poda de pesos pequenos com a seletividade da norma ℓ_0. Além disso, evita-se também o problema da atração a zero evanescente da regularização por norma ℓ_2. Contudo, a ausência de qualquer penalização para pesos maiores implica a necessidade de uso de algum outro tipo de regularização para evitar o *overfitting*.

5.4 COMPRESSÃO DE REDES NEURAIS USANDO A APROXIMAÇÃO LINEARIZADA DA NORMA ℓ_0

Conforme evidenciado anteriormente, a regulização baseada na aproximação linearizada da norma ℓ_0 tende a não ser eficiente em evitar o problema de *overfitting* por atuar apenas sobre pesos com valores próximos de zero. Para contornar esse problema, segue-se aqui a ideia descrita na Seção 4.3, combinando tal regularizaçao com a regularização por norma ℓ_2. Dessa forma, considerando $\Omega(\mathbf{w}) = ||\mathbf{w}||_2^2 + ||\mathbf{w}||_{\tilde{0}}$, a seguinte regra de atualização de pesos da rede pode ser obtida:

$$w_j \leftarrow \begin{cases} w_j - 2\eta\alpha_{\ell_2} w_j - \eta\alpha_{\ell_0}\beta\,\text{sign}(w_j) - \eta\frac{\partial J(\mathbf{w};\mathbf{X},\mathbf{y})}{\partial w_j}, & \text{se } |w_j| \leq \frac{1}{\beta} \\ w_j - 2\eta\alpha_{\ell_2} w_j - \eta\frac{\partial J(\mathbf{w};\mathbf{X},\mathbf{y})}{\partial w_j} & \text{para outros valores.} \end{cases} \tag{41}$$

O esquema de atualização de pesos descrito em (41) é claramente delimitado em duas regiões distintas de valores de w_j para a regularização por norma ℓ_0, sendo tal limite definido pelo valor de β. Mais especificamente, β define a região em torno de zero onde a penalização baseada na norma ℓ_0 é aplicada, enquanto α_{ℓ_0} controla a intensidade com que os pesos dentro desse limiar serão penalizados e atraídos para zero. Por outro lado, a regularização por norma ℓ_2, controlada pelo parâmetro de penalização α_{ℓ_2}, atua sobre todos os parâmetros da rede, ainda que com intensidade

decrescente à medida que a magnitude do peso se aproxima de zero. Com isso, busca-se eficiência tanto na prevenção de *overfitting* quanto na indução de esparsidade, sendo que esta última pode ser explorada por uma estratégia de poda de pesos para obtenção de redução de complexidade.

5.5 RESULTADOS EXPERIMENTAIS

Nesta seção, o objetivo é comparar o desempenho do esquema de compressão de redes apresentado no presente capítulo, o qual utiliza a aproximação linearizada da norma ℓ_0, com o apresentado no Capítulo 4. Nesse contexto, são apresentados resultados envolvendo experimentos de compressão de redes Lenet-300-100 e ResNet-50 aplicadas aos conjuntos de dados MNIST e CIFAR-100, respectivamente. Em todos os casos, a abordagem de poda utilizada para compresão das redes é a poda global (*global prunning*) e considerando o critério de poda por limiar de magnitude. Além disso, poda é realizada em uma etapa subsequente ao treinamento. As demais especificidades de cada um dos casos são descritas nas seções a seguir em conjunto com os resultados experimentais obtidos.

5.5.1 Experimento 1. Compressão da Rede Lenet-300-100 para o MNIST

Neste primeiro experimento, o problema considerado é o de compressão da rede Lenet-300-100 [14] aplicada ao problema do *dataset* MNIST [14]. Os hiperparâmetros relativos à regularização foram variados conforme descrito a seguir:

- α_{ℓ_2} com valores 1×10^{-5}, 5×10^{-5}, 1×10^{-4}, 5×10^{-4} e 1×10^{-3};
- α_{ℓ_0} com valores 1×10^{-6}, 5×10^{-6}, 1×10^{-5}, 5×10^{-5}, 1×10^{-4}, 5×10^{-4}, 1×10^{-3} e 5×10^{-3};
- β com valores 1, 2, 5, 10, 20, 50 e 100.

Assim, foram realizados 280 treinamentos para cada uma das estratégias de regularização baseadas na norma ℓ_0, além de 5 treinamentos com a regularização baseada na norma ℓ_2 para fins de comparação. Em todos os casos, utilizou-se o otimizador Adam, com taxa de aprendizado de 1×10^{-3}, *mini-batches* de tamanho 64, além de um total de 32 épocas, sendo que o modelo resultante de cada treinamento, em princípio, é o obtido ao final dessas 32 épocas. Um método de seleção alternativo foi também explorado com as mesmas configurações de treinamento. Nesse método alternativo, ao invés de selecionar o modelo resultante ao final das 32 épocas, o modelo com melhor acurácia em qualquer época do treinamento foi selecionado. O objetivo dessa abordagem é avaliar o desempenho máximo de cada abordagem, bem como verificar se o treinamento continuado com a norma ℓ_0 leva a uma degradação do modelo ou da

capacidade de compressão do mesmo. Nas figuras com resultados experimentais mostradas a seguir, as diferentes abordagens de compressão são referidas da seguinte forma:

- ℓ_2 corresponde à compressão com regularização por norma ℓ_2 exclusivamente;
- $\ell_2\ell_0$ corresponde à compressão com regularização combinada do Capítulo 4, a qual usa a aproximação exponencial para a norma ℓ_0;
- $\ell_2\ell_0$L corresponde à compressão com regularização combinada usando a aproximação linearizada para a norma ℓ_0.

Além disso, $\ell_2\ell_0$-M e $\ell_2\ell_0$L-M referem-se às abordagens que consideram o melhor modelo obtido ao longo do treinamento ao invés do modelo obtido ao final das 32 épocas.

Os resultados de acurácia máxima obtida por cada abordagem em função do nível de compressão da rede estão apresentados na Figura 21. A partir dessa figura, observa-se que os níveis de acurácia obtidos para taxas de compressão de até 64 vezes são bastante próximos para ambas abordagens baseadas na norma ℓ_0. Somente para maiores níveis de compressão é que a abordagem baseada na aproximação linearizada começa a apresentar alguma desvantagem. Visando quantificar melhor essa diferença, a Figura 22 apresenta o total de pesos após a poda da rede em função da acurácia obtida. A partir dessa figura, observa-se por exemplo que, para níveis de acurácia de 96.6%, apesar da diferença de níveis de compressão ser relativamente grande (em torno de 215$\times$ para a aproximação exponencial e 150$\times$ para a aproximação linearizada), a diferença em termos de número absoluto de pesos é de cerca de apenas 600 pesos, em uma rede que originalmente possui 266.610 pesos. Outro ponto importante a se destacar nessas figuras é que não foram observadas diferenças relevantes entre os resultados obtidos considerando o modelo ao final das 32 épocas ou o melhor modelo ao longo do treinamento.

Outro aspecto importante aqui analisado é a robustez das abordagens de compressão em relação à escolha dos hiperparâmetros. Nesse contexto, a Figura 23 apresenta curvas de percentual de configurações de hiperparâmetros em função da acurácia mínima obtida, visando fornecer uma noção de qual o percentual das configurações consideradas para hiperparâmetros que permitiram obter uma acurácia acima de um determinado valor. A partir de tal figura, nota-se que a abordagem de compressão que usa a regularização baseada na aproximação linearizada da norma ℓ_0 demonstrou-se mais robusta à escolha dos hiperparâmetros. Por exemplo, considerando um nível de acurácia mínima de 97,6% (1% abaixo da máxima obtida em todos experimentos), 54% das configurações de hiperparâmetros proporcionaram desempenho aceitável para a abordagem baseada na aproximação linearizada da norma ℓ_0, percentual esse que foi de 44% para a abordagem baseada na aproximação exponencial.

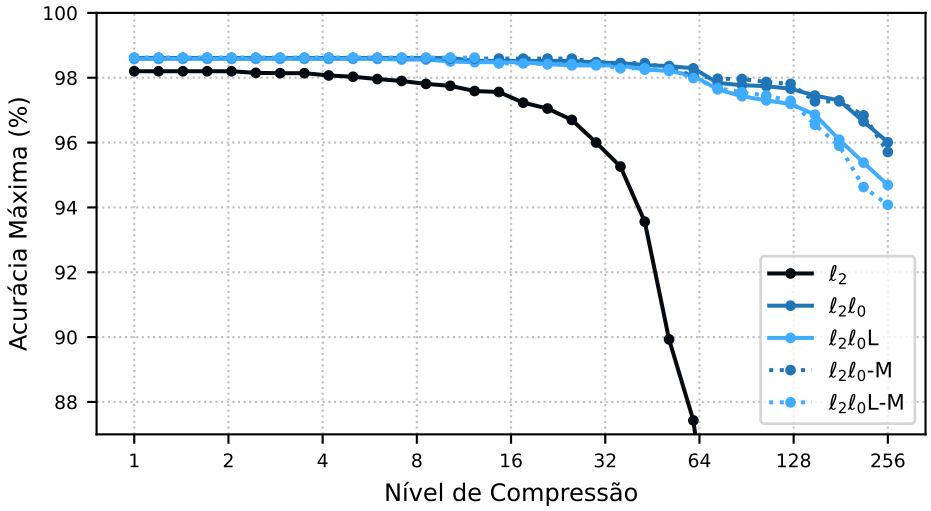


Figura 21 – Acurácia máxima obtida por cada estratégia em função do nível de compressão da rede Lenet-300-100 aplicada ao *dataset* MNIST.

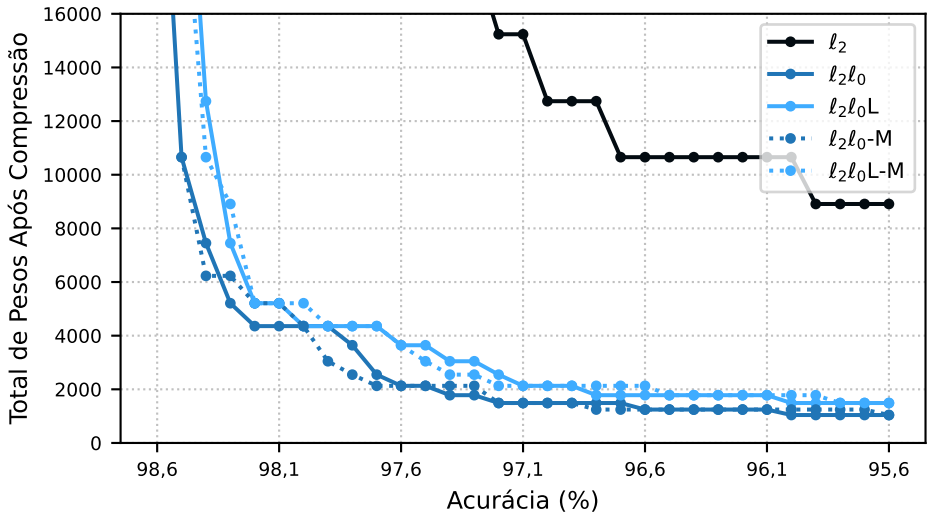


Figura 22 – Total de pesos após a compressão em função da acurácia para a rede Lenet-300-100 aplicada ao *dataset* MNIST.

5.5.2 Experimento 2. Compressão da Rede ResNet-50 para o CIFAR100

Neste segundo experimento, o objetivo é comparar o desempenho dos diferentes esquemas de compressão, agora para o problema mais complexo de compressão da rede ResNet-50 [55] (que possui cerca de 23,5 milhões de parâmetros) aplicada ao problema do *dataset* CIFAR-100 [117]. Por se tratar de um problema muito mais complexo computacionalmente, considera-se um número menor de combinações de valores dos hiperparâmetros relativos à regularização, os quais foram variados conforme descrito a seguir:

- α_{ℓ_0} com valores 1×10^{-7}, 5×10^{-7} e 1×10^{-6};
- β com valores 5, 20, 50, 75 e 100.

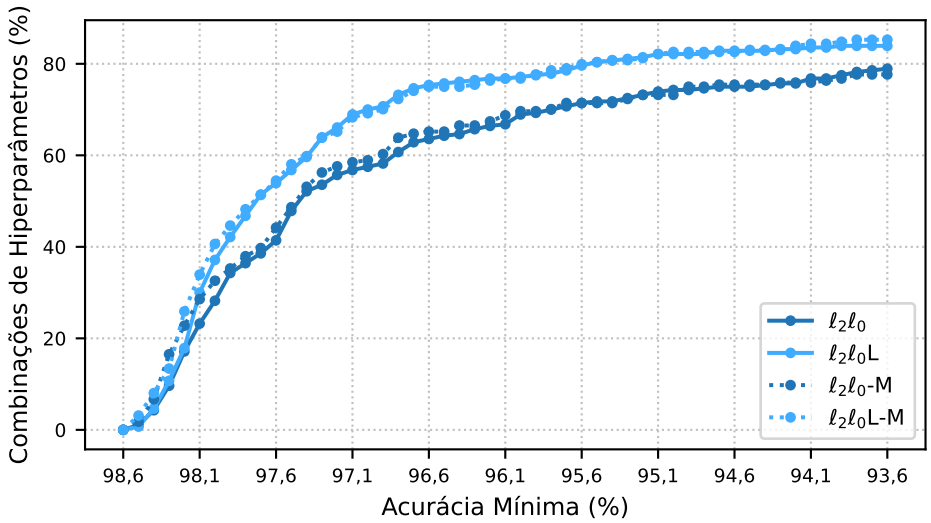


Figura 23 – Percentual das combinações consideradas de hiperparâmetros que proporcionaram uma certa acurácia mínima para a rede Lenet-300-100 aplicada ao *dataset* MNIST.

Ainda, para α_{ℓ_0}, foi utilizada a estratégia NORM, descrita na Seção 4.4.2, para definição de valores específicos para cada camada da rede. Com respeito a α_{ℓ_2}, o valor considerado foi 1×10^{-4} para os casos envolvendo regularizações por norma ℓ_0, enquanto no caso envolvendo regularização apenas por norma ℓ_2, foram considerados os valores 1×10^{-5}, 1×10^{-4}, 1×10^{-3}, 1×10^{-2} e 1×10^{-1}. Assim, tem-se um total de 15 treinamentos para cada uma das estratégias de regularização baseadas na norma ℓ_0, além de 5 treinamentos com a regularização baseada na norma ℓ_2. Em todos os casos, utilizou-se o otimizador baseado no algoritmo do gradiente descendente estocástico, com passo inicial de $0,1$ e decaimento de valor de passo baseado em *cosine decay*. O tamanho de *mini-batch* considerado foi 128 e os treinamentos foram feitos sempre com um total de 200 épocas. O modelo resultante de cada treinamento é o obtido ao final dessas 200 épocas.

A Figura 24 apresenta os resultados de acurácia máxima obtida para cada abordagem em função do nível de compressão da rede. O primeiro aspecto a se destacar nos resultados apresentados nessa figura é que, sem considerar qualquer compressão de rede (nível de compressão igual a 1,0), a abordagem baseada na aproximação linearizada da norma ℓ_0 foi a que obteve o melhor desempenho (76,16%), seguida pela abordagem baseada na aproximação exponencial (75,66%) e, por último, a abordagem baseada apenas na norma ℓ_2 (75,07%). Esses resultados superiores das abordagens baseadas em norma ℓ_0 sugerem que o próprio processo de compressão via poda da rede pode estar tendo um efeito adicional de regularização, promovendo uma melhor generalização para o conjunto de testes. Além disso, a maior robustez da abordagem baseada na aproximação linearizada da norma ℓ_0 em relação à escolha dos hiperparâmetros, já observada no experimento anterior, também pode estar contribuindo para

esse melhor desempenho, uma vez que o número de combinações de hiperparâmetros considerado é relativamente pequeno. Em relação ao desempenho das abordagens para níveis de compressão maiores do que 1,0, observa-se um comportamento similar ao já observado no experimento anterior (similaridade de acurácia até um certo nível e aproximação exponencial levando a um desempenho ligeiramente superior para níveis de compressão mais elevados).

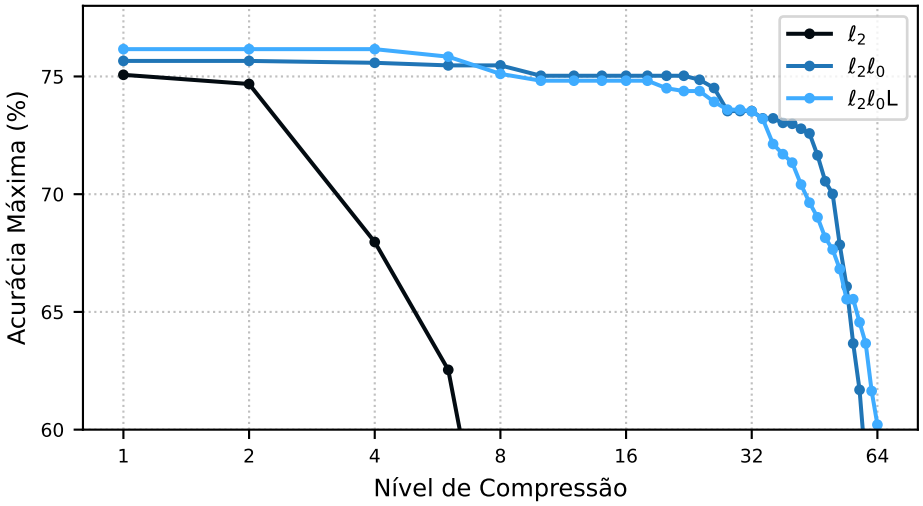


Figura 24 – Acurácia máxima de cada estratégia em função do nível de compressão para a rede ResNet-50 aplicada ao *dataset* CIFAR-100.

A Figura 25 apresenta o total de pesos após a poda da rede em função da acurácia obtida. Também dessa figura, nota-se o melhor desempenho de compressão da abordagem baseada na aproximação exponencial para níveis de acurácia mais baixos, apesar de que, para níveis de acurácia mais altos, o melhor desempenho foi o da abordagem baseada na aproximação linearizada em função da maior acurácia máxima obtida. Contudo, essa diferença de desempenho entre as duas abordagens baseadas na norma ℓ_0 é pequena. Por exemplo, considerando níveis de acurácia próximos de 74,1% (2% de redução em relação à máxima obtida), a diferença em termos de número absoluto de pesos entre as duas abordagens baseadas na norma ℓ_0 é de cerca de 75.000 pesos, em uma rede que originalmente possui cerca de 23,5 milhões de pesos.

Visando agora avaliar a robustez das abordagens de compressão em relação à escolha dos hiperparâmetros, a Figura 26 mostra curvas de percentual de configurações de hiperparâmetros em função da acurácia mínima obtida. Nessa figura, nota-se uma pequena vantagem da abordagem baseada na aproximação linearizada da norma ℓ_0. Entretanto, essa vantagem não é tão destacada quanto àquela observada no experimento anterior por conta do número relativamente pequeno de combinações de hiperparâmetros considerado no presente experimento.

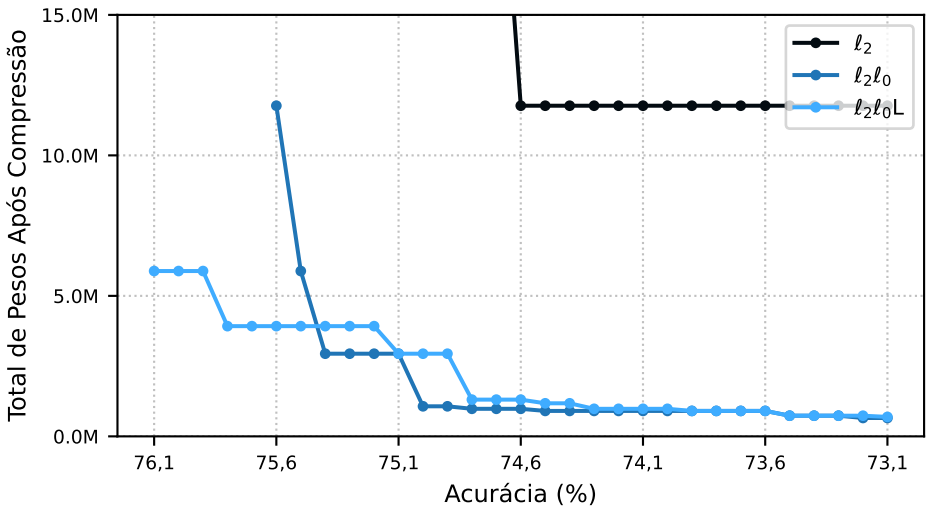


Figura 25 – Total de pesos após a compressão em função da acurácia para a rede ResNet-50 aplicada ao *dataset* CIFAR-100.

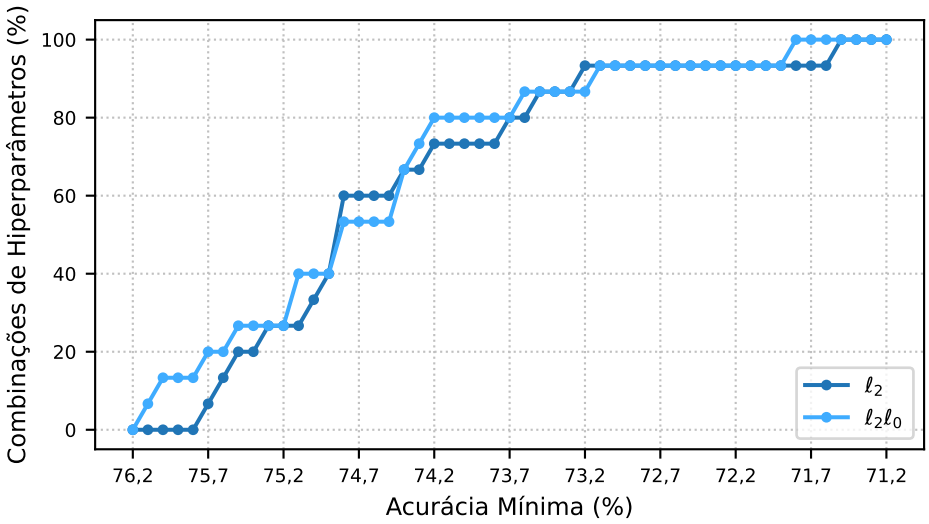


Figura 26 – Percentual das combinações consideradas de hiperparâmetros que proporcionaram uma acurácia mínima para a rede ResNet-50 aplicada ao *dataset* CIFAR-100.

5.6 CONCLUSÕES

O presente capítulo foi dedicado à apresentação de uma nova estratégia de compressão de redes neurais baseada em regularização combinada por normas ℓ_2 e ℓ_0 usando uma aproximação linearizada da norma ℓ_0. Em comparação com a estratégia do Capítulo 4, a nova abordagem envolve uma simplificação matemática e computacional do processo de regularização. Os resultados experimentais apresentados mostram que a regularização com aproximação linear da norma ℓ_0 é capaz de atingir um desempenho de compressão e acurácia próximo ao do esquema baseado na aproximação exponencial do Capítulo 4. Além disso, a nova abordagem mostrou-se mais robusta à escolha dos hiperparâmetros, o que é uma característica desejável em qualquer estratégia de compressão de redes neurais.

6 CONCLUSÕES FINAIS

Neste capítulo, as conclusões finais deste trabalho de pesquisa são apresentadas. Inicialmente, uma discussão acerca de cada capítulo e dos resultados obtidos é realizada. Em seguida, os esquemas de compressão propostos e os resultados experimentais obtidos são sumarizados. As considerações finais e ideias para continuação da pesquisa são abordadas ao final do capítulo.

6.1 SUMÁRIO E DISCUSSÃO DOS RESULTADOS

O Capítulo 1 deste trabalho abordou o crescente interesse e utilização das DNNs e destacou as propriedades das redes que têm contribuído para o bom desempenho em diversas aplicações. Na sequência, os elevados custos computacionais e ambientais foram discutidos como problemas comuns a serem tratados em aplicações práticas que envolvem uso de DNNs, uma vez que a complexidade dos modelos atuais impõe restrições para implantação em sistemas embarcados e em aplicações que exigem processamento em tempo real. Nesse contexto, o objetivo central desta tese é apresentado, tendo como foco estratégias que busquem a redução de complexidade da rede especialmente durante a inferência.

O Capítulo 2 tratou da fundamentação teórica para o desenvolvimento da tese, estruturando-se em torno das arquiteturas de redes neurais, do processo de aprendizagem e das estratégias de controle de complexidade. Em um primeiro momento, uma breve descrição histórica do desenvolvimento de redes neurais foi apresentada. Na sequência, os modelos de MLPs e CNNs foram detalhados. É salientado que as CNNs utilizam conectividade esparsa e compartilhamento de pesos, propriedades que garantem eficiência paramétrica e as consolidam como arquitetura dominante em visão computacional, em contraste com a alta complexidade das MLPs. Em seguida, o capítulo abordou o processo de treinamento supervisionado, focando no algoritmo *backpropagation* baseado no algoritmo do gradiente descendente. Por fim, foram discutidos o conceito de generalização e as técnicas de regularização, essenciais para a manutenção da capacidade preditiva e desempenho dos modelos considerando conjuntos de dados não vistos durante treinamento.

No Capítulo 3, técnicas de compressão de redes neurais profundas foram descritas. Tais técnicas têm como principal objetivo endereçar as demandas de inferência e treinamento dos modelos atuais, principalmente em aplicações envolvendo processamento em tempo real e/ou em sistemas embarcados. Nesse contexto, as decomposições matriciais e técnicas que exploram características específicas de hardware foram discutidas, sendo citados os benefícios e os desafios ao adotar essas formas de compressão de redes. Na sequência, a poda de redes neurais foi apresentada como uma importante e eficiente estratégia de compressão que baseia-se em identificar pesos

e/ou neurônios com menor relevância para o desempenho do modelo, de forma que a remoção de tais parâmetros não represente degradação significativa de desempenho. Alguns critérios para identificar e remover os pesos da rede foram apresentados, como é o caso da poda baseada na magnitude absoluta do peso em nível global ou por camada. Na sequência do capítulo, foi estabelecida uma classificação das técnicas de poda em termos da abordagem de remoção: a poda estruturada e a não estruturada. A poda não estruturada consiste na remoção de pesos individuais e resulta em uma rede altamente esparsa, com elevado potencial de compressão e com mínima degradação de desempenho. Conforme abordado, embora o processamento eficiente dessas estruturas irregulares seja um desafio para as arquiteturas de *hardware* convencionais, tais como as GPUs que são otimizadas para operações densas, plataformas de processamento dedicadas podem ser consideradas para implantação de forma eficiente dos modelos esparsos, economizando recursos como memória e consumo de unidades de processamento. Ao final do capítulo, estratégias de regularização baseadas em penalização por norma vetorial foram apresentadas, dando luz às estratégias de indução de esparsidade na rede usadas durante o processo de treinamento.

O tema central do Capítulo 4 foi uma nova abordagem de regularização baseada na norma ℓ_0. Essa abordagem foi desenvolvida considerando uma aproximação exponencial diferenciável da norma ℓ_0. Assim, no início do Capítulo 4, tem-se o desenvolvimento da equação de atualização dos pesos a partir da regularização por norma ℓ_0 proposta. Na sequência, uma análise teórica acerca das características das regularizações ℓ_2, ℓ_1 e ℓ_0 foi realizada, na qual comparou-se o efeito dessas normas durante o treinamento de redes. Cada uma dessas normas apresenta importantes diferenças nos efeitos de atração para zero durante treinamento do modelo, destacando-se que: a penalização pela norma ℓ_2 depende linearmente da magnitude do peso e, portanto, resulta em uma atração para zero decrescente à medida que o peso aproxima-se de zero; e a penalização pela norma ℓ_1 tem efeito de penalização constante controlado por parâmetros definidos durante treinamento. Menciona-se que a penalização por norma ℓ_1 tem um problema de objetivos conflitantes, pois o valor do coeficiente de penalização contribui de maneira inversa para evitar o *overfitting* ou para favorecer a atração para zero visando a poda de pesos. Finalmente, mostrou-se que a penalização pela norma ℓ_0 tende a ser muito pequena à medida que a magnitude do peso aumenta (devido ao termo $e^{-\beta|w_j|}$) e, assim, a regularização proposta não tem capacidade de tratar o *overfitting*. Por outro lado, em termos de atração a zero para os coeficientes menores, o efeito é mais forte e aumenta à medida que o peso se torna menor resultando em uma atração a zero que aumenta à medida que o peso se aproxima de zero. Assim, redes neurais com características naturalmente esparsas são induzidas durante treinamento, de forma que o modelo resultante tem alta capacidade de compressão associada a uma pequena, ou desprezível, perda de acurácia. Ainda no Capítulo 4, um esquema

de compressão de redes com normas ℓ_2 e ℓ_0 combinadas foi proposto, com o objetivo de endereçar tanto o problema de *overfitting* quanto induzir redes esparsas. Na última parte do capítulo diversos resultados experimentais foram apresentados, confrontando a compressão e a acurácia obtidas com o esquema proposto e com de outros trabalhos da literatura, comprovando a eficácia da nova abordagem apresentada.

No Capítulo 5, uma aproximação linearizada da norma ℓ_0 foi explorada para o desenvolvimento de um novo esquema de regularização. Essa aproximação foi obtida a partir da aproximação da função exponencial centrada em zero através da série de Taylor. Para fins de simplicidade matemática e computacional, apenas o primeiro termo da série de Taylor foi considerado para obter a referida aproximação. A partir de tal aproximação, desenvolveu-se a equação de atualização dos pesos considerando a regularização proposta. Ainda no Capítulo 5, foi feita uma análise do comportamento de atração para zero da nova regularização baseada na aproximação linearizada da norma ℓ_0, comparando-a com as demais regularizações por norma consideradas neste trabalho. Dessa análise, conclui-se que a abordagem de regularização obtida com a aproximação linearizada exibe um comportamento híbrido, combinando as características da regularização por norma ℓ_1 para pesos de menor magnitude e ℓ_0 para pesos de maior magnitude. Na sequência do capítulo, resultados experimentais foram apresentados, comprovando que a regularização baseada na aproximação linearizada da norma ℓ_0 leva a níveis de compressão próximos aos da regularização discutida no Capítulo 4, com a vantagem de uma maior robustez em relação à escolha dos hiperparâmetros de treinamento.

6.2 CONSIDERAÇÕES FINAIS

Esta tese de doutorado foi dedicada ao desenvolvimento de novas estratégias de compressão de redes neurais profundas, com foco na indução de esparsidade durante o processo de treinamento via regularização por norma ℓ_0, seguida de poda não estruturada. As contribuições originais principais foram detalhadas nos Capítulos 4 e 5, sendo elas centradas em duas novas estratégias de regularização obtidas a partir de diferentes aproximações diferenciáveis da norma ℓ_0. Para ambas estratégias, foram realizadas análises teóricas detalhadas sobre os efeitos de atração para zero, envolvendo também comparações com as regularizações por norma ℓ_1 e ℓ_2. Tais análises estabeleceram os benefícios e desafios intrínsecos a cada tipo de regularização por norma no contexto da indução de esparsidade. Por fim, a eficácia das abordagens propostas foi verificada por meio de resultados experimentais robustos obtidos no treinamento de modelos de redes neurais profundas, onde as métricas de compressão e acurácia corroboraram as premissas e conclusões da análise teórica conduzida.

6.3 SUGESTÕES PARA TRABALHOS FUTUROS

Para trabalhos futuros sugere-se a avaliação de ganhos de treinamento e inferência de redes neurais considerando plataformas de *hardware* específicas, tais como FPGAs. Nessa avaliação, métricas de compressão da rede associadas à demanda por unidades lógicas, blocos de processamento digitais, memória e uso de flip-flops podem ser consideradas. Além disso, a redução no tempo de treinamento obtido ao utilizar-se a aproximação linearizada da norma ℓ_0 pode ser considerada no escopo de plataformas sem acesso a *hardware* dedicado para cálculo de funções exponenciais. Também sugere-se a exploração de técnicas de aproximação da norma ℓ_0 almejando poda estrutural. Uma forma para alcançar esse objetivo seria utilizar a regularização esparsa de grupo [40], em conjunto com a aproximação da norma ℓ_0 aqui apresentada, de forma a potencializar o efeito de esparsidade em tal regularização.

REFERÊNCIAS

[1] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*. MIT press Cambridge, 2016, vol. 1.

[2] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.

[3] K. P. Murphy, *Machine learning : a probabilistic perspective*. Cambridge, Mass. [u.a.]: MIT Press, 2013.

[4] R. Akita, A. Yoshihara, T. Matsubara, and K. Uehara, "Deep learning for stock prediction using numerical and textual information," in *2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS)*. IEEE, 2016, pp. 1–6.

[5] M. R. Vargas, B. S. De Lima, and A. G. Evsukoff, "Deep learning for stock market prediction from financial news articles," in *2017 IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA)*. IEEE, 2017, pp. 60–65.

[6] A. Tsantekidis, N. Passalis, A. Tefas, J. Kanniainen, M. Gabbouj, and A. Iosifidis, "Forecasting stock prices from the limit order book using convolutional neural networks," in *2017 IEEE 19th Conference on Business Informatics (CBI)*, vol. 1. IEEE, 2017, pp. 7–12.

[7] Z. Jianqiang, G. Xiaolin, and Z. Xuejun, "Deep convolution neural networks for twitter sentiment analysis," *IEEE Access*, vol. 6, pp. 23 253–23 260, 2018.

[8] A. Hassan and A. Mahmood, "Deep learning approach for sentiment analysis of short texts," in *2017 3rd international conference on control, automation and robotics (ICCAR)*. IEEE, 2017, pp. 705–710.

[9] T. Young, D. Hazarika, S. Poria, and E. Cambria, "Recent trends in deep learning based natural language processing," *ieee Computational intelligenCe magazine*, vol. 13, no. 3, pp. 55–75, 2018.

[10] P. Kłosowski, "Deep learning for natural language processing and language modelling," in *2018 Signal Processing: Algorithms, Architectures, Arrangements, and Applications (SPA)*. IEEE, 2018, pp. 223–228.

[11] A. B. Nassif, I. Shahin, I. Attili, M. Azzeh, and K. Shaalan, "Speech recognition using deep neural networks: A systematic review," *IEEE Access*, 2019.

[12] Y.-H. Liao, H.-y. Lee, and L.-s. Lee, “Towards structured deep neural network for automatic speech recognition,” in *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE, 2015, pp. 137–144.

[13] T. Kosaka, K. Konno, and M. Kato, “Deep neural network-based speech recognition with combination of speaker-class models,” in *2015 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*. IEEE, 2015, pp. 1203–1206.

[14] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner *et al.*, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[15] A. Bandhu and S. S. Roy, “Classifying multi-category images using deep learning: A convolutional neural network model,” in *2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)*. IEEE, 2017, pp. 915–919.

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[17] S. Han, H. Mao, and W. J. Dally, “Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding,” in *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2016. [Online]. Available: http://arxiv.org/abs/1510.00149

[18] Y. Cheng, D. Wang, P. Hu, P. Zhou, C. Huang, J. Li, L. Chen, and X. Qian, “A survey of model compression and acceleration for deep neural networks,” *Journal of Big Data*, vol. 5, no. 1, pp. 1–33, 2018.

[19] E. Strubell, A. Ganesh, and A. McCallum, “Energy and policy considerations for deep learning in nlp,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 3645–3650. [Online]. Available: https://aclanthology.org/P19-1355

[20] R. Schwartz, J. Dodge, N. A. Smith, and O. Etzioni, “Green ai,” *Communications of the ACM*, vol. 63, no. 12, pp. 54–63, 2020. [Online]. Available: https://doi.org/10.1145/3381831

[21] L. H. Kaack, S. Stoll, F. van der Plas *et al.*, “Aligning artificial intelligence with climate change mitigation,” *Nature Climate Change*, vol. 12, pp. 518–527, 2022. [Online]. Available: https://doi.org/10.1038/s41558-022-01377-7

[22] D. Patterson, J. Gonzalez, U. Hölzle, Q. Le, C. Liang, L.-M. Munguia, D. Rothchild, D. R. So, M. Texier, and J. Dean, "The carbon footprint of machine learning training will plateau, then shrink," *Computer*, vol. 55, no. 7, pp. 18–28, 2022.

[23] N. C. Thompson, K. Greenewald, K. Lee, and G. F. Manso, "The computational limits of deep learning," *ArXiv preprint arXiv:2007.05558*, 2020. [Online]. Available: https://arxiv.org/abs/2007.05558

[24] M. Denil, B. Shakibi, L. Dinh, M. A. Ranzato, and N. de Freitas, "Predicting parameters in deep learning," in *Advances in Neural Information Processing Systems 26*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2013, pp. 2148–2156. [Online]. Available: http://papers.nips.cc/paper/5025-predicting-parameters-in-deep-learning.pdf

[25] T. N. Sainath, B. Kingsbury, V. Sindhwani, E. Arisoy, and B. Ramabhadran, "Low-rank matrix factorization for deep neural network training with high-dimensional output targets," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, May 2013, pp. 6655–6659.

[26] X. Zhang, , , K. He, and J. Sun, "Efficient and accurate approximations of nonlinear convolutional networks," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 1984–1992.

[27] Y. Sun, X. Liu, and L. Liang, "Retrain-free fully connected layer optimization using matrix factorization," in *2017 IEEE International Conference on Image Processing (ICIP)*, Sep. 2017, pp. 3914–3918.

[28] V. Vanhoucke, A. Senior, and M. Z. Mao, "Improving the speed of neural networks on cpus," in *Proc. NIPS2011 Workshop on Deep Learn. Unsupervised Feature Learn.*, Granada, Spain, Dec. 2011, pp. 1–8.

[29] K. Hwang and W. Sung, "Fixed-point feedforward deep neural network design using weights +1, 0, and -1," in *2014 IEEE Workshop on Signal Processing Systems (SiPS)*, Oct 2014, pp. 1–6.

[30] S. Anwar, K. Hwang, and W. Sung, "Fixed point optimization of deep convolutional neural networks for object recognition," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, April 2015, pp. 1131–1135.

[31] Y. LeCun, J. Denker, and S. Solla, "Optimal brain damage," *Advances in neural information processing systems*, vol. 2, 1989.

[32] B. Hassibi and D. G. Stork, "Second order derivatives for network pruning: Optimal brain surgeon," in *Advances in Neural Information Processing*

Systems 5, S. J. Hanson, J. D. Cowan, and C. L. Giles, Eds. Morgan-Kaufmann, 1993, pp. 164–171. [Online]. Available: http://papers.nips.cc/paper/647-second-order-derivatives-for-network-pruning-optimal-brain-surgeon.pdf

[33] Y. Guo, A. Yao, and Y. Chen, "Dynamic network surgery for efficient dnns," *Advances in neural information processing systems*, vol. 29, 2016.

[34] D. Yu, F. Seide, G. Li, and L. Deng, "Exploiting sparseness in deep neural networks for large vocabulary speech recognition," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2012, pp. 4409–4412.

[35] L. Mauch and B. Yang, "A novel layerwise pruning method for model reduction of fully connected deep neural networks," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2017, pp. 2382–2386.

[36] S. Srinivas and R. V. Babu, "Data-free parameter pruning for deep neural networks," in *Proceedings of the British Machine Vision Conference (BMVC)*, 2015, pp. 31.1–31.12. [Online]. Available: https://dx.doi.org/10.5244/C.29.31

[37] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network," in *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds. Curran Associates, Inc., 2015, pp. 1135–1143. [Online]. Available: http://papers.nips.cc/paper/5784-learning-both-weights-and-connections-for-efficient-neural-network.pdf

[38] W. Pan, H. Dong, and Y. Guo, "Dropneuron: Simplifying the structure of deep neural networks," *arXiv preprint arXiv:1606.07326*, 2016.

[39] C. Louizos, M. Welling, and D. P. Kingma, "Learning sparse neural networks through *l_*0 regularization," in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. [Online]. Available: https://openreview.net/forum?id=H1Y8hhg0b

[40] S. Scardapane, D. Comminiello, A. Hussain, and A. Uncini, "Group sparse regularization for deep neural networks," *Neurocomputing*, vol. 241, pp. 81–89, 2017.

[41] D. L. Donoho, "Compressed sensing," *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1289–1306, April 2006.

[42] Y. Li and S. Ji, “L0-arm: Network sparsification via stochastic binary optimization,” in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery in Databases*, Würzburg, Germany, Sept. 2019, pp. 432–448.

[43] T. Zhang, S. Ye, K. Zhang, J. Tang, W. Wen, M. Fardad, and Y. Wang, “A systematic DNN weight pruning framework using alternating direction method of multipliers,” in *Computer Vision – ECCV 2018*, 2018, pp. 191–207.

[44] M. A. Carreira-Perpinan and Y. Idelbayev, “"learning-compression"algorithms for neural net pruning,” in *Proc. 2018 IEEE/CVF Conf. Comput. Vision Pattern Recognit.*, June 2018, pp. 8532–8541.

[45] Y. Idelbayev and M. A. Carreira-Perpinan, “Exploring the effect of l0/l2 regularization in neural network pruning using the LC toolkit,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Singapore, Singapore, May 2022, pp. 3373–3377.

[46] Y. Gu, J. Jin, and S. Mei, “*l*_{0} norm constraint lms algorithm for sparse system identification,” *IEEE Signal Processing Letters*, vol. 16, no. 9, pp. 774–777, 2009.

[47] X. Wang and W. Jia, “Optimizing edge ai: A comprehensive survey on data, model, and system strategies,” *arXiv preprint arXiv:2501.03265*, 2025.

[48] H. Cheng, M. Zhang, and J. Q. Shi, “A survey on deep neural network pruning: Taxonomy, comparison, analysis, and recommendations,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[49] T. Liang, J. Glossner, L. Wang, S. Shi, and X. Zhang, “Pruning and quantization for deep neural network acceleration: A survey,” *Neurocomputing*, vol. 461, pp. 370–403, 2021.

[50] R. Abdulkadirov, P. Lyakhov, and N. Nagornov, “Survey of optimization algorithms in modern neural networks,” *Mathematics*, vol. 11, no. 11, p. 2466, 2023.

[51] L. S. Barth and P. von Petersenn, “Efficient compression of neural networks and datasets,” *arXiv preprint arXiv:2505.17469*, 2025.

[52] W. S. McCulloch and W. Pitts, “A logical calculus of the ideas immanent in nervous activity,” *The bulletin of mathematical biophysics*, vol. 5, pp. 115–133, 1943.

[53] F. Rosenblatt, “The perceptron: a probabilistic model for information storage and organization in the brain.” *Psychological review*, vol. 65, no. 6, p. 386, 1958.

[54] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors,” *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[55] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2016-Decem, pp. 770–778, 2016.

[56] F. Rosenblatt, “Perceptron simulation experiments,” *Proceedings of the IRE*, vol. 48, no. 3, pp. 301–309, 1960.

[57] H.-D. Block, “The perceptron: A model for brain functioning. i,” *Reviews of Modern Physics*, vol. 34, no. 1, p. 123, 1962.

[58] A. G. Konheim, “A geometric convergence theorem for the perceptron,” *Journal of the Society for Industrial and Applied Mathematics*, vol. 11, no. 1, pp. 1–14, 1963.

[59] A. E. MURRAY, “Perceptron applications in photo interpretation,” *Photogramm. Engng. 27 (4), 627*, vol. 637, 1961.

[60] M. L. Minsky and S. A. Papert, *Perceptrons: An Introduction to Computational Geometry*. MIT Press, 1969.

[61] C. C. Aggarwal *et al.*, *Neural networks and deep learning*. Springer, 2018.

[62] G. Cybenko, “Approximation by superpositions of a sigmoidal function,” *Mathematics of control, signals and systems*, vol. 2, no. 4, pp. 303–314, 1989.

[63] K. Hornik, “Approximation capabilities of multilayer feedforward networks,” *Neural networks*, vol. 4, no. 2, pp. 251–257, 1991.

[64] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[65] T. Kohonen, “Self-organized formation of topologically correct feature maps,” *Biological cybernetics*, vol. 43, no. 1, pp. 59–69, 1982.

[66] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” in *International Conference on Learning Representations (ICLR)*, 2014. [Online]. Available: https://arxiv.org/abs/1312.6114

[67] Y. Bengio, A. Courville, and P. Vincent, “Representation learning: A review and new perspectives,” *IEEE TPAMI*, vol. 35, no. 8, pp. 1798–1828, 2013.

[68] T. Talaei Khoei, H. Ould Slimane, and N. Kaabouch, “Deep learning: systematic review, models, challenges, and research directions,” *Neural Computing and Applications*, vol. 35, no. 31, pp. 23 103–23 124, 2023.

[69] L. Bottou, F. E. Curtis, and J. Nocedal, “Optimization methods for large-scale machine learning,” *SIAM Review*, vol. 60, no. 2, pp. 223–311, 2018.

[70] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015. [Online]. Available: http://arxiv.org/abs/1412.6980

[71] S. Raschka, “Model evaluation, model selection, and algorithm selection in machine learning,” *arXiv preprint arXiv:1811.12808*, 2018.

[72] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: a simple way to prevent neural networks from overfitting,” *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[73] J. Gusak, M. Kholiavchenko, E. Ponomarev, L. Markeeva, P. Blagoveschensky, A. Cichocki, and I. Oseledets, “Automated multi-stage compression of neural networks,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, 2019, pp. 0–0.

[74] M. Tukan, A. Maalouf, M. Weksler, and D. Feldman, “No fine-tuning, no cry: Robust svd for compressing deep networks,” *Sensors*, vol. 21, no. 16, p. 5599, 2021.

[75] C. Runkel, N. Kuete Meli, J. Lukasik, A. Biguri, C.-B. Schönlieb, and M. Moeller, “Smooth model compression without fine-tuning,” *arXiv preprint arXiv:2505.24469*, 05 2025.

[76] B. Zhang, D. Cheng, Y. Zhang, F. Liu, and J. Tian, “Lossless model compression via joint low-rank factorization optimization,” *arXiv preprint arXiv:2412.06867*, 2024.

[77] H. Liu, Z. Jiang, B. Liu, L. Li, and X. Zhang, “Low-rank compression of cnn models based on efficient rank parameter learning,” in *2024 China Automation Congress (CAC)*, 2024, pp. 1670–1675.

[78] R. Banner, Y. Nahshan, and D. Soudry, “Post training 4-bit quantization of convolutional networks for rapid-deployment,” *Advances in neural information processing systems*, vol. 32, 2019.

[79] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko, “Quantization and training of neural networks for efficient integer-arithmetic-only inference,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 2704–2713.

[80] D. Blalock, J. J. Gonzalez Ortiz, J. Frankle, and J. Guttag, “What is the state of neural network pruning?” *Proceedings of machine learning and systems*, vol. 2, pp. 129–146, 2020.

[81] Z. Li, H. Li, and L. Meng, “Model compression for deep neural networks: A survey,” *Computers*, vol. 12, no. 3, p. 60, 2023.

[82] Y. LeCun, J. Denker, and S. Solla, “Optimal brain damage,” *Advances in neural information processing systems*, vol. 2, 1990.

[83] K. Neklyudov, D. Molchanov, A. Ashukha, and D. P. Vetrov, “Structured bayesian pruning via log-normal multiplicative noise,” in *Advances in Neural Information Processing Systems*, 2017, pp. 6775–6784.

[84] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, “Pruning filters for efficient convnets,” in *International Conference on Learning Representations (ICLR)*, 2017. [Online]. Available: https://arxiv.org/abs/1608.08710

[85] H. Tanaka, D. Kunin, D. L. Yamins, and S. Ganguli, “Pruning neural networks without any data by iteratively conserving synaptic flow,” *Advances in neural information processing systems*, vol. 33, pp. 6377–6389, 2020.

[86] Z. Liu, M. Liang, S.-H. Zhan, J. Sheth, and C.-L. Yuan, “The unreasonable effectiveness of random pruning: Return of the most naive baseline for sparse training,” in *International Conference on Learning Representations (ICLR)*, 2022.

[87] L. Deng, G. Li, S. Han, L. Shi, and Y. Xie, “Model compression and hardware acceleration for neural networks: A comprehensive survey,” *Proceedings of the IEEE*, vol. 108, no. 4, pp. 485–532, 2020.

[88] B. Liu, M. Wang, H. Foroosh, M. Tappen, and M. Pensky, “Sparse convolutional neural networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 806–814.

[89] Y. Choi, M. El-Khamy, and J. Lee, “Compression of deep convolutional neural networks under joint sparsity constraints,” *arXiv preprint arXiv:1805.08303*, 2018.

[90] C. Lin, Z. Zhong, W. Wei, and J. Yan, “Synaptic strength for convolutional neural network,” *advances in neural information processing systems*, vol. 31, 2018.

[91] Y. Ji, L. Liang, L. Deng, Y. Zhang, Y. Zhang, and Y. Xie, “Tetris: Tile-matching the tremendous irregular sparsity,” *Advances in neural information processing systems*, vol. 31, 2018.

[92] J. Yu, A. Lukefahr, D. Palframan, G. Dasika, R. Das, and S. Mahlke, "Scalpel: Customizing dnn pruning to the underlying hardware parallelism," *ACM SIGARCH Computer Architecture News*, vol. 45, no. 2, pp. 548–560, 2017.

[93] T. Zhang, S. Ye, K. Zhang, X. Ma, N. Liu, L. Zhang, J. Tang, K. Ma, X. Lin, M. Fardad *et al.*, "Structadmm: A systematic, high-efficiency framework of structured weight pruning for dnns," *arXiv preprint arXiv:1807.11091*, 2018.

[94] X. Ma, W. Niu, T. Zhang, S. Liu, S. Lin, H. Li, W. Wen, X. Chen, J. Tang, K. Ma *et al.*, "An image enhancing pattern-based sparsity for real-time inference on mobile devices," in *European Conference on Computer Vision*. Springer, 2020, pp. 629–645.

[95] Y. Wang, Z. Lu, P. Zhou, and H. Ren, "Sconv: A sparse cnn accelerator on fpga for resource-constrained edge devices," in *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2020, pp. 1–5.

[96] L. Mancera and J. Portilla, "L0-norm-based sparse representation through alternate projections," in *Proc. IEEE Int. Conf. Image Process.*, Atlanta, GA, Oct. 2006, pp. 2089–2092.

[97] Q. Xie, C. Li, B. Diao, Z. An, and Y. Xu, "L0 regularization based fine-grained neural network pruning method," in *2019 11th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)*. IEEE, 2019, pp. 1–4.

[98] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. B. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *CoRR*, vol. abs/1408.5093, 2014. [Online]. Available: http://arxiv.org/abs/1408.5093.

[99] C. Louizos, K. Ullrich, and M. Welling, "Bayesian compression for deep learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, 2017. [Online]. Available: https://doi.org/10.48550/arXiv.1705.08665

[100] D. Molchanov, A. Ashukha, and D. Vetrov, "Variational dropout sparsifies deep neural networks," in *Proc. Int. Conf. Mach. Learn.*, Sydney, Australia, Aug. 2017, pp. 2498–2507.

[101] D. T. Phan, L. M. Nguyen, N. H. Nguyen, and J. R. Kalagnanam, "Pruning deep neural networks with ℓ_0-constrained optimization," in *Proc. IEEE Int. Conf. Data Mining*, Sorrento, Italy, Nov. 2020, pp. 1214–1219.

[102] X. Ding, G. Ding, X. Zhou, Y. Guo, J. Han, and J. Liu, "Global sparse momentum SGD for pruning very deep neural networks," in *Proc. Int. Conf. Neural Inf. Proc. Syst.*, Vancouver, Canada, Dec. 2019, pp. 1–13.

[103] X. Xiao and Z. Wang, "Autoprune: Automatic network pruning by regularizing auxiliary parameters," in *Proc. Int. Conf. Neural Inf. Proc. Syst.*, Vancouver, Canada, Dec. 2019, pp. 1–11.

[104] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. 3rd Int. Conf. Learning Representations*, San Diego, CA, May 2015, pp. 1–14.

[105] Z. Huang and N. Wang, "Data-driven sparse structure selection for deep neural networks," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Munich, Germany, Sep. 2018, pp. 304–320.

[106] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, "Pruning filters for efficient ConvNets," in *Proc. Int. Conf. Learn. Representations*, Toulon, France, Apr. 2017, pp. 1–13.

[107] C. Zhao, B. Ni, J. Zhang, Q. Zhao, W. Zhang, and Q. Tian, "Variational convolutional neural network pruning," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2019-June, pp. 2775–2784, 2019.

[108] P. Singh, V. K. Verma, P. Rai, and V. P. Namboodiri, "Play and prune: Adaptive filter pruning for deep model compression," in *Proc. 28th Int. Joint Conf. Artificial Intelligence*, Macao, China, 2019, p. 3460–3466.

[109] B. O. Ayinde, T. Inanc, and J. M. Zurada, "Redundant feature pruning for accelerated inference in deep neural networks," *Neural Networks*, vol. 118, pp. 148–158, 2019.

[110] T. Li, B. Wu, Y. Yang, Y. Fan, Y. Zhang, and W. Liu, "Compressing convolutional neural networks via factorized convolutional filters," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2019-June, pp. 3972–3981, 2019.

[111] J. Ye, X. Lu, Z. Lin, and J. Z. Wang, "Rethinking the smaller-norm-less-informative assumption in channel pruning of convolution layers," in *Proc. Int. Conf. Learn. Representations*, Vancouver, Canada, Apr. 2018, pp. 1–11.

[112] Y. Li, S. Gu, K. Zhang, L. Van Gool, and R. Timofte, "DHP: Differentiable meta pruning via hypernetworks," in *Proc. Eur. Conf. Comput. Vision*, Online, Aug. 2020, pp. 608–624.

[113] L.-N. Wang, W. Liu, X. Liu, G. Zhong, P. P. Roy, J. Dong, and K. Huang, "Compressing deep networks by neuron agglomerative clustering," *Sensors*, vol. 20, no. 21, pp. 1–16, 2020.

[114] M. Wielgosz, E. Jamro, and K. Wiatr, “Highly efficient structure of 64-bit exponential function implemented in fpgas,” vol. 4943, 03 2008, pp. 272–277.

[115] E. Beck, E. L. O. Batista, and R. Seara, “Norm-constrained adaptive algorithms for sparse system identification based on projections onto intersections of hyperplanes,” *Signal Processing*, vol. 118, pp. 259–271, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0165168415002182

[116] D. Bertsekas, *Convex optimization algorithms*. Athena Scientific, 2015.

[117] A. Krizhevsky, G. Hinton *et al.*, “Learning multiple layers of features from tiny images.(2009),” 2009.