UNIVERSIDADE FEDERAL DO MARANHÃO

Programa de Pós-Graduação em Ciência da Computação

Carlos Eduardo Nascimento Cajado

Treinamento de funções de ativação em redes neurais artificiais: uma abordagem com regressão polinomial

São Luís
2026

CARLOS EDUARDO NASCIMENTO CAJADO

TREINAMENTO DE FUNÇÕES DE ATIVAÇÃO EM REDES NEURAIS ARTIFICIAIS: UMA ABORDAGEM COM REGRESSÃO POLINOMIAL

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal do Maranhão, como parte dos requisitos necessários para obtenção do grau de Mestre em Ciência da Computação.

Orientador: Prof. Dr. Areolino de Almeida Neto

SÃO LUÍS - MA
2026

Ficha gerada por meio do SIGAA/Biblioteca com dados fornecidos pelo(a) autor(a).
Diretoria Integrada de Bibliotecas/UFMA

Cajado, Carlos Eduardo Nascimento.
Treinamento de funções de ativação em redes neurais artificiais: uma abordagem com regressão polinomial / Carlos Eduardo Nascimento Cajado. - 2026.
82 f.

Orientador(a): Areolino de Almeida Neto.
Dissertação (Mestrado) - Programa de Pós-graduação em Ciência da Computação/ccet, Universidade Federal do Maranhão, São Luís, 2026.

1. Deep Learning. 2. Funções de Ativação Treináveis. 3. Regressão Polinomial. I. de Almeida Neto, Areolino. II. Título.

CARLOS EDUARDO NASCIMENTO CAJADO

TREINAMENTO DE FUNÇÕES DE ATIVAÇÃO EM REDES NEURAIS ARTIFICIAIS: UMA ABORDAGEM COM REGRESSÃO POLINOMIAL

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal do Maranhão, como parte dos requisitos necessários para obtenção do grau de Mestre em Ciência da Computação.

Prof. Dr. Areolino de Almeida Neto
Universidade Federal do Maranhão
Orientador

Prof. Dr. Omar Andres Carmona Cortes
Universidade Federal do Maranhão
Examinador Interno

Prof. Dr. Alexandre Cesar Muniz de Oliveira
Universidade Federal do Maranhão
Examinador Interno

São Luís - MA
2026

Dedico este trabalho a Deus, pela força, sabedoria e por me guiar em cada passo desta jornada. À minha esposa, pelo apoio, paciência e compreensão durante todo o processo. Ao meu orientador, pela orientação, confiança e contribuição essencial para a realização deste trabalho.

AGRADECIMENTOS

Primeiramente, agradeço a Deus por me conceder saúde, força e sabedoria para superar todas as dificuldades encontradas ao longo desta jornada acadêmica, iluminando meu caminho e fortalecendo minha fé nos momentos de incerteza.

À minha família, em especial aos meus pais Domingos Ferreira Cajado e Francisca das Chagas Cajado, pelo apoio constante, dedicação e pelos valores transmitidos, fundamentais para minha formação pessoal e profissional.

À minha esposa amada, Adrielle Campelo Cunha Cajado, por fazer parte da minha caminhada, não apenas com compreensão, mas também com incentivo nos momentos de dificuldade. Fique registrado que esta conquista também é sua.

Ao meu orientador, professor Areolino de Almeida Neto, pelo suporte, pelas correções e pelo incentivo ao desenvolvimento deste e de outros projetos, cuja parceria espero que continue forte e duradoura.

Aos colegas do Laboratório InovTec, Samuel Pereira e José Raimundo, pela ajuda e suporte durante a pesquisa. Por fim, agradeço a todos que, de alguma forma, contribuíram para a minha formação, direta ou indiretamente. Meus sinceros agradecimentos.

“Fé em Deus, que ele é justo. Ei, irmão, nunca se esqueça: na guarda, guerreiro, levanta a cabeça.”
(Racionais MC’s, V.L.)

RESUMO

As redes neurais artificiais são amplamente aplicadas a problemas complexos, contudo, a escolha da função de ativação (FA) ideal ainda representa um desafio, especialmente devido à ausência de critérios consistentes que relacionem as características dos dados à função mais adequada. À luz dessas limitações, avanços recentes destacam o potencial das FA treináveis, que se ajustam dinamicamente durante o treinamento para aprimorar o desempenho da rede. Tal mecanismo ainda pode ser explorado na inserção de novas camadas ocultas em redes do tipo *stacked autoencoder*, a fim de favorecer maior profundidade estrutural sem comprometer a estabilidade ou a eficiência do aprendizado. Em vista disso, este trabalho propõe uma nova abordagem, baseada em regressão polinomial, para o desenvolvimento de funções de ativação treináveis. O método introduz um mecanismo de inserção de novas camadas escondidas, de modo a colaborar com as camadas já existentes, caracterizando-o como colaborativo. Isso permite a integração das novas camadas sem degradar o conhecimento adquirido pelas camadas anteriores. Deste modo, simultaneamente, potencializando a capacidade adaptativa e o desempenho da rede por meio do cálculo da estimativa do erro. Esse cálculo é utilizado para ajustar dinamicamente a saída dos neurônios ocultos, em comparação ao uso de FA fixas. O método é avaliado em oito bases de dados de *benchmarking* extraídas da plataforma OpenML, totalizando 4.800 experimentos, a fim de analisar seu impacto na estabilidade do treinamento, na adaptabilidade da estrutura e no desempenho da rede em comparação com a função de ativação sigmoide tradicional.

Palavras-chave: Deep Learning, Funções de Ativação Treináveis, Regressão Polinomial.

ABSTRACT

Artificial neural networks are widely applied to complex problems; however, selecting an optimal activation function (AF) remains a challenge, particularly due to the lack of consistent criteria for relating data characteristics to the most appropriate function. In light of these limitations, recent advances highlight the potential of trainable activation functions, which dynamically adjust during training to improve network performance. Such a mechanism can also be leveraged to insert new hidden layers into stacked autoencoder networks, promoting greater structural depth without compromising training stability. In this context, this work proposes a novel approach based on polynomial regression to develop trainable activation functions. The methodology introduces a collaborative layer insertion mechanism, enabling the integration of new layers without degrading the knowledge acquired in previous ones, while simultaneously enhancing the network's adaptive capacity and performance through the use of error estimation as a criterion for dynamically adjusting the outputs of hidden neurons, in comparison with fixed activation functions. The proposed approach is evaluated on eight benchmarking datasets extracted from the OpenML platform, totaling 4,800 experiments, analyzing its impact on training stability, adaptability, and overall network performance compared to the traditional sigmoid activation function.

Keywords: Deep Learning, Trainable Activation Functions, Polynomial Regression.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ACU	Acurácia
API	Interface de Programação de Aplicações, em inglês
CNNs	Redes Neurais Convolucionais, em inglês
DL	Aprendizado profundo, em inglês
DNNs	Redes neurais profundas, em inglês
DSN	*Stacked autoencoder* tradicional
ELU	Unidade Exponencial linear, em inglês
FAs	Funções de ativação
FAt	Função de ativação treinada
FSC	Métrica F1-Score
GPUs	Unidades de processamento gráfico, em inglês
GUI	Interface gráfica do usuário, em inglês
KANs	Redes Kolmogorov-Arnold Networks, em inglês
MLGs	Modelos lineares generalizados
MLP	Perceptron de múltiplas camadas, em inglês
MRI	Imagens de ressonância magnética, em inglês
MSE	Erro quadrático médio, em inglês
NLM	Médias não locais, em inglês
PRE	Precisão
PReLU	Unidade Linear Retificada Paramétrica, em inglês
PYPI	Índice de pacotes python, em inglês
ReLU	Unidade Linear Retificada, em inglês
RMSE	Raiz do Erro Quadrático Médio, em inglês
RNAs	Redes neurais artificiais
RWN	Rede neural artificial com pesos aleatórios, em inglês
SEN	Sensibilidade
SiLU	Unidade Linear Sigmoide, em inglês
StatLog	Aprendizagem Baseada em Estatística e Lógica, em inglês
TPUs	Unidades de processamento tensorial, em inglês

SUMÁRIO

1 INTRODUÇÃO

Segundo Squire e Kandel (2003), a formação de memórias de longo prazo envolve modificações estruturais e funcionais nos neurônios, particularmente nas sinapses, que passam por alterações morfológicas, formação de novas conexões e fortalecimento das já existentes. Essas modificações são conhecidas como plasticidade sináptica, observadas em todas as regiões do cérebro humano. Analogamente, o aprendizado de uma rede neural supervisionada ocorre tradicionalmente por meio do ajuste iterativo dos pesos sinápticos, com base na apresentação de exemplos de treinamento rotulados, aspirando minimizar a diferença entre a resposta desejada e a resposta real da rede. Embora eficaz, o método pode ser suscetível a problemas como o sobreajuste (*overfitting*) e a escolha inadequada de hiperparâmetros, exigindo técnicas adicionais de regularização e validação para garantir a generalização do modelo em dados não vistos (Haykin, 2001).

Ademais, as redes neurais artificiais (RNAs) têm se mostrado bem-sucedidas na resolução de diversos problemas, especialmente com o avanço do aprendizado profundo (*deep learning*) (Ruedas *et al.*, 2024; Krari *et al.*, 2024; Chen *et al.*, 2024). No entanto, pesquisas na literatura indicam que ainda não existe uma relação consistente entre as propriedades estatísticas dos dados e a escolha ideal dos hiperparâmetros da rede, especialmente da função de ativação. Em outras palavras, uma função de ativação eficaz para um problema específico pode apresentar limitações significativas em outros problemas(Jankowski; Duch, 1997).

As funções de ativação (FAs) são fundamentais em redes neurais, pois permitem o aprendizado de características abstratas por meio de transformações não lineares (Dubey; Singh; Chaudhuri, 2022). Dada a forte influência da função de ativação na taxa de sucesso de uma RNA, a busca pela função ideal é comumente realizada por meio de experimentação, uma abordagem que pode ser complexa e demorada (Sharma; Venugopalan, 2014; Dorofki *et al.*, 2012).

Visto isso, nos últimos anos, em muitos trabalhos na literatura, a comunidade científica busca melhorar o desempenho de uma rede neural por meio do estudo das funções de ativação treináveis. Essas funções também são conhecidas por aprendíveis ou adaptáveis e levam, geralmente, a um desempenho superior da rede (Apicella *et al.*, 2021). Já existem técnicas que abordam o treinamento e a manipulação dessas funções, embora cada uma possua suas limitações específicas.

Além disso, redes mais profundas enfrentam limitações significativas, como o problema do desaparecimento do gradiente. Esse problema consiste em um fenômeno em que o gradiente diminui progressivamente à medida que é retropropagado pelas camadas da rede, resultando em atualizações extremamente pequenas dos pesos iniciais (Hochreiter, 1998). Como resultado, os pesos das camadas mais distantes da saída são atualizados muito mais lentamente do que os das camadas mais próximas da saída, o que compromete o processo de aprendizado geral da rede (Rosa *et al.*, 2019).

Outro empecilho que pode afetar o treinamento de RNAs profundas é que a retropropa-

gação ajusta os parâmetros a partir do gradiente do erro empírico calculado sobre a amostra de treinamento. Ao inserir novas camadas ocultas, o sinal de erro precisa atravessar múltiplos níveis de transformações não lineares, sendo sucessivamente alterado pelo produto das derivadas locais (Pascanu; Mikolov; Bengio, 2013). Esse processo pode levar a fenômenos conhecidos como *exploding* ou *vanishing gradients*, nos quais o gradiente cresce ou diminui exponencialmente à medida que é propagado pelas camadas, afetando a eficácia da atualização de pesos e, em última instância, a convergência do treinamento (Philipp; Song; Carbonell, 2018; Hochreiter, 1991). Em termos mais claros, o efeito acumulativo dessas transformações sucessivas pode gerar discrepâncias entre o erro empírico minimizado e o erro verdadeiro da distribuição de dados, especialmente em redes muito profundas.

Uma técnica comumente adotada para mitigar esses problemas consiste em adicionar uma camada por vez, ajustando apenas os pesos conectados à camada oculta recém-inserida. No entanto, como efeito colateral, a inclusão de uma nova camada afeta negativamente o aprendizado previamente alcançado pela rede (Viana *et al.*, 2025). Como solução para esse problema, foram propostos estudos focados na inserção colaborativa de novas camadas ocultas em redes neurais do tipo *Stacked Autoencoder*. Essas abordagens não apenas aumentam a profundidade da arquitetura enquanto preservam o conhecimento adquirido previamente, mas também evitam a degradação do desempenho ao longo do processo de treinamento.

Nesse contexto, propõe-se uma nova técnica de inserção colaborativa de camadas ocultas, associada a funções de ativação treináveis, para superar as limitações atuais, ajustando as FA dos neurônios ocultos a partir da estimativa do erro, para gerar a melhor saída relacionada aos dados de entrada. Essa técnica integra a regressão polinomial como ferramenta central no desenvolvimento de uma rede neural profunda. Nesse cenário, a regressão, amplamente utilizada na análise estatística, permite uma interpretação detalhada das relações entre variáveis e oferece uma base robusta para decisões fundamentadas em dados, permitindo a previsão dos valores da variável de interesse com base nos fatores identificados (Hoffmann, 2016).

1.1 Objetivos

O objetivo geral desta pesquisa foi desenvolver uma nova técnica de função de ativação treinável para neurônios ocultos com ajuste por meio da estimativa do erro visando à realizar inserção colaborativa de novas camadas ocultas em redes do tipo *stacked autoencoders* sem degradar o conhecimento prévio e corrigindo aprendizados errôneos.

Os objetivos específicos foram:

1. Propor uma função de ativação treinável baseada em regressão polinomial para neurônios das camadas ocultas;

2. Definir um modelo matemático para estimar o erro nos neurônios ocultos (E_h), orientando o ajuste dinâmico da função de ativação;

3. Integrar regressão polinomial ao treinamento de redes neurais profundas (DNNs), ampliando a capacidade de representação do modelo neural;

4. Implementar a inserção colaborativa e progressiva de camadas ocultas em arquiteturas *Stacked Autoencoder*;

5. Avaliar o desempenho da função de ativação proposta em comparação com uma função tradicional utilizando diferentes conjuntos de dados e configurações de rede neural.

1.2 Justificativas e Motivações

As arquiteturas de redes neurais profundas convencionais fazem uso de FAs analíticas fixas, cuja definição é realizada *a priori*, o que pode limitar a capacidade do modelo de adaptar-se de forma eficiente a diferentes distribuições e padrões dos dados. Essa limitação motiva o desenvolvimento de FAs treináveis, ajustadas dinamicamente durante o processo de aprendizado para produzir saídas mais adequadas nos neurônios das camadas ocultas. Dessa forma, o comportamento dos neurônios passa a ser diretamente influenciado pelo erro de estimação, ampliando o poder de representação da rede e reduzindo a dependência de escolhas empíricas previamente estabelecidas.

Em consonância com isso, as FAs fixas apresentam limitações estruturais que impactam diretamente a eficiência do treinamento em RNAs. Por não possuírem capacidade de adaptação automática ao comportamento do erro, tendem a impor uma forma funcional rígida ao processo de otimização. Essa rigidez pode dificultar o alinhamento entre as características dos dados e o formato da FA, resultando em maior tempo de convergência ou instabilidade no ajuste dos pesos. Como consequência, o treinamento pode exigir maior intervenção manual, especialmente na calibração de hiperparâmetros arquiteturais.

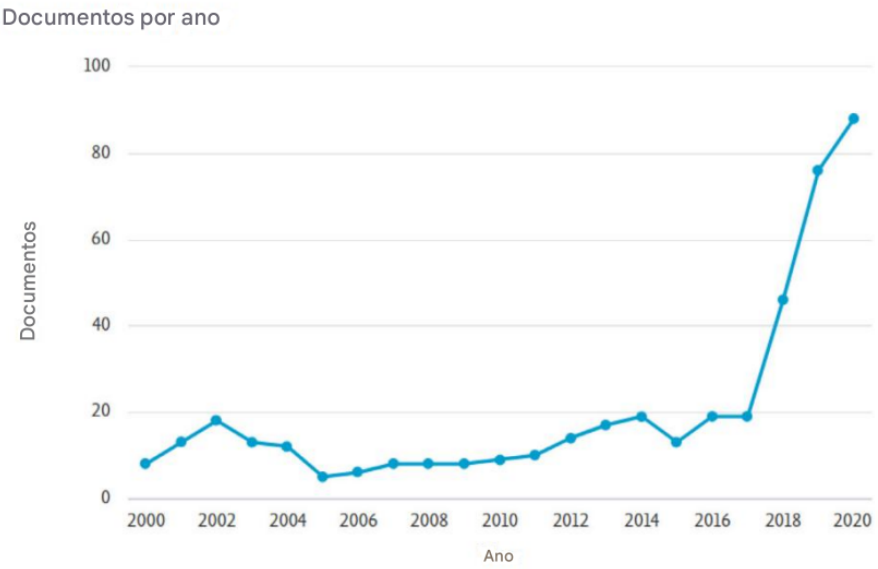
De forma semelhante, o aumento da profundidade em RNAs nem sempre conduz à melhoria de desempenho. A inserção arbitrária de camadas ocultas eleva a complexidade do modelo e pode provocar degradação do erro, instabilidade no treinamento e perda de capacidade de generalização. Além disso, alterações estruturais realizadas durante o processo de aprendizado podem comprometer o conhecimento previamente consolidado, tornando o equilíbrio entre capacidade representacional e estabilidade um desafio recorrente (dessa forma, uma mudança estrutural essas con).

Em aplicações reais, esses problemas tornam-se ainda mais evidentes. A obtenção de previsões precisas depende, frequentemente, de configurações empíricas complexas, tanto na escolha das FAs quanto na definição da estrutura interna das RNAs. Essa dependência reduz a reprodutibilidade experimental e aumenta a suscetibilidade a erros de modelagem. Soma-se a isso o fato de que dados reais apresentam ruído, incompletude e variações temporais, fatores que exigem maior flexibilidade estrutural do modelo. Nesse cenário, evidencia-se a necessidade de abordagens que conciliem precisão, adaptabilidade e menor dependência de ajustes manuais.

1.3 Trabalhos Relacionados

As funções de ativação treináveis, em redes neurais, vêm despertando interesse devido à sua potencial capacidade de proporcionar uma aprendizagem superior em comparação aos modelos tradicionais. Elas se destacam principalmente por sua capacidade de adaptação durante o treinamento, o que contribui significativamente para o aprimoramento do desempenho da rede. Esse fator tem impulsionado o crescimento dos estudos relacionados às funções de ativação treináveis (Apicella *et al.*, 2021), como ilustrado na Figura 1.

Figura 1 – Aumento das pesquisas sobre funções de ativação treináveis entre 2000 e 2020.



Fonte: Adaptado de Apicella *et al.* (2021)

Além disso, o trabalho de Apicella compara funções de ativação tradicionais, definidas como fixas (como a ReLU, a sigmoide (SIG) e a tangente hiperbólica), com funções de ativação treináveis, divididas em funções parametrizadas (derivadas de funções de ativação fixas padrão, com a adição de um conjunto de parâmetros treináveis) e funções baseadas em métodos de ensemble (misturas de várias funções distintas).

Outro trabalho relevante é *"Activation functions in deep learning: A comprehensive survey and benchmark"* que enfatiza o papel das funções de ativação (FA) em redes neurais, além de revisar diversas FA populares, classificando-as em categorias como funções baseadas em sigmoid/tanh, retificação, exponenciais e adaptativas, e examinando propriedades essenciais, incluindo monotonicidade, suavidade e capacidade de adaptação aos dados.

Por meio dessas características, busca-se garantir que a função de ativação contribua para a convergência eficaz durante o treinamento, mantenha a simplicidade computacional e preserve o fluxo de gradiente necessário para o aprendizado. A pesquisa conclui que, embora FAs clássicas ainda sejam amplamente utilizadas, FAs adaptativas baseadas na complexidade dos dados podem melhorar o aprendizado profundo e representam uma área promissora para investigações futuras (Dubey; Singh; Chaudhuri, 2022).

Ademais, as redes Kolmogorov-Arnold (KANs), que propõem a substituição dos pesos lineares por funções de ativação univariadas ajustáveis, buscam servir como uma alternativa aos Perceptrons Multi-Camada (MLPs) com o objetivo de melhorar a PRE e a interpretabilidade em tarefas específicas. Baseadas no teorema de Kolmogorov-Arnold, essas redes são modeladas a partir de curva definida matematicamente por dois ou mais pontos de controle (*splines*), permitindo que cada neurônio capture relações complexas de formas mais eficientes. Elas se mostraram especialmente eficazes em conjuntos de dados de baixa dimensão e no aprendizado contínuo (Liu *et al.*, 2024).

Ainda na literatura, destaca-se o trabalho de Ertuğrul (2018), que propõe o uso de uma função de ativação treinada (FAt), ajustada individualmente para cada neurônio na camada oculta por meio de regressão linear, com base nos dados de treinamento. Essa abordagem elimina a necessidade de seleção manual ou de tentativa e erro com funções de ativação tradicionais. A FAt personalizada foi aplicada a uma rede neural artificial com pesos aleatórios (RWN) e validada em 50 conjuntos de dados de classificação e regressão, demonstrando desempenho superior em relação às funções de ativação convencionais.

Por outro lado, a melhoria no desempenho de redes neurais também pode envolver, atualmente, abordagens como a de inserção colaborativa de camadas em redes neurais do tipo *Stacked Autoencoder*, como, por exemplo, no reconhecimento de sílabas fonéticas (Viana *et al.*, 2025). Nesse método, propõe-se que, após o treinamento de uma camada existente, uma nova camada seja inserida por meio de um ramo paralelo às camadas ocultas e de saída já treinadas, para posteriormente ser incorporada à arquitetura principal da rede, deixando de existir apenas como ramo paralelo.

Ademais, essa estratégia é associada à extração de características de sinais de sílabas fonéticas pela Transformada Rápida de Fourier (FFT), representadas por barras verticais agrupadas em figuras irregulares formadas principalmente pela união de retângulos, que, por sua vez, são utilizadas para o cálculo do centroide, permitindo a compactação dos dados sem perda de informações relevantes. Como consequência, esse método de inserção colaborativa viabiliza a introdução de novos conhecimentos sem comprometer os já adquiridos, gerando bom desempenho e redução no tempo de projeto, especialmente na definição do número de camadas e de neurônios.

Outro trabalho relevante é *C-SAN: Convolutional Stacked Autoencoder Network for Brain Tumor Detection Using MRI* (Gayathiri; Santhanam, 2025), no qual o autor combina Redes Neurais Convolucionais (CNN) com *deep stacked autoencoders* no contexto de pesquisas recentes sobre detecção de tumores cerebrais por imagens de ressonância magnética (MRI). A proposta visa à extração hierárquica mais robusta das características relevantes da região tumoral, partindo de etapas de pré-processamento com *Non-Local Means* (NLM) e segmentação por V-Net. Consequentemente, a integração dessas técnicas resulta em maior precisão na detecção, com métricas elevadas de acurácia, sensibilidade e especificidade, consolidando o C-SAN como uma abordagem eficiente entre as técnicas modernas de diagnóstico assistido por computador.

No mesmo contexto, destaca-se ainda a proposta por Hoon Chung, Sung Joo Lee e Jeon Gue Park (2016). A ideia fundamental consiste em representar FA não lineares convencionais em formas treináveis e, posteriormente, recondicioná-las por meio de uma estrutura de retro-propagação do erro. Para isso, as funções não lineares tradicionais são aproximadas por uma série de Taylor, cujos coeficientes são retreinados simultaneamente com os demais parâmetros da rede. A eficácia da abordagem proposta foi avaliada no domínio do reconhecimento de dígitos manuscritos utilizando o conjunto de dados MNIST (Chung; Lee; Park, 2016).

No estado da arte, também é relevante o artigo de Silhan, Oehmcke e Kramer (2019), que apresenta uma técnica para inicializar e ajustar automaticamente os hiperparâmetros durante o processo de treinamento de Redes Neurais Artificiais (RNAs) baseadas em *stacked autoencoders*. Nesse método, uma população de *autoencoders* é treinada com atualizações de pesos baseadas em gradiente, enquanto os hiperparâmetros são modificados e os pesos são herdados de forma Lamarckiana. O treinamento é realizado camada por camada, com cada nova camada iniciando um novo processo de otimização neuroevolutiva. A função de aptidão da abordagem evolutiva incorpora uma medida de qualidade de redução de dimensionalidade, para uma avaliação mais eficaz do desempenho.

Visto isso, embora os modelos baseados em Regressão Linear sejam fundamentais como método preditivo elementar que relaciona uma variável dependente Y com uma variável independente X por meio de uma reta (Chein, 2019), a regressão linear simples possui limitações, como o fato de não demonstrar causalidade. Diferente do modelo linear simples, que só pode representar relações lineares, a regressão polinomial, modelada como um polinômio de grau n, é capaz de capturar relações mais complexas e curvilíneas entre as variáveis (Queiroz *et al.*, 2024). Como, a exemplo, nos trabalhos (Jabeen; Ahmad; Zaman, 2024; Queiroz *et al.*, 2024).

1.4 Contribuições

As principais contribuições deste trabalho foram:

- Construção de uma função de ativação treinável baseada em regressão polinomial para neurônios das camadas ocultas;
- Obtenção de uma estimativa do erro nos neurônios ocultos (E_h) para o ajuste dinâmico da função de ativação;
- Obtenção da inserção colaborativa de camadas ocultas em redes *Stacked Autoencoder*.

1.5 Organização do Trabalho

Esta dissertação está estruturada em seis capítulos. Após a introdução, o Capítulo 2 – Redes Neurais Artificiais aborda os fundamentos teóricos da área. O Capítulo 3 – Regressão discute os conceitos fundamentais e os principais tipos de regressão, com ênfase na regressão

polinomial, que contribui fundamentalmente para a proposta deste trabalho. O Capítulo 4 – Metodologia descreve os materiais e os métodos empregados, incluindo as linguagens e as plataformas utilizadas, as bases de dados analisadas, o pré-processamento dos dados e a proposta de redes *stacked autoencoders* com inserção colaborativa via regressão polinomial. O Capítulo 5 – Resultados e Discussão apresenta a análise e a interpretação dos resultados obtidos. Por fim, o Capítulo 6 – Conclusão sintetiza os principais resultados do trabalho.

2 REDES NEURAIS ARTIFICIAIS

Este capítulo apresenta os fundamentos teóricos relacionados às RNAs, partindo de sua definição formal e da descrição dos elementos que compõem esses modelos computacionais. A partir dessa base conceitual, são discutidas as FAs. Em seguida, são explorados os aspectos arquiteturais dessas redes, destacando como a organização das camadas e os padrões de conexão entre neurônios influenciam sua capacidade de representação. Nesse contexto, também são abordados os principais métodos de treinamento e aprendizagem utilizados para ajustar os parâmetros do modelo e permitir a extração de padrões a partir dos dados. Por fim, é apresentado o conceito de *Stacked Autoencoders*, uma arquitetura que exemplifica a aplicação desses fundamentos no desenvolvimento de modelos mais profundos.

2.1 Definições

Uma rede neural artificial (RNA) é um modelo computacional inspirado no funcionamento do cérebro humano, projetado para resolver problemas complexos por meio de aprendizado automático. Ela é composta por unidades chamadas neurônios artificiais, organizados em camadas, que recebem, processam e transmitem informações (Guadalupe; Aguilar, 2025).

Em consonância com isso, Haykin (2001) descreve uma rede neural como uma máquina adaptativa, na qual os neurônios artificiais estão extensivamente interconectados:

> Uma rede neural é um processador maciçamente paralelo distribuído constituído de unidades de processamento simples, que têm a propensão natural para armazenar conhecimento experimental e torná-lo disponível para uso. Ela se assemelha ao cérebro em dois aspectos: 1. O conhecimento é adquirido pela rede a partir de seu ambiente através de um processo de aprendizagem. 2. Forças de conexão entre neurônios, conhecidas como pesos sinápticos, são utilizadas para armazenar o conhecimento adquirido (Haykin, 2001, p. 28).

Os conceitos de inteligência artificial e redes neurais artificiais possuem como marcos fundamentais uma série de trabalhos históricos. Cronologicamente, destaca-se o trabalho de Warren S. McCulloch e Walter Pitts (1943), que introduziram uma abordagem formal para descrever o comportamento do sistema nervoso por meio da lógica proposicional, tratando a atividade neural como um processo “tudo ou nada” que pode ser modelado logicamente. Ou seja, as atividades de cada neurônio podem ser representadas como proposições, associadas a conceitos como inibição relativa e absoluta, oferecendo um modelo para simular e analisar atividades cognitivas complexas (McCulloch; Pitts, 1943).

Já em 1958, conforme introduzido na seção anterior, Rosenblatt (1958) propôs um modelo probabilístico para a organização e armazenamento de informações no cérebro, baseado em redes de conexões neurais. A ideia central do modelo é que, ao invés de armazenar informações sensoriais de forma codificada ou representacional, o perceptron utiliza conexões entre

unidades de processamento para criar padrões de respostas em função dos estímulos recebidos. A aprendizagem, portanto, ocorre pela adaptação dessas conexões, as quais são fortalecidas ou enfraquecidas com base em reforços positivos ou negativos, permitindo ao perceptron não somente armazenar informações, mas também generalizar e reconhecer padrões a partir de estímulos inéditos.

Rosenblatt também explorou a eficácia de diferentes sistemas de aprendizado no modelo do perceptron, incluindo as variantes alpha, beta e gamma. Cada um desses sistemas apresenta características distintas no processo de modificação das conexões neurais, com ênfase no equilíbrio entre estímulos excitantes e inibitórios. O modelo destaca que, em sistemas mais complexos, como os que lidam com estímulos diferenciados (por exemplo, formas geométricas), o desempenho do perceptron pode ser significativamente impactado pela maneira como as respostas são atribuídas a estímulos previamente experimentados. A capacidade de generalização do perceptron é essencial para sua eficiência, ao permitir que o sistema aprenda e responda corretamente a novos estímulos, sem a necessidade de reprogramação completa (Rosenblatt, 1958).

Dois anos depois, Bernard Widrow e Marcian Hoff (1960) realizaram outro trabalho fundamental para a compreensão das redes neurais atuais, marcado pela criação do ADALINE, um neurônio artificial capaz de "aprender" a classificar padrões, ajustando sua própria estrutura com base no *feedback* de erro. Além disso, eles inventaram o Memistor, um componente eletroquímico que altera sua resistência de forma reversível por meio da eletrodeposição de cobre em um substrato de grafite, dando margem para o armazenamento estável das experiências de treino do sistema (Widrow; Hoff, 1960). De maneira análoga, como o ADALINE foi desenvolvido para reconhecer padrões binários, por exemplo, ao analisar bits de transmissão em uma linha telefônica, ele poderia estimar o próximo bit de um conjunto. (Academy, 2025).

Com base nesses trabalhos, atualmente, identificam-se sete elementos fundamentais que compõem o neurônio artificial, conforme ilustrado na Figura 2, em consonância com os conceitos apresentados por Silva e Santos (2015 apud Campos, 2018).

- **Sinais de entrada** $(x_1, x_2, x_3, \ldots, x_n)$: correspondem a sinais ou medições provenientes do meio externo, representando os valores assumidos pelas variáveis de uma aplicação específica, ou de outros neurônios anteriores a este.

- **Pesos sinápticos** $(w_1, w_2, \ldots, w_n)$: são coeficientes associados a cada sinal de entrada ou sinal de saída de um neurônio anterior, responsáveis por ponderar a influência relativa de cada variável no processamento do neurônio.

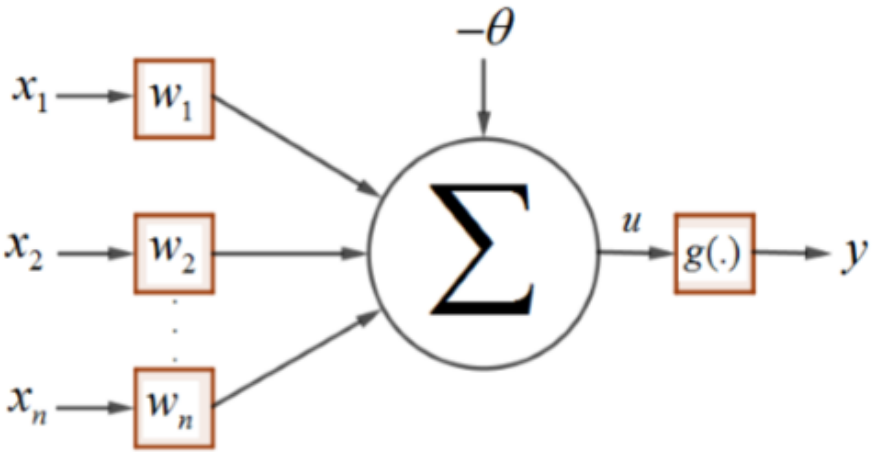
- **Combinador linear** $(\sum)$: tem a função de agregar os sinais de entrada ponderados pelos respectivos pesos sinápticos, resultando em um valor que representa o potencial interno do neurônio.

- **Limiar de ativação** (θ): consiste em um parâmetro que define o patamar necessário para que o valor produzido pelo combinador linear seja suficiente para provocar a ativação do

neurônio em direção à saída definido como *bias*.

- **Potencial de ativação** (u): usualmente denotado por u, é definido como o resultado da combinação linear dos sinais de entrada ponderados pelos respectivos pesos sinápticos, subtraído do limiar de ativação.
- **Função de ativação** (g): tem como finalidade restringir a saída do neurônio a um intervalo de valores apropriado, de acordo com sua definição matemática e com o comportamento esperado do modelo.
- **Sinal de saída** (y): representa o valor final produzido pelo neurônio em resposta a um determinado conjunto de sinais de entrada.

Figura 2 – Estrutura de um neurônio artificial



Fonte: Campos (2018)

Como saída, o neurônio artificial apresenta a seguinte representação matemática, expressa nas Equações 1.

$$\begin{cases} u = \sum_{i=1}^{n} w_i x_i - \theta \\ y = g(u) \end{cases} \tag{1}$$

2.2 Funções de Ativação

As funções de ativação são responsáveis por transformar as entradas e produzir saídas que representam, geralmente de forma não linear, as relações existentes nos dados. Ao realizar essa transformação, definem se e como a informação processada por um neurônio será propagada para os neurônios subsequentes. Em algoritmos de aprendizado profundo (DL), essas funções são projetadas considerando aspectos como a condução eficiente do processo de aprendizado, a

mitigação do *overfitting*, o aumento da precisão dos modelos e a redução do custo computacional (Kılıçarslan; Adem; Çelik, 2021).

Nesse contexto, a função de ativação determina se um neurônio será ativado a partir do valor de entrada recebido. Em abordagens baseadas em limiar, o neurônio é ativado apenas quando o valor da transformação linear ultrapassa um determinado limite; caso contrário, permanece inativo. Também denominada função de limiar ou função de transferência, trata-se de uma transformação escalar–escalar aplicada à saída do neurônio. Na ausência de uma função de ativação não linear, a rede neural reduz-se a um modelo puramente linear, no qual o sinal de saída corresponde a um polinômio de primeiro grau. Assim, o classificador baseado em limiar exerce papel fundamental ao definir se o resultado da transformação linear é suficiente para ativar o neurônio (Sharma; Sharma; Athaiya, 2017).

2.2.1 FA Tradicionais

Primeiramente, segundo Apicella *et al.* (2021), as funções de ativação de forma fixa compreendem todas aquelas cujo formato é previamente definido, incluindo, por exemplo, as funções sigmoide, tangente hiperbólica e ReLU. Essas funções não possuem parâmetros ajustáveis durante o treinamento. As funções de ativação de forma fixa podem ser subdivididas em duas categorias: funções clássicas de ativação e funções baseadas em retificação (Apicella *et al.*, 2021).

Funções Clássicas

As funções de ativação clássicas correspondem às primeiras abordagens empregadas no desenvolvimento das redes neurais artificiais. Entre as mais conhecidas, destacam-se a função sigmoide e a tangente hiperbólica, utilizadas principalmente em redes rasas devido à sua suavidade e diferenciabilidade. A sigmoide é tradicionalmente aplicada em problemas de classificação binária, enquanto a tangente hiperbólica apresenta média próxima de zero, o que favorece a dinâmica do treinamento. Entretanto, ambas apresentam limitações relevantes, como o problema do gradiente desvanecente em regiões de saturação e o maior custo computacional decorrente do uso de funções exponenciais (Mercioni; Holban, 2020). As funções são definidas pelas seguintes equações:

- **Sigmoide**:

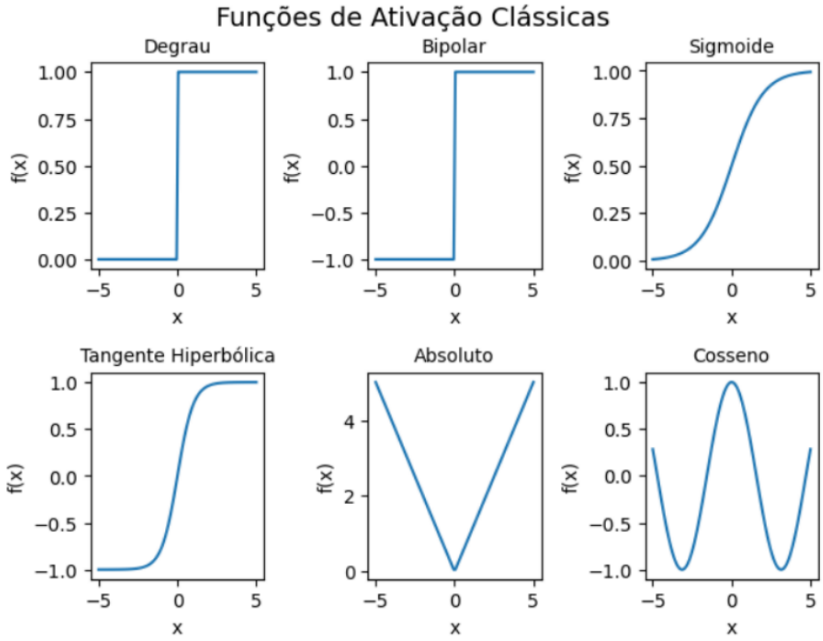
$$\sigma(x) = \frac{1}{1+e^{-x}} \tag{2}$$

- **Tangente hiperbólica (tanh)**:

$$\tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \tag{3}$$

Ademais, também podem ser encontradas outras funções clássicas, como as funções degrau, bipolar, valor absoluto e cosseno (Apicella *et al.*, 2021). Essas funções apresentam comportamentos característicos, os quais são ilustrados na Figura 3.

Figura 3 – Comportamento das funções clássicas de ativação.



Fonte: Adaptado de Apicella *et al.* (2021)

Funções baseadas em retificação

Já as funções baseadas em retificação possuem a capacidade de transformar valores negativos em zero ou em pequenos valores, introduzindo uma não linearidade simples e mitigando o problema do gradiente que desaparece. Incluem as funções da família de retificação, como ReLU, Leaky ReLU (LReLU), ELU e SiLU, entre outras. Essas funções podem ser definidas pelas seguintes equações:

- **ReLU** (*Rectified Linear Unit*):

$$\text{ReLU}(u) = \max(0, u) \tag{4}$$

- **Leaky ReLU**:

$$\text{Leaky ReLU}(u, \alpha) = \begin{cases} u, & \text{se } u > 0 \\ \alpha \cdot u, & \text{se } u \leq 0 \end{cases} \tag{5}$$

- **ELU** (*Exponential Linear Unit*):

$$\text{ELU}(u, \alpha) = \begin{cases} u, & \text{se } u \geq 0 \\ \alpha \cdot (e^u - 1), & \text{se } u < 0 \end{cases} \tag{6}$$

- **SiLU** (*Sigmoid Linear Unit*):

$$\text{SiLU}(u) = \frac{u}{1+e^{-u}} \tag{7}$$

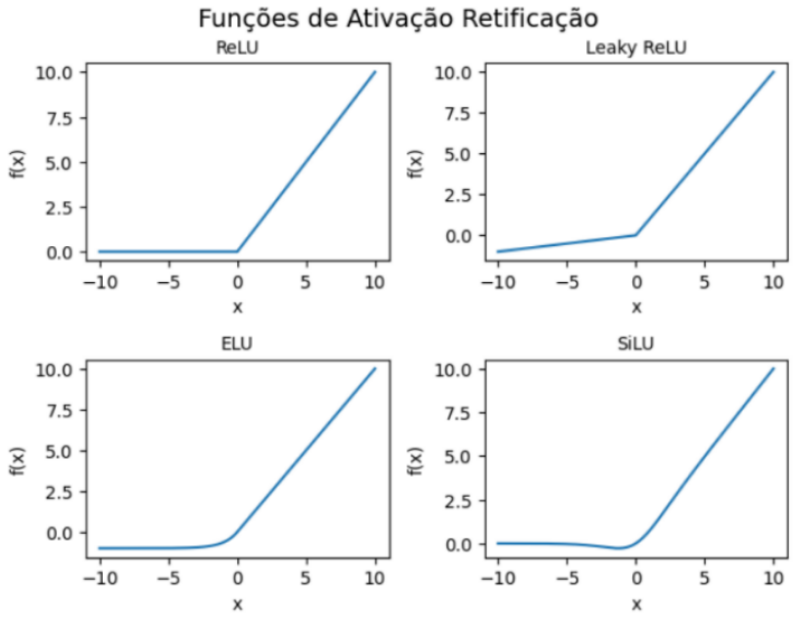
onde:

- $u \in \mathbb{R}$ representa o pontecial de ativação do neurônio;
- $\alpha > 0$ é um parâmetro escalar que controla o comportamento das funções de ativação para valores negativos da entrada, atuando como coeficiente de inclinação na Leaky ReLU e como fator de escala na ELU.

A ReLU é uma das funções de ativação mais utilizadas devido à sua simplicidade computacional e à capacidade de mitigar o problema do gradiente desvanecente, pois ativa apenas neurônios cujas entradas assumem valores positivos. Contudo, pode levar ao problema de neurônios inativos quando o gradiente se torna nulo para entradas negativas, o que motiva o uso de variantes como a Leaky ReLU, a qual introduz uma inclinação não nula nesse regime (Sharma; Sharma; Athaiya, 2017; Apicella *et al.*, 2021).

A ELU combina comportamento linear para entradas positivas com uma função exponencial para valores negativos, o que resulta em maior suavidade e estabilidade no processo de otimização. Já a função SiLU, também conhecida como Swish, definida como o produto entre a entrada e a função sigmoide, destaca-se por ser suave e não monótona, com desempenho competitivo e, em alguns casos, superior ao da ReLU em tarefas de DL (Sharma; Sharma; Athaiya, 2017). O comportamento dessas funções de ativação é apresentado na Figura 4.

Figura 4 – Comportamento das funções de ativação ReLU, Leaky ReLU, ELU e SiLU.



Fonte: Adaptado de Apicella *et al.* (2021)

2.2.2 FA Treinadas

As funções de ativação treináveis referem-se a abordagens em que o formato da função é ajustado durante a fase de treinamento. Nessa perspectiva, o objetivo consiste em aprender uma forma funcional adequada com base nos dados utilizados no processo de aprendizado.

A partir dessa concepção, diversos tipos de FA treináveis têm sido propostos, não apenas na forma de sub-redes *feed-forward* compostas por neurônios clássicos, mas também como extensões de FA fixas. No caso de sub-redes, elas são integradas à rede neural principal por meio da adição de novas camadas. Esse tipo de abordagem aumenta a expressividade da rede ao permitir ajustes dinâmicos durante o treinamento. As funções de ativação treináveis podem ser classificadas em duas famílias:

- **Funções padrão parametrizadas**: derivadas de funções fixas clássicas, com a adição de um conjunto de parâmetros ajustáveis durante o treinamento. Essas funções podem ser expressas como versões parametrizadas de funções padrão, cujos valores dos parâmetros são aprendidos a partir dos dados (Chung; Lee; Park, 2016). As FA padrão parametrizadas incluem parâmetros treináveis que ajustam sua forma. Por exemplo, a ReLU com vazamento paramétrico tem os coeficientes de sua parte negativa aprendidos adaptativamente e permite um aprendizado automático de retificação (Zhang *et al.*, 2014). Alguns exemplos são:
  - **Sigmoide Generalizada Ajustável**: adiciona os parâmetros α e β à função sigmoide clássica, permitindo controlar sua suavidade e amplitude.
  - **ReLU Paramétrica (PReLU)**: introduz um parâmetro α com o objetivo de ajustar a inclinação da parte negativa da função ReLU.
  - **ELU Paramétrica**: adiciona os parâmetros β e γ para controlar o comportamento da função ELU.
  - **Swish**: combina a multiplicação da entrada pela função sigmoide, incorporando um parâmetro α para ajustar sua forma.
- **Funções baseadas em métodos de *ensemble***: fundamentam-se na combinação linear de diferentes FA. Essas combinações são modeladas como funções lineares de uma variável e, em alguns casos, incluem parâmetros adicionais ajustados durante o treinamento (Apicella *et al.*, 2021). Diferentemente da abordagem tradicional, na qual uma única FA é escolhida para todos os neurônios, o *activation ensemble* permite que diferentes funções atuem simultaneamente, explorando suas características complementares na extração de atributos. Essa combinação pode ocorrer no nível do neurônio, integrando múltiplas ativações na geração das representações internas, ou no nível de decisão, onde as saídas produzidas por diferentes funções são agregadas por meio de mecanismos como votação majoritária (Nandi; Jana; Das, 2020).

Como exemplos, destacam-se:

  - **Mistura Adaptativa**: combinação de funções como LReLU e ELU, ponderadas por um parâmetro p aprendido durante o treinamento.
  - **Função de Ativação Hierárquica**: caracteriza-se pela organização de funções de ativação em uma hierarquia de múltiplos níveis, utilizando combinações adaptativas.
  - **Combinações Lineares**: representadas por funções definidas como somas ponderadas de ativações básicas, conforme a Equação 8.

$$f(a) = \sum_{i=1}^{k} \alpha_i g_i(a) \tag{8}$$

- **Casos específicos**: Apicella *et al.* (2021) também identificam uma categoria de casos específicos (*outliers*), que engloba abordagens menos convencionais. Embora menos frequentes, essas alternativas se destacam por oferecer soluções relevantes para cenários particulares, tais como:

  - **Funções Polinomiais**: modelam a ativação como uma combinação polinomial, embora apresentem limitações relacionadas à universalidade.
  - **Funções Fuzzy**: utilizam lógica fuzzy para modelar funções de ativação com maior flexibilidade.
  - **Expansões de Fourier**: aplicam métodos matemáticos, como a expansão em séries de Fourier, para ajustar a forma da função de ativação.
  - **Métodos Baseados em Interpolação**: ajustam a ativação por meio de interpolação ou tabelas de consulta (*look-up tables*), permitindo controle direto sobre sua forma.

2.3 Arquitetura das Redes Neurais

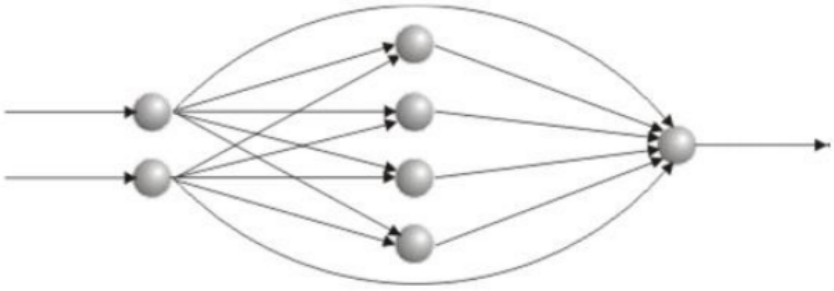
Uma RNA é composta, fundamentalmente, por um conjunto de camadas organizadas de forma hierárquica. Em sua estrutura, necessariamente há ao menos uma camada de entrada e uma camada de saída, podendo existir uma ou mais camadas intermediárias. A camada de entrada é responsável por receber os dados provenientes do ambiente externo. As camadas intermediárias, também denominadas camadas escondidas, ocultas ou invisíveis, realizam o processamento interno das informações. Por fim, a camada de saída é responsável por produzir os resultados do processamento.

Nesse processo, os neurônios de diferentes camadas são interconectados por meio de conexões, que possuem valores associados chamados de pesos sinápticos, ou simplesmente

pesos. Cada neurônio recebe informações ponderadas dessas conexões, sejam elas provenientes de entradas externas ou de saídas de outros neurônios. Essas informações são processadas por meio da função de ativação.

Entre as arquiteturas de redes neurais artificiais mais utilizadas, destacam-se as redes *feedforward* e as redes recorrentes. A principal diferença entre elas está nas conexões entre os neurônios: nas redes *feedforward*, não há realimentações das saídas para as entradas ao longo da rede, ou seja, as informações seguem em uma direção, da entrada para a saída (Figura 5). As redes *feedforward* podem ser divididas em duas categorias, dependendo do número de camadas: camada única ou multicamadas (Sazlı, 2006).

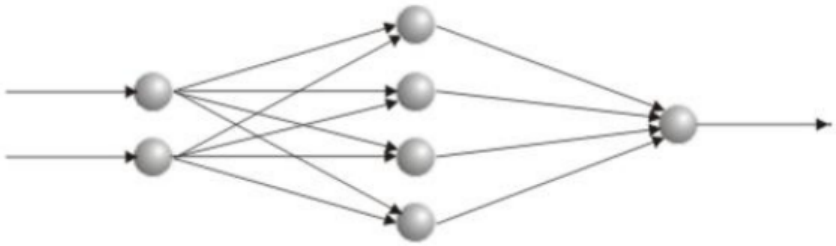
Figura 5 – Topologia *feedforward*.



Fonte: de Almeida Neto (2003 apud Teixeira, 2017)

Além disso, existe a topologia estritamente *feedforward*, como representada na Figura 6, caracterizada pela propagação unidirecional dos sinais da camada de entrada para a camada de saída, sem a presença de ciclos ou conexões de realimentação. Já nas redes recorrentes, também chamadas de *feedback*, conforme mostrado na Figura 7, algumas ou todas as saídas podem ser conectadas novamente às entradas de neurônios de camadas anteriores (Teixeira, 2017; Sazlı, 2006).

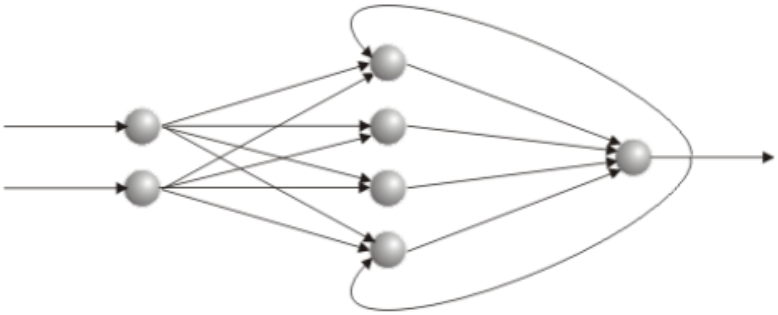
Figura 6 – Topologia estritamente *feedforward*.



Fonte: de Almeida Neto (2003 apud Teixeira, 2017)

Adicionalmente, ainda é possível somar à estrutura das RNAs um elemento denominado *bias*, um parâmetro interno à rede capaz de interagir com um conjunto de neurônios, cuja função consiste em aumentar ou reduzir a entrada líquida da função de ativação, conforme o seu valor

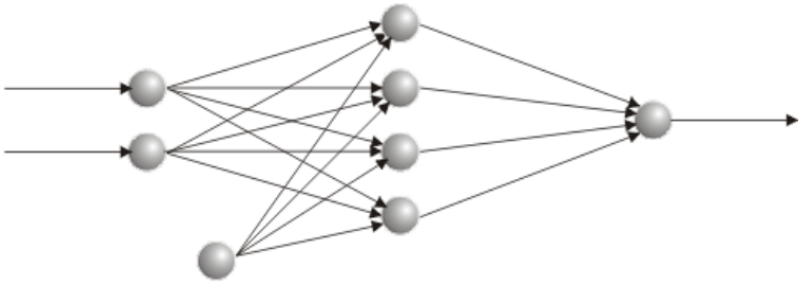
Figura 7 – Topologia *feedback*.



Fonte: de Almeida Neto (2003 apud Teixeira, 2017)

seja positivo ou negativo. A representação desta arquitetura de RNA pode ser visualizada na Figura 8.

Figura 8 – RNA com *bias*.



Fonte: de Almeida Neto (2003 apud Teixeira, 2017)

2.4 Métodos de Treinamento

O processo de treinamento das RNAs fundamenta-se na capacidade de aprender a partir de amostras que representam o comportamento de um sistema, permitindo, após a etapa de aprendizado, a generalização de soluções para entradas não previamente apresentadas. Esse processo consiste, principalmente, na aplicação de algoritmos de aprendizagem responsáveis pela sintonização dos pesos e limiares dos neurônios, de modo a minimizar a diferença entre as saídas produzidas pela rede e aquelas consideradas desejadas (Silva; Spatti; Flauzino, 2016).

No que se refere aos tipos de treinamento, destacam-se três abordagens principais: o treinamento supervisionado, o semissupervisionado e o não supervisionado, os quais são:

- **Supervisionado:** Consiste na utilização de um conjunto de dados previamente rotulado, no qual cada amostra de treinamento é composta por sinais de entrada e suas respectivas saídas desejadas. Para isso, faz-se necessária a disponibilização de uma tabela de dados representativa do processo, também denominada tabela de atributos/valores, no caso de sistemas *off-line*. A partir da qual as estruturas neurais formulam hipóteses sobre o

comportamento a ser aprendido. Em sistemas físicos, os valores de referência podem ser obtidos por uma equação ou por sensores.

Nesse contexto, o processo de aprendizagem ocorre como se houvesse um "professor" orientando a rede quanto às respostas corretas associadas a cada amostra apresentada. Posteriormente, os pesos sinápticos e os limiares são ajustados continuamente por meio de ações comparativas executadas pelo próprio algoritmo de aprendizagem, que avalia a defasagem entre as saídas produzidas pela rede e aquelas desejadas. A rede é considerada treinada quando essa defasagem se encontra dentro de limites aceitáveis, previamente definidos (Silva; Spatti; Flauzino, 2016).

- **Semissupervisionado:** O aprendizado semissupervisionado fundamenta-se na hipótese de que a distribuição dos dados de entrada $p(x)$ carrega informações estruturais relevantes sobre a distribuição condicional $p(y \mid x)$. Em outras palavras, assume-se que a forma como os dados estão distribuídos no espaço de entrada reflete, ao menos parcialmente, a separação entre as classes.

  Para que essa premissa seja válida, os dados não rotulados utilizados no treinamento devem ser consistentes com o domínio do problema, ou seja, devem pertencer às mesmas classes que se deseja modelar. A inclusão de amostras provenientes de classes não relacionadas pode distorcer essa estrutura e comprometer o aprendizado, podendo inclusive degradar o desempenho do modelo. Dessa forma, a relação entre $p(x)$ e $p(y \mid x)$ constitui a base conceitual que sustenta as principais hipóteses exploradas pelos métodos de aprendizado semissupervisionado (Bergmann, 2024).

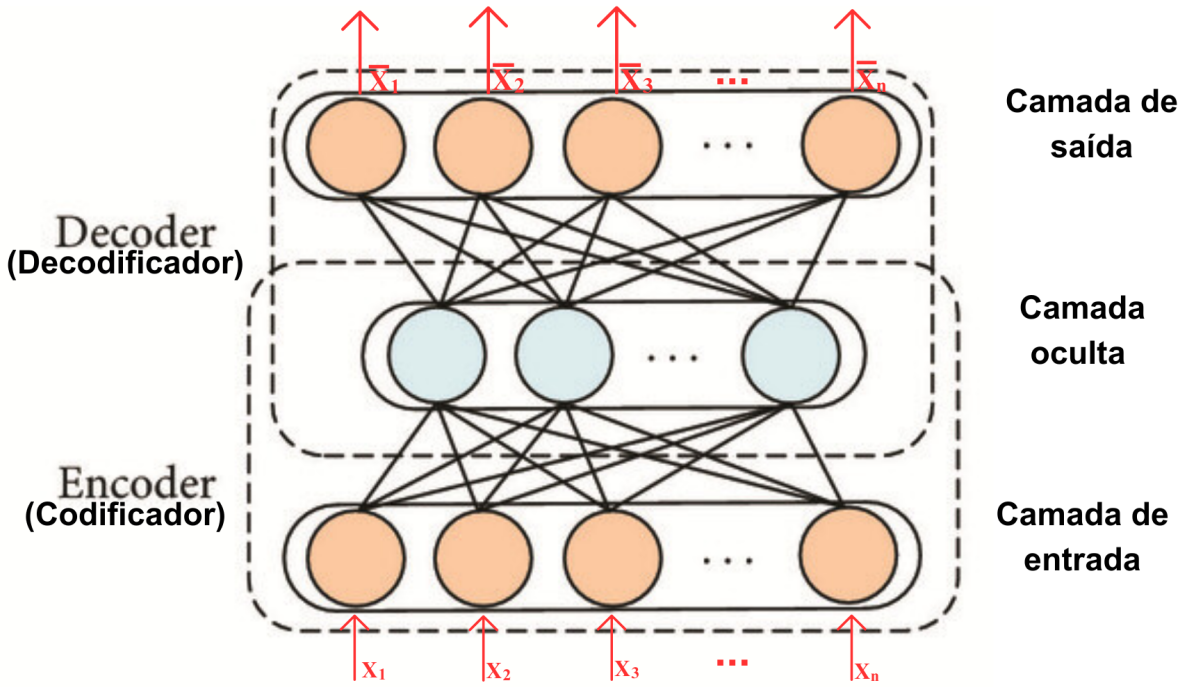
- **Não supervisionado:** Caracteriza-se pela ausência de um supervisor que forneça saídas desejadas durante o processo de aprendizagem. Sob essa ótica, apenas os dados de entrada são disponibilizados ao algoritmo, sem rótulos ou classificações prévias. Dessa forma, a RNA deve identificar autonomamente padrões, correlações ou estruturas presentes nos dados, organizando-os em grupos ou representações que revelem suas relações internas (Bishop, 1995 apud Fleck *et al.*, 2016).

2.5 Redes *Stacked Autoencoders*

A rede *Stacked Autoencoder* (DSN) é uma arquitetura de rede neural profunda formada pelo empilhamento de múltiplos *autoencoders*, geralmente do tipo esparso, em que a saída de cada camada é utilizada como entrada para a camada subsequente. Essa organização permite a extração hierárquica de características associada há uma escalabilidade do aprendizado superior (Chaudhary, 2019 apud Pereira, 2023). Cada *autoencoder* é composto por três partes principais: o *encoder*, a camada oculta intermediária (*bottleneck*) e o *decoder*. O *autoencoder* tem como objetivo reconstruir a entrada original a partir de uma representação interna compacta, aprendida de forma não supervisionada (Deng; Yu, 2014; Liu; Bao; Han, 2018).

O treinamento ocorre em duas etapas complementares. O *encoder* é responsável por mapear os dados de entrada em uma representação oculta, enquanto o *decoder* reconstrói os dados originais a partir dessa codificação (Liu; Bao; Han, 2018). A estrutura de um *autoencoder* é apresentada na Figura 9.

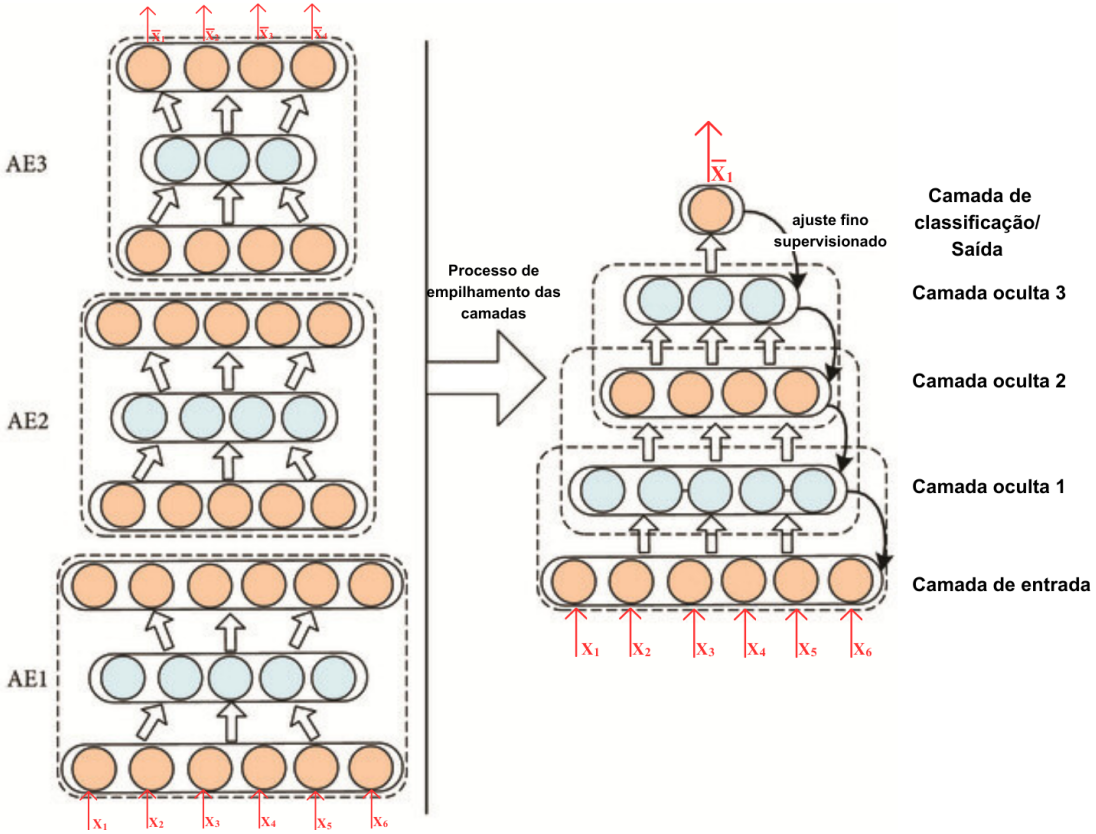
Figura 9 – Estrutura *Autoencoder*



Fonte: Adaptado de Liu, Bao e Han (2018)

Alguns conjuntos de dados apresentam relações complexas entre suas características; portanto, o uso de um único *autoencoder* pode não ser suficiente para capturar adequadamente tais padrões ou reduzir de forma eficiente a dimensionalidade das entradas. Nesses casos, utilizam-se *autoencoders* empilhados (*stacked autoencoders*), que consistem na organização sequencial de múltiplos codificadores em camadas sucessivas. Esse empilhamento permite a construção de modelos profundos capazes de aprender representações progressivamente mais abstratas e discriminativas dos dados (Bakshi, 2021; Liu; Bao; Han, 2018). Um *stacked autoencoder* com três codificadores é apresentado na Figura 10.

Figura 10 – Redes *stacked autoencoders*



Fonte: Adaptado de Liu, Bao e Han (2018)

3 REGRESSÃO

Os fundamentos da análise de regressão remontam ao século XIX, especialmente aos estudos sobre hereditariedade desenvolvidos por Sir Francis Galton (1822–1911), que investigou a relação entre a estatura média dos pais e a altura dos filhos adultos. Ao organizar os dados em tabelas de contingência e representá-los em gráficos de dispersão, identificou uma tendência aproximadamente linear com inclinação inferior a um. Esse resultado levou à formulação do conceito de regressão à média, segundo o qual valores extremos tendem a aproximar-se da média populacional nas gerações subsequentes. Posteriormente, seu trabalho recebeu formalização matemática por Karl Pearson, Francis Ysidro Edgeworth e George Udny Yule, consolidando os fundamentos da regressão linear (Fahrmeir *et al.*, 2022).

A regressão constitui, na atualidade, uma metodologia estatística utilizada na análise de problemas empíricos nas ciências sociais, econômicas e da vida. Essa abordagem permite compreender e interpretar relações entre fatores, fornecendo suporte quantitativo para a tomada de decisões fundamentadas em dados. Seu objetivo consiste em avaliar, do ponto de vista estatístico, a existência e a magnitude da associação entre variáveis, bem como possibilitar a predição de valores da variável de interesse.

Em termos formais, a regressão modela a relação entre uma variável resposta e uma ou mais variáveis explicativas, viabilizando inferência estatística e previsão. Ao longo do tempo, essa metodologia incorporou modelos lineares clássicos e extensões paramétricas, não paramétricas e semiparamétricas, ampliando sua capacidade de representação e análise de fenômenos complexos (Hoffmann; Vieira, 1987; Fahrmeir *et al.*, 2022).

3.1 Tipos de Regressão

Devido à sua relevância, é possível encontrar uma ampla variedade de modelos de regressão, cada qual com particularidades e características próprias. Entre os principais, conforme Hoffmann e Vieira (1987), destacam-se:

3.1.1 Regressão Linear Simples

A regressão linear simples permite estimar, de forma aproximada, a relação entre uma variável dependente e uma única variável independente. Partindo do pressuposto de que os dados apresentam comportamento constante na natureza, constrói-se uma reta hipotética que melhor se ajusta ao conjunto de pontos observados.

Considerando um conjunto de n pares de valores de duas variáveis, X_i e Y_i, com $i = 1, 2, \ldots, n$, e admitindo-se que Y seja uma função linear de X, pode-se estabelecer um modelo de regressão linear simples, cujo modelo estatístico é dado por:

$$Y_i = \alpha + \beta X_i + u_i \tag{9}$$

em que α e β são parâmetros do modelo, X representa a variável explanatória, Y corresponde à variável dependente e u_i denota o termo de erro aleatório. O coeficiente angular da reta, representado por β, é denominado coeficiente de regressão, enquanto α corresponde ao coeficiente linear da reta, também conhecido como termo constante da equação de regressão.

Uma das principais limitações da regressão linear reside na possibilidade de discrepâncias entre os dados observados e a reta ajustada. Por esse motivo, esse método é frequentemente empregado em conjuntos reduzidos de dados ou em análises exploratórias, visto que alguns pontos podem não se ajustar perfeitamente à linha de regressão estimada.

3.1.2 Regressão Linear Múltipla

A regressão linear múltipla é empregada para modelar a relação entre uma variável resposta e duas ou mais variáveis explicativas, permitindo avaliar o efeito parcial de cada regressor, mantendo os demais constantes. Diferentemente da regressão linear simples, esse modelo possibilita o controle simultâneo de múltiplos fatores, o que favorece uma representação mais adequada de fenômenos empíricos complexos, desde que seus pressupostos sejam satisfeitos.

O modelo populacional de regressão linear múltipla com k variáveis explicativas pode ser expresso por:

$$Y_j = \alpha + \beta_1 X_{j1} + \beta_2 X_{j2} + \cdots + \beta_k X_{jk} + u_j, \tag{10}$$

ou, de forma compacta,

$$Y_j = \alpha + \sum_{i=1}^{k} \beta_i X_{ji} + u_j. \tag{11}$$

Nesse modelo, u_j representa o termo de erro aleatório associado à j-ésima observação. Sob a formulação clássica, assumem-se linearidade nos parâmetros, exogeneidade, homocedasticidade, ausência de multicolinearidade perfeita e independência dos erros. A hipótese de normalidade dos erros é necessária para a realização de inferência estatística paramétrica exata, mas não constitui condição obrigatória para a estimação dos coeficientes pelo método dos mínimos quadrados (Fahrmeir *et al.*, 2022).

Na literatura, a regressão linear múltipla tem sido aplicada em diferentes contextos, como na estimativa da produtividade da soja, empregada por Mercante *et al.* (2010); na combinação de regressão linear e redes neurais para aprimorar previsões de radiação solar apresentada por Guarnieri (2006) e na análise da influência do acesso ao saneamento básico sobre a incidência e mortalidade por COVID-19, realizada por Aquino (2020). Tais aplicações evidenciam a versatilidade do modelo, cuja adequação depende da fundamentação teórica, da qualidade dos dados e da verificação criteriosa de seus pressupostos.

3.1.3 Regressão logística

A regressão logística é um modelo estatístico utilizado para analisar a relação entre uma variável dependente e uma ou mais variáveis independentes. A variável dependente é de natureza categórica, sendo comumente binária (com duas categorias). Quando apresenta apenas

duas categorias, utiliza-se a regressão logística binária; quando possui mais de duas categorias, emprega-se a regressão logística multinomial (Hoffmann; Vieira, 1987).

Diferentemente da regressão linear, a regressão logística é adequada para variáveis resposta limitadas ao intervalo entre 0 e 1, permitindo modelar adequadamente a relação entre as variáveis independentes e a probabilidade da variável dependentes (Domínguez-Almendros; Benítez-Parejo; Gonzalez-Ramirez, 2011).

Como representação do modelo multivariado, tem-se:

$$p = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_k X_k)}} \tag{12}$$

onde:

- $p = P(Y = 1 \mid X)$ é a probabilidade de ocorrência do evento;
- β_0 é o intercepto do modelo;
- β_i são os coeficientes associados às variáveis explicativas;
- X_i são as variáveis independentes (ou preditoras).

Entre suas principais características, destaca-se a possibilidade de estimar diretamente a probabilidade de ocorrência do evento de interesse e interpretar os coeficientes do modelo em termos de razão de chances (*odds ratio*), como também permite o ajuste simultâneo de múltiplos fatores, para o controle de variáveis de confusão e a análise de interações (Domínguez-Almendros; Benítez-Parejo; Gonzalez-Ramirez, 2011).

3.1.4 Regressão de Poisson

A regressão de Poisson é usada quando a variável dependente possui dados de contagem. Este modelo faz parte dos Modelos Lineares Generalizados (MLG), sendo ideal para prever o número de eventos que ocorrem em um intervalo de tempo ou espaço específico. Por exemplo, pode ser utilizado para prever o número de chamadas no atendimento ao cliente relacionadas a um produto específico ou estimar o número de chamadas de serviço de emergência durante um evento. A variável dependente deve atender às seguintes condições: possuir uma distribuição de Poisson, as contagens não podem ser negativas e o método não é adequado para números não inteiros.

Os Modelos Lineares Generalizados (MLG) representam uma união de modelos lineares e não-lineares, partindo de uma distribuição da família exponencial, que inclui distribuições como normal, Poisson, binomial, gama e normal inversa. Esses modelos abrangem desde os modelos lineares tradicionais (com erros distribuídos normalmente) até os modelos logísticos (Schmidt, 2003).

Os MLG consistem em três componentes principais: a componente aleatória, que envolve variáveis explicativas de uma variável resposta com distribuição exponencial e valor esperado:

$$E(y_i) = \mu \tag{13}$$

a componente sistemática, que forma uma estrutura linear para o modelo de regressão, expressa como:

$$\eta = \beta x^T \tag{14}$$

onde x^T representa o vetor de variáveis explicativas e g denota a função de ligação, uma função monótona e diferenciável que relaciona a média da variável resposta μ à estrutura linear η, descrita por:

$$g(\mu) = \eta \tag{15}$$

A função de ligação canônica, que simplifica a relação para $\eta = \theta_i$, utiliza o parâmetro de localização ou canônico θ (Tadano; Ugaya; FrAnCo, 2009).

Um exemplo de aplicação do método de regressão de Poisson é a metodologia para avaliação do impacto da poluição atmosférica na saúde populacional, descrita por Tadano, Ugaya e FrAnCo (2009). Outro exemplo é encontrado nos estudos de Schmidt (2003), onde o modelo de regressão de Poisson é aplicado para analisar diversos aspectos da saúde, como o número de internações hospitalares ou visitas a pronto-socorros relacionadas a fatores ambientais e de saúde pública.

3.1.5 Regressão LASSO

Regularização é uma técnica que ajuda a resolver problemas de ajuste excessivo, onde um modelo pode ter um bom desempenho nos dados de treinamento, mas um desempenho ruim nos dados de validação (teste). Esse problema pode ser resolvido pela regularização ao adicionar um termo de penalidade à função objetivo, controlando assim a complexidade do modelo. Além disso, existem dois tipos principais de regularização: L1 e L2.

- L1: método de menor desvio absoluto, no qual o termo de penalidade corresponde à soma dos valores absolutos dos coeficientes utilizados na regressão LASSO;
- L2: método de mínimos quadrados, no qual o termo de penalidade é a soma dos quadrados dos coeficientes, sendo utilizado na regressão Ridge, apresentada na Subseção 3.1.6.

Deste modo, a regressão LASSO, ou operador de redução e seleção menos absoluta (LASSO, sigla em inglês), utiliza regularização e funções objetivas ao restringir o tamanho dos coeficientes de regressão. Diferentemente da regressão Ridge, que apenas reduz a magnitude dos coeficientes, a regressão LASSO pode levar alguns coeficientes exatamente a zero, promovendo a seleção de um conjunto de características do banco de dados para construir modelos mais enxutos. Como apenas os recursos necessários são utilizados na regressão LASSO, enquanto todos os outros são ajustados para zero, muitas vezes é possível evitar o ajuste excessivo do modelo. Além disso, a regressão LASSO geralmente requer a padronização dos dados para melhor desempenho. A formulação matemática para estimação dos coeficientes $\boldsymbol{\beta}$ na regressão LASSO é apresentada na Equação 16. Os coeficientes estimados $\hat{\boldsymbol{\beta}}$ são então utilizados na

equação clássica da regressão linear múltipla, apresentada na Equação 11, caracterizando a regressão LASSO.

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \left[\sum_{i=1}^{n} \left(Y_i - \left(\beta_0 + \sum_{j=1}^{k} \beta_j X_{ij} \right) \right)^2 + \lambda \sum_{j=1}^{k} |\beta_j| \right] \tag{16}$$

- Y_i: valor observado da variável resposta na i-ésima observação;
- X_{ij}: são os valores dos preditores para a i-ésima observação e a j-ésima variável preditora;
- $\beta_0, \beta_1, \ldots, \beta_k$: são os coeficientes a serem estimados;
- λ: Parâmetro de regularização, que controla a força da penalização aplicada aos coeficientes do modelo.

Essa regressão é amplamente utilizada em diversos campos para aprimorar previsões e interpretar resultados, ao reduzir a complexidade do modelo e selecionar variáveis significativas. Por exemplo, no trabalho de Miquelluti, Ozaki e Miquelluti (2022), a técnica é aplicada ao seguro de índice climático usando uma abordagem de LASSO quantílico geograficamente ponderado. Nesse contexto, a regressão é ajustada para considerar variações espaciais e utiliza informações de locais vizinhos para gerar estimativas robustas, melhorando a precisão e a relevância das previsões.

3.1.6 Regressão Ridge

A regressão Ridge, também conhecida como regressão de encolhimento ou regressão de cume, utiliza a regularização L2 para minimizar a função objetivo ao adicionar um termo de penalidade à soma dos quadrados dos coeficientes. Essa abordagem não somente ajuda a controlar a complexidade do modelo, evitando o ajuste excessivo como também, melhora o desempenho em dados de validação. É outra técnica de aprendizado de máquina que pode ser utilizada quando há uma forte correlação entre variáveis independentes, conhecida como multicolinearidade. Isso significa que, à medida que uma variável independente muda, outras podem mudar junto à ela.

Normalmente, as estimativas de mínimos quadrados produzem valores imparciais; entretanto, com dados multicolineares, essas estimativas podem ser distorcidas. Se a relação colinear for extremamente alta, a análise poderá produzir um valor de viés, ou seja, uma diferença entre o valor esperado e o valor verdadeiro de uma variável. Além disso, os modelos podem ajustar-se excessivamente aos dados de treinamento.

Um exemplo de aplicação pode ser encontrado no artigo intitulado “Matriz de Oferta Agropecuária: Uma Aplicação de Novas Técnicas de Regressão de Cume ” de Yamaguchi *et al.* (2019). Nesse estudo, é utilizado um modelo multiequacional de oferta de produtos agropecuários específicos para o Estado de São Paulo. O objetivo consiste em estimar, por meio da regressão

de cume, as elasticidades da oferta agropecuária em relação ao preço, bem como as elasticidades cruzadas, no curto e no longo prazo.

Esta regressão, por definição, é eficaz em contextos com alta multicolinearidade, nos quais as variáveis independentes estão fortemente correlacionadas. De forma adicional, em comparação com a regularização L1, a L2 tende a apresentar melhor desempenho e maior eficiência computacional.

A regularização L2 não altera a estrutura do modelo apresentado na Equação 11, mas modifica o processo de estimação dos coeficientes. Nesse contexto, os parâmetros $\boldsymbol{\beta}$ são obtidos pela minimização de uma função de custo penalizada, definida por:

$$\min\left(\sum_{i=1}^{n}\varepsilon_i^2+\lambda\beta^2\right)=\min\left(\sum_{i=1}^{n}(Y_i-(\beta_0+\beta_1X_{i1}+\beta_2X_{i2}+\ldots+\beta_kX_{ik}))^2+\lambda\sum_{j=1}^{k}\beta_j^2\right) \quad (17)$$

com os componentes:

- Y_i: São os valores observados da variável;
- X_{ij}: Valores preditores para a i-ésima observação e a j-ésima variável preditora;
- $\beta_0,\beta_1,\ldots,\beta_k$: Coeficientes a serem estimados;
- λ: Parâmetro de regularização, que controla a força da penalização aplicada aos coeficientes (ou pesos) do modelo.

3.1.7 Regressão polinomial

Semelhante à regressão linear múltipla, a análise de regressão polinomial é modificada e pode ser resumida como uma técnica para ajustar uma equação não linear tomando funções polinomiais da variável independente. Frequentemente, a linha de melhor ajuste gerada pela regressão polinomial é uma linha curva, em vez de reta. Assim, tem-se um polinômio de grau k em uma variável descrito como:

$$Y=\beta_0+\beta_1X_1+\beta_2X_2+\cdots+\beta_kX_k+e \quad (18)$$

sabendo que:

- Y: Variável dependente que se está tentando prever.
- β_0: Intercepto, valor de Y quando todas as variáveis X são zero;
- $\beta_1,\beta_2,\ldots,\beta_k$: Coeficientes de regressão que indicam a influência de cada variável independente $X_1,X_2,\ldots,X_k$;
- $X_1,X_2,\ldots,X_k$: Variáveis independentes ou preditoras;

- e: Termo de erro ou resíduo (diferença entre o valor observado e o valor previsto).

Ademais, uma preocupação ao utilizar regressão polinomial em modelos preditivos, que precisa ser contornada, é o ajuste excessivo. Esse fenômeno ocorre quando o modelo se adapta excessivamente bem aos dados de treinamento que passa a capturar não apenas os padrões reais, mas também o ruído e as flutuações aleatórias presentes nos dados. Como consequência, embora o modelo apresente um desempenho excelente durante o treinamento, sua capacidade de generalização para novos dados torna-se limitada, comprometendo a validade dos resultados em aplicações práticas.

4 METODOLOGIA

Este capítulo apresenta a metodologia adotada para o desenvolvimento e avaliação da proposta deste trabalho. Primeiramente, são descritos os materiais e ferramentas computacionais utilizados na implementação dos experimentos, incluindo as linguagens de programação, os ambientes de desenvolvimento e as plataformas de obtenção de dados. Em seguida, são detalhados os métodos empregados, contemplando as etapas de extração de dados da rede neural, a definição da nova função de ativação proposta e o processo de inclusão de novas camadas ocultas em uma rede *Stacked Autoencoder*. Por fim, são apresentadas as bases de dados utilizadas e os procedimentos experimentais adotados para a validação da abordagem proposta.

4.1 Materiais

4.1.1 Linguagem Python

O Python é uma linguagem de programação interpretada, interativa e orientada a objetos, que combina simplicidade com escalabilidade no desenvolvimento. Ele incorpora módulos, exceções, tipagem dinâmica, tipos de dados de alto nível e classes e fornece ao programador uma sintaxe clara e de fácil aprendizado. A linguagem é extensível, o que permite integração com outras como C e C++, sendo utilizada como linguagem de extensão para aplicativos como servidores *web*, sistemas de automação, ferramentas de análise de dados e jogos. Além disso, o Python é portátil, podendo ser executado em várias variantes do Unix, macOS e Windows 2000 ou versões posteriores. Seu uso é comprovado em sistemas empresariais de grande porte como YouTube, Google, Journyx, IronPort, EVE Online, HomeGain, Thawte Consulting, University of Maryland, Firaxis Games, o que contribui para a criação de aplicações globalmente expressivas, como plataformas de vídeo, produção de filmes, ferramentas de busca, jogos multiplayer massivos e sistemas de gerenciamento de recursos empresariais.

Aprender Python é acessível tanto para iniciantes quanto para programadores experientes, com muitos recursos de apoio, como a documentação oficial detalhada, cursos online gratuitos, comunidades ativas em fóruns e listas de discussão, além de conferências e encontros de programadores que promovem o aprendizado colaborativo. A linguagem também conta com um ecossistema que inclui o Índice de Pacotes Python (PyPI), que hospeda milhares de módulos de terceiros. O Python é desenvolvido sob uma licença de código aberto aprovada pela OSI, o que permite seu uso e distribuição livremente, inclusive para fins comerciais. Sua licença é gerida pela Python Software Foundation, o que garante a continuidade e evolução da linguagem (Foundation, 2026a; Foundation, 2026b).

Neste trabalho, a seleção, o armazenamento local e o pré-processamento das bases de dados, além da criação de gráficos preliminares, foram realizados utilizando a versão 3.12.4 da linguagem Python. Para isso, foram empregadas bibliotecas essenciais, como Pandas, Matplotlib.pyplot, Os, NumPy e Sklearn, além da conexão com a plataforma OpenML por meio de sua

biblioteca proprietária. Essas bibliotecas foram fundamentais nas etapas de pré-processamento, com os algoritmos implementados utilizando as ferramentas Jupyter Notebook e o Google Colaboratory.

4.1.2 Google Colaboratory e Jupyter Notebook

O Google Colaboratory, também denominado Colab, é um serviço de *notebooks* Jupyter hospedado, mantido pela Google Research, que possibilita a escrita e a execução de código Python por meio de um navegador *web*, dispensando a configuração prévia de ambientes locais. Conforme descrito na documentação oficial, a plataforma fornece um ambiente de execução previamente configurado, com as principais bibliotecas da linguagem Python, integração nativa com o Google Drive e suporte opcional a aceleradores de *hardware*, como unidades de processamento gráfico (GPUs) e unidades de processamento tensorial (TPUs). Tais características tornam o Colab uma ferramenta adotada nos setores público, privado e acadêmico, em contextos de pesquisa científica, ensino e experimentação em aprendizado de máquina (Google Research, 2026).

Nesse contexto, o Jupyter Notebook constitui a base tecnológica sobre a qual o Colab é construído e configura-se como uma aplicação *web* de código aberto, sem fins lucrativos, desenvolvida e mantida pelo Projeto Jupyter, cujo propósito é oferecer um ambiente de computação interativa que integre código executável, texto descritivo, visualizações e outros elementos multimídia em um único documento. Ademais, a ferramenta suporta múltiplas linguagens de programação por meio de kernels especializados, sendo considerada uma das principais ferramentas em ciência de dados, computação científica e educação, por favorecer fluxos de trabalho exploratórios, documentados e reprodutíveis (Project Jupyter, 2026).

4.1.3 Linguagem MATLAB

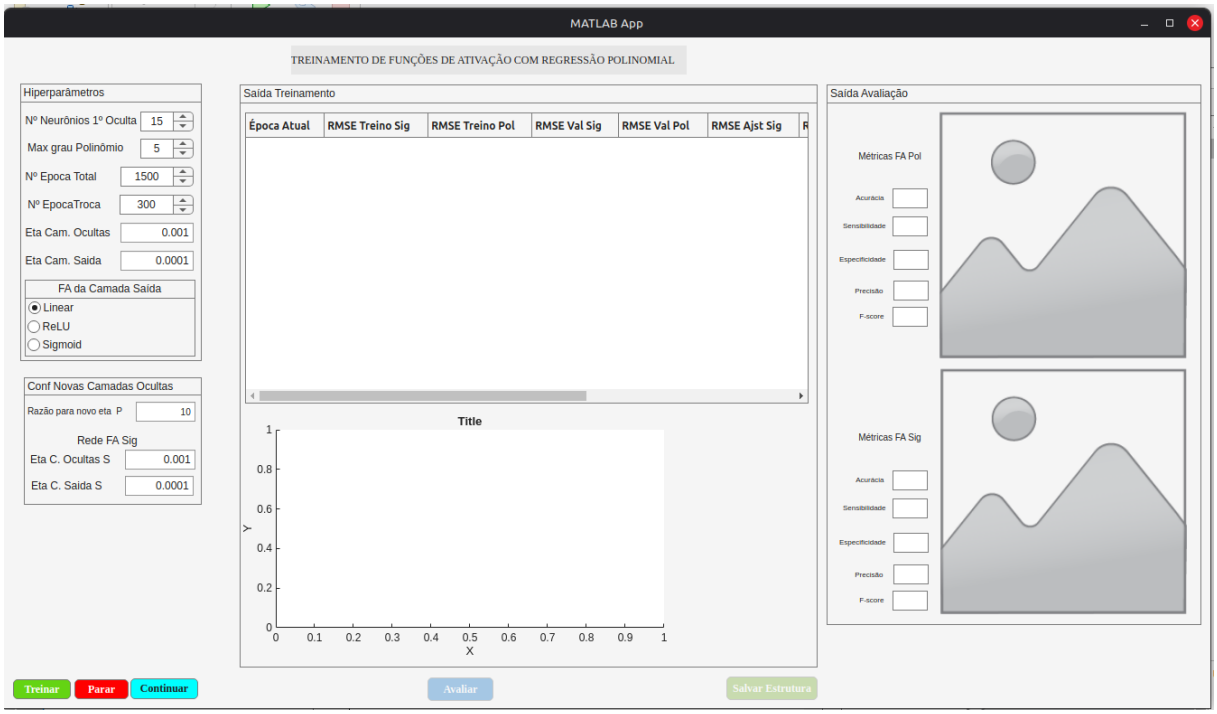
O MATLAB é uma plataforma de programação e computação numérica projetada especificamente para aplicações de engenharia e ciências, como análise de dados, processamento de sinais e imagens, controle de sistemas, comunicações sem fio e robótica. A plataforma fornece uma linguagem de programação de alto nível, que facilita a expressão direta de cálculos matriciais e de álgebra linear, essenciais para a modelagem matemática e simulação de sistemas complexos. Por conseguinte, o MATLAB conta com ferramentas especializadas que permitem escalar análises para clusters, GPUs e ambientes em nuvem, com mínima alteração de código, tornando-o altamente flexível para o processamento de grandes volumes de dados e aplicações em tempo real (MathWorks, 2026a).

Outrossim, o MATLAB ainda facilita a criação de aplicativos dinâmicos por meio de elementos visuais, a Interface Gráfica do Usuário (GUI), como botões e controles deslizantes, para que usuários interajam com os dados sem a necessidade de programação direta. Existem três maneiras principais de criar GUIs na linguagem: converter *scripts* em aplicativos simples

com o Editor ao Vivo, para que usuários modifiquem variáveis por meio de controles interativos; criar interfaces via ferramenta de arrastar e soltar; ou desenvolver a interface programaticamente, com controle total sobre o código. Essas abordagens tornam o MATLAB acessível não somente para iniciantes, mas também para desenvolvedores que buscam personalizar e criar soluções complexas (MathWorks, 2026b).

Em consonância com essas características do MATLAB, o algoritmo principal proposto neste trabalho foi desenvolvido paralelamente à implementação de uma interface gráfica do usuário (GUI), com o objetivo de proporcionar controle completo sobre os experimentos realizados. A Figura 11 apresenta a interface desenvolvida, destacando suas principais funcionalidades, enquanto a Figura 12 ilustra as informações disponibilizadas pela aplicação após a execução de um experimento.

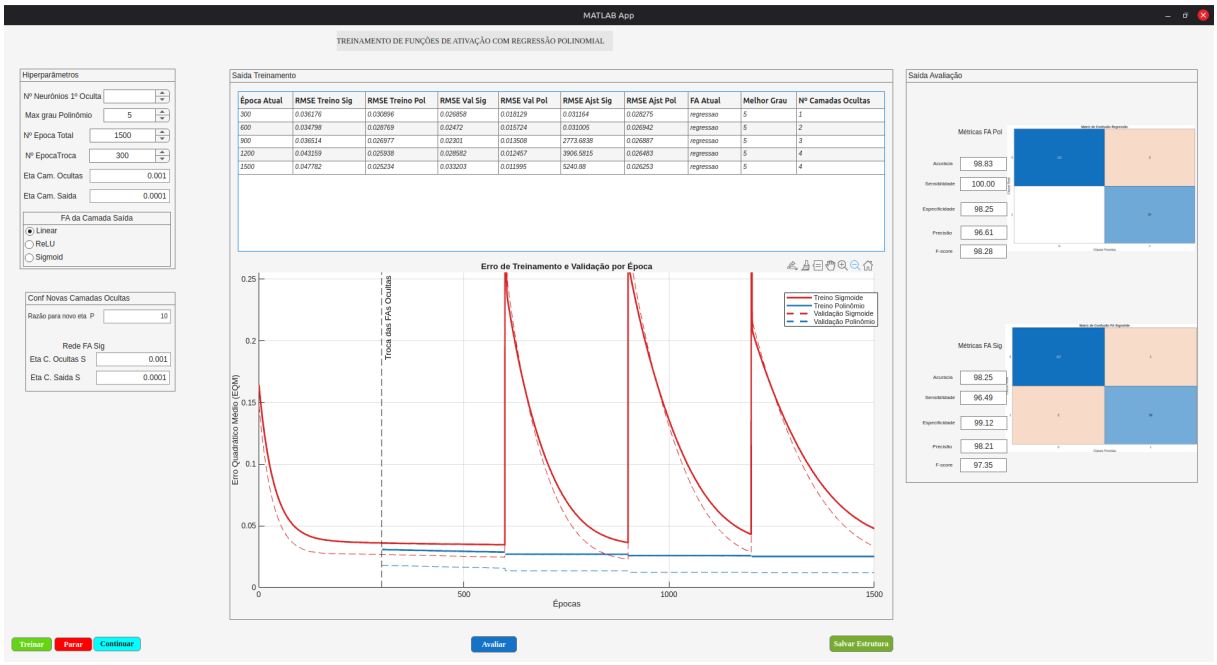
Figura 11 – Interface gráfica para realização dos experimentos no MATLAB



Fonte: Elaborado pelo autor (2025)

A interface desenvolvida tem como objetivo centralizar, de forma estruturada, a definição dos hiperparâmetros do modelo. Por meio do painel de entrada, o usuário pode configurar o número de neurônios ocultos, o número total de épocas, os critérios de troca das FAs, as taxas de aprendizado e os parâmetros associados à inserção de novas camadas. Essa organização assegura controle preciso sobre o comportamento da rede e promove a padronização dos experimentos, o que resulta em maior reprodutibilidade dos resultados e na redução de intervenções manuais no código.

Figura 12 – Informações fornecidas pela aplicação após sua execução



Fonte: Elaborado pelo autor (2025)

Adicionalmente, a interface apresenta a visualização gráfica do erro de treinamento e validação ao longo das épocas, o que possibilita a análise do processo de convergência e da estabilidade do modelo da RNA. As métricas de desempenho são exibidas juntamente com as matrizes de confusão, para uma avaliação quantitativa da capacidade de classificação da rede. Os botões de controle, como treinar, parar, continuar, avaliar e salvar estrutura, organizam a condução dos experimentos, sendo que a opção salvar estrutura registra a configuração da rede, os pesos obtidos e as métricas correspondentes. Dessa forma, os experimentos permanecem sistematizados e devidamente documentados no próprio ambiente computacional.

4.1.4 Plataforma OpenML

A OpenML é uma plataforma de código aberto para compartilhamento de *datasets*, algoritmos e experimentos em aprendizado de máquina. Ela suporta tarefas específicas de *machine learning*, como classificação, regressão e aprendizado profundo, e assegura transparência e reprodutibilidade nas pesquisas. Sua interface de programação de aplicações (API) aberta permite a integração com ferramentas como Python e R. Sua natureza colaborativa possibilita que usuários contribuam com dados, algoritmos e resultados.

Conforme proposto por Bischl *et al.*(2021), o OpenML fornece conjuntos de dados e divisões padronizadas, conhecidos como "*Benchmark Suites*", utilizados em experimentos para avaliar algoritmos de aprendizado de máquina. Eles simplificam o intercâmbio de configurações e resultados ao fornecer conjuntos de dados consistentes, com divisões de treino e teste padronizadas, com foco na reprodutibilidade dos experimentos.

O principal exemplo é o OpenML-CC18, que consiste em 72 conjuntos de dados de classificação, cuidadosamente selecionados para *benchmarking* pragmático, os dados incluem entre 500 e 100.000 instâncias e no máximo 5.000 atributos, atendendo a critérios específicos para garantir eficiência nos testes (OpenML, 2025).

4.1.5 Base de dados

Mushroom

A base de dados "Mushroom", originalmente disponibilizada pela UCI e atualmente acessível por meio da plataforma OpenML como uma base de dados no conjunto de *benchmark* OpenML100, descreve amostras hipotéticas de cogumelos com lamelas, estruturas finas e alongadas, dispostas radialmente na parte inferior do chapéu dos cogumelos, responsáveis pela produção e liberação dos esporos, pertencentes às famílias *Agaricus* e *Lepiota*.

Construído a partir das informações apresentadas na obra *The Audubon Society Field Guide to North American Mushrooms* (Lincoff, 1997) o conjunto compreende 23 espécies caracterizadas por atributos físicos e sensoriais, incluindo formato, superfície e cor do chapéu, propriedades das lamelas, do caule, do véu, do anel, do esporo, bem como odor, habitat e população. A base contém 8.124 instâncias, cada uma rotulada quanto à comestibilidade em duas classes, comestível ou venenosa, sendo a categoria originalmente definida como de comestibilidade desconhecida incorporada à classe venenosa.

DNA

Outra base de dados utilizada é denominada "DNA", originalmente derivada do projeto Statistical and Logic Based Learning (StatLog) e posteriormente disponibilizada por meio da biblioteca MLbench, com os registros brutos mantidos no repositório da UCI Machine Learning. O conjunto também está disponível na plataforma OpenML, integrando o *benchmark* OpenML-CC18. A base contém 3.186 instâncias correspondentes a junções de *splicing*, cada uma descrita por 180 atributos binários. O problema consiste na classificação de três tipos de fronteiras genéticas: exon–intron (*ei*), intron–exon (*ie*) e nenhuma das anteriores. As variáveis simbólicas originais, que representavam os nucleotídeos A, G, T e C, foram transformadas em atributos binários, resultando na conversão de 60 atributos simbólicos em 180 variáveis indicadoras. Exemplos ambíguos foram removidos, totalizando a exclusão de quatro registros.

German Credit (Credit-g)

Já a base de dados "German Credit", disponibilizada também pelo repositório UCI Machine Learning e integrante do conjunto de *benchmark* OpenML-CC18, destina-se à classificação de indivíduos quanto ao risco de crédito. Cada instância descreve um solicitante por meio de atributos socioeconômicos e financeiros, incluindo posição financeira da conta corrente, duração do crédito, histórico de crédito, finalidade, valor solicitado, situação de poupança, tempo de emprego, idade, tipo de moradia, número de créditos existentes. A variável alvo categoriza os

indivíduos como bons ou maus pagadores. O conjunto incorpora ainda uma matriz de custos associada a erros de classificação, na qual classificar um cliente de alto risco como bom implica custo superior ao erro inverso, refletindo cenários reais de tomada de decisão em análise de crédito. O conjunto é composto por 1.000 instâncias, com 20 atributos e classe binária.

PC4

A base de dados "PC4" integra o programa NASA Metrics Data e foi desenvolvida por Mike Chapman, sendo posteriormente disponibilizada pelo repositório tera-PROMISE. O conjunto foi obtido a partir de sistemas de *software* embarcados em satélites em órbita terrestre e tem como objetivo a predição de defeitos em código-fonte. As instâncias são descritas por métricas extraídas automaticamente do código, incluindo medidas de complexidade e tamanho baseadas nos modelos de McCabe e Halstead, amplamente utilizados para caracterização objetiva da qualidade de *software*. O conjunto PC4 é empregado em estudos de engenharia de *software* empírica e avaliação de modelos preditivos, discutido em diversos trabalhos relacionados à qualidade e confiabilidade de dados de defeitos de *software*.

KR-VS-KP

A base de dados "KR-VS-KP", outro conjunto disponibilizado pelo UCI Machine Learning e originalmente desenvolvida por Alen Shapiro, representa posições de tabuleiro em um fim de jogo de xadrez no qual o lado branco possui um rei e uma torre, enquanto o lado preto possui um rei e um peão localizado na casa a7, a um lance de ser promovido. Cada instância descreve uma configuração do tabuleiro por meio de 36 atributos simbólicos, além de uma variável de classe que indica se o lado branco pode vencer (*won*) ou não (*nowin*). O conjunto contém 3.196 instâncias, sem valores ausentes, e apresenta distribuição de classes relativamente equilibrada. Essa base foi utilizada em estudos clássicos de aprendizado de conceitos e indução estruturada em sistemas especialistas.

Steel Plates Faults

A base de dados "Steel Plates Faults", fornecida pelo centro de pesquisa Semeion, contém informações sobre defeitos industriais em chapas de aço. Cada instância é descrita por 27 atributos numéricos relacionados a propriedades geométricas e de luminosidade, bem como por atributos binários que indicam a ocorrência de diferentes tipos de falhas. Os defeitos são classificados em sete categorias distintas, incluindo riscos, manchas, sujeira e outras anomalias. Em aplicações práticas, essas classes podem ser agrupadas para formulações de classificação binária. O conjunto é empregado em estudos de reconhecimento de padrões e controle de qualidade industrial.

Bioresponse

A base de dados "Bioresponse", disponibilizada pela empresa Boehringer Ingelheim por meio da plataforma Kaggle, tem como objetivo a predição da resposta biológica de moléculas com base em suas propriedades químicas. Cada instância representa uma molécula, sendo a va-

riável alvo definida por um valor binário que indica a presença ou ausência de resposta biológica observada experimentalmente. As demais colunas correspondem a 1.776 descritores moleculares normalizados, que capturam características estruturais e físico-químicas das moléculas. O conjunto original de treinamento e teste foi unificado, tornando a base adequada para a avaliação de modelos de classificação em contextos de alta dimensionalidade.

Tabela 1 – Resumo das bases de dados utilizadas

ID OpenML	Nome	N.º Atributos	N.º Instâncias	N.º Classes	Domínio
3	kr-vs-kp	36	3 196	2	Jogos / Xadrez
15	Breast-Cancer-W	10	699	2	Doença/Medicina
24	mushroom	22	8 124	2	Biologia / Classificação de Espécies
31	credit-g	20	1 000	2	Finanças / Análise Bancária
1049	pc4	37	1 458	2	Engenharia de *software*
1504	steel-plates-fault	33	1 941	2	Engenharia / Qualidade Industrial
4134	bioresponse	1 777	3 751	2	Química / Bioinformática
40670	DNA	180	3 186	3	Biologia Molecular

Fonte: Elaborado pelo Autor (2025)

4.2 Métodos

O desenvolvimento deste trabalho foi estruturado em três etapas principais. Primeiramente, são extraídos os dados referentes à FA de um modelo de rede neural MLP. Em seguida, é criada uma nova função de ativação baseada em regressão polinomial, a qual é treinada individualmente para cada neurônio da camada oculta. Na terceira etapa, novas camadas ocultas são adicionadas, de modo que os elementos da regressão polinomial são decompostos e distribuídos nessas camadas. Essas etapas são detalhadas nas subseções 4.2.2, 4.2.3 e 4.2.4, respectivamente.

4.2.1 Pré-processamento

O pré-processamento dos dados foi realizado de forma padronizada para todas as bases utilizadas neste estudo. Inicialmente, utilizou-se a biblioteca Pandas, em Python, para organizar os conjuntos de dados no formato tabular. Posteriormente, um algoritmo desenvolvido em MATLAB foi empregado para processar os arquivos no formato CSV, com o objetivo de garantir consistência experimental e evitar vieses durante o treinamento dos modelos.

Inicialmente, cada base de dados foi carregada e separada em uma matriz de atributos de entrada $\mathbf{X}$ e um vetor de rótulos $\mathbf{Y}$, sendo a última coluna correspondente à variável alvo e as demais às características do problema.

Em seguida, foi realizada uma etapa de embaralhamento aleatório das amostras, utilizando uma semente fixa (42), com o objetivo de garantir a reprodutibilidade dos experimentos. A partir dessa permutação, os dados foram divididos em quatro subconjuntos: treinamento, validação, ajuste_FA e teste. Considerando as particularidades de cada conjunto de dados, os percentuais de divisão foram definidos com base em experimentos preliminares, conforme apresentado na Tabela 2.

Tabela 2 – Divisão percentual dos conjuntos de dados

ID OpenML	Treino (%)	Validação (%)	Ajuste FAs (%)	Teste (%)
3	70%	6%	4%	20%
15	60%	7,5%	7,5%	25%
24	60%	7,5%	7,5%	25%
31	64%	8%	8%	20%
1049	64%	8%	8%	20%
1504	60%	7,5%	7,5%	25%
4134	64%	8%	8%	20%
40670	72%	4%	4%	20%

Fonte: Elaborado pelo Autor (2026)

Após a divisão, foi aplicada a normalização dos atributos por meio da técnica de escalonamento Min-Max, conforme a Equação 19:

$$X_{\text{norm}} = \frac{X - X_{\text{min}}}{X_{\text{max}} - X_{\text{min}}} \tag{19}$$

Por fim, os parâmetros X_{min} e X_{max} foram obtidos exclusivamente a partir do conjunto de treinamento e posteriormente reutilizados para normalizar os conjuntos de validação, ajuste FAs e teste. Esse procedimento evita o vazamento de informação (*data leakage*) e garante que os dados de teste e o processo de ajuste das FAs permaneçam independentes durante o treinamento.

4.2.2 Extração de dados da rede neural

Com base nos fundamentos teóricos presentes na literatura, o treinamento da rede neural inicia-se com a utilização de uma função de ativação fixa, neste caso, adotou-se a função sigmoide. Durante esse processo, além da determinação dos pesos, são armazenados os parâmetros adicionais, como a saída da camada oculta Y_h e o erro E, os quais são fundamentais para a definição do valor estimado de D_h (valor desejado) para cada neurônio da camada oculta. Esse valor é calculado a partir das definições a seguir.

Sabe-se que a saída de um neurônio na camada de saída pode ser expressa pela seguinte equação:

$$Y_o = W_{oh} \cdot Y_h \tag{20}$$

considerando que a camada de saída possui apenas um neurônio e a camada escondida n neurônios, onde Y_o representa a saída da rede, W_{oh} é o vetor de pesos de dimensão $1 \times n$ entre a camada oculta e a camada de saída e Y_h é a saída da camada oculta, representada por um vetor de dimensão $n \times 1$. Ou seja, a rede produz uma saída Y_o tal que:

$$Y_o = \sum_{i=1}^{n} {}^{i}W_{oh} \cdot {}^{i}Y_h \tag{21}$$

Por sua vez, o erro E na saída é definido por:

$$E = D - Y_o \quad \Rightarrow \quad D = Y_o + E \tag{22}$$

onde D é a saída desejada da rede e Y_o é a saída produzida pela rede.

Ademais, pode-se considerar que existe um certo valor de saída da camada escondida que produz o valor desejado D na saída da rede. Esse valor de saída da camada escondida é nomeado como D_h. Deste modo, D_h é o valor desejado de saída camada oculta. Embora esse valor não seja conhecido diretamente, ele pode ser estimado por definição, conforme expresso a seguir:

$$W_{oh} \cdot D_h = D \quad \Rightarrow \quad \sum_{i=1}^{n} {}^{i}W_{oh} \cdot {}^{i}D_h = D \tag{23}$$

Assim, a relação entre o valor desejado D e o valor obtido Y_o, ao substituir as Eq. 21 e Eq. 23 na Eq. 22, pode ser reescrita como:

$$\sum_{i=1}^{n} {}^{i}W_{oh} \cdot {}^{i}D_h = \sum_{i=1}^{n} {}^{i}W_{oh} \cdot {}^{i}Y_h + E \tag{24}$$

Como o erro final E, Eq. 22, depende diretamente das ativações da camada escondida e dos pesos associados, uma vez que estas são utilizadas como entrada para a camada de saída, cada neurônio da camada escondida é responsável, indiretamente, pelo erro E. Entretanto, a contribuição individual de cada neurônio oculto para esse erro não é explicitamente conhecida.

Diante dessa limitação, adotou-se, neste trabalho, uma abordagem estimada para a decomposição do erro na camada de saída. Considerando cada neurônio i da camada oculta, assume-se que o erro de saída da rede pode ser distribuído entre as conexões provenientes da camada escondida por meio de um fator de proporcionalidade ou por meio de uma distribuição.

Individualizando a Eq. 24 para cada neurônio da camada oculta, para uma formulação desacoplada por neurônio, considerou-se que o erro total E é distribuído entre as conexões provenientes da camada escondida de modo uniforme. Dessa forma, estabeleceu-se a seguinte relação para cada neurônio escondido:

$${}^{i}W_{oh} \cdot {}^{i}D_h = {}^{i}W_{oh} \cdot {}^{i}Y_h + \frac{E}{n} \tag{25}$$

em que $\frac{E}{n}$ representa uma parcela do erro E, sendo n um fator de distribuição e um número inteiro. A função desse fator é repartir o erro associado ao neurônio de saída entre as conexões provenientes dos neurônios da última camada escondida.

A parcela de contribuição de cada neurônio para o erro final pode ser definida de diferentes formas, de modo que a contribuição de cada conexão para o erro total da rede seja ponderada de maneira distinta, ou com base em medidas específicas ou por meio de funções de distribuição. Porém, neste trabalho, adotou-se uma distribuição uniforme, na qual n corresponde ao número de neurônios na camada escondida, implicando que cada conexão recebe igualmente a mesma parcela do erro E. Então, a saída desejada para qualquer neurônio da camada escondida ${}^{i}D_h$, que será utilizada na criação da nova função de ativação baseada em regressão polinomial, é considerando ${}^{i}W_{oh} \neq 0$:

$${}^{i}D_h = {}^{i}Y_h + \frac{E}{n \cdot {}^{i}W_{oh}} \tag{26}$$

em que:

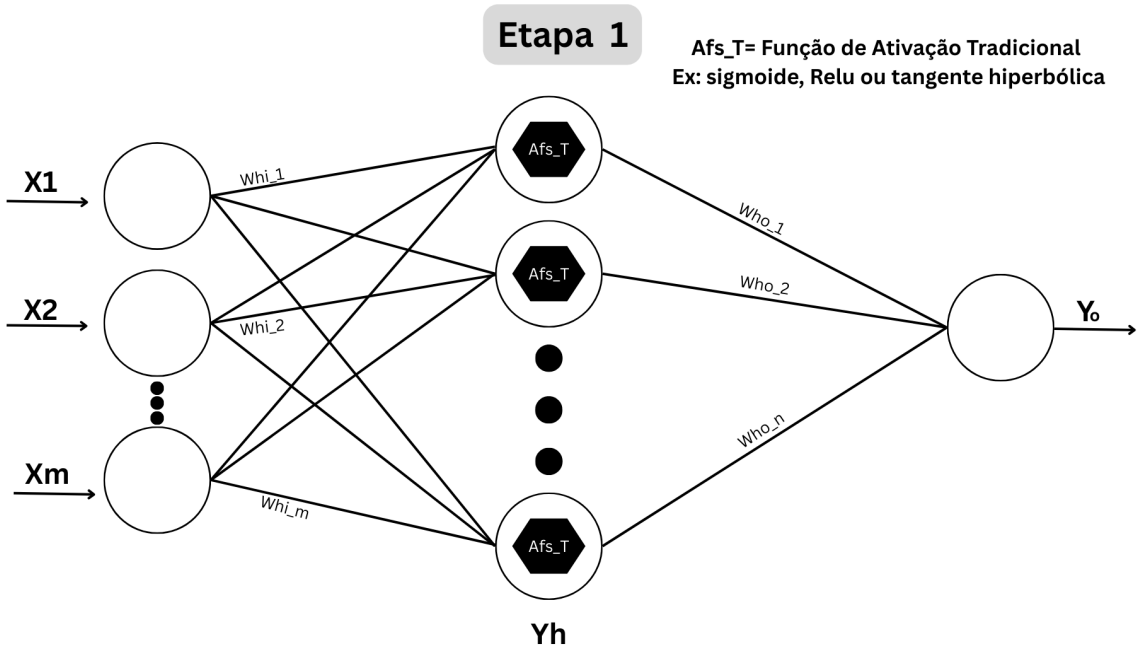
- ${}^{i}D_h$ é o vetor de saída desejada para o neurônio oculto i;
- ${}^{i}Y_h$ representa a saída atual do neurônio oculto i;
- E é o vetor de erro;
- n é o número total de neurônios na camada oculta;
- ${}^{i}W_{oh}$ é o vetor de pesos entre o neurônio oculto i e os neurônios da camada de saída;

Ressalta-se que outras estratégias de distribuição não foram investigadas neste estudo. Como exemplos, o fator de distribuição poderia ser definido de forma proporcional aos pesos das conexões, atribuindo maior parcela do erro às conexões de maior magnitude, ou ainda baseado em distribuições não uniformes, como uma distribuição normal, em que determinadas conexões teriam maior influência no erro total.

4.2.3 Nova função de ativação

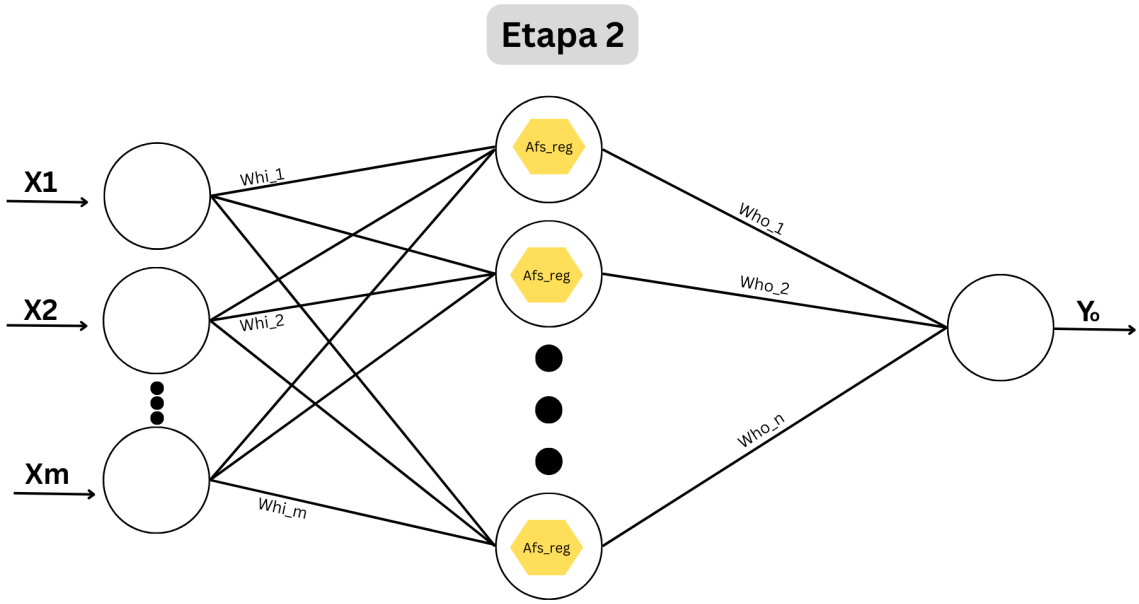
A nova FA parte da necessidade de automatizar a escolha do grau polinomial mais adequado para um modelo de regressão. Para isso, foram utilizadas as funções polyfit e polyval do MATLAB, pois permitem a adaptação dinâmica do modelo durante o treinamento da RNA. Inicialmente, a rede é treinada com FAs sigmoide, conforme a Figura 13. Em seguida, ocorre a transição para a função de regressão, buscando otimizar a precisão do ajuste, Figura 14.

Figura 13 – Extração de dados da rede neural - etapa 1.



Fonte: Elaborado pelo Autor (2025)

Figura 14 – Substituição da função de ativação tradicional pela função de regressão — etapa 2.



Fonte: Elaborado pelo Autor (2025)

Durante esse processo, os dados são normalizados para garantir estabilidade numérica e minimizar variações indesejadas. A busca pelo grau polinomial ótimo é realizada de forma automatizada, iniciando em 1 e aumentando progressivamente até o limite da capacidade da função polyfit para os dados de entrada. Entretanto, como a seleção do grau polinomial baseada exclusivamente nas métricas de erro do treinamento tende a superestimar o desempenho da rede, esse procedimento pode levar ao problema de *overfitting*, comprometendo a capacidade de generalização da rede.

Por essa razão, o grau polinomial final de cada neurônio na camada oculta é selecionado com base na avaliação em instâncias não utilizadas no treinamento, considerando aquele que apresenta o menor erro quadrático médio (RMSE). Esse procedimento demanda não apenas selecionar o grau mais adequado para a generalização da rede, como também explorar diversos mínimos locais ao testar múltiplas configurações polinomiais.

4.2.4 Novas camadas ocultas

A inserção de novas camadas ocultas ocorre com a introdução de uma nova camada por vez. Para a inserção de uma nova camada oculta, após os processos descritos nas Subseções 4.2.2 e 4.2.3, adota-se uma técnica inovadora baseada na reestruturação da FA da camada anterior. Essa abordagem tem como objetivo promover uma redistribuição gradual da complexidade da rede, de modo não apenas a controlá-la, mas também a permitir a inserção sucessiva de novas camadas ocultas sem comprometer a estabilidade do modelo e o aprendizado já obtido.

Inicialmente, a função de regressão polinomial associada à camada oculta, conforme

descrita na Eq. 27, é decomposta em dois blocos principais: o termo de menor grau é isolado e mantido na camada original, enquanto os demais termos do polinômio são ajustados para compor as FAs da camada oculta recém-inserida.

A regressão polinomial original é definida por:

$$AFs\_reg(z) = \sum_{i=1}^{n} a_i z^i + a_0 \tag{27}$$

O processo de decomposição baseia-se em funções compostas, sendo modelado da seguinte forma: seja $y = h(z)$, pode-se reescrever y como $y = f(x)$ e $x = g(z)$. Nesse contexto, a função $g(z)$ é responsável por introduzir progressivamente transformações não lineares da entrada, enquanto a função $f(x)$ realiza a combinação final dos termos polinomiais.

Para os neurônios da primeira camada oculta, a função de ativação é denotada por $\text{AFs}_{1\text{reg}}$ e corresponde à aplicação direta da entrada líquida, isto é, ao termo linear do polinômio, definindo-se:

$$\text{AFs}_{1\text{reg}}(z) = z = x \tag{28}$$

onde z representa a entrada líquida do neurônio e x é uma variável intermediária utilizada na composição das camadas subsequentes.

Para os neurônios da camada recém-inserida, a função de ativação (FA) recebe os termos polinomiais restantes. A fim de preservar o aprendizado já alcançado, o novo polinômio, quando avaliado com o valor fornecido pela Eq. 28, deve necessariamente reproduzir a mesma saída da Eq. 27. Seja $\text{AFs}_{2\text{reg}}$ a FA da segunda camada oculta. Como $x = g(z) = z$, segue que:

$$\text{AFs}_{2\text{reg}}(x) = \sum_{i=1}^{n} a_i x^i + a_0 \tag{29}$$

Essa condição é válida, pois $n > 0$. Além disso, ao substituir a Eq. 28 na Eq. 29, obtém-se novamente a Eq. 27.

Para a inserção de uma terceira camada oculta a partir do isolamento do termo quadrático, surge uma limitação importante: a transformação quadrática elimina a informação de sinal, de modo que valores negativos passam a ser representados como positivos após a segunda camada. Como consequência, o polinômio aplicado sobre essa saída passa a operar sobre uma representação ambígua, produzindo resultados distintos dos esperados para o valor original. Para contornar esse problema, é necessário garantir que a informação de sinal da segunda camada seja propagada para a terceira camada.

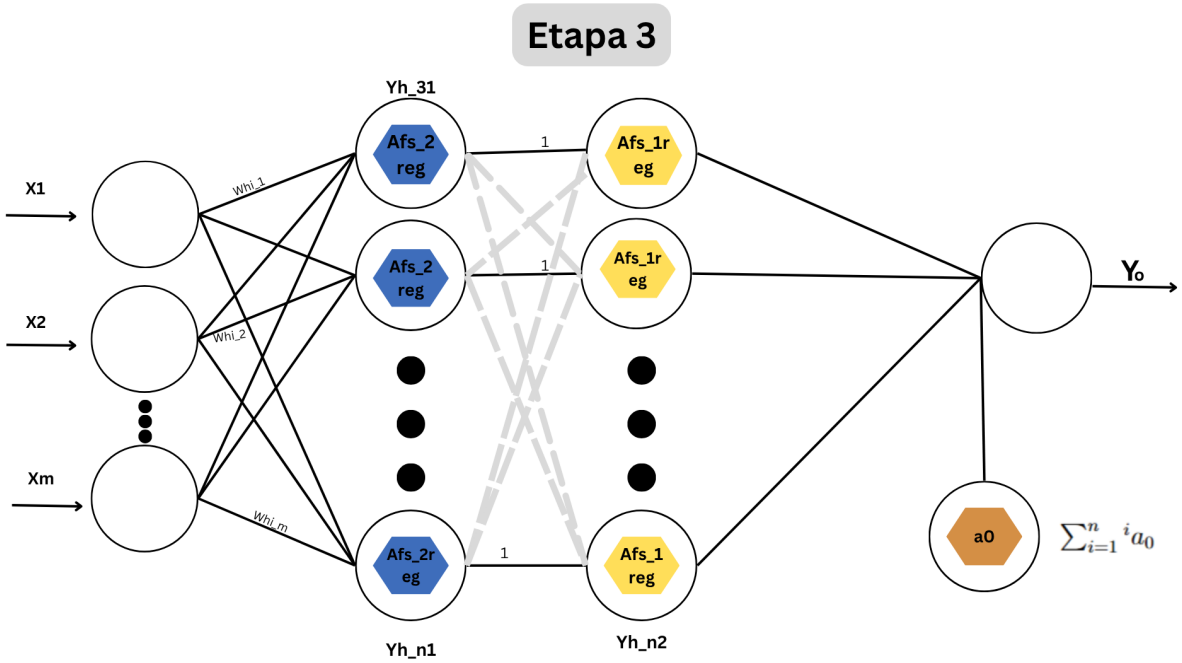
Além disso, durante o cálculo das ativações nas camadas ocultas, foi necessário aplicar um processo de radiciação aos termos intermediários associados à decomposição do polinômio utilizado como FA. Entretanto, quando o índice da raiz é par, a operação de radiciação não está definida no conjunto dos números reais para valores negativos, resultando em números complexos. Como a rede neural utilizada neste trabalho opera exclusivamente no domínio dos números reais, a presença de valores complexos poderia comprometer a estabilidade numérica da rede e inviabilizar o processo de treinamento.

Para contornar esse problema, adotou-se uma transformação que preserva o sinal da entrada. Nesse procedimento, calcula-se inicialmente a raiz sobre o valor absoluto do valor de entrada e, posteriormente, o sinal original é reintroduzido no resultado. Essa estratégia permite que a operação de radiciação seja aplicada independentemente do sinal da entrada, evitando a geração de números complexos. Além disso, essa abordagem mantém a simetria da transformação em torno da origem, ou seja, preserva a polaridade do valor aplicado às FAs e possibilita a utilização consistente da FA baseada em termos polinomiais ao longo das camadas ocultas da rede neural.

Outra característica essencial do modelo diz respeito à forma como a segunda camada oculta recebe os dados da camada anterior. Para garantir que a saída da primeira camada oculta seja corretamente propagada até a camada de saída, a matriz de pesos entre as duas últimas camadas ocultas deve ser a matriz identidade. Essa configuração preserva o conhecimento previamente adquirido pela rede, o que não ocorreria caso a matriz de pesos fosse inicializada de outra maneira.

Adicionalmente, no processo anteriormente descrito, o termo constante do polinômio das FAs da segunda camada pode ser separado e incorporado como *bias* da camada de saída. A Figura 15 ilustra o processo de inserção de uma nova camada oculta.

Figura 15 – Processo de inserção de uma nova camada oculta — etapa 3.



Fonte: Elaborado pelo Autor (2025)

No esquema da Figura 15, a nova camada escondida é representada na cor azul. As conexões entre as camadas de entrada e escondida, bem como entre a escondida e a de saída, não são alteradas durante a inserção de uma nova camada. As conexões na cor cinza são inicializadas com valor nulo, de modo que, no início do treinamento da nova camada escondida, apenas os

neurônios da camada escondida anterior alimentam os neurônios correspondentes na camada seguinte.

O processo acima pode ser repetido para a inserção de novas camadas ocultas. As novas camadas adicionadas recebem os termos do polinômio, que representam contribuições progressivamente maiores em grau, onde os termos isolados são crescentes em potência a cada camada. Cada camada nova recebe a saída da penúltima camada escondida anterior, de modo a preservar a saída da função original, ao mesmo tempo em que busca expandir a capacidade de generalização da rede. Nesse processo repetitivo, as FAs da última camada escondida correspondem sempre ao polinômio ajustado, enquanto a camada anterior a essa última camada escondida retém um termo isolado do polinômio.

Esse procedimento ocorre de forma iterativa, respeitando a estrutura hierárquica da rede, de modo que cada camada adicionada corresponda a um termo do polinômio. A inserção e, consequentemente, a divisão dos termos são ajustadas durante o treinamento, conforme as necessidades do modelo. Ou seja, à medida que o RMSE se estabiliza, a estrutura da rede é modificada para permitir a adição de uma nova camada.

5 RESULTADOS E DISCUSSÃO

Os resultados foram obtidos considerando variações na configuração da rede, nos conjuntos de dados e nas inicializações dos pesos. A combinação de oito bases de dados, três variantes de FAs sigmoide, polinomial baseada em Y_h e polinomial baseada em D_h, quatro quantidades de neurônios ocultos (3, 10, 15 e 20) e 50 diferentes inicializações de pesos totalizou 4.800 treinamentos independentes.

Como exemplo ilustrativo, as Figuras 16, 17, 18 e 19 apresentam o comportamento típico do erro quadrático médio (RMSE) durante o treinamento das RNAs. Nessas figuras, são exibidas curvas correspondentes a três redes distintas, diferenciadas pelas FAs adotadas nas camadas ocultas:

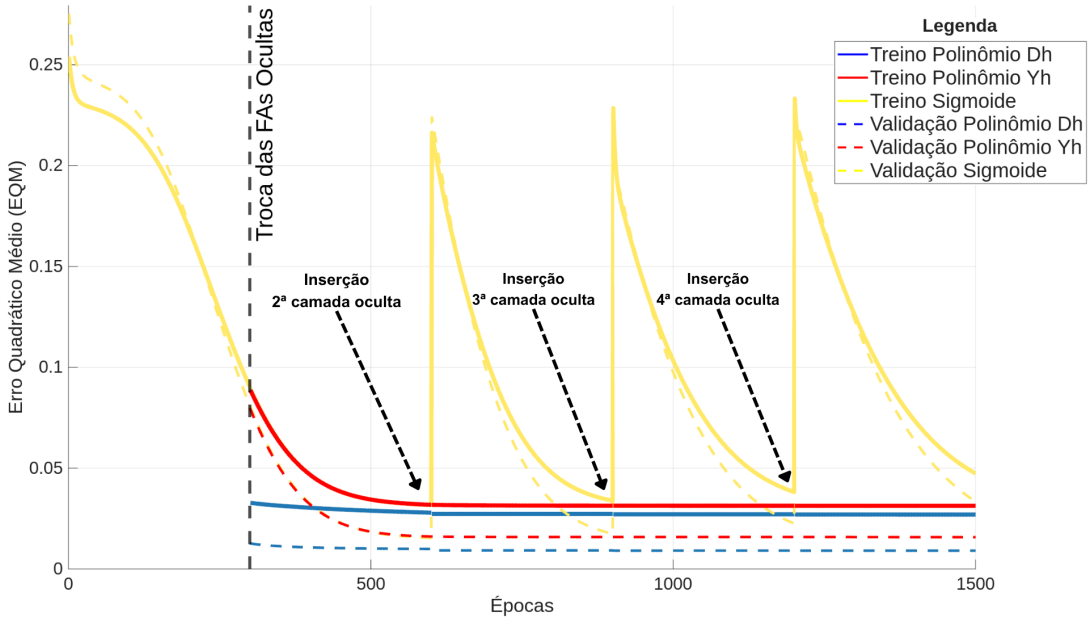
(i) uma rede com FAs baseadas na função sigmoide (SIG);

(ii) uma rede com FAs definidas por uma função polinomial obtida via regressão, tendo como referência a saída dos neurônios originais Y_h, definida neste trabalho como POL-Yh;

(iii) uma rede cujas FAs são modeladas por uma função polinomial obtida via regressão, considerando como alvo a saída D_h, definida como POL-Dh.

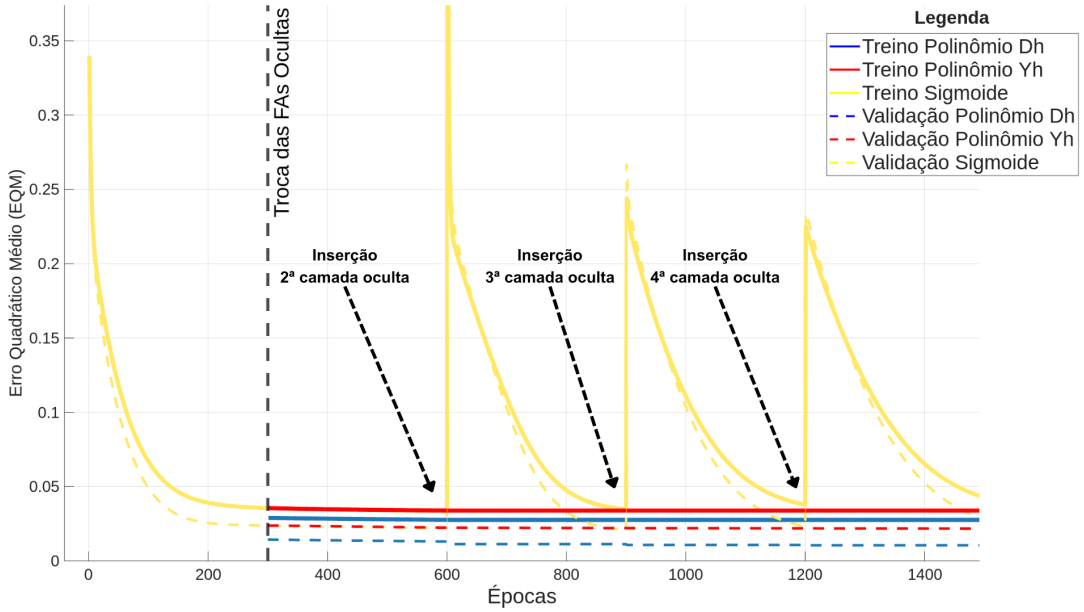
O experimento foi conduzido utilizando o conjunto de dados Kr-Ks-Kp, com 3, 10, 15 e 20 neurônios ocultos, conforme indicado nas respectivas figuras, totalizando 1.500 épocas de treinamento e taxa de aprendizagem inicial de 1×10^{-4}. As redes neurais foram modificadas na época 300, com a substituição das FAs obtidas por regressão associadas a POL-Dh e POL-Yh A cada 300 épocas, uma nova camada oculta foi adicionada, com redução sucessiva da taxa de aprendizagem (η) em 10^{-2}, até que quatro camadas ocultas fossem incorporadas à arquitetura.

Figura 16 – Evolução do MSE com novas FA e inserção de camadas ocultas (3 neurônios)



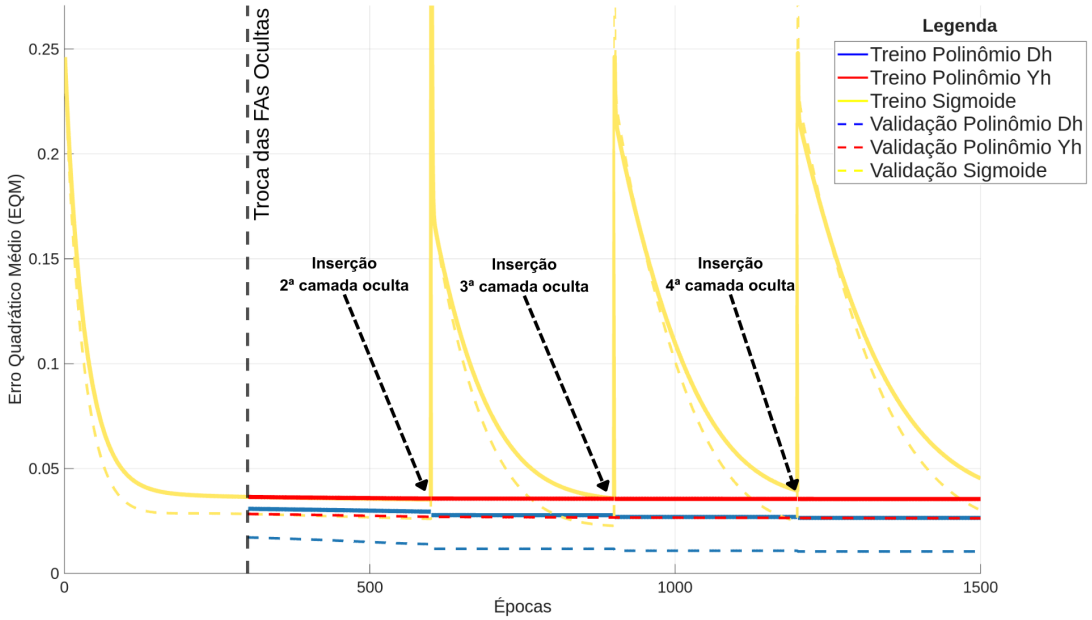
Fonte: Elaborado pelo Autor (2025)

Figura 17 – Evolução do MSE com novas FA e inserção de camadas ocultas (10 neurônios)



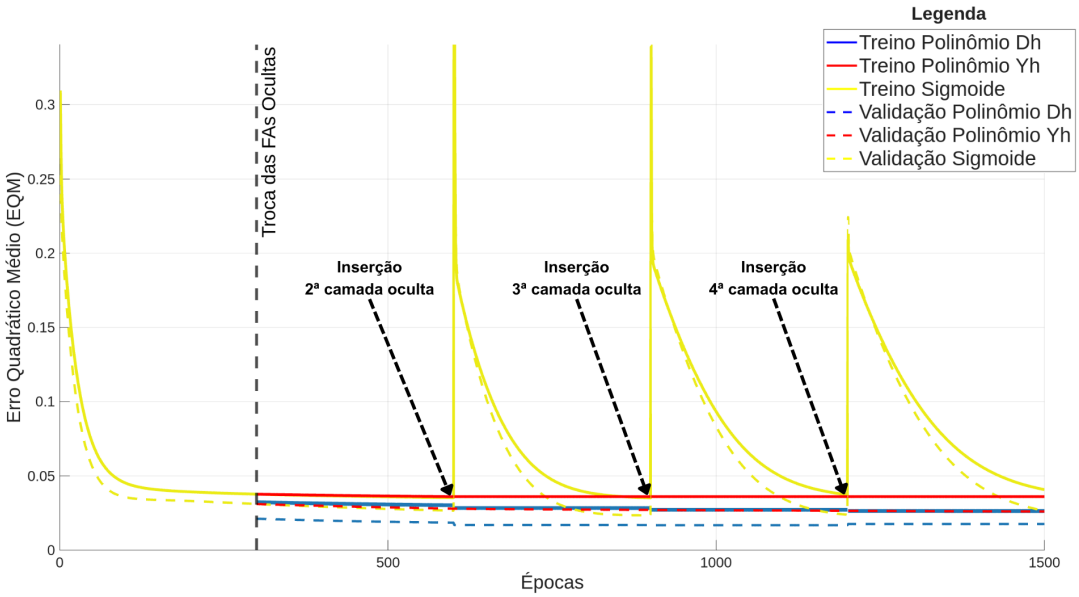
Fonte: Elaborado pelo Autor (2025)

Figura 18 – Evolução do MSE com novas FA e inserção de camadas ocultas (15 neurônios)



Fonte: Elaborado pelo Autor (2025)

Figura 19 – Evolução do MSE com novas FA e inserção de camadas ocultas (20 neurônios)



Fonte: Elaborado pelo Autor (2025)

Ao comparar as três funções de ativação distintas, a sigmoide (curvas em amarelo), a polinomial POL-Yh (curvas em vermelho) e a polinomial POL-Dh (curvas em azul), observa-se que as linhas contínuas representam o erro para os dados de treinamento, enquanto as linhas tracejadas representam o erro para os dados de validação. A função sigmoide inicia com valores de erro significativamente mais elevados, pois constitui a função base para a criação das novas FAs ao final da época 300, neste conjunto de dados. As redes que utilizam a função sigmoide apresentam oscilações acentuadas sempre que ocorre a inserção de uma nova camada oculta. Em contrapartida, as funções polinomiais apresentam maior estabilidade, redução progressiva do erro e menor discrepância entre os conjuntos de treinamento e validação. Ainda se destaca a presença de curvas com quedas mais acentuadas ao longo do treino e da validação, especialmente observadas nas Figuras 18 e 19.

A cada inserção de uma nova camada oculta, indicada como 2ª, 3ª e 4ª camadas nas Figuras 16, 17, 18 e 19, ocorre um aumento do erro nas redes com função sigmoide, seguido de nova convergência, como na Figura 16, o que acarreta degradação do aprendizado adquirido. Esse comportamento evidencia maior sensibilidade à alteração da arquitetura.

Já as funções polinomiais POL-Dh e POL-Yh, por sua vez, mantêm curvas mais suaves e estáveis. Destaca-se que a função POL-Dh apresenta, de forma consistente, um comportamento em formato de escada decrescente nos pontos de inserção de novas camadas, alcançando os menores valores finais de RMSE, tanto no treinamento, quanto na validação. Assim, a comparação evidencia que as funções polinomiais, especialmente POL-Dh, proporcionam desempenho mais estável e menor erro ao longo do processo incremental de aprofundamento da rede.

Além disso, por meio de experimentação, verificou-se que a adição de camadas acima da quarta camada oculta não proporcionou melhorias significativas no desempenho do modelo.

Esse comportamento pode ser explicado, principalmente, pela pequena magnitude da taxa de aprendizado após a quarta camada oculta. Como um polinômio de grau mais elevado é empregado a cada inserção de nova camada, valores elevados de η podem provocar instabilidade no processo de aprendizagem, comprometendo a convergência da rede.

De forma complementar, as Figuras 20 a 27 apresentam a frequência relativa de ocorrência do menor RMSE entre as três FAs avaliadas (POL-Dh, POL-Yh e SIG). Em cada experimento, a técnica que obteve o menor RMSE recebeu um ponto; dessa forma, a soma percentual entre as três técnicas totaliza 100% em cada intervalo de épocas analisado.

Adicionalmente, as Tabelas 3 a 10 apresentam as principais métricas utilizadas na avaliação das técnicas, a saber: acurácia (ACU), sensibilidade (SEN), especificidade (ESP), precisão (PRE) e F1-Score (FSC). Para cada métrica, são reportadas a média aritmética (Med), o desvio padrão (DP), o maior valor obtido (Máx) e o menor valor (Mín) observado dentre os 50 experimentos realizados para cada configuração de neurônios ocultos e para cada arquitetura de FA. Considerando-se as três funções avaliadas e as diferentes configurações estruturais, totalizam-se 600 experimentos por base de dados.

Tabela 3 – Métricas para 3, 10, 15 e 20 neurônios ocultos - base breast-cancer-w.

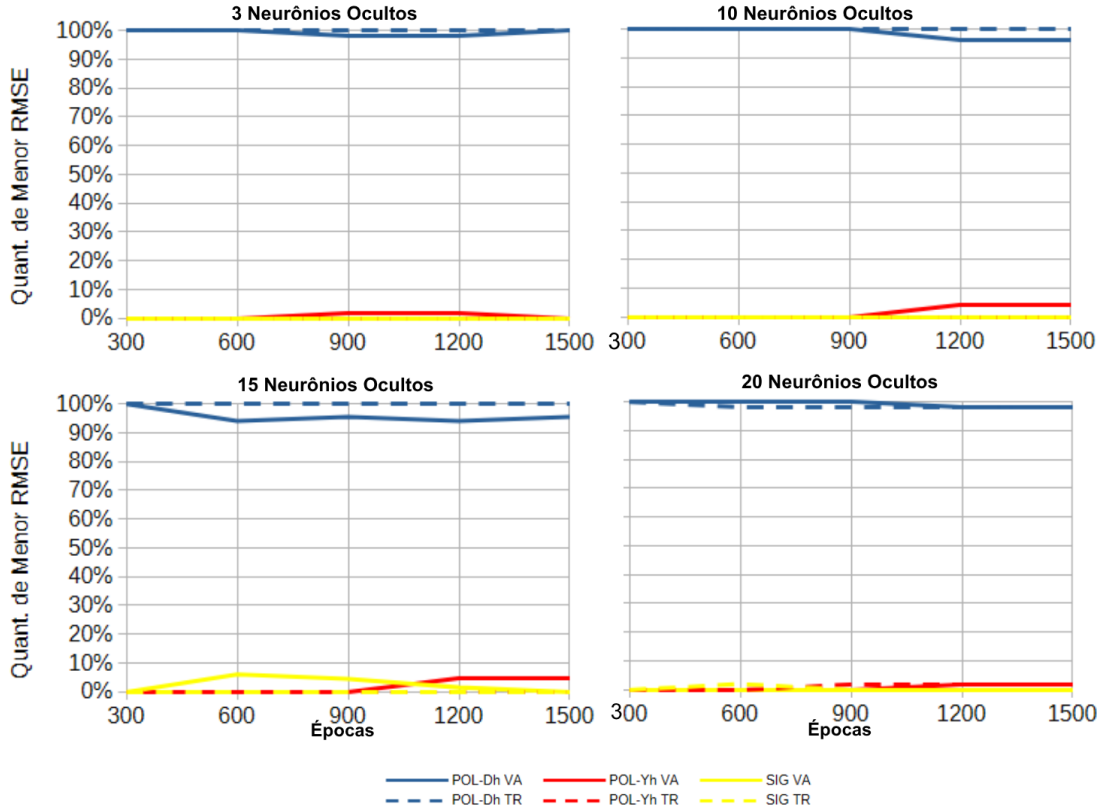
	ACURÁCIA			SENSIBILIDADE			ESPECIFICIDADE			PRECISÃO			F1 SCORE		
	POL-Dh	POL-Yh	SIG	POL-Dh	POL-Yh	SIG	POL-Dh	POL-Yh	SIG	POL-Dh	POL-Yh	SIG	POL-Dh	POL-Yh	SIG
3 Neurônios Ocultos															
Méd	**98,44**	97,91	94,17	**98,81**	97,19	84,6	98,25	98,27	**98,95**	**96,57**	96,55	89,87	**97,67**	96,87	86,11
DP	0,27	0,29	9,57	0,82	0,87	29,53	0	0,12	0,59	0,03	0,24	26,79	0,42	0,45	28,51
Máx	**98,83**	98,25	**98,83**	**100**	98,25	98,25	98,25	99,12	**100**	96,61	98,21	**100**	**98,28**	97,39	98,25
Mín	**98,25**	97,66	66,67	**98,25**	96,49	0	**98,25**	**98,25**	**98,25**	**96,55**	96,49	0	**97,39**	96,49	0
10 Neurônios Ocultos															
Méd	**98,49**	97,98	98,05	**99,26**	96,67	96,28	98,11	98,63	**98,93**	96,33	97,25	**97,83**	**97,77**	96,96	97,05
DP	0,37	0,3	0,31	1,01	0,53	0,57	0,33	0,44	0,36	0,6	0,86	0,72	0,56	0,44	0,46
Máx	**98,83**	98,25	98,25	**100**	98,25	96,49	98,25	**99,12**	**99,12**	96,61	**98,21**	**98,21**	**98,28**	97,39	97,35
Mín	97,08	**97,66**	97,08	**96,49**	**96,49**	94,74	97,37	**98,25**	**98,25**	94,83	**96,49**	96,43	95,65	**96,49**	95,58
15 Neurônios Ocultos															
Méd	**98,48**	98,03	97,92	**99,29**	96,52	96,85	98,08	**98,78**	95,84	96,27	97,55	**98,96**	97,76	97,03	**97,89**
DP	0,32	0,3	0,36	0,92	0,78	0,55	0,35	0,43	1,05	0,63	0,84	0,34	0,48	0,46	0,66
Máx	**98,83**	**98,83**	98,25	**100**	**100**	97,39	98,25	**99,12**	98,25	96,61	98,21	**99,12**	**98,28**	**98,28**	98,21
Mín	**97,66**	**97,66**	97,08	**96,49**	94,74	95,5	97,37	**98,25**	92,98	94,92	**96,49**	**98,25**	**96,49**	96,43	96,43
20 Neurônios Ocultos															
Méd	**98,34**	98,02	97,95	**98,8**	96,25	95,63	98,11	98,9	**99,1**	96,31	97,79	**98,16**	**97,54**	97,01	96,88
DP	0,39	0,39	0,4	1,14	1,16	1,13	0,32	0,74	0,12	0,59	1,33	0,25	0,6	0,57	0,62
Máx	**98,83**	**98,83**	98,25	**100**	**100**	96,49	98,25	**99,12**	**99,12**	96,61	**98,25**	98,21	**98,28**	98,25	97,35
Mín	**97,08**	96,49	**97,08**	**96,49**	94,74	92,98	97,37	94,74	**98,25**	94,83	90,48	**96,43**	**95,65**	95	95,5

Fonte: Elaborado pelo Autor (2026)

Para a Figura 20, observa-se que, para 3, 10, 15 e 20 neurônios ocultos, há clara predominância da função POL-Dh, a qual concentra praticamente 100% das ocorrências de menor RMSE em todas as épocas, tanto no conjunto de treinamento (TR) quanto no de validação (VA). A função sigmoide, por sua vez, demonstra comportamento instável. Esse resultado é coerente com as métricas apresentadas na Tabela 3, na qual a sigmoide apresenta um desvio padrão elevado, por exemplo, 9,57 de acurácia e 29,53 para sensibilidade com 3 neurônios ocultos, além de apresentar valores mínimos iguais a zero em algumas métricas. Em contraste, a

POL-Dh combina acurácia média (98,44%) com F1-Score de 97,67%, indicando desempenho não somente consistente, mas também escalável ao número de neurônios.

Figura 20 – Evolução quantitativa de menor RMSE - base breast-cancer-w.



Fonte: Elaborado pelo Autor (2026)

À medida que o número de neurônios ocultos aumenta para 15 e 20, o cenário torna-se mais equilibrado tanto nos gráficos quanto nas métricas médias, com diferenças reduzidas entre as FAs. Ainda assim, a predominância da POL-Dh permanece evidente sob o critério de frequência do menor RMSE, embora a sigmoide passe a apresentar maior estabilidade, refletida na redução significativa do desvio padrão.

Desse modo, mesmo com médias mais próximas entre as técnicas, a POL-Dh mantém maior sensibilidade média (98,8%) e maior frequência de ocorrência do menor RMSE com 20 neurônios ocultos. Esses resultados indicam que, embora o aumento da complexidade reduza as discrepâncias médias entre as abordagens, o critério baseado na consistência do menor erro evidencia vantagem sistemática da função POL-Dh na maioria dos experimentos realizados.

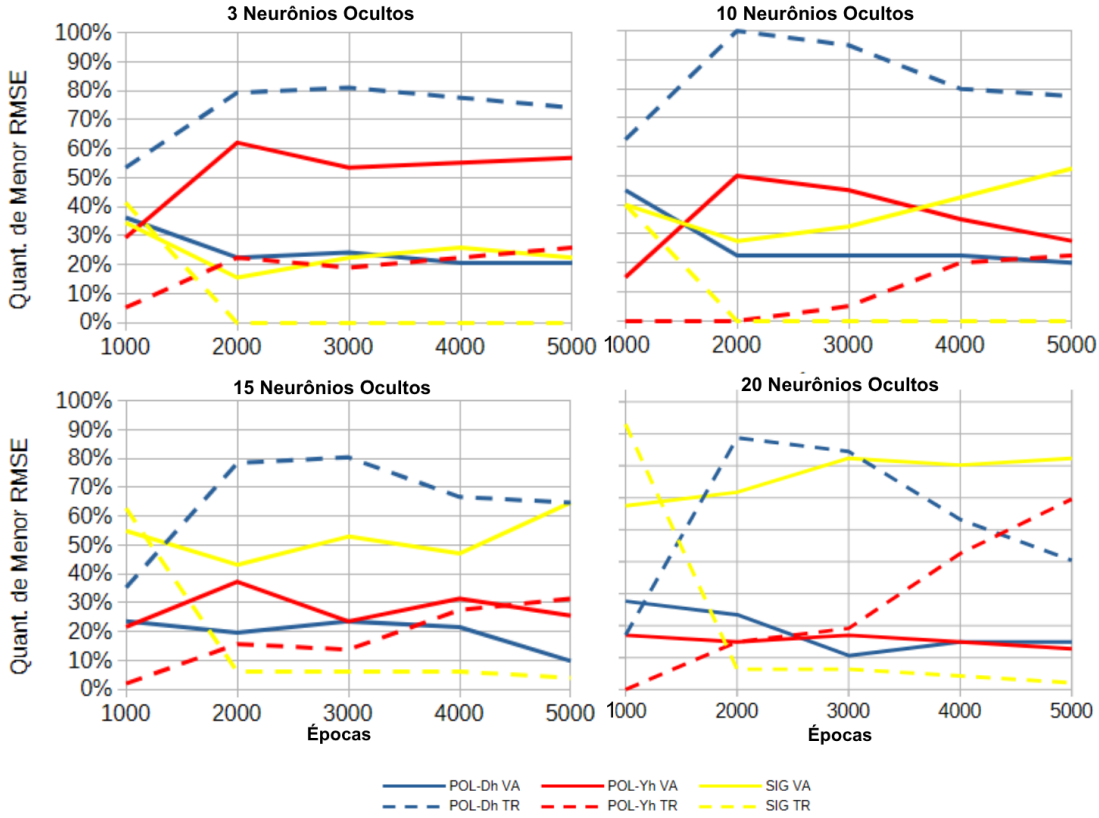
Já a Figura 21, referente à base bioresponse, evidencia um comportamento mais dinâmico e competitivo entre as técnicas, especialmente com o aumento do número de neurônios ocultos. A função POL-Dh apresenta predominância no conjunto de treinamento, sobretudo entre 2000 e 5000 épocas, com percentuais próximos ou superiores a 75% de ocorrência do menor RMSE com 3, 10 e 15 neurônios ocultos.

Tabela 4 – Métricas para 3, 10, 15 e 20 neurônios ocultos - base bioresponse

	ACURÁCIA			SENSIBILIDADE			ESPECIFICIDADE			PRECISÃO			F1 SCORE		
	POL-Dh	POL-Yh	SIG	POL-Dh	POL-Yh	SIG	POL-Dh	POL-Yh	SIG	POL-Dh	POL-Yh	SIG	POL-Dh	POL-Yh	SIG
3 Neurônios Ocultos															
Méd	74,47	**74,57**	72,01	82,52	82,55	**82,99**	65,36	**65,55**	59,59	72,94	**73,06**	70,03	77,42	**77,50**	75,89
DP	1,17	1,00	1,64	1,87	2,05	3,20	2,02	2,06	5,66	1,11	1,02	2,22	1,09	1,00	1,12
Máx	**76,93**	76,80	75,20	86,43	86,93	**92,21**	70,74	**71,59**	68,47	75,76	**76,13**	73,77	**79,67**	79,58	78,18
Mín	72,00	**72,80**	66,40	**77,89**	76,38	72,11	59,94	**61,08**	42,61	70,70	**71,21**	63,73	75,09	**75,15**	72,11
10 Neurônios Ocultos															
Méd	**75,14**	75,07	73,20	82,07	**82,16**	81,74	**67,30**	67,05	63,54	**73,96**	73,84	71,73	**77,79**	77,76	76,39
DP	0,78	1,04	0,94	1,42	1,63	1,83	1,95	2,24	2,06	1,00	1,20	0,98	0,69	0,92	0,90
Máx	77,07	**77,20**	74,80	85,18	85,18	**85,43**	**71,88**	**71,88**	67,33	76,43	**76,60**	73,68	**79,76**	**79,76**	78,01
Mín	**74,00**	72,93	70,53	78,64	**78,89**	77,39	63,07	**63,35**	58,24	**72,04**	71,52	68,79	**76,62**	75,81	74,51
15 Neurônios Ocultos															
Méd	**74,98**	74,91	73,31	81,70	81,63	**81,71**	**67,37**	67,31	63,81	**73,91**	73,86	71,90	**77,60**	77,54	76,47
DP	1,02	0,96	1,34	1,94	1,65	2,22	1,88	1,57	3,47	1,00	0,89	1,49	1,02	0,94	1,10
Máx	77,20	**77,60**	76,00	85,43	85,18	**91,71**	**71,02**	70,45	68,18	**76,28**	76,20	74,37	79,91	**80,14**	78,82
Mín	**72,80**	**72,80**	68,80	76,88	**77,64**	76,38	63,64	**64,49**	42,90	72,09	**72,44**	64,49	**75,48**	75,24	73,52
20 Neurônios Ocultos															
Méd	**74,83**	74,80	72,92	81,86	**81,89**	80,25	**66,87**	66,78	64,64	**73,65**	73,60	72,57	**77,53**	77,52	75,05
DP	1,11	0,99	3,96	1,89	1,64	11,91	1,56	1,62	5,68	0,93	0,92	4,25	1,11	0,96	10,93
Máx	**77,60**	76,67	77,20	**85,93**	85,43	85,43	71,31	69,89	**100,00**	76,62	75,51	**100,00**	**79,81**	79,19	79,47
Mín	72,27	**72,67**	47,47	76,38	**76,88**	1,01	**63,35**	63,07	59,66	**71,99**	71,83	69,57	74,60	**74,91**	1,99

Fonte: Elaborado pelo Autor (2026)

Figura 21 – Evolução quantitativa de menor RMSE - base bioresponse.



Fonte: Elaborado pelo Autor (2026)

Entretanto, no conjunto de validação, observa-se maior equilíbrio, com participação expressiva tanto da POL-Yh quanto da sigmoide, especialmente nas épocas iniciais. Em relação às métricas médias de acurácia, embora os valores sejam bastante próximos entre si, a POL-

Dh apresenta desempenho ligeiramente superior. Por sua vez, a sigmoide evidencia resultado marginalmente inferior e maior variabilidade entre os experimentos, refletida em valores de DP mais elevados, o que indica menor estabilidade estatística. De modo geral, as métricas médias permanecem bastante próximas, sugerindo que as diferenças observadas na frequência do menor erro são sutis, porém sistemáticas. Assim, para a base bioresponse, não se verifica dominância tão expressiva quanto a observada na Figura 20; contudo, a POL-Dh demonstra maior consistência no treinamento, ao passo que a etapa de validação revela um cenário mais competitivo e equilibrado entre as três FAs.

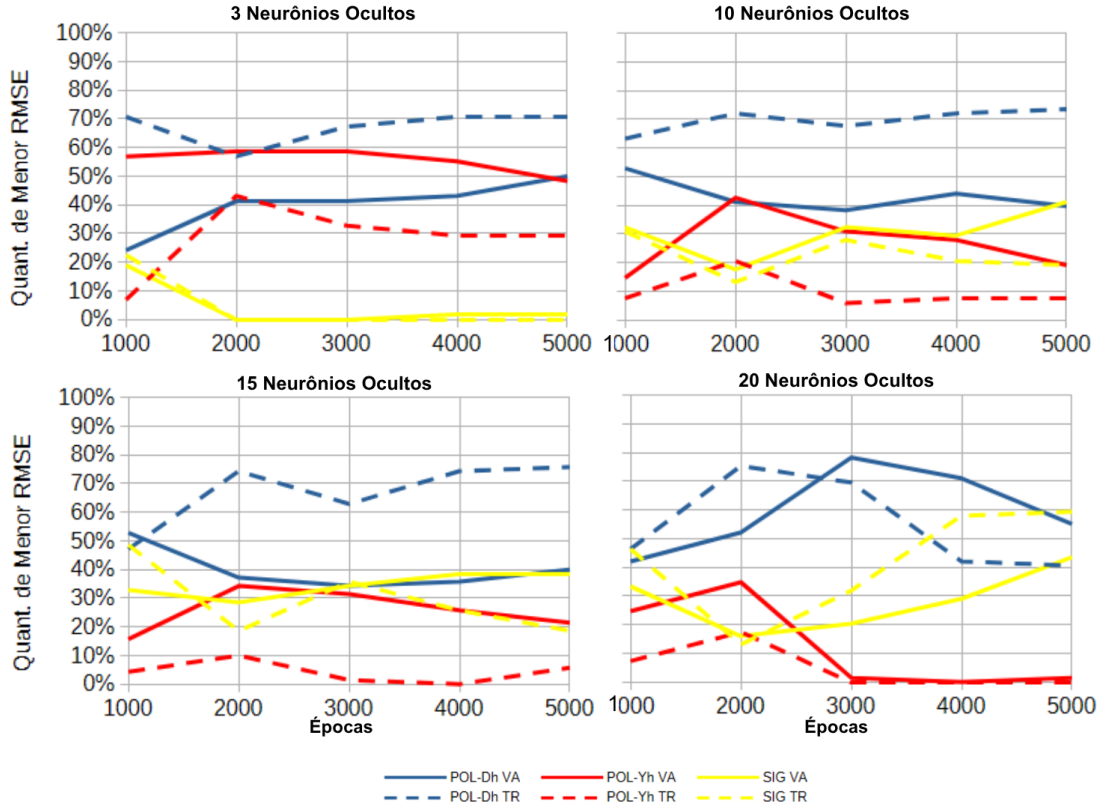
Tabela 5 – Métricas para 3, 10, 15 e 20 neurônios ocultos - base DNA.

	ACURÁCIA			SENSIBILIDADE			ESPECIFICIDADE			PRECISÃO			F1 SCORE		
	POL-Dh	POL-Yh	SIG	POL-Dh	POL-Yh	SIG	POL-Dh	POL-Yh	SIG	POL-Dh	POL-Yh	SIG	POL-Dh	POL-Yh	SIG
3 Neurônios Ocultos															
Méd	79,51	**79,56**	77,17	66,93	67,22	**72,18**	86,46	**87,69**	77,84	75,33	**79,30**	67,48	70,51	**72,51**	69,14
DP	1,21	1,18	2,27	8,61	8,01	8,25	3,53	2,50	11,95	3,64	2,50	6,54	4,62	4,87	3,87
Máx	**83,54**	82,73	80,40	87,27	87,04	**99,07**	**92,90**	92,20	88,11	83,33	**85,45**	76,34	81,34	**86,24**	79,22
Mín	76,38	**76,77**	66,96	53,15	**55,21**	55,14	79,67	**81,43**	20,86	65,69	**73,40**	41,73	61,54	**64,63**	58,73
10 Neurônios Ocultos															
Méd	**80,10**	80,02	78,57	75,19	75,36	**75,48**	86,62	**86,85**	81,96	80,47	**80,81**	75,21	77,55	**77,82**	75,29
DP	0,96	0,87	1,09	6,90	6,77	3,85	2,77	2,71	3,02	2,70	2,74	3,34	4,06	4,12	2,88
Máx	**83,36**	83,05	80,85	**85,58**	84,47	83,00	**92,75**	92,20	89,05	85,85	**86,27**	83,33	**84,65**	**84,26**	80,00
Mín	**78,02**	**78,02**	75,51	58,95	59,57	**66,02**	**80,14**	79,43	75,89	**75,00**	73,87	67,00	**67,07**	**67,07**	66,67
15 Neurônios Ocultos															
Méd	**79,98**	79,97	77,55	**77,07**	76,91	76,45	86,24	**86,27**	79,54	80,71	**80,75**	73,48	**78,70**	78,63	74,59
DP	0,98	1,05	3,77	6,01	6,32	6,72	2,67	2,64	4,64	2,72	2,83	4,64	3,58	3,84	3,88
Máx	82,57	**82,89**	81,16	**86,67**	86,11	87,13	92,86	92,91	**98,59**	86,25	86,84	**96,23**	85,45	**85,84**	83,81
Mín	**78,18**	77,71	48,04	**59,80**	58,16	35,66	**78,57**	**78,57**	65,25	**73,91**	73,45	60,48	**69,51**	**69,09**	52,04
20 Neurônios Ocultos															
Méd	**82,32**	79,99	81,15	**80,30**	79,47	79,09	**89,84**	86,35	86,63	**85,82**	81,55	81,58	**82,91**	80,46	80,27
DP	1,89	0,98	1,51	4,55	3,32	4,58	1,81	2,25	1,75	2,47	2,63	2,18	3,12	2,58	3,08
Máx	**89,01**	82,57	84,93	**91,89**	85,00	89,09	**95,00**	92,86	92,14	**92,39**	85,92	88,04	**90,27**	84,11	**86,73**
Mín	**79,75**	77,86	78,65	64,76	**69,57**	65,05	**85,11**	81,56	81,56	**79,35**	74,51	75,00	**71,96**	**73,08**	71,28

Fonte: Elaborado pelo Autor (2026)

Para a base DNA, inicialmente se observa predominância da POL-Dh no treinamento em todas as configurações, especialmente após 3000 épocas com 3 neurônios ocultos. Na validação, verifica-se maior equilíbrio entre POL-Dh e POL-Yh, com participação reduzida da sigmoide nas épocas intermediárias. Com 3 neurônios, POL-Dh e POL-Yh apresentam acurácias médias muito próximas (79,51% e 79,56%), enquanto a sigmoide atinge maior sensibilidade (72,18%), porém com elevada variabilidade em especificidade (DP = 11,95), indicando menor estabilidade. À medida que o número de neurônios aumenta para 10 e 15, o cenário torna-se mais equilibrado entre as técnicas, embora a POL-Dh mantenha leve vantagem em F1-Score médio (78,70%) e a sigmoide apresente maior variabilidade. Por fim, com 20 neurônios, a POL-Dh passa a se destacar também nas métricas médias, alcançando maior acurácia (82,32%) e F1-Score (82,91%), consolidando vantagem tanto na frequência do menor RMSE, quanto no desempenho médio.

Figura 22 – Evolução quantitativa de menor RMSE - base DNA.



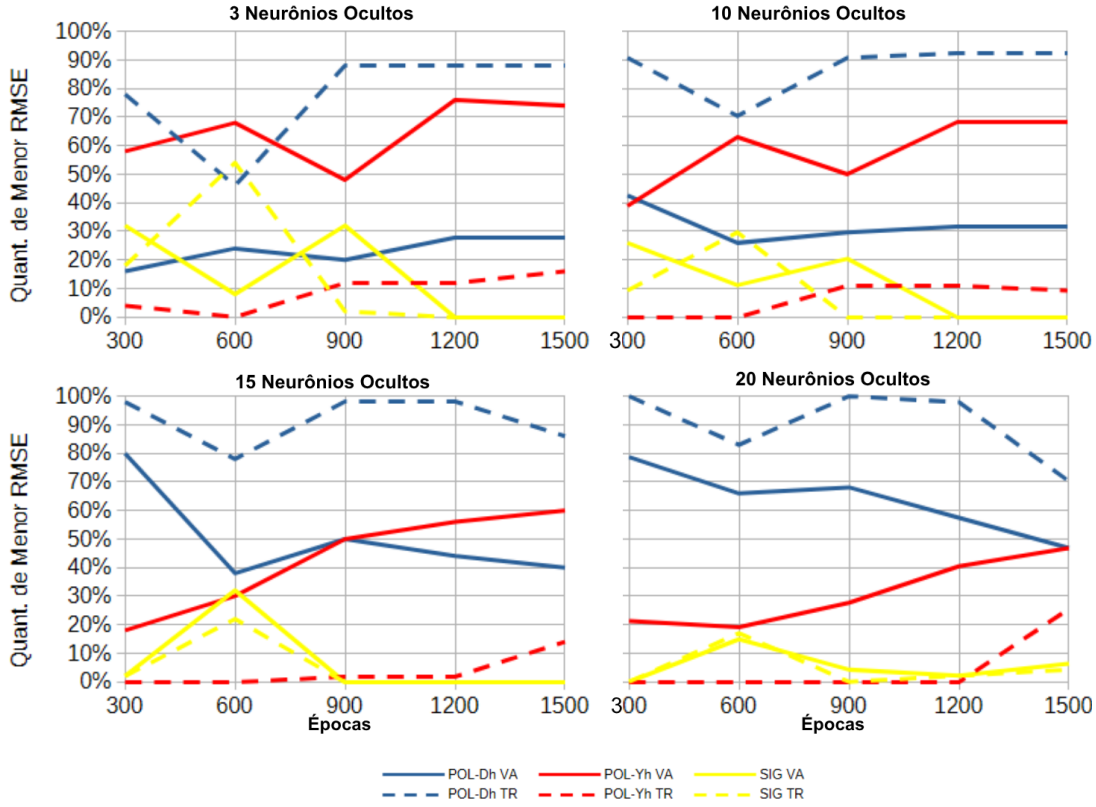
Fonte: Elaborado pelo Autor (2026)

Tabela 6 – Métricas para 3, 10, 15 e 20 neurônios ocultos - base credit-g.

	ACURÁCIA			SENSIBILIDADE			ESPECIFICIDADE			PRECISÃO			F1 SCORE		
	POL-Dh	POL-Yh	SIG	POL-Dh	POL-Yh	SIG	POL-Dh	POL-Yh	SIG	POL-Dh	POL-Yh	SIG	POL-Dh	POL-Yh	SIG
3 Neurônios Ocultos															
Méd	**69,96**	69,91	68,00	89,24	88,47	**100,00**	29,00	**30,47**	0,00	72,80	**73,02**	68,00	80,15	79,99	**80,95**
DP	1,09	1,11	0,00	2,46	1,37	0,00	5,89	3,81	0,00	1,23	0,95	0,00	0,76	0,71	0,00
Máx	**72,00**	71,50	68,00	**96,32**	91,91	**100,00**	**46,88**	37,50	0,00	**76,87**	74,84	68,00	**81,94**	81,06	80,95
Mín	**68,00**	67,50	**68,00**	83,09	85,29	**100,00**	9,38	**17,19**	0,00	69,31	**70,06**	**68,00**	78,67	78,38	**80,95**
10 Neurônios Ocultos															
Méd	**69,70**	69,63	68,00	88,65	88,88	**100,00**	**29,43**	28,72	0,00	**72,84**	72,68	68,00	79,89	79,91	**80,95**
DP	1,39	1,31	0,00	3,67	3,33	0,00	8,01	7,89	0,00	1,72	1,58	0,00	1,15	0,99	0,00
Máx	**72,00**	**72,00**	68,00	95,59	**100,00**	**100,00**	**60,94**	56,25	0,00	**79,34**	78,12	68,00	**81,70**	81,58	80,95
Mín	65,50	66,00	**68,00**	70,59	73,53	**100,00**	**14,06**	1,56	0,00	**69,27**	68,34	**68,00**	74,71	75,76	**80,95**
15 Neurônios Ocultos															
Méd	68,80	**68,89**	68,00	89,37	95,00	**100,00**	**25,09**	13,41	0,00	**71,96**	70,08	68,00	79,49	80,58	**80,95**
DP	2,22	1,53	0,00	6,89	4,12	0,00	14,25	10,08	0,00	2,72	1,74	0,00	2,12	1,03	0,00
Máx	**74,50**	72,50	68,00	**100,00**	**100,00**	**100,00**	**60,94**	32,81	0,00	**79,34**	73,94	68,00	**83,01**	82,61	80,95
Mín	63,50	64,50	**68,00**	66,91	86,03	**100,00**	**0,00**	**0,00**	**0,00**	**68,00**	67,57	**68,00**	71,37	77,88	**80,95**
20 Neurônios Ocultos															
Méd	67,72	**68,33**	68,00	95,06	98,28	**100,00**	**9,64**	4,69	0,00	**69,16**	68,71	68,00	79,97	80,83	**80,95**
DP	2,23	1,07	0,00	5,50	3,25	0,00	9,57	7,47	0,00	1,61	1,18	0,00	1,94	0,82	0,00
Máx	**72,50**	71,50	68,00	**100,00**	**100,00**	**100,00**	**39,06**	25,00	0,00	**74,25**	72,57	68,00	82,21	**82,67**	80,95
Mín	58,50	64,50	**68,00**	76,47	83,82	**100,00**	**0,00**	**0,00**	**0,00**	66,88	**67,68**	**68,00**	71,67	76,25	**80,95**

Fonte: Elaborado pelo Autor (2026)

Figura 23 – Evolução quantitativa de menor RMSE - base credit-g.



Fonte: Elaborado pelo Autor (2026)

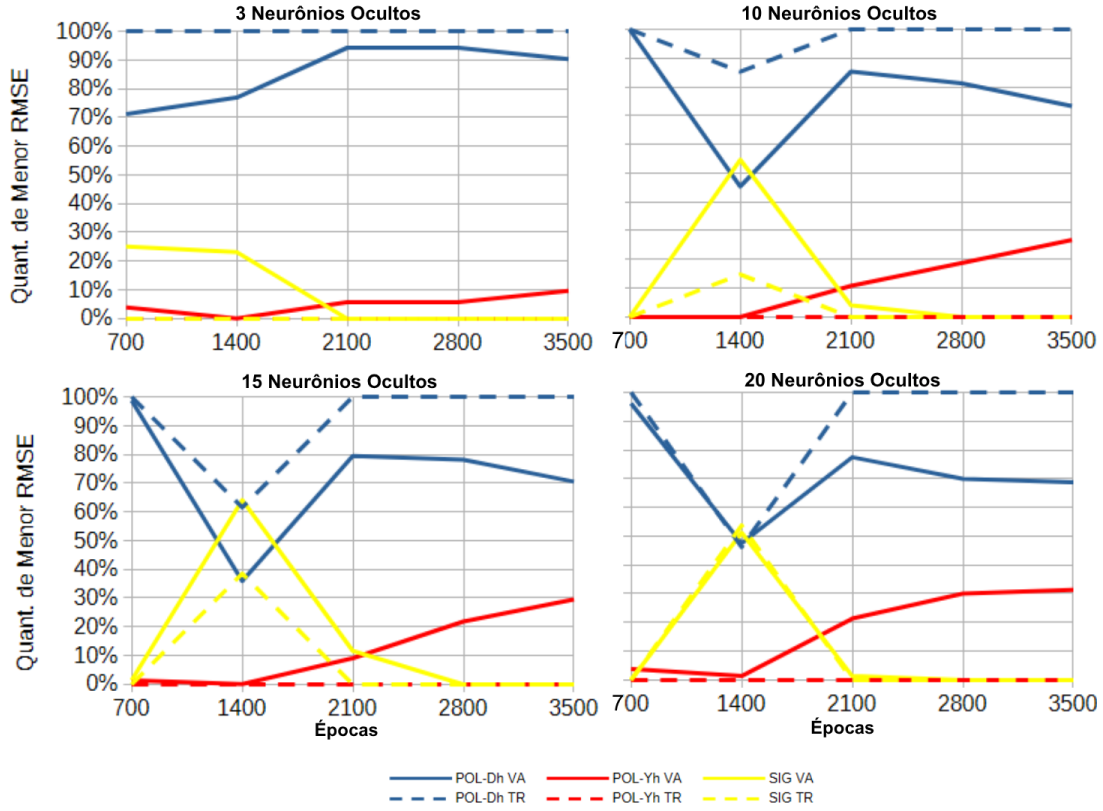
No caso da base credit-g, de modo consistente, as funções polinomiais superam a sigmoide na maioria das configurações. Com 3 e 10 neurônios ocultos, essas funções apresentam maior acurácia média, melhor equilíbrio entre sensibilidade e especificidade (29%) e menores desvios-padrão. Em contrapartida, a sigmoide, embora atinja 100% de sensibilidade, apresenta especificidade igual a 0%, caracterizando incapacidade de identificar a classe negativa. Ademais, o F1-Score mostra-se mais consistente nas abordagens polinomiais. Quando se aumentam os neurônios para 15 e 20, não se observam ganhos proporcionais, havendo aumento da variabilidade, o que sugere sobreajuste. A análise da frequência do menor RMSE confirma a predominância da POL-Dh, sobretudo nas configurações menos complexas.

Tabela 7 – Métricas para 3, 10, 15 e 20 neurônios ocultos - base pc4.

	ACURÁCIA			SENSIBILIDADE			ESPECIFICIDADE			PRECISÃO			F1 SCORE		
	POL-Dh	POL-Yh	SIG	POL-Dh	POL-Yh	SIG	POL-Dh	POL-Yh	SIG	POL-Dh	POL-Yh	SIG	POL-Dh	POL-Yh	SIG
3 Neurônios Ocultos															
Méd	89,79	**90,19**	89,38	**20,22**	8,50	0,00	98,05	99,89	**100,00**	56,43	**88,11**	0,00	**29,44**	15,25	0,00
DP	0,43	0,48	0,00	3,79	4,33	0,00	0,62	0,17	0,00	8,61	23,12	0,00	4,00	7,28	0,00
Máx	90,41	**91,10**	89,38	**35,48**	16,13	0,00	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	0,00	**42,31**	27,78	0,00
Mín	**89,04**	**89,04**	**89,38**	**9,68**	0,00	0,00	96,17	99,62	**100,00**	**45,45**	0,00	0,00	**17,65**	0,00	0,00
10 Neurônios Ocultos															
Méd	88,99	**90,72**	89,38	**24,13**	14,62	0,04	96,69	99,76	**100,00**	52,64	**89,19**	0,04	**31,19**	24,94	0,08
DP	1,80	0,35	0,04	8,75	2,97	0,37	2,81	0,26	0,00	14,44	11,27	0,37	4,96	4,57	0,72
Máx	**91,44**	91,10	89,73	**61,29**	19,35	3,23	99,62	**100,00**	**100,00**	**87,50**	**100,00**	3,23	**46,58**	31,58	6,25
Mín	83,90	**89,73**	89,38	**16,13**	3,23	0,00	86,59	99,23	**100,00**	26,47	**66,67**	0,00	**21,74**	6,25	0,00
15 Neurônios Ocultos															
Méd	89,26	**90,99**	89,44	**24,90**	16,67	0,58	96,90	99,81	**100,00**	55,06	**92,38**	16,67	**32,41**	28,15	1,12
DP	1,60	0,29	0,15	8,17	2,04	1,35	2,57	0,24	0,00	14,19	9,31	37,51	4,70	3,00	2,59
Máx	91,10	**91,44**	90,07	**54,84**	19,35	6,45	99,62	**100,00**	**100,00**	85,71	**100,00**	**100,00**	**47,62**	32,43	12,12
Mín	83,22	**90,41**	89,38	**12,90**	9,68	0,00	86,59	99,23	**100,00**	30,00	**71,43**	0,00	**21,05**	17,65	0,00
20 Neurônios Ocultos															
Méd	89,24	**91,16**	89,54	**28,10**	18,42	1,45	96,50	99,79	**100,00**	56,58	**92,26**	32,50	**34,79**	30,65	2,77
DP	2,05	0,28	0,25	10,92	1,55	2,29	3,40	0,25	0,00	14,28	9,14	47,13	5,54	2,30	4,33
Máx	91,10	**91,44**	90,07	**54,84**	19,35	6,45	99,62	**100,00**	**100,00**	83,33	**100,00**	**100,00**	**47,27**	32,43	12,12
Mín	81,16	**90,41**	89,38	**16,13**	12,90	0,00	84,29	99,23	**100,00**	29,31	**71,43**	0,00	**24,39**	22,22	0,00

Fonte: Elaborado pelo Autor (2026)

Figura 24 – Evolução quantitativa de menor RMSE - base pc4.



Fonte: Elaborado pelo Autor (2026)

Por sua vez, na base pc4, caracterizada por forte desbalanceamento, cerca de 87,8% das instâncias concentradas em uma única classe, observa-se impacto direto na convergência das redes. Esse cenário favorece configurações que tendem a predizer apenas a classe majoritária, o

que se reflete em valores de precisão muito baixos. Assim, a acurácia isolada (90%) deixa de ser uma métrica discriminativa adequada.

Nesse contexto, a função sigmoide mantém especificidade próxima de 100%, porém apresenta sensibilidade praticamente nula, evidenciando falha na detecção da classe minoritária. Em contraste, as funções polinomiais exibem ganhos progressivos em sensibilidade, especialmente a POL-Dh, o que se reflete em melhor desempenho no F1-Score, com seus maiores valores observados na configuração com 20 neurônios ocultos.

Enquanto a POL-Yh se destaca em termos de precisão e acurácia média, a POL-Dh demonstra melhor equilíbrio global entre sensibilidade e F1-Score, sendo mais adequada para cenários desbalanceados. Por fim, os gráficos de menor RMSE corroboram esses resultados, indicando predominância das funções polinomiais ao longo das épocas e maior estabilidade de generalização nesse tipo de problema.

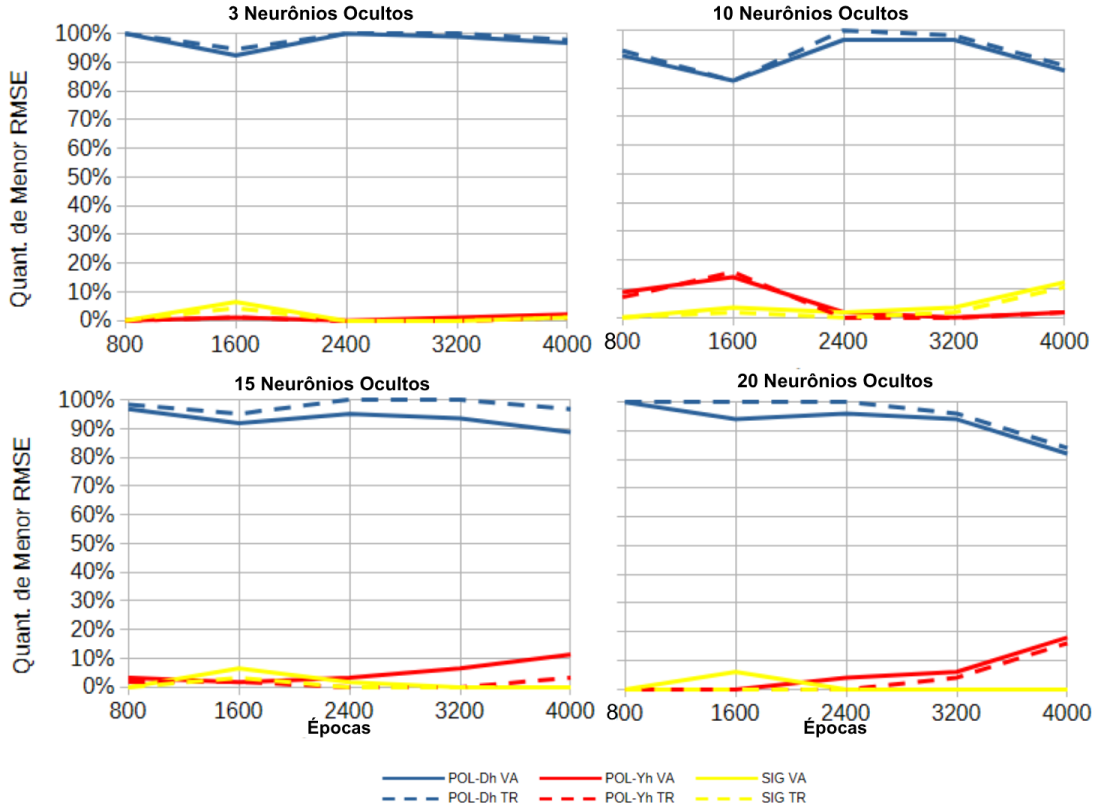
Tabela 8 – Métricas para 3, 10, 15 e 20 neurônios ocultos - base mushrooms.

	ACURÁCIA			SENSIBILIDADE			ESPECIFICIDADE			PRECISÃO			F1 SCORE		
	POL-Dh	POL-Yh	SIG	POL-Dh	POL-Yh	SIG	POL-Dh	POL-Yh	SIG	POL-Dh	POL-Yh	SIG	POL-Dh	POL-Yh	SIG
3 Neurônios Ocultos															
Méd	**95,96**	88,07	93,37	**96,60**	92,72	91,76	**95,36**	83,69	94,89	**95,30**	88,22	94,41	**95,88**	89,20	93,06
DP	1,51	12,84	0,66	2,25	8,41	1,32	3,50	26,82	0,36	3,17	14,07	0,38	1,45	10,08	0,73
Máx	**98,33**	95,91	95,42	**100,00**	**100,00**	95,94	**99,81**	98,47	95,79	**99,79**	98,29	95,41	**98,29**	95,67	95,29
Mín	**88,82**	48,50	91,58	**91,78**	19,59	88,32	**78,30**	0,00	93,98	**81,27**	48,50	93,75	**89,67**	32,27	91,05
10 Neurônios Ocultos															
Méd	**96,77**	89,60	93,58	**96,95**	93,53	91,87	**96,59**	85,89	95,18	**96,42**	89,67	94,72	**96,67**	90,76	93,27
DP	0,95	11,99	0,63	1,85	3,39	1,22	1,34	25,44	0,38	1,35	13,10	0,40	0,99	7,68	0,69
Máx	**98,67**	96,85	95,67	99,90	**100,00**	96,55	**99,42**	97,99	95,79	**99,37**	97,70	95,43	**98,65**	96,77	95,58
Mín	**94,88**	48,55	92,42	**92,99**	89,85	90,05	**93,59**	0,10	94,65	**93,62**	48,52	94,06	**94,68**	65,34	92,01
15 Neurônios Ocultos															
Méd	**97,09**	88,44	93,28	**97,73**	94,82	92,08	**96,50**	82,44	94,40	**96,37**	87,90	94,22	**97,03**	90,21	93,07
DP	1,25	13,89	2,74	1,67	3,21	1,76	1,97	29,10	6,07	1,91	15,09	3,90	1,26	9,03	2,10
Máx	**99,26**	96,16	96,21	**100,00**	**100,00**	**100,00**	**99,71**	97,51	95,79	**99,68**	97,22	95,58	**99,24**	96,10	96,11
Mín	**93,85**	48,50	72,97	**93,20**	90,46	87,92	**88,15**	0,00	47,51	**88,82**	48,50	64,21	**93,74**	65,32	78,21
20 Neurônios Ocultos															
Méd	**96,79**	77,04	76,47	**97,87**	94,43	56,45	**95,77**	60,67	95,31	**95,67**	76,30	91,34	**96,73**	82,32	84,34
DP	1,24	20,14	21,27	2,04	7,63	42,12	2,22	41,47	11,64	2,10	21,17	18,85	1,25	13,32	24,98
Máx	**99,31**	96,06	94,24	**100,00**	**100,00**	93,40	99,71	96,27	**100,00**	**99,67**	95,66	**100,00**	**99,29**	95,99	99,14
Mín	**93,50**	45,00	8,67	**90,86**	55,13	0,00	**87,67**	0,00	16,25	**88,42**	44,58	0,00	**93,21**	49,30	0,64

Fonte: Elaborado pelo Autor (2026)

Na base mushrooms, a função POL-Dh apresenta desempenho consistentemente superior com 3, 10 e 15 neurônios ocultos, mantendo acurácia acima de 95% e atingindo 97,09% com 15 neurônios, acompanhada de alta sensibilidade (97,73%) e especificidade (96,50%). O F1-Score permanece elevado (97%) e com baixos desvios-padrão, indicando elevada estabilidade. A sigmoide mantém desempenho competitivo até 15 neurônios, porém com médias ligeiramente inferiores. Em contraste, a POL-Yh apresenta maior variabilidade, com desvios-padrão elevados e valores mínimos reduzidos (incluindo especificidade mínima de 0%), sugerindo instabilidade em determinadas inicializações.

Figura 25 – Evolução quantitativa de menor RMSE - base mushrooms.



Fonte: Elaborado pelo Autor (2026)

Com 20 neurônios ocultos, observa-se degradação significativa da sigmoide (sensibilidade de 56,45% e F1-Score de 84,34%) e da POL-Yh, ambas com aumento de variabilidade. A POL-Dh, por sua vez, mantém desempenho robusto (acurácia de 96,79% e F1-Score de 96,73%). A análise da frequência do menor RMSE confirma esses resultados, evidenciando ampla predominância da POL-Dh em treinamento e validação ao longo das épocas, especialmente nas configurações menos complexas, enquanto a sigmoide apresenta participação residual e a POL-Yh não converte maior frequência nas arquiteturas mais profundas em superioridade nas métricas finais.

Para a base kr-vs-kp, observa-se inicialmente desempenho equilibrado entre as três funções com 3 neurônios ocultos, todas com acurácia média próxima de 95,5% e F1-Score superior a 95%, sendo a sigmoide a mais estável nessa configuração. Entretanto, com 10 neurônios, a sigmoide passa a apresentar instabilidade significativa, evidenciada por sensibilidade mínima de 0% e acurácia mínima de 48,04%, elevando substancialmente o desvio padrão. As funções polinomiais mantêm maior regularidade estatística. Com 15 neurônios, a POL-Dh apresenta melhor equilíbrio global (acurácia de 95,42% e F1-Score de 95,64%), enquanto, com 20 neurônios, POL-Dh e POL-Yh alcançam desempenhos praticamente equivalentes e superiores à sigmoide em termos de estabilidade, mantendo F1-Score acima de 95,6%.

A análise da frequência do menor RMSE para kr-vs-kp reforça esses resultados, evidenciando alternância predominante entre POL-Dh e POL-Yh ao longo das épocas, especialmente

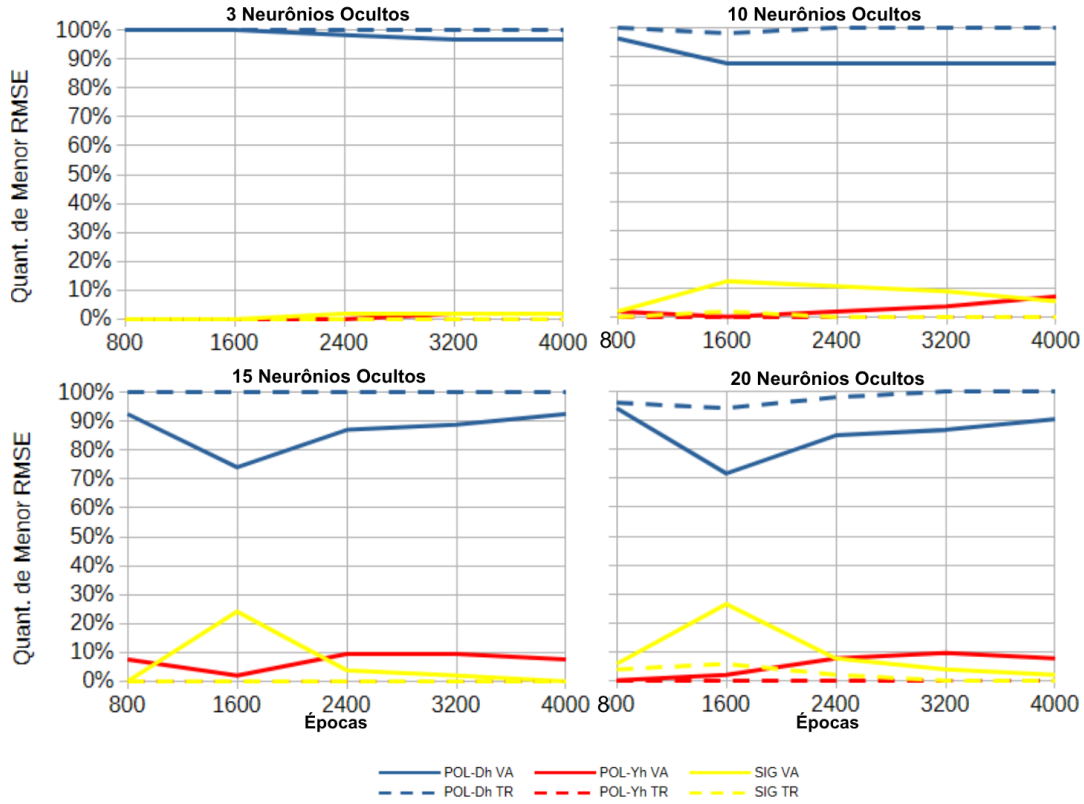
nas configurações com 10 e 20 neurônios. A sigmoide apresenta participação pontual e menos consistente. Ademais, as funções polinomiais demonstram convergência mais estável entre treinamento e validação, com menor oscilação percentual, corroborando a robustez observada nas métricas médias.

Tabela 9 – Métricas para 3, 10, 15 e 20 neurônios ocultos - base kr-vs-kp

	ACURÁCIA			SENSIBILIDADE			ESPECIFICIDADE			PRECISÃO			F1 SCORE		
	POL-Dh	POL-Yh	SIG	POL-Dh	POL-Yh	SIG	POL-Dh	POL-Yh	SIG	POL-Dh	POL-Yh	SIG	POL-Dh	POL-Yh	SIG
3 Neurônios Ocultos															
Méd	**95,53**	95,29	95,51	96,37	96,31	**96,44**	**94,63**	94,19	94,53	**95,23**	94,79	95,02	**95,75**	95,52	95,71
DP	1,41	1,49	0,43	1,34	0,69	0,62	3,82	3,44	0,84	2,96	2,37	0,71	1,19	1,21	0,40
Máx	**96,87**	96,40	96,56	**99,10**	**99,10**	97,89	**99,67**	96,09	96,09	**99,68**	96,40	96,35	**96,90**	96,54	96,71
Mín	89,05	84,35	**94,52**	92,47	**95,18**	94,58	78,18	68,40	**92,51**	83,08	77,23	**93,29**	90,38	86,81	**94,78**
10 Neurônios Ocultos															
Méd	94,01	**94,98**	93,96	**97,05**	96,26	93,84	90,72	93,60	**94,08**	92,20	**94,40**	94,37	94,47	**95,27**	94,72
DP	2,98	2,46	6,93	1,19	0,80	14,28	7,03	5,58	1,57	4,75	3,50	1,81	2,35	1,87	4,72
Máx	**96,71**	96,56	96,24	**99,70**	99,40	97,29	96,74	96,09	**100,00**	**96,98**	96,41	**100,00**	**96,83**	96,71	96,42
Mín	77,46	77,15	48,04	**94,58**	94,28	0,00	**54,07**	53,09	90,23	**70,00**	69,62	**84,02**	**82,04**	81,89	59,62
15 Neurônios Ocultos															
Méd	**95,42**	94,64	94,23	**96,54**	96,17	94,46	**94,21**	92,97	93,99	**94,77**	94,25	94,55	**95,64**	95,08	93,83
DP	0,63	5,18	5,67	0,78	0,82	11,63	1,55	11,35	1,72	1,29	5,50	1,47	0,58	3,37	10,32
Máx	**96,87**	96,09	96,56	98,49	**100,00**	97,59	96,74	96,42	**100,00**	96,94	96,64	**100,00**	**97,00**	96,26	96,67
Mín	**94,21**	57,43	53,68	**94,88**	94,58	10,84	**90,88**	11,40	88,27	**92,05**	54,97	90,00	**94,55**	70,94	19,57
20 Neurônios Ocultos															
Méd	**95,53**	95,45	93,24	**96,42**	96,15	93,45	94,56	**94,69**	93,01	95,05	**95,15**	93,80	**95,73**	95,64	94,29
DP	0,62	0,49	7,17	0,79	0,53	14,19	1,07	0,94	5,97	0,92	0,81	3,70	0,58	0,46	3,92
Máx	**96,71**	**96,71**	96,24	**98,19**	97,29	**100,00**	**96,74**	**96,74**	**100,00**	**96,94**	96,93	**100,00**	**96,85**	96,84	96,43
Mín	**94,52**	94,37	48,04	93,98	**95,18**	0,00	91,21	**92,18**	51,79	92,31	**93,02**	69,17	**94,69**	94,67	69,55

Fonte: Elaborado pelo Autor (2025)

Figura 26 – Evolução quantitativa de menor RMSE - base kr-vs-kp.



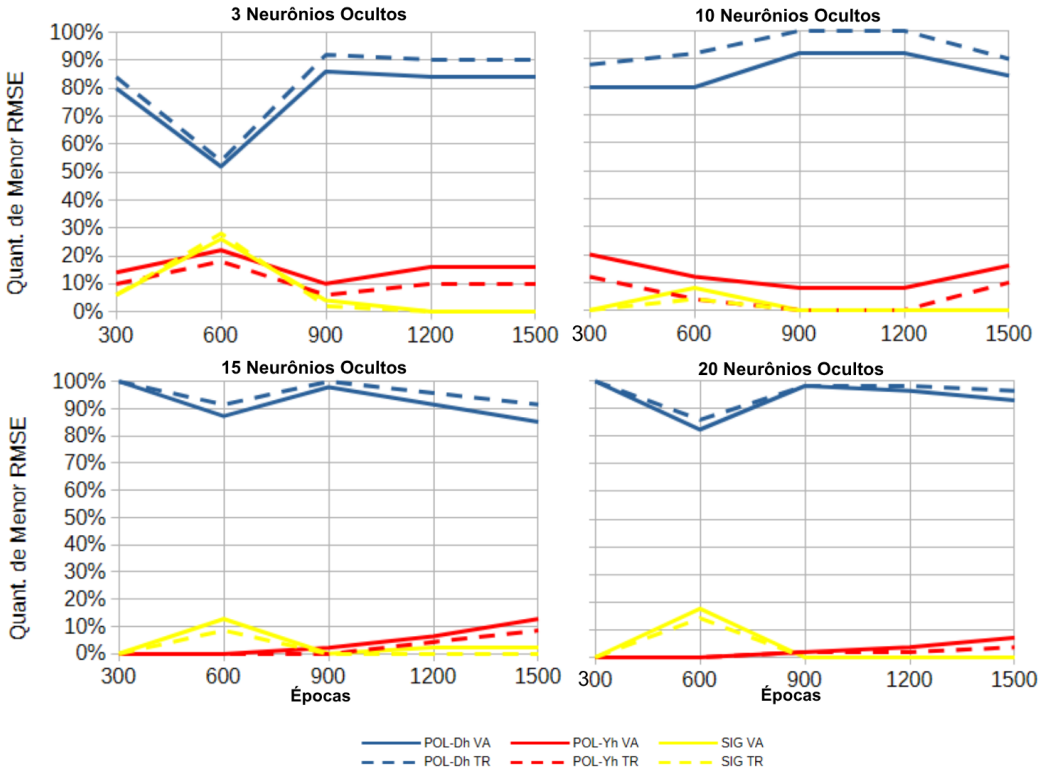
Fonte: Elaborado pelo Autor (2026)

Tabela 10 – Métricas para 3, 10, 15 e 20 neurônios ocultos - base steel-plates-fault.

	ACURÁCIA			SENSIBILIDADE			ESPECIFICIDADE			PRECISÃO			F1 SCORE		
	POL-Dh	POL-Yh	SIG	POL-Dh	POL-Yh	SIG	POL-Dh	POL-Yh	SIG	POL-Dh	POL-Yh	SIG	POL-Dh	POL-Yh	SIG
3 Neurônios Ocultos															
Méd	**72,14**	72,00	66,80	**39,16**	34,96	0,00	88,53	90,40	**100,00**	**65,11**	64,44	0,00	**47,18**	45,27	0,00
DP	1,93	0,83	0,00	12,83	2,60	0,00	7,40	1,03	0,00	6,57	2,35	0,00	8,25	2,42	0,00
Máx	**76,29**	73,81	66,80	**81,99**	42,24	0,00	**99,69**	92,59	**100,00**	**90,91**	70,00	0,00	**60,55**	51,13	0,00
Mín	64,54	**69,69**	66,80	6,21	**25,47**	0,00	55,86	**87,65**	**100,00**	48,00	**60,29**	0,00	11,63	**35,81**	0,00
10 Neurônios Ocultos															
Méd	**70,86**	67,56	66,80	**28,71**	6,69	0,00	91,81	97,81	**100,00**	**65,73**	37,22	0,00	**38,31**	9,79	0,00
DP	2,76	1,41	0,00	11,13	10,53	0,00	5,84	3,92	0,00	9,71	39,82	0,00	10,75	14,55	0,00
Máx	**76,29**	71,55	66,80	**54,66**	32,30	0,00	99,38	**100,00**	**100,00**	93,10	**100,00**	0,00	**56,48**	41,60	0,00
Mín	62,47	64,54	**66,80**	**1,24**	0,00	0,00	69,14	**87,96**	**100,00**	**41,73**	0,00	0,00	**2,42**	0,00	0,00
15 Neurônios Ocultos															
Méd	**67,31**	66,30	66,80	**12,45**	2,26	0,00	94,58	98,12	**100,00**	**49,42**	11,91	0,00	**17,80**	1,59	0,00
DP	1,99	3,93	0,00	12,14	13,67	0,00	6,03	12,65	0,00	18,04	31,34	0,00	14,10	7,58	0,00
Máx	**70,93**	68,66	66,80	50,93	**93,79**	0,00	**100,00**	**100,00**	**100,00**	80,00	**100,00**	0,00	47,81	**50,93**	0,00
Mín	61,86	40,00	**66,80**	**0,00**	**0,00**	**0,00**	**69,14**	13,27	**100,00**	**0,00**	**0,00**	**0,00**	**0,00**	**0,00**	**0,00**
20 Neurônios Ocultos															
Méd	**67,02**	66,75	66,80	**8,07**	0,48	0,00	96,31	99,69	**100,00**	**46,43**	4,59	0,00	**12,26**	0,78	0,00
DP	1,63	0,55	0,00	10,05	2,11	0,00	3,89	1,72	0,00	20,76	18,59	0,00	12,36	3,24	0,00
Máx	**71,55**	68,25	66,80	**47,20**	14,29	0,00	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	0,00	**52,41**	20,54	0,00
Mín	**64,95**	63,30	66,80	**0,00**	**0,00**	**0,00**	79,63	**87,65**	**100,00**	**0,00**	**0,00**	**0,00**	**0,00**	**0,00**	**0,00**

Fonte: Elaborado pelo Autor (2026)

Figura 27 – Evolução quantitativa de menor RMSE - base steel-plates-fault.



Fonte: Elaborado pelo Autor (2026)

Em contraste, na base steel-plates-fault, a sigmoide apresenta colapso completo em todas as configurações, com sensibilidade, precisão e F1-Score médios iguais a 0% e especificidade de 100%, caracterizando classificação exclusiva da classe majoritária. As funções polinomiais superam esse comportamento degenerado, atingindo, com 3 neurônios, acurácia de 72,14% e FSC

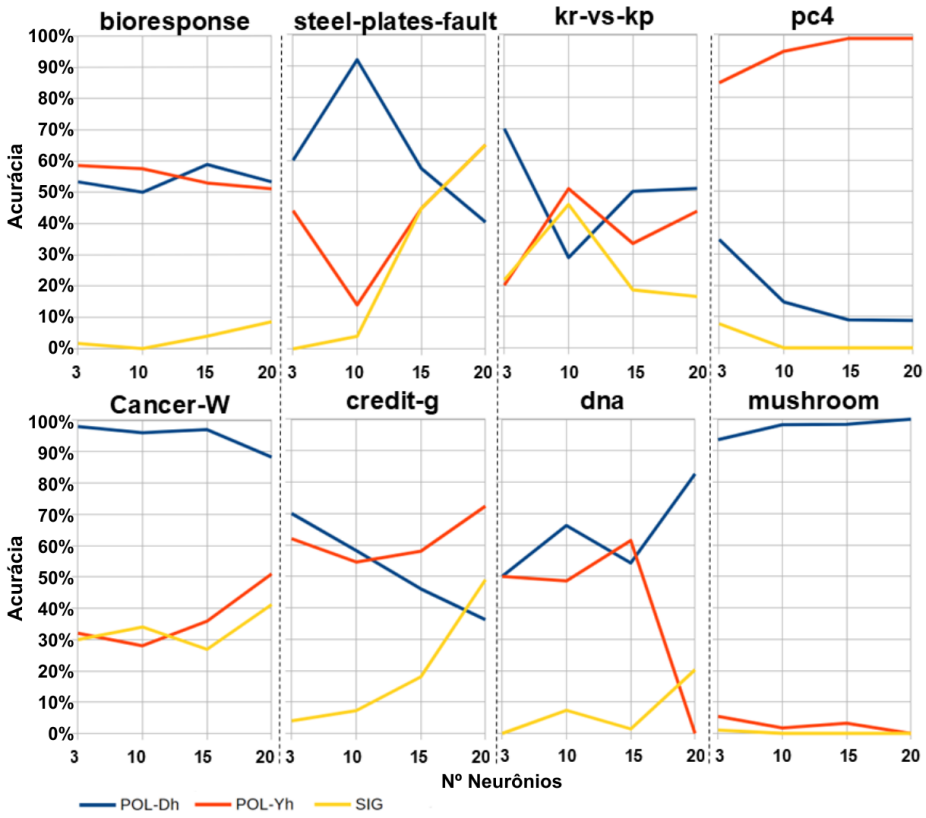
de 47,18% (POL-Dh), porém exibem degradação progressiva com o aumento da complexidade. Com 20 neurônios, a POL-Dh reduz a sensibilidade para 8,07% e o F1-Score para 12,26%, enquanto a POL-Yh apresenta desempenho ainda inferior. Os gráficos de menor RMSE indicam predominância inicial da POL-Dh nas configurações menos complexas, mas com perda de estabilidade e maior dispersão entre treinamento e validação à medida que a rede se torna mais profunda, evidenciando limitações de generalização em cenário mais desafiador e desbalanceado.

Adicionalmente, para comparar as três variantes de FAs, os valores, expressos em porcentagem, que representam o desempenho máximo alcançado por cada método são apresentados nas Figuras 28 a 32. As métricas analisadas incluem: acurácia (Fig. 28), sensibilidade (Fig. 29), especificidade (Fig. 30), precisão (Fig. 31) e F1-Score (Fig. 32). O objetivo dessa análise é identificar qual modelo se destaca em termos de desempenho para cada uma dessas métricas.

Para cada experimento, o método que alcança o melhor desempenho em uma métrica específica recebe 1 ponto, sendo posteriormente o resultado normalizado para percentual relativo ao total de experimentos. Nos casos em que dois ou mais métodos apresentam o mesmo valor máximo para determinada métrica, todos receberam pontuação equivalente.

Cabe destacar que, embora a fórmula de cálculo da métrica não realize a soma direta dos valores de desempenho, ela apresenta o melhor desempenho individual de cada modelo em relação a cada métrica. Assim, a pontuação total pode ultrapassar 100%, especialmente em situações de empate entre os modelos.

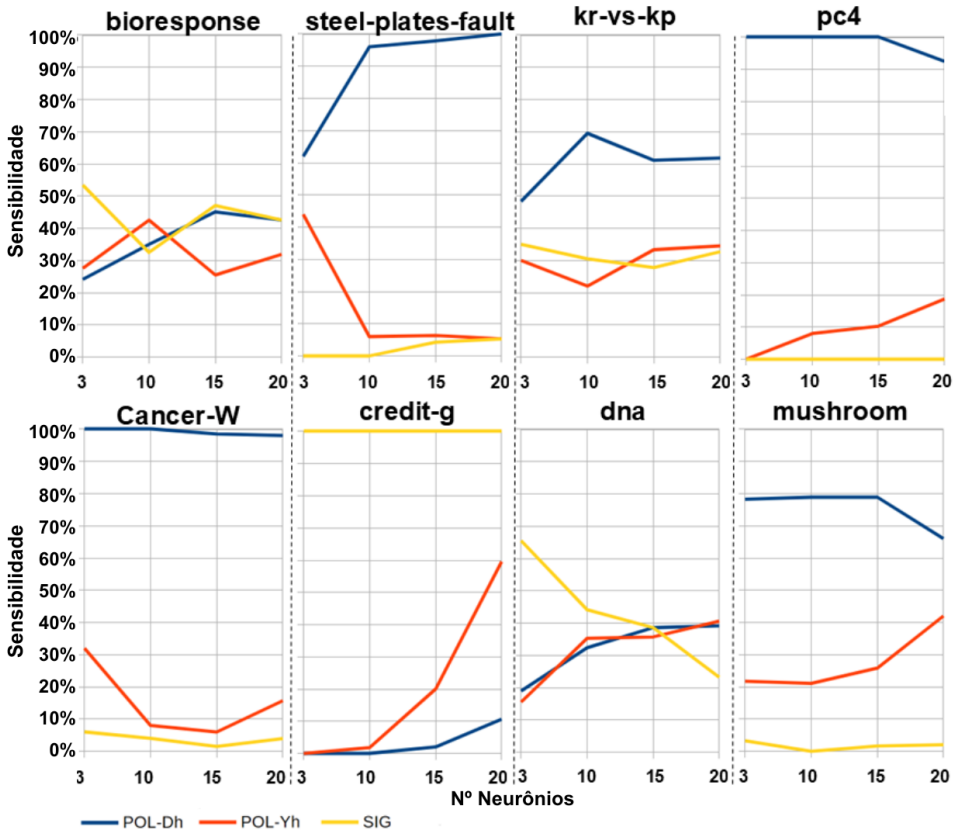
Figura 28 – Acurácia para diferentes conjuntos de dados e configurações de neurônios.



Fonte: Elaborado pelo Autor (2025)

O método que utiliza regressão polinomial POL-Dh nas FAs supera consistentemente as demais abordagens, demonstrando desempenho superior na maioria dos conjuntos de dados analisados. No gráfico, a linha azul, representando o POL-Dh, apresenta resultados mais elevados, particularmente nas bases bioresponse, Cancer-W e mushroom, além de superar os demais modelos na maioria dos conjuntos com menor número de neurônios na camada oculta. Em comparação, o método baseado na regressão polinomial Y_h (POL-Yh, linha vermelha) apresenta variações significativas de desempenho dependendo do conjunto de dados, superando os demais métodos em alguns casos, como na base PC4. Por outro lado, o modelo que utiliza a função de ativação sigmoide (SIG, linha amarela) tende a apresentar desempenho inferior na maioria das bases, embora obtenha resultados satisfatórios na base steel-plates-fault com 20 neurônios ocultos. Isso sugere que o modelo apresenta limitações quando aplicado a conjuntos de dados com características significativamente distintas entre si.

Figura 29 – Sensibilidade para diferentes conjuntos de dados e configurações de neurônios.



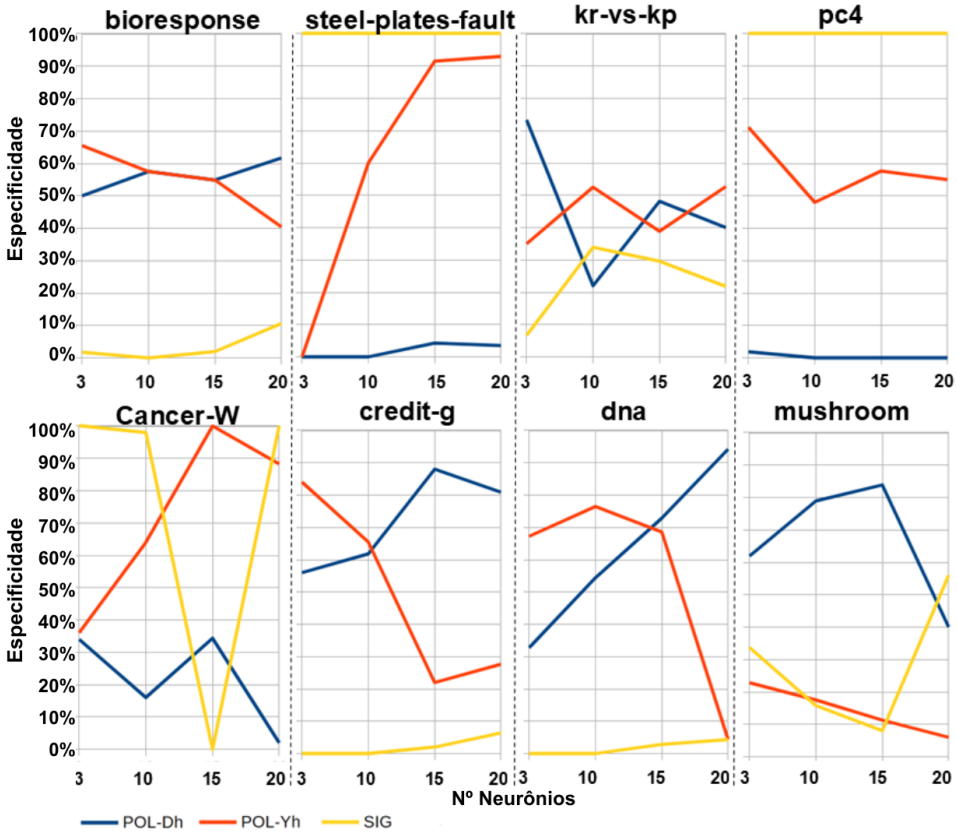
Fonte: Elaborado pelo Autor (2025)

Os gráficos de sensibilidade (Fig. 29) revelam como diferentes RNAs e suas variações de funções de ativação se comportam durante o treinamento em diversos conjuntos de dados. A função de ativação com POL-Dh apresenta redução contínua do erro à medida que camadas adicionais são inseridas, destacando-se especialmente nas bases bioresponse e steel-plates-fault, além de liderar com margem significativa nessa métrica nas bases kr-vs-kp, pc4, Cancer-W e mushroom. Em contraste, o método que utiliza a função de ativação POL-Yh exibe variações

mais acentuadas na sensibilidade, com oscilações perceptíveis em vários conjuntos de dados, sugerindo menor consistência. A função de ativação sigmoide, embora amplamente utilizada, demonstra degradação na sensibilidade, conforme observado nas bases steel-plates-fault, pc4 e mushroom.

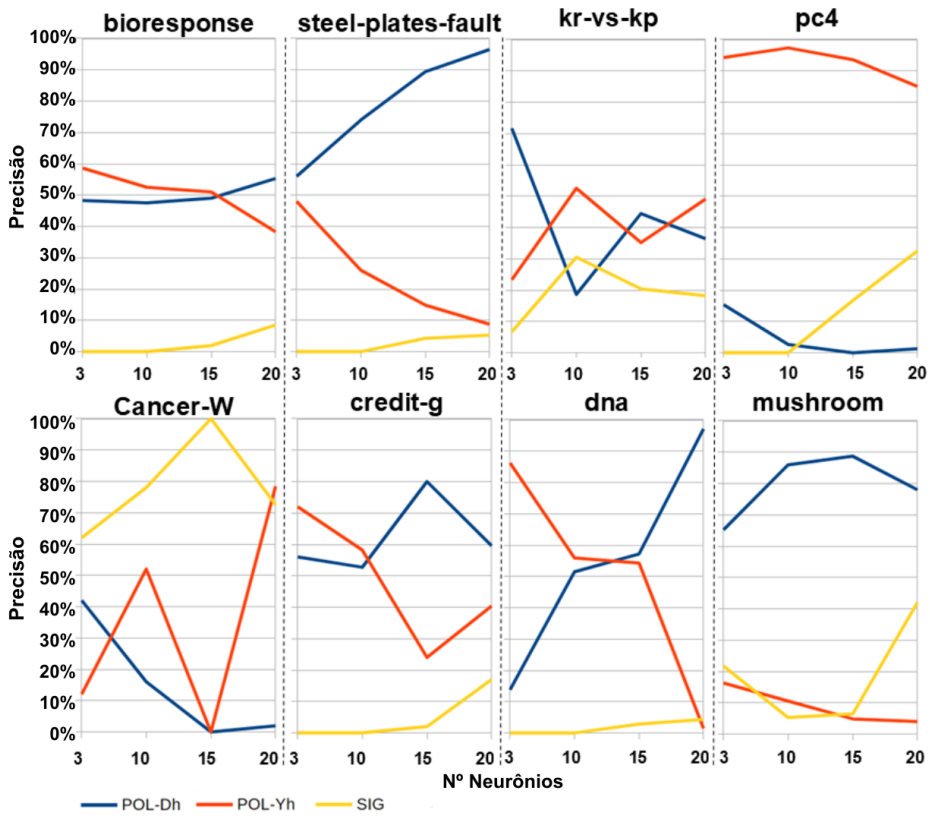
De forma semelhante, em termos de especificidade (Fig. 30), os resultados indicam que a função de ativação com POL-Dh apresenta desempenho superior em diversas bases, como DNA e credit-g, nas quais a taxa de verdadeiros negativos é mais elevada. Entretanto, a função POL-Yh demonstra maior instabilidade, com quedas significativas de especificidade em determinadas configurações de camadas, sugerindo menor confiabilidade nesses cenários. A função sigmoide, embora menos impactada pela adição de camadas, apresenta desempenho inferior na maioria dos casos, especialmente em bases mais complexas, como DNA e credit-g. De modo geral, esses resultados indicam que a função de ativação baseada em POL-Dh é a mais eficaz tanto em sensibilidade quanto em especificidade, configurando-se como uma alternativa promissora para modelos de redes neurais em diferentes contextos.

Figura 30 – Especificidade para diferentes conjuntos de dados e configurações de neurônios.



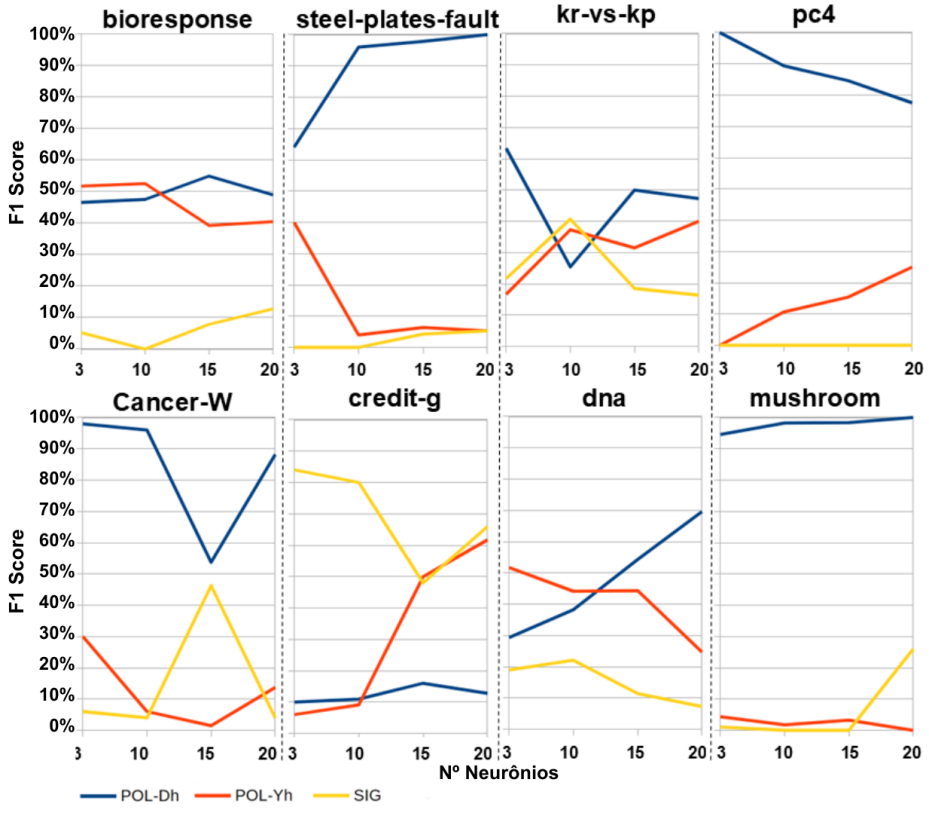
Fonte: Elaborado pelo Autor (2025)

Figura 31 – Precisão para diferentes conjuntos de dados e configurações de neurônios.



Fonte: Elaborado pelo Autor (2025)

Figura 32 – F1-Score para diferentes conjuntos de dados e configurações de neurônios.



Fonte: Elaborado pelo Autor (2025)

A consolidação dessa análise, refletida no F1-Score (Fig. 32), reforça o POL-Dh como a abordagem mais robusta e eficaz. O método mantém desempenho elevado e consistente, especialmente nas bases mushroom e steel-plates-fault. Por outro lado, os métodos POL-Yh e SIG demonstram maior sensibilidade a variações nos hiperparâmetros (eixo X) e nas características dos dados, apresentando quedas abruptas de desempenho, o que indica menor capacidade de generalização.

A fim de estabelecer uma comparação com a literatura, a Tabela 11 apresenta trabalhos que empregam diferentes modelos e técnicas de aprendizagem de máquina. Esses métodos são aplicados às bases de dados de *benchmarks* da OpenML-CC18, também utilizadas nesta dissertação. Nessas tabelas, são apresentadas as acurácias médias comparativas entre oito modelos e/ou técnicas distintas. À luz desse critério, para assegurar a padronização experimental, os resultados do modelo POL-Dh foram obtidos com a configuração de hiperparâmetros composta por três neurônios ocultos.

Tabela 11 – Comparação com os trabalhos presentes na literatura: acurácia média (%)

ID OpenML	**Este trabalho**	**Moussa *et al.* (2024)**	**Briola *et al.* (2023)**		**Jagadish *et al.* (2024)**				
	POL-Dh	**QNNs**	**HCNN-B**	**HCNN-M**	**RL**	**SVM**	**XGBoost**	**TabPFN**	**ERMI**
3	**95,53%**	–	–	–	82,57%	85,14%	79,86%	86,64%	84,50%
15	**98,44%**	98%	97%	97%	–	–	–	–	–
31	**69,96%**	–	–	–	64,21%	63,57%	63,50%	60,36%	61,50%
1049	89,79%	–	**90%**	**90%**	71,36%	75,21%	78,86%	77,07%	77,14%

Fonte: Elaborado pelo Autor (2025)

A partir dos resultados apresentados na Tabela 11, ressalta-se que o objetivo dessa comparação não é estabelecer uma avaliação direta entre os métodos, uma vez que os experimentos foram conduzidos em ambientes computacionais distintos, com diferentes configurações de hiperparâmetros e protocolos experimentais. Assim, os resultados devem ser interpretados apenas como uma referência comparativa, permitindo observar o comportamento geral do método proposto neste trabalho em relação a técnicas recentes presentes na literatura.

Observa-se que o modelo POL-Dh apresenta desempenho competitivo e, em alguns casos, superior às abordagens comparadas. No conjunto de dados da base kr-vs-kp (ID 3) da coleção OpenML-CC18, por exemplo, o método proposto alcança uma acurácia de 95,53%, superando os modelos apresentados em Jagadish *et al.* (2024).

De forma semelhante, no conjunto de dados credit-g (ID 31), o POL-Dh obtém 69,96%, valor superior aos métodos comparativos. Para o conjunto de dados Breast-Cancer-W (ID 15), o resultado obtido (98,44%) também mostra-se levemente superior ao apresentado por Moussa *et al.* (2024) e Briola *et al.* (2023). Apenas no conjunto de dados pc4 (ID 1049) o desempenho do método proposto é ligeiramente inferior ao apresentado pelos modelos HCNN, embora ainda permaneça superior aos métodos clássicos analisados em Jagadish *et al.* (2024).

Nota: QNNs – Redes Neurais Quânticas; RL – Regressão Logística; HCNN-B – *BootstrapNet*; HCNN-M – *MeanSimMatrix*; ERMI – *Entropy Regularized Mutual Information*;

6 CONCLUSÃO

Este trabalho investigou, com base em 4.800 experimentos realizados, o comportamento da técnica proposta, especialmente na variante POL-Dh, como uma alternativa às limitações associadas ao uso de FAs fixas, bem como aos problemas relacionados à inserção de novas camadas ocultas.

Os resultados indicam que a abordagem permite expandir a arquitetura da rede sem comprometer o conhecimento previamente adquirido, inclusive corrigindo algum aprendizado ruim incorporados pela rede. Adicionalmente, o método reduz o número de épocas necessárias para o treinamento da RNA em diversos cenários avaliados. Enquanto a função sigmoide apresentou, em alguns casos, distorções no processo de aprendizado e aumento inicial do erro ao adicionar novas camadas, as FAs polinomiais mostraram tendência de redução mais contínua do RMSE.

A análise das métricas de desempenho — acurácia, sensibilidade, especificidade, precisão e F1-score — indicam que a variante POL-Dh apresentou resultados favoráveis na maioria dos conjuntos de dados avaliados, sugerindo bom comportamento de generalização diante de variações na complexidade e nas características dos dados. Ressalta-se que os resultados obtidos referem-se às métricas calculadas sobre os dados de teste e não apenas sobre os dados de treinamento. Observou-se menor erro quadrático médio (RMSE) ao longo das épocas de treinamento, valores médios competitivos para as métricas e desvio padrão relativamente reduzido, além de equilíbrio entre sensibilidade e F1-score, especialmente nas bases Breast-Cancer-W, Mushrooms e DNA.

Já a variante POL-Yh também apresentou desempenho superior ao método baseado na função sigmoide em alguns cenários, embora tenha obtido resultados mais modestos em comparação à POL-Dh, possivelmente devido à ausência de um mecanismo explícito de estimativa de erro para ajuste da saída dos neurônios nas camadas ocultas.

Em bases com maior complexidade ou desbalanceamento, como credit-g, pc4 e kr-vs-kp, as funções polinomiais apresentaram, em geral, comportamento mais estável que a função sigmoide. Nesses casos, a sigmoide mostrou maior variabilidade ou desempenho irregular em determinadas configurações experimentais. As variantes POL-Dh e, em menor grau, POL-Yh, demonstraram maior equilíbrio entre as métricas analisadas, particularmente em termos de FSC, que tende a ser mais informativo em cenários desbalanceados.

Por outro lado, na base steel-plates-fault observou-se maior dificuldade para todas as abordagens analisadas. Nesse conjunto de dados, verificou-se degradação progressiva do desempenho à medida que a complexidade do problema aumenta. Ainda assim, na análise global baseada no melhor desempenho por métrica, a variante POL-Dh apresentou resultados competitivos em comparação às demais FA avaliadas.

Aditivamente, a comparação com resultados reportados na literatura para bases da coleção OpenML-CC18 indica que a abordagem proposta apresenta desempenho no mínimo comparável

e em diversos casos superior ao desempenho de diferentes técnicas de aprendizado de máquina, incluindo modelos clássicos e arquiteturas mais recentes. Em alguns conjuntos de dados, como kr-vs-kp e breast-cancer-w, foram observadas acurácias semelhantes ou superiores às relatadas em trabalhos anteriores, enquanto em cenários mais complexos, como na base pc4, o desempenho manteve-se próximo ao de abordagens recentes da literatura.

Em síntese, os resultados obtidos sugerem que o uso de regressão polinomial com ajuste da saída, por meio da inclusão de uma estimativa de erro na evolução das FAs, pode contribuir para tornar o processo de aprendizagem mais ágil e estável em RNAs profundas. Esses achados indicam que a abordagem proposta pode ser uma alternativa potencialmente interessante para investigações futuras envolvendo FAt em redes DSN.

6.1 Limitações

O sistema, embora apresente bons resultados, ainda demanda aprimoramentos. Um dos principais desafios observados refere-se à explosão do gradiente, que, nesta pesquisa, foi mitigada por meio da redução da taxa de aprendizagem; contudo, essa abordagem não elimina o problema, indicando a necessidade de investigações mais aprofundadas.

Como consequência direta dessa redução na taxa de aprendizagem, surge uma limitação importante: à medida que novas camadas ocultas são adicionadas, a contribuição individual de cada uma tende a se tornar progressivamente menos significativa. Em cenários mais profundos, isso pode levar a um ponto em que a inclusão de novas camadas deixa de ser estrategicamente eficiente.

Além disso, tem-se outra limitação o fato de o sistema ter sido avaliado apenas em bases de dados de pequeno e médio porte, devido, principalmente, à escassez de tempo e à indisponibilidade de recursos computacionais mais avançados, o que restringe a análise de seu desempenho em cenários de maior escala.

6.2 Trabalhos Futuros

Os resultados obtidos nesta pesquisa abrem perspectivas para o aprofundamento das investigações relacionadas às FAt e à inserção colaborativa de camadas ocultas em RNAs. O conhecimento consolidado ao longo do desenvolvimento do método proposto permite explorar novas estratégias de aprimoramento algorítmico, bem como sua aplicação em novos conjuntos de dados mais complexos e reais.

Um dos caminhos promissores consiste em investigar diferentes estratégias para o isolamento dos termos do polinômio no processo de inserção de novas camadas. No presente trabalho, os termos polinomiais foram isolados de forma crescente em potência. Como extensão natural, propõe-se avaliar o impacto do isolamento em ordem decrescente ou aleatória, analisando seus efeitos sobre a estabilidade do treinamento, a dinâmica de convergência e o desempenho final da rede.

Outra abordagem relevante envolve a utilização de diferentes funções de ativação fixas como base para a construção dos polinômios treináveis. Nesta pesquisa, a função sigmoide foi adotada como referência comparativa. Estudos futuros podem considerar a tangente hiperbólica e a ReLU, investigando como as propriedades intrínsecas dessas funções influenciam o comportamento dos polinômios gerados e sua capacidade de adaptação aos dados.

Soma-se, ainda, a proposta de avaliação do método em bases de dados mais complexas e de maior escala, utilizando conjuntos de *benchmarks*, como os de processamento de imagens e séries temporais, a fim de analisar sua capacidade de generalização, robustez e escalabilidade em cenários mais desafiadores.

Por fim, mostra-se pertinente explorar a viabilidade do emprego de funções não polinomiais e não lineares no mecanismo proposto. A incorporação de funções racionais, exponenciais ou híbridas pode ampliar a flexibilidade representacional do modelo, para novas formas de ajuste dinâmico da saída dos neurônios ocultos. Tais investigações poderão contribuir para o fortalecimento e a ampliação da proposta apresentada, buscando elevar o potencial de generalização da abordagem colaborativa desenvolvida.

REFERÊNCIAS

ACADEMY, Data Science. **Uma breve história das redes neurais artificiais**. 2025. https://www.deeplearningbook.com.br/uma-breve-historia-das-redes-neuraisartificiais/. Acesso em: 09 Dezembro 2025.

APICELLA, Andrea *et al.* A survey on modern trainable activation functions. **Neural Networks**, Elsevier, v. 138, p. 14–32, 2021.

AQUINO, Davi Santiago. Influência do acesso a saneamento básico na incidência e na mortalidade por covid-19: análise de regressão linear múltipla nos estados brasileiros. **Revista Thema**, v. 18, p. 319–331, 2020.

BAKSHI, Rajas. **Stacked Autoencoders**. 2021. Acesso em: 23 fev. 2026. Disponível em: https://towardsdatascience.com/stacked-autoencoders-f0a4391ae282/.

BERGMANN, Dave. **O que é aprendizado semissupervisionado?** 2024. Acesso em: 18 mar. 2026. Disponível em: https://www.ibm.com/br-pt/think/topics/semi-supervised-learning.

BISCHL, Bernd *et al.* Openml benchmarking suites. **arXiv**, 2021. Disponível em: https://arxiv.org/abs/1708.03731.

BISHOP, Christopher M. **Neural networks for pattern recognition**. [S.l.]: Oxford university press, 1995.

BRIOLA, Antonio *et al.* Homological convolutional neural networks. **arXiv preprint arXiv:2308.13816**, 2023.

Campos, FERNANDO AUGUSTO BESSA. **SOLUÇÃO NUMÉRICA DE EQUAÇÕES DIFERENCIAIS PARCIAIS VIA REDES NEURAIS ARTIFICIAIS DE LEGENDRE E CHEBYSCHEV**. Dissertação (Mestrado) — Programa de Pós-Graduação em Matemática e Estatística (PPGME), 2018. Disponível em: https://ppgme.propesp.ufpa.br/ARQUIVOS/dissertacoes/2018/Fernando%20Augusto%20Bessa%20Campos.pdf.

CHAUDHARY, Shreya. **A High-Level Guide to Autoencoders**. 2019. Towards Data Science. Acesso em: 15 fev. 2026. Disponível em: https://towardsdatascience.com/a-high-level-guide-to-autoencoders-b103ccd45924.

CHEIN, Flávia. **Introdução aos modelos de regressão linear: um passo inicial para compreensão da econometria como uma ferramenta de avaliação de políticas públicas**. [S.l.]: Escola Nacional de Administração Pública (Enap), 2019.

CHEN, Weimin *et al.* A patch-based deep learning mri segmentation model for improving efficiency and clinical examination of the spinal tumor. **Journal of Bone Oncology**, Elsevier, p. 100649, 2024.

CHUNG, Hoon; LEE, Sung Joo; PARK, Jeon Gue. Deep neural network using trainable activation functions. In: **2016 International Joint Conference on Neural Networks (IJCNN)**. [S.l.: s.n.], 2016. p. 348–352.

de Almeida Neto, Areolino. **Aplicações de Múltiplas Redes Neurais em Sistemas Mecatrônicos**. Tese (Doutorado) — Instituto Tecnológico de Aeronáutica, 2003.

DENG, Li; YU, Dong. Deep learning: methods and applications. **Foundations and Trends& in Signal Processing**, Emerald Publishing Limited, v. 7, n. 3-4, p. 197–387, 2014.

DOMíNGUEZ-ALMENDROS, S.; BENíTEZ-PAREJO, N.; GONZALEZ-RAMIREZ, A. R. Logistic regression models. **Allergologia et Immunopathologia**, v. 39, n. 5, p. 295–305, 2011.

DOROFKI, Mohammad *et al.* Comparison of artificial neural network transfer functions abilities to simulate extreme runoff data. **International Proceedings of Chemical, Biological and Environmental Engineering**, v. 33, p. 39–44, 2012.

DUBEY, Shiv Ram; SINGH, Satish Kumar; CHAUDHURI, Bidyut Baran. Activation functions in deep learning: A comprehensive survey and benchmark. **Neurocomputing**, Elsevier, v. 503, p. 92–108, 2022.

ERTUĞRUL, Ömer Faruk. A novel type of activation function in artificial neural networks: Trained activation function. **Neural Networks**, Elsevier, v. 99, p. 148–157, 2018.

FAHRMEIR, Ludwig *et al.* **Regression: Models, Methods and Applications**. 2. ed. Berlin; Heidelberg: Springer, 2022. ISBN 978-3-662-63881-1.

FLECK, Leandro *et al.* Redes neurais artificiais: Princípios básicos. **Revista Eletrônica Científica Inovação e Tecnologia**, v. 1, n. 13, p. 47–57, 2016.

FOUNDATION, Python Software. **General Python FAQ**. 2026. Acesso em: 07 jan. 2026. Disponível em: https://docs.python.org/3/faq/general.html#what-is-python.

FOUNDATION, Python Software. **Quotes about Python**. 2026. Acesso em: 07 jan. 2026. Disponível em: https://www.python.org/about/quotes/.

GAYATHIRI, R; SANTHANAM, Suganthi. C-san: Convolutional stacked autoencoder network for brain tumor detection using mri. **Biomedical Signal Processing and Control**, Elsevier, v. 99, p. 106816, 2025.

Google Research. **Google Colaboratory**. 2026. https://colab.google. Acesso em: jan. 2026.

GUADALUPE, Inês Maria Maia Pinto de; AGUILAR, Juan Carlos Zavaleta. Pensamento computacional no ensino médio: Uma sequência didática sobre inteligência artificial e redes neurais artificiais. **ARACÊ**, v. 7, n. 10, p. e9200–e9200, 2025.

GUARNIERI, Ricardo André. Emprego de redes neurais artificiais e regressão linear múltipla no refinamento das previsões de radiação solar do modelo eta. **Insituto Nacional de Pesquisas Espaciais, 171pp**, 2006.

HAYKIN, Simon. **Neural Networks: A Comprehensive Foundation**. Upper Saddle River, NJ: Prentice Hall, 2001.

HOCHREITER, Sepp. Untersuchungen zu dynamischen neuronalen netzen. **Diploma, Technische Universität München**, v. 91, n. 1, p. 31, 1991.

HOCHREITER, Sepp. The vanishing gradient problem during learning recurrent neural nets and problem solutions. **International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems**, World Scientific, v. 6, n. 02, p. 107–116, 1998.

HOFFMANN, Rodolfo. **Análise de regressão: uma introdução à econometria**. [S.l.]: Escola Superior de Agricultura Luiz de Queiroz, Universidade de São Paulo, 2016.

HOFFMANN, Rodolfo; VIEIRA, Sônia. Análise de regressão; uma introdução à econometria. 1987.

JABEEN, Asma; AHMAD, Shahzad; ZAMAN, Shahid. The study of regression model based on com-polynomial in blood cancer drug properties. **Partial Differential Equations in Applied Mathematics**, Elsevier, v. 9, p. 100648, 2024.

JAGADISH, Akshay K *et al.* Human-like category learning by injecting ecological priors from large language models into neural networks. **arXiv preprint arXiv:2402.01821**, 2024.

JANKOWSKI, Norbert; DUCH, Wlodzislaw. New neural transfer functions. **Applied Mathematics and Computer Science**, 1997.

KILIÇARSLAN, Serhat; ADEM, Kemal; ÇELIK, Mete. An overview of the activation functions used in deep learning algorithms. **Journal of New Results in Science**, Tokat Gaziosmanpasa University, v. 10, n. 3, p. 75–88, 2021.

KRARI, Ayoub *et al.* Securing iot networks: A deep learning strategy against rpl selective forwarding attacks. **International Journal of Engineering Trends and Technology**, 2024.

LINCOFF, Gary H. **Field guide to North American mushrooms**. [S.l.]: Knopf National Audubon Society, 1997.

LIU, Guifang; BAO, Huaiqian; HAN, Baokun. A stacked autoencoder-based deep neural network for achieving gearbox fault diagnosis. **Mathematical Problems in Engineering**, Wiley Online Library, v. 2018, n. 1, p. 5105709, 2018.

LIU, Ziming *et al.* Kan: Kolmogorov-arnold networks. **arXiv preprint arXiv:2404.19756**, 2024.

MATHWORKS. **MATLAB**. 2026. Acesso em: 11 jan. 2026. Disponível em: https://www.mathworks.com/products/matlab.html.

MATHWORKS. **MATLAB GUI**. 2026. Acesso em: 11 jan. 2026. Disponível em: https://www.mathworks.com/discovery/matlab-gui.html.

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

MERCANTE, Erivelto *et al.* Modelos de regressão lineares para estimativa de produtividade da soja no oeste do paraná, utilizando dados espectrais. **Engenharia Agrícola**, SciELO Brasil, v. 30, n. 3, p. 504–517, 2010.

MERCIONI, Marina Adriana; HOLBAN, Stefan. The most used activation functions: Classic versus current. In: IEEE. **2020 International Conference on Development and Application Systems (DAS)**. [S.l.], 2020. p. 141–145.

MIQUELLUTI, Daniel Lima; OZAKI, Vitor Augusto; MIQUELLUTI, David José. Uma aplicação do lasso quantílico geograficamente ponderado ao seguro de índice climático. **Revista de Administração Contemporânea**, SciELO Brasil, v. 26, p. e200387, 2022.

MOUSSA, Charles *et al.* Hyperparameter importance and optimization of quantum neural networks across small datasets. **Machine Learning**, Springer, v. 113, n. 4, p. 1941–1966, 2024.

NANDI, Arijit; JANA, Nanda Dulal; DAS, Swagatam. Improving the performance of neural networks with an ensemble of activation functions. In: **2020 International Joint Conference on Neural Networks (IJCNN)**. [S.l.: s.n.], 2020. p. 1–7.

OPENML. **OpenML-CC18 Benchmark Suite**. 2025. Acesso em: 11 jan. 2026. Disponível em: https://docs.openml.org/benchmark/#openml-cc18.

PASCANU, Razvan; MIKOLOV, Tomas; BENGIO, Yoshua. On the difficulty of training recurrent neural networks. In: PMLR. **International conference on machine learning**. [S.l.], 2013. p. 1310–1318.

PEREIRA, Bianca Valéria Lopes. **Reconhecimento de fonemas com compactação das frequências via centroide e redes stacked autoencoders**. Dissertação (Dissertação (Mestrado em Ciência da Computação)) — Universidade Federal do Maranhão, 2023. Programa de Pós-Graduação em Ciência da Computação. Acesso em: 15 fev. 2026. Disponível em: https://tedebc.ufma.br/jspui/handle/tede/5486.

PHILIPP, George; SONG, Dawn; CARBONELL, Jaime G. **The exploding gradient problem demystified: definition, prevalence, impact, origin, tradeoffs, and solutions**. 2018. Disponível em: https://arxiv.org/abs/1712.05577. Acesso em: 24 fev. 2026.

Project Jupyter. **About Jupyter**. 2026. https://jupyter.org/about. Acesso em: jan. 2026.

QUEIROZ, Fellipe *et al.* Predição de consumo energético de aplicações openmp em máquinas multi-core com aprendizado de máquina. In: SBC. **Simpósio em Sistemas Computacionais de Alto Desempenho (SSCAD)**. [S.l.], 2024. p. 129–136.

ROSA, João PS *et al.* Overview of artificial neural networks. In: **Using artificial neural networks for analog integrated circuit design automation**. [S.l.]: Springer, 2019. p. 21–44.

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

RUEDAS, Cristina Martínez *et al.* Assessment of cnn-based methods for discrimination of olive planting systems with sentinel-2 images. **Rafael and Gutiérrez-Reina, Daniel and Castillejo-Gonzalez, Isabel Luisa, Assessment of Cnn-Based Methods for Discrimination of Olive Planting Systems with Sentinel-2 Images**, 2024.

SAZLı, Murat H. A brief review of feed-forward neural networks. **Communications Faculty of Sciences University of Ankara Series A2-A3 Physical Sciences and Engineering**, Ankara University, v. 50, n. 01, 2006.

SCHMIDT, C. M. C. **Modelo de regressão de Poisson aplicado à área da saúde**. 98 p. Dissertação (Dissertação (Mestrado em Modelagem Matemática)) — Universidade Regional do Noroeste do Estado do Rio Grande do Sul, Ijuí, 2003.

SHARMA, Bhavna; VENUGOPALAN, KJIJCE. Comparison of neural network training functions for hematoma classification in brain ct images. **IOSR Journal of Computer Engineering**, v. 16, n. 1, p. 31–35, 2014.

SHARMA, Sagar; SHARMA, Simone; ATHAIYA, Anidhya. Activation functions in neural networks. **Towards Data Sci**, v. 6, n. 12, p. 310–316, 2017.

SILHAN, Tim; OEHMCKE, Stefan; KRAMER, Oliver. Evolution of stacked autoencoders. In: IEEE. **2019 IEEE Congress on Evolutionary Computation (CEC)**. [S.l.], 2019. p. 823–830.

SILVA, Ivan Nunes da; SPATTI, Danilo Hernane; FLAUZINO, Rogério Andrade. **Redes Neurais Artificiais para Engenharia e Ciências Aplicadas: Fundamentos Teóricos e Aspectos Práticos**. 2. ed. São Paulo, SP, Brasil: Artliber Editora, 2016. ISBN 978-85-88098-87-9.

SILVA, Mariane Tavares; SANTOS, Charles Morphy D. Uma análise histórica sobre a seleção natural: de darwin-wallace à síntese estendida da evolução. **Amazônia: Revista de Educação em Ciências e Matemáticas**, Universidade Federal do Pará, v. 11, n. 22, p. 46–61, 2015.

TADANO, Yara de Souza; UGAYA, Cássia Maria Lie; FRANCO, AdMiLSon TeixeirA. Método de regressão de poisson: metodologia para avaliação do impacto da poluição atmosférica na saúde populacional. **Ambiente & Sociedade**, SciELO Brasil, v. 12, n. 2, p. 241–255, 2009.

TEIXEIRA, Aline Porfiro. **Múltiplas Redes Neurais Utilizando Redes MLP e RBF.** Tese (Doutorado) — UEMA, 2017.

VIANA, Francisco dos Santos *et al.* Classificação de sílabas fonéticas via centroides e inserção colaborativa de camadas em redes stacked autoencoders. **Revista Delos**, v. 18, n. 71, p. e6481–e6481, 2025.

WIDROW, Bernard; HOFF, Marcian. **An Adaptive "Adaline"Neuron Using Chemical "Memistors"**. Stanford, CA, 1960. Prepared under Office of Naval Research Contract Nonr 225(24).

YAMAGUCHI, L. C. Takao *et al.* Matriz de oferta agropecuária: uma aplicação de novas técnicas de regressão de cume. **Revista de Economia e Sociologia Rural**, Sociedade Brasileira de Economia, Administração e Sociologia Rural (SOBER), v. 23, n. 2, p. 235–249, 2019.

ZHANG, Xiaohui *et al.* Improving deep neural network acoustic models using generalized maxout networks. In: IEEE. **2014 IEEE international conference on acoustics, speech and signal processing (ICASSP)**. [S.l.], 2014. p. 215–219.