Parametrização automática de redes neurais utilizando algoritmos evolutivos

https://doi.org/10.56238/sevened2024.007-089

Ádamo Henrique Rocha de Oliveira
Grau de formação mais alto: mestrado
Instituição acadêmica: Instituto Federal do Maranhão

Yrlles Araujo Moraes
Grau de formação mais alto: mestrado
Instituição acadêmica: Instituto Federal do Maranhão

Rayssa Pereira Moraes Rego Sobrinho
Grau de formação mais alto: graduação
Instituição acadêmica: Instituto Federal do Maranhão

Jorleanderson Moraes Maia
Grau de formação mais alto: graduação
Instituição acadêmica: Instituto Federal do Maranhão

Marcos Silva Chaves
Grau de formação mais alto: graduação
Instituição acadêmica: Instituto Federal do Maranhão

RESUMO
A Inteligência Artificial consiste em uma área onde são desenvolvidos métodos ou sistemas que atuam de maneira inteligente, aproximando-se do comportamento humano, em situações que envolvem resolução de problemas, aquisição e representação de conhecimento, reconhecimento padrões, etc. Dentro deste contexto, um tipo de modelo computacional que ganhou destaque são as Redes Neurais Artificiais (RNA's), que são formadas por blocos básicos inspirados no neurônio biológico. Uma RNA possui capacidade de atuar em diversas aplicações como, por exemplo, aproximação universal de funções, controle de processos, reconhecimento e classificação de padrões e agrupamento de dados. Para atender uma ampla gama de aplicações, uma RNA exige a determinação de uma série de parâmetros, dentre eles: topologia, número de camadas, quantidade de neurônios, função de ativação, método de treinamento, etc. Ou seja, o projeto de uma RNA com a configuração mais adequada para cada tipo de problema requer uma série de escolhas, testes preliminares e experiência do projetista. Contudo, para evitar que tais escolhas sejam realizadas de maneira empírica, é possível tratar esta parametrização como um problema de otimização, permitindo sua resolução através da utilização de algoritmos evolutivos, que são ferramentas de otimização desenvolvidas para simular diversos processos evolutivos naturais. Neste trabalho foram aplicados o Algoritmo Genético e a Evolução Diferencial com codificação binária para parametrizar automaticamente redes neurais de camada oculta única aplicadas na modelagem de um conversor buck e na previsão de resistência à compressão de concreto autoadensável (SCC) com adição de fibras. As redes neurais utilizadas foram treinadas com o algoritmo Máquina de Aprendizado Extremo e os resultados das simulações mostram que o Algoritmo Genético foi a técnica que apresentou o melhor desempenho ao parametrizar a rede no processo de modelagem do conversor buck, enquanto a Evolução Diferencial combinada com a codificação binária GVP foi a melhor estratégia para parametrizar a rede neural no processo de previsão de resistência à compressão de SCC.

Palavras-chave: Inteligência Artificial, Redes Neurais Artificiais, Parametrização, Algoritmos Evolutivos, Codificação Binária.

1 INTRODUÇÃO

A Inteligência Artificial (IA) é uma área que constitui vários procedimentos computacionais cujas funções realizadas, caso um ser humano as executasse, seriam consideradas inteligentes. O principal propósito desta área é buscar métodos ou sistemas computacionais que possuam ou reforcem a capacidade de comportamentos inteligentes do ser humano, como a de resolver problemas, adquirir e representar conhecimento, reconhecer padrões, dentre outros (Lima; Pinheiro; Santos, 2016).

Os sistemas considerados inteligentes, são aqueles que apresentam determinadas capacidades como: aquisição de conhecimento, planejamento de eventos, resolução de problemas, representações de informações, armazenamento de conhecimento, comunicação através de linguagens coloquiais e aprendizado.

Conforme afirmam Lima, Pinheiro e Santos (Lima; Pinheiro; Santos, 2016), o aumento do poder computacional ocorrido nas últimas décadas permitiu o destaque de uma linha de pesquisa da IA conhecida como Aprendizado de Máquina. Esta linha de pesquisa tem por objetivo estudar e desenvolver métodos computacionais para a obtenção de sistemas capazes de adquirir conhecimento de forma automática.

O desafio principal dos algoritmos de aprendizagem é maximizar a capacidade de generalização de seu aprendizado. Dentre as diversas alternativas promissoras na resolução deste desafio, encontram-se as Redes Neurais Artificiais.

Redes Neurais Artificiais (RNA) têm sido, ao longo dos últimos anos, uma área da IA de grande desenvolvimento. As RNA’s podem ser caracterizadas como modelos computacionais capazes de adaptar, aprender, generalizar, agrupar ou organizar dados. Tal capacidade é adquirida através dos modelos inspirados no sistema nervoso de seres vivos (Lima; Pinheiro; Santos, 2016; Silva; Spatti; Flauzino, 2010).

As redes neurais possuem algumas características, como não linearidade e adaptabilidade, que permitem a sua aplicação em diversas áreas. As principais aplicações das redes neurais são: aproximação universal de funções, controle de processos, reconhecimento e classificação de padrões, agrupamento de dados (clusterização), sistemas de previsão, otimização de sistemas e memórias associativas.

Uma RNA é formada por um conjunto de unidades básicas de processamento que se comunicam enviando informações uma para a outra por meio de determinadas conexões. Estas unidades de processamento são chamadas de neurônios e são modelos matemáticos inspirados no neurônio biológico (Lima; Pinheiro; Santos, 2016).

A forma de interconexão dos neurônios em uma RNA define a sua arquitetura, sendo as principais: feedforward de camada simples, feedforward de camadas múltiplas, recorrente e estrutura reticulada. Uma vez definida a arquitetura, haverá ainda a segmentação da RNA em camadas: a

primeira camada recebe os dados a serem tratados; as camadas intermediárias fazem o tratamento destes, através da extração de características pertinentes; e a última camada reproduz os resultados gerados (Silva; Spatti; Flauzino, 2010).

A forma como uma RNA adquire conhecimento sobre um determinado processo é conhecida como treinamento e pode ser feita através de vários algoritmos, dentre os quais podem ser citados: backpropagation, método de Newton, Levenberg–Marquardt e Extreme Learning Machine (Huang; Zhu; Siew, 2004; Silva; Spatti; Flauzino, 2010; Wilamowski; Irwin, 2018).

A aplicação de uma RNA para resolver um determinado problema, além da escolha da topologia e do método de treinamento, envolve ainda uma série de determinações relacionadas a quantidade de neurônios, funções de ativação, pesos iniciais, etc. Para aplicações nas áreas de identificação de sistemas e previsão de séries temporais, parâmetros adicionais podem surgir como, por exemplo: quantidade de termos aplicados na camada de entrada, tipos de regressores, atrasos dos sinais de entrada e/ou saída, etc. Cada conjunto de parâmetros escolhido pode levar a RNA a ter um desempenho melhor ou pior, dependendo do tipo de problema a ser tratado.

Em outras palavras, projetar uma RNA com a configuração mais adequada para cada tipo de problema requer uma série de escolhas, testes preliminares e experiência do projetista. Entretanto, para que determinados parâmetros não sejam determinados de maneira empírica, estes podem ser tratados como variáveis de um problema de otimização, que por sua vez é possível de ser resolvido através de diversas técnicas, dentre elas os algoritmos evolutivos (Gaspar-Cunha; Takahashi; Antunes, 2013).

Desta forma, este trabalho tem por finalidade o desenvolvimento de uma estratégia de projeto de RNA, utilizando algoritmos evolutivos em algumas etapas para que a parametrização seja realizada de forma automática, visando a praticidade no projeto e a obtenção do melhor desempenho possível por parte da RNA, atendendo a critérios de performance a serem selecionados.

Este trabalho está organizado conforme itens a seguir. Na seção 2 será apresentada a fundamentação teórica sobre o tema estudado. Na seção 3 será tratada a estratégia proposta. Na seção 4 são apresentados os estudos de caso utilizados no trabalho. Na seção 5 são apresentados e discutidos os resultados obtidos. E, por fim, na seção 6 estão as conclusões.

2 FUNDAMENTAÇÃO TEÓRICA

Esta seção apresenta os principais conceitos relacionados aos assuntos necessários para elaboração da estratégia proposta para parametrização de redes neurais utilizando algoritmos evolutivos. São tratados conceitos sobre redes neurais artificiais, algoritmos evolutivos e codificações binárias.

2.1 REDES NEURAIS ARTIFICIAIS

Redes neurais artificiais (RNA's) são modelos computacionais inspirados no sistema nervoso de seres vivos. A célula elementar do sistema nervoso cerebral é o neurônio e seu papel se resume a conduzir impulsos sob determinadas condições de operação (Silva; Spatti; Flauzino, 2010). Assim como as redes neurais biológicas, as RNA's também são compostas por unidades básicas: os neurônios artificiais (denominados apenas como "neurônios").

Um neurônio é uma unidade de processamento de informação que é fundamental para a operação de uma rede neural (Haykin, 2001). O diagrama de blocos apresentado na Figura 1 mostra o modelo de um neurônio, que forma a base para o projeto de RNA's.

Três elementos básicos devem ser destacados neste modelo:

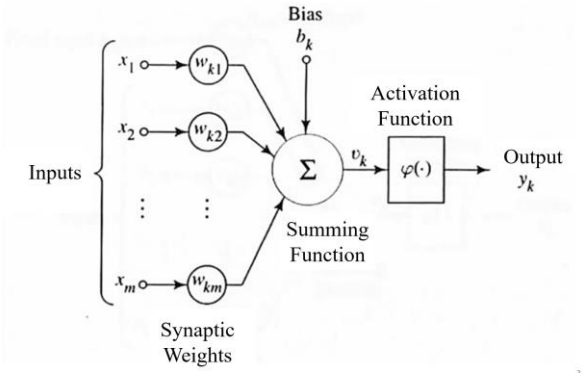
1 – Um conjunto de sinapses, cada uma caracterizada por um peso. Especificamente, um sinal x_j na entrada da sinapse j conectada ao neurônio k é multiplicada pelo peso sináptico w_{kj}.

2 – Um somador para somar os sinais de entrada, ponderados pelas respectivas sinapses do neurônio; as operações descritas aqui constituem um combinador linear.

3 – Uma função de ativação para restringir a amplitude da saída de um neurônio.

O modelo neuronal da Figura 1 inclui também um bias aplicado externamente, representado por b_k. Este bias tem o efeito de aumentar ou diminuir a entrada líquida da função de ativação, dependendo se ele é positivo ou negativo, respectivamente.

Figura 1 – Modelo não linear de um neurônio.



Fonte: Haykin (2001).

Em termos matemáticos, podemos descrever um neurônio k escrevendo o seguinte par de equações:

$$u_k = \sum_{j=1}^{m} w_{kj} x_j \qquad (1)$$

e

$$y_k = \varphi(u_k + b_k) \qquad (2)$$

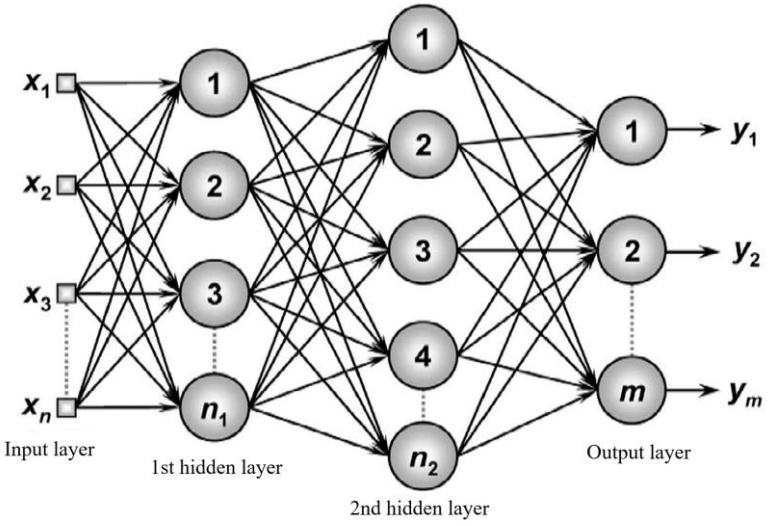
Onde, x_1, x_2, ..., x_m são os sinais de entrada; w_{k1}, w_{k2}, ..., w_{km} são os pesos sinápticos do neurônio k; u_k é a saída do combinador linear devido aos sinais de entrada; b_k é o bias; $\varphi(\cdot)$ é a função de ativação; e y_k é o sinal de saída do neurônio (Haykin, 2001).

As maneiras como os neurônios são interconectados definem as topologias ou arquiteturas das redes neurais. As principais arquiteturas de redes neurais são apresentadas por Silva, Spatti e Flauzino (2010), como, por exemplo, arquitetura *feedforward* de camada simples, arquitetura *feedforward* de camadas múltiplas, arquitetura recorrente ou realimentada e arquitetura em estrutura reticulada.

Basicamente, uma rede neural artificial pode ser dividida em três partes, denominadas de camadas, as quais podem ser constituídas por uma quantidade variável de neurônios e são nomeadas da seguinte forma: camada de entrada, responsável por receber informações (dados), sinais, características ou medições do meio externo; camadas ocultas, compostas de neurônios responsáveis por extrair as características associadas ao processo ou sistema a ser inferido; e camada de saída, responsável pela produção e apresentação dos resultados finais da rede (Silva; Spatti; Flauzino, 2010).

Na Figura 2 é possível observar um exemplo de rede neural com a camada de entrada, duas camadas ocultas e a camada de saída, constituídas por n, n_1, n_2 e m neurônios, respectivamente. A camada de entrada recebe o sinal representado pelo conjunto de dados $\boldsymbol{x} = [x_1, x_2, x_3, \ldots, x_n]$, as camadas intermediárias processam estes dados e a camada de saída apresenta o resultado final da rede, ou seja, o conjunto de dados $\boldsymbol{y} = [y_1, y_2, \ldots, y_m]$. Especificamente, esta rede do exemplo possui arquitetura *feedforward* de múltiplas camadas.

Figura 2 – Exemplo de rede *feedforward* de camadas múltiplas.



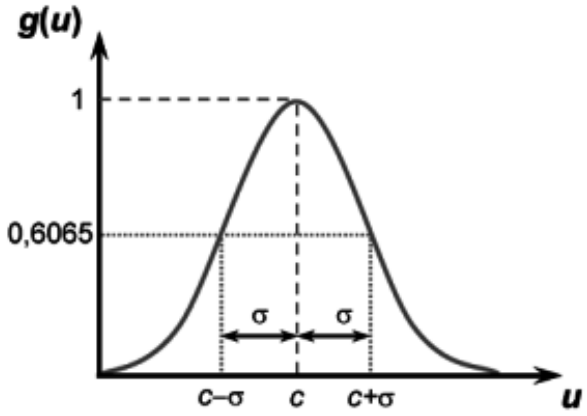
Fonte: Silva, Spatti e Flauzino (2010).

Uma determinada arquitetura de rede neural pode possuir diferentes tipos de topologias, as quais podem ser definidas pelas quantidades de neurônios utilizados ou, ainda, pelos diferentes tipos de funções de ativação. Alguns exemplos de funções de ativação apresentados por Silva, Spatti e Flauzino (2010) são: função degrau, função sinal, função rampa simétrica, função logística, função tangente hiperbólica e função gaussiana. Na Figura 3 é possível observar a função de ativação gaussiana, dada por:

$$g(u) = e^{-\frac{(u-c)^2}{2\sigma^2}} \qquad (3)$$

onde, c é um parâmetro que define o centro da função gaussiana e σ denota o desvio padrão associado à mesma, isto é, o quão espalhada está a curva em relação ao seu centro. Ainda observando o gráfico da Figura 3, é possível constatar que o valor do desvio padrão está diretamente associado com o ponto de inflexão da função gaussiana, sendo que σ^2 indica a sua respectiva variância.

Figura 3 – Função de ativação gaussiana.



Fonte: Silva, Spatti e Flauzino (2010).

Outro aspecto relevante das redes neurais, é a maneira como o conhecimento é adquirido e armazenado, ou seja, o seu processo de treinamento, que consiste na aplicação de um conjunto de passos ordenados com o intuito de ajustar os pesos dos neurônios (Silva; Spatti; Flauzino, 2010). Tal processo de ajuste, também conhecido como algoritmo de aprendizagem, visa então sintonizar a rede para que as suas respostas estejam próximas dos valores desejados. Os principais métodos de aprendizado são: supervisionado, não supervisionado e com reforço.

Existem diversos tipos de topologias de redes neurais e algumas que merecem destaque são: *Perceptrons* de Camada Única, *Adaline*, *Perceptrons* de Múltiplas Camadas, Redes de Função de Base Radial, Máquinas de Vetor de Suporte, Máquinas de Comitê, Redes recorrentes de Hopfield, Redes auto-organizáveis de Kohonen, Redes LVQ e Redes ART (Haykin, 2001; Silva; Spatti; Flauzino, 2010).

Tratando-se de redes neurais do tipo *feedforward* da camada única (SLFN, do inglês *Single Hidden Layer Feedforward Neural Network*), é possível elencar alguns algoritmos de treinamento, como *backpropagation*, método de Newton e Levenberg–Marquardt (Silva; Spatti; Flauzino, 2010; Wilamowski; Irwin, 2018), porém um algoritmo que vem ganhando destaque na literatura é o chamado Máquina de Aprendizado Extremo (ELM, do inglês *Extreme Learning Machine*) (Huang; Zhu; Siew, 2004). Este algoritmo realiza o treinamento de redes SLFN de maneira rápida e simplificada, fazendo uso da inversão generalizada de Moore-Penrose (Serre, 2002) para calcular analiticamente os pesos de saída da rede. Conforme apresentado por Huang, Zhu e Siew (2004), esta técnica permite obter a menor norma dos pesos, evita a convergência para mínimos locais e não necessita de muitos passos iterativos para obter o melhor desempenho de aprendizagem, diferente do que ocorre nos métodos de baseados em gradiente descendente.

2.2 ALGORITMOS EVOLUTIVOS

Os algoritmos evolutivos são técnicas heurísticas inspiradas em mecanismos de adaptação dos seres vivos, conforme observados na natureza, onde determinados mecanismos produzem respostas adequadas para problemas de grande complexidade. Este efeito, no qual o desenvolvimento dos algoritmos evolutivos é baseado, ocorre como consequência da execução de ações simples por múltiplos agentes individuais que interagem entre si e com o ambiente, produzindo coletivamente a solução do problema de adaptação (Gaspar-Cunha; Takahashi; Antunes, 2013).

Conforme afirmam Gaspar-Cunha, Takahashi e Antunes (Gaspar-Cunha; Takahashi; Antunes, 2013), vale também ressaltar que a classe de algoritmos evolutivos, nos últimos anos, tem crescido em convergência com linhas de pesquisa provenientes do campo da Pesquisa Operacional, que há muito tem se dedicado ao estudo de técnicas heurísticas estocásticas de propósito geral, conhecidas como metaheurísticas. Isto permite que a gama de algoritmos possíveis de serem aplicados seja ainda maior, enriquecendo cada vez mais o desenvolvimento de técnicas para solução de problemas complexos.

Alguns exemplos de algoritmos evolutivos são: Algoritmo Genético (Holland, 1975), Evolução Diferencial (Storn; Price, 1997), Estratégias Evolutivas (Rechenberg, 1965; Schwefel, 1965), Programação Genética (Koza, 1992), Colônia de Formigas (Dorigo, 1992) e Algoritmos Imunoinspirados (De Castro; Timmis, 2002). Neste trabalho serão utilizados o Algoritmo Genético (GA) e a Evolução Diferencial (DE) como os algoritmos aplicados na parametrização automática de redes neurais.

2.3 CODIFICAÇÕES BINÁRIAS

O problema de parametrizar redes neurais apresenta alguns aspectos caracterizados como otimização binária. Por exemplo, a decisão de quantas e quais entradas a rede irá possuir. Algoritmos

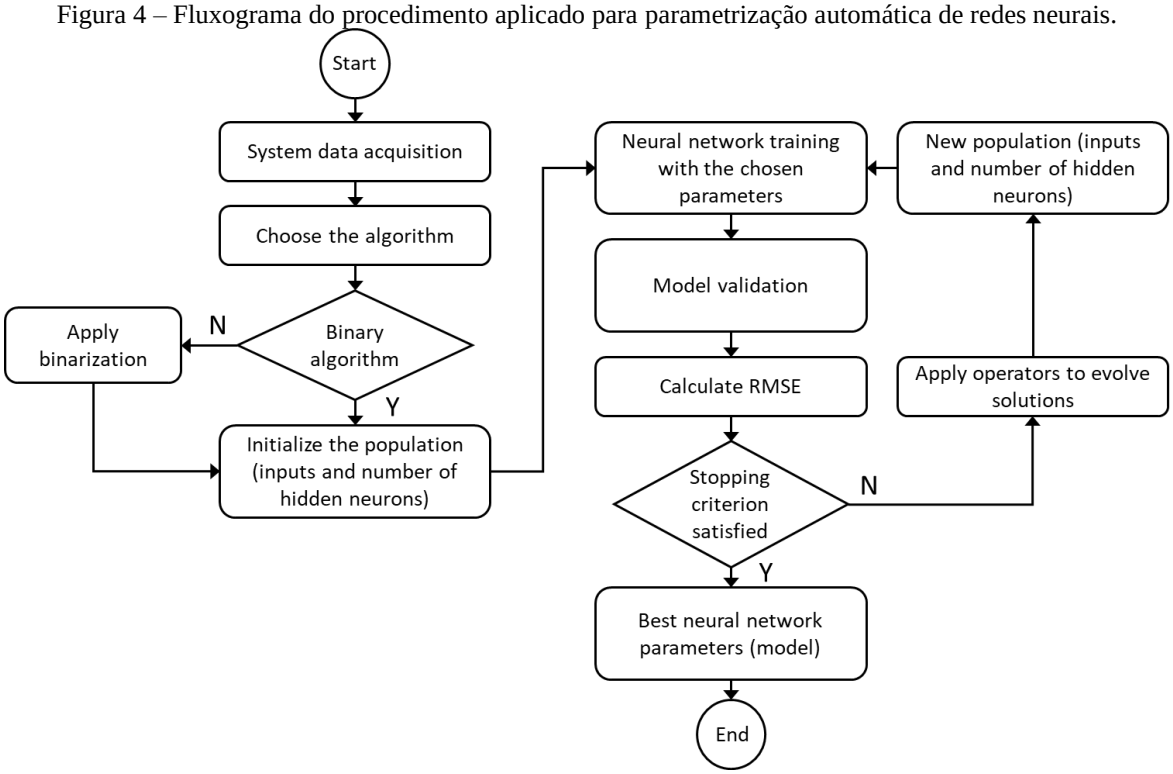
evolutivos vêm sendo cada vez mais aplicadas para resolver problemas de otimização binária. Alguns deles foram originalmente criadas para lidar com problemas binários, como o Algoritmo Genético, enquanto outros foram projetadas para trabalhar com variáveis contínuas e necessitam de alguma adaptação para tratar deste tipo de problema, como a Evolução Diferencial (Dahi; Mezioud; Draa, 2015).

Neste trabalho, as codificações binárias utilizadas para adaptar a Evolução Diferencial serão: Codificação binária por Função de Transferência (TF, do inglês *Transfer Function*); Codificação binária por Prioridade do Maior Valor (GVP, do inglês *Great Value Priority*); e Codificação binária por Modulação em Ângulo (AM, do inglês *Angle Modulation*). Todas as codificações utilizadas na pesquisa encontram-se detalhadas na literatura (Crawford *et al.*, 2017).

3 ESTRATÉGIA PROPOSTA

A estratégia proposta neste trabalho para parametrização de redes neurais utilizando algoritmos evolutivos consiste na aplicação de alguns passos, que podem ser observados no fluxograma da Figura 4.

Figura 4 – Fluxograma do procedimento aplicado para parametrização automática de redes neurais.



Fonte: Os autores (2024).

Inicialmente, deve-se dispor de dados experimentais obtidos do sistema estudado. Em seguida deve-se determinar qual algoritmo evolutivo será utilizado, o qual irá manipular soluções para realizar a parametrização automática da rede neural. Os parâmetros a serem determinados neste estudo são a quantidade e os tipos de entradas e a quantidade de neurônios na camada oculta.

Caso seja selecionado o GA, este possui todas as estruturas para determinar estes parâmetros. Tanto a determinação das entradas, quanto a escolha da quantidade de neurônios, podem ser codificadas por um vetor em formato binário. Contudo, para a DE, será necessário aplicar codificações binárias para a determinação das entradas, uma vez que este algoritmo evolutivo é do tipo contínuo e, na sua configuração padrão, só conseguiria determinar a quantidade de neurônios.

Considerando todas as combinações possíveis entre o algoritmo DE e as codificações binárias (TF, GVP e AM), além de incluir na análise o algoritmo GA, obtém-se um total de 4 técnicas a serem avaliadas no processo de parametrização automática. Estas técnicas encontram-se listadas na Tabela 1.

Tabela 1 – Técnicas aplicadas na estratégia de parametrização automática

Metaheurística	**Codificação Binária**	**Técnica**
GA	-	GA
DE	TF	DE-TF
	GVP	DE-GVP
	AM	DE-AM

Fonte: Os autores (2024).

A etapa seguinte consiste na criação de uma população inicial de soluções e no cálculo da aptidão de cada uma delas através da função objetivo. Esta função será composta pela rede neural sendo aplicada na predição do comportamento do sistema em estudo, através das etapas de treinamento com Máquina de Aprendizado Extremo e validação do modelo.

O desempenho de uma solução será baseado na validação do modelo gerado por esta solução, sendo aplicado como índice de desempenho a raiz quadrada do erro médio quadrático (RMSE, do inglês *Root Mean Square Error*) calculado por meio da seguinte equação (Peñaranda; Saavedra-Montes, 2012):

$$RMSE = \sqrt{\frac{\sum_{k=1}^{N}\left(y(k) - \hat{y}(k)\right)^2}{N}} \quad (4)$$

Onde $y(k)$ e $\hat{y}(k)$ são, respectivamente, as saídas observada e predita utilizando os dados de validação, e N é a quantidade de dados utilizados.

A partir deste ponto, inicia-se o ciclo iterativo dos algoritmos evolutivos, onde são aplicados os operadores para evoluir a população de soluções. A cada vez que este ciclo é executado, conta-se uma geração e o mesmo se repetirá até que algum critério de parada seja atingido. Neste trabalho, o critério de parada utilizado será a quantidade máxima de gerações.

Uma vez atingido o critério de parada, o algoritmo encerrará a busca e o melhor conjunto de parâmetros para a rede neural será aquele que gerar um modelo com menor RMSE. Com a finalização

desta etapa e seleção dos melhores parâmetros, tem-se a conclusão de uma simulação. Para fins de análise do desempenho das técnicas apresentadas, cada simulação será repetida várias vezes.

Para avaliar o desempenho de cada técnica, inicialmente será verificado o comportamento médio da curva de convergência da função objetivo. Para isto, em cada geração serão tomados os valores obtidos por todas as simulações e tirada a média.

Outra maneira de avaliar o desempenho das técnicas será através do valor da função objetivo alcançado ao final de cada simulação. Neste caso, serão verificados os valores mínimo, médio, máximo e desvio padrão dentre todas as simulações realizadas por cada técnica.

Todos os códigos utilizados neste trabalho foram implementados no software Matlab e as simulações foram executadas em um notebook com processador Intel Core i7 e 16GB de memória RAM (do inglês *Random Access Memory*).

4 ESTUDOS DE CASO

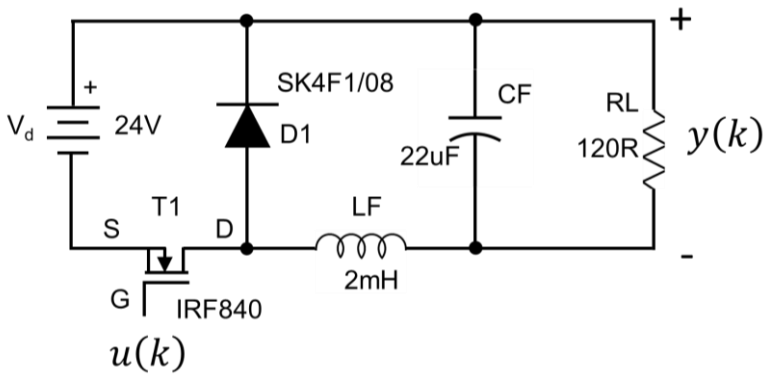
Esta seção apresenta os sistemas de teste que serão utilizados como estudos de caso para validar a estratégia proposta.

4.1 MODELAGEM DE UM CONVERSOR *BUCK*

O primeiro estudo de caso deste trabalho consiste em utilizar uma rede neural para modelar um conversor *buck*. A estratégia de modelagem consiste em aplicar técnicas de identificação de sistemas utilizando uma rede neural de camada oculta única treinada com o algoritmo Máquina de Aprendizado Extremo, conforme estudo conduzido por Oliveira e Leandro (2019).

O conversor *buck* que servirá de objeto de estudo foi discutido inicialmente no trabalho de Aguirre, Donoso-Garcia e Santos-Filho (2000) e frequentemente é utilizado como sistema de teste na área de identificação de sistemas. O diagrama do conversor *buck* é apresentado na Figura 5.

Figura 5 – Diagrama do conversor *buck*.

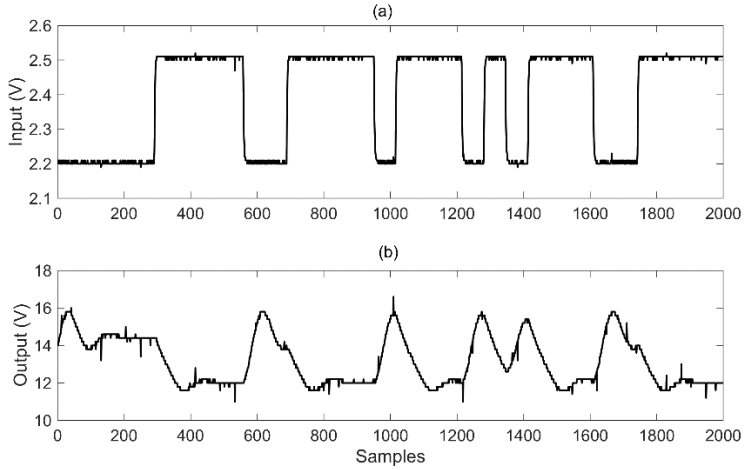


Fonte: Aguirre, Donoso-Garcia e Santos-Filho (2000).

O sistema de regulação de tensão de carga não é mostrado na Figura 5. No experimento realizado com este conversor, a fonte de alimentação v_d foi mantida constante e igual a $24V$. Para excitar a dinâmica do conversor, optou-se por um sinal de entrada do tipo PRBS, limitado entre $2{,}2V$ e $2{,}5V$, aplicado ao *buck* usando-se um conversor Digital/Analógico (Aguirre; Donoso-Garcia; Santos-Filho, 2000).

Neste experimento, a tensão que define a razão cíclica do conversor foi considerada como sinal de entrada, $u(k)$, e a tensão elétrica na saída do conversor foi adotada como sinal de saída, $y(k)$. Foram coletados 2000 pares de amostras, que podem ser observados na Figura 6.

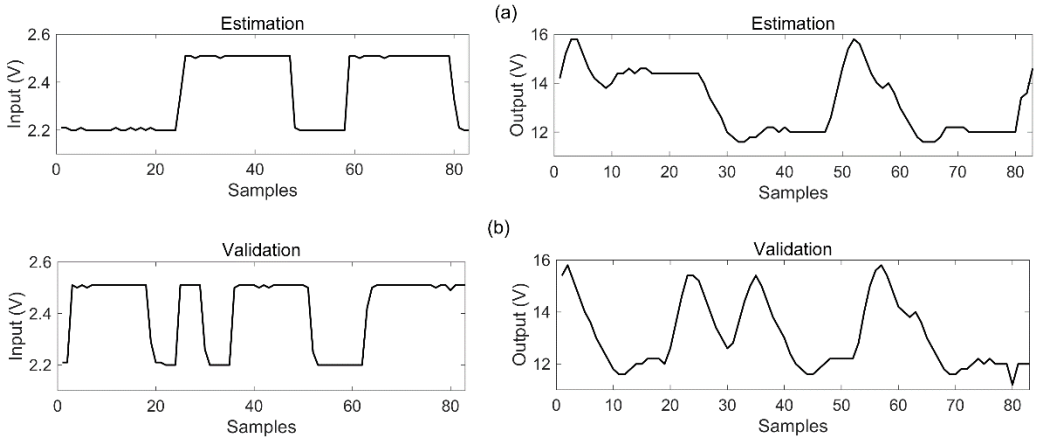
Figura 6 – Dados coletados do conversor *buck*. (a) sinal de entrada e (b) sinal de saída.



Fonte: Aguirre, Donoso-Garcia e Santos-Filho (2000).

Como os dados foram amostrados a uma taxa mais alta do que a necessária, optou-se por aplicar um tempo de amostragem de trabalho igual a $T_{s^*} = 120\mu s$, resultando num total de 166 pares de amostras. Além disso, os dados foram divididos em dois conjuntos distintos, cada um com 50% dos 166 pares de amostras. A primeira metade dos dados foi utilizada para estimação dos modelos e a segunda metade foi reservada para a etapa de validação, conforme observado na Figura 7.

Figura 7 – Sinais de entrada e saída do conversor *buck*. Dados de estimação (a) e validação (b).



Fonte: Os autores (2024).

Os dados apresentados na FIGURA 8 foram utilizados por Aguirre, Donoso-Garcia e Santos-Filho (2000) para obtenção de modelos para o conversor *buck*. Estes mesmos dados foram disponibilizados na internet pelos autores, o que permitiu sua utilização neste estudo de caso.

4.2 PREVISÃO DA RESISTÊNCIA À COMPRESSÃO DE CONCRETO AUTOADENSÁVEL (SCC) COM ADIÇÃO DE FIBRAS

O segundo estudo de caso deste trabalho consiste na utilização da rede neural para prever a resistência à compressão do concreto autoadensável (SCC) com adição de fibras. A literatura aponta diversos estudos que aplicam abordagens semelhantes (Balasubramaniam; Padmanaban, 2022; Gholamzadeh-Chitgar; Berenjian, 2019; Najm; Mohamed; Alzard, 2023; Saha; Prasad; Kumar, 2017; Tavakoli *et al.*, 2014), uma vez que a previsão de propriedades de materiais empregados na construção civil utilizando técnicas computacionais têm apresentado resultados muito promissores.

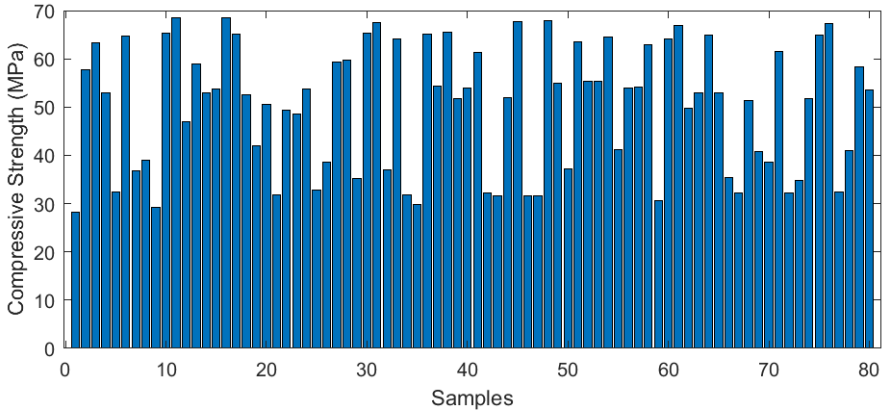
O SCC é um material utilizado na construção civil e caracteriza-se por sua capacidade de fluir através da seção fortemente reforçada com a viscosidade necessária sem segregação (Balasubramaniam; Padmanaban, 2022). Além disso, tende a apresentar maiores resistências a partir da adição de fibras, quando comparados à concretos sem adições.

A base de dados utilizada foi desenvolvida experimentalmente por Saha, Prasad e Kumar (2017) contendo um total de 99 amostras. Para a previsão de uma única saída (resistência à compressão), foram utilizados nove parâmetros de entrada que incluem quantidades de cimento, agregado miúdo (areia), agregado graúdo, cinzas volantes, fibras de vidro, fibras de polipropileno, água, super plastificante e aditivo modificador de viscosidade (VMA).

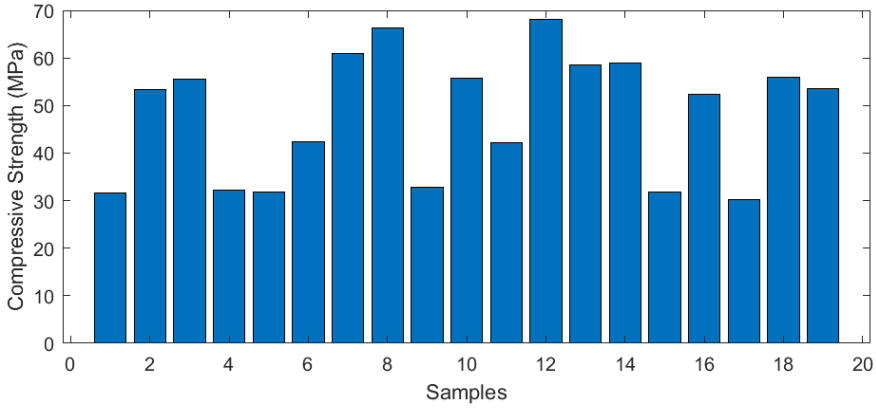
Os dados foram divididos em dois conjuntos: o conjunto de treinamento, com 80% dos dados (80 amostras), e o conjunto de teste com 20% dos dados (19 amostras). Na Figura 8, estão apresentados os valores de saída destes dois conjuntos

Figura 8 – Valores de resistência à compressão do SCC. Dados de treinamento (a) e teste (b).

(a)



(b)



Fonte: Os autores (2024).

5 RESULTADOS

Nesta seção serão apresentados e discutidos os resultados obtidos pela aplicação da estratégia proposta nos estudos de casos indicados na seção 4.

5.1 CONFIGURAÇÃO DOS PARÂMETROS

Para aplicar a estratégia proposta na resolução dos estudos de caso apresentados, inicialmente foi necessário determinar o tipo de rede neural que será utilizado. Foi selecionada a topologia de camada oculta única, utilizando função de ativação sigmóide e treinada com o algoritmo Máquina de Aprendizado Extremo.

Em seguida, deve-se determinar as variáveis que servirão de entrada para a rede neural. No primeiro estudo de caso, como se trata de um problema de modelagem, é necessário selecionar o conjunto de termos candidatos para o modelo do conversor buck. Este conjunto está apresentado na Figura 9 (Oliveira; Leandro, 2019). Já no segundo estudo de caso, o problema consiste na previsão da resistência à compressão de SCC com adição de fibras, o que torna necessário indicar quais materiais servirão como entrada. No caso deste material as entradas são: cimento, agregado miúdo (areia),

agregado graúdo, cinzas volantes, fibras de vidro, fibras de polipropileno, água, super plastificante, aditivo modificador de viscosidade (VMA).

Figura 9 – Conjunto de termos candidatos para entrada da rede neural do conversor buck.

1º	2º	3º	4º	5º	6º	7º	8º
$y(k-1)$	$y(k-2)$	$y(k-3)$	$u(k-1)$	$u(k-2)$	$u(k-3)$	$y^2(k-1)$	$y(k-1)\,y(k-2)$

9º	10º	11º	12º	13º	14º
$y(k-1)\,y(k-3)$	$y(k-1)\,u(k-1)$	$y(k-1)\,u(k-2)$	$y(k-1)\,u(k-3)$	$y(k-2)\,y(k-3)$	$y^2(k-2)$

15º	16º	17º	18º	19º	20º
$y(k-2)\,u(k-1)$	$y(k-2)\,u(k-2)$	$y(k-2)\,u(k-3)$	$y^2(k-3)$	$y(k-3)\,u(k-1)$	$y(k-3)\,u(k-2)$

21º	22º	23º	24º	25º	26º	27º
$y(k-3)\,u(k-3)$	$u^2(k-1)$	$u(k-1)\,u(k-2)$	$u(k-1)\,u(k-3)$	$u^2(k-2)$	$u(k-2)\,u(k-3)$	$u^2(k-3)$

Fonte: Oliveira e Leandro (2019).

Além das entradas da rede neural, os algoritmos deverão selecionar a quantidade de neurônios na camada oculta da rede. Este número irá variar de 1 a 30 e o valor ideal será determinado pelos algoritmos, através da execução de 10 simulações para cada quantidade de neurônios testada e escolhendo aquela quantidade que apresentar melhor desempenho para a rede.

Para o tamanho das populações dos algoritmos evolutivos foi adotado o valor de 50. O número máximo de gerações adotado foi o valor de 100, considerado como critério de parada do algoritmo.

Valores de parâmetros específicos de cada algoritmo foram selecionados heuristicamente através de testes independentes. Os melhores valores encontrados para estes parâmetros estão apresentados na Tabela 2.

Tabela 2 – Parâmetros dos algoritmos evolutivos

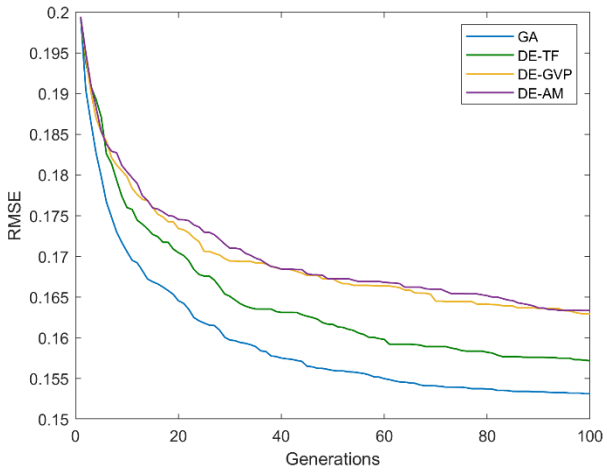
Metaheurística	**Parâmetro**	**Valor**
GA	t_{sn}	0,6
	t_c	0,7
	p_m	0,05
DE	F_m	0,7
	C_r	0,6
	Estratégia	DE/rand/1/bin

Fonte: Os autores (2024).

5.2 RESULTADOS PARA A MODELAGEM DO CONVERSOR BUCK

Utilizando as configurações apresentadas na seção 5.1, a estratégia apresentada na seção 3 foi aplicada para cada uma das 4 opções de técnicas listadas na Tabela 1, sendo realizado um total de 30 simulações para cada técnica. As curvas de convergência média, resultantes das 30 simulações de cada técnica, podem ser observadas na Figura 10. Já os resultados estatísticos do *RMSE* para todas as técnicas estão apresentados na Tabela 3, sendo que o valor médio indicado corresponde aos apresentados na Figura 10.

Figura 10 – Curvas de convergência média das 4 técnicas aplicadas na modelagem do conversor buck.



Fonte: Os autores (2024).

Tabela 3 – Valores de RMSE alcançados pelas técnicas na modelagem do conversor buck

Técnica	Mínimo	Médio	Máximo	Desvio Padrão
GA	**0,14884**	**0,15314**	**0,15640**	**0,0018789**
DE-TF	0,15357	0,15718	0,16403	0,0020830
DE-GVP	0,15381	0,16294	0,16807	0,0035657
DE-AM	0,15605	0,16336	0,16950	0,0033986

Fonte: Os autores (2024).

Analisando a Figura 10 e a Tabela 3, é possível observar que o GA foi a técnica que alcançou o melhor desempenho médio (menor valor médio de *RMSE*) para o problema de modelagem do conversor *buck*, com o valor de 0,15314. Em segundo lugar ficou a técnica DE-TF, com um *RMSE* médio 2,64% maior que o alcançado pelo GA. As técnicas com piores desempenhos médios foram DE-GVP e DE-AM, com *RMSE* médio 6,40% e 6,67% maiores que o do GA, respectivamente. Em relação às velocidades de convergência, todas as 4 técnicas tiveram desempenho semelhante, sem que nenhuma parasse em mínimos locais.

A partir destas considerações, é possível verificar que, como a maior parte dos parâmetros determinados para a rede estão em formato binário (entradas), o GA apresentou um melhor desempenho na determinação de tais parâmetros, pois esta técnica é um algoritmo nativamente binário. Foi possível observar também que a escolha da codificação binária influenciou no desempenho da DE, sendo a codificação TF aquela que proporcionou à DE o seu melhor desempenho médio.

Observando, agora, a coluna de “Mínimo” da Tabela 3, é possível observar os valores de *RMSE* do melhor modelo encontrado por cada técnica. Estes modelos são consequência do melhor conjunto de parâmetros (entradas e neurônios) que cada técnica selecionou para a rede neural. Estes parâmetros estão apresentados na Tabela 4.

Tabela 4 – Parâmetros selecionados pelas técnicas no problema de modelagem do conversor buck

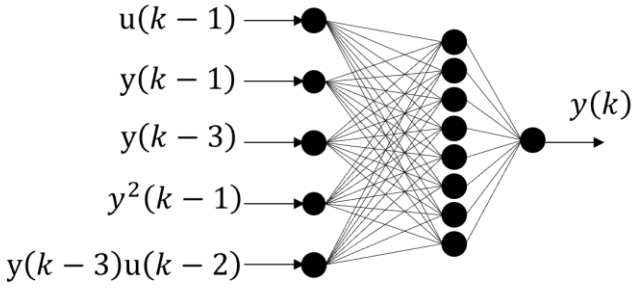
Técnica	Nº de entradas	Termos escolhidos	Nº de neurônios	RMSE do modelo
GA	**5**	**1º, 3º, 4º, 7º e 20º**	**8**	**0,14884**
DE-TF	13	1º, 3º, 4º, 6º, 7º, 9º, 10º, 15º, 16º, 17º, 18º, 23º e 24º	9	0,15357
DE-GVP	15	1º, 3º, 4º, 6º, 7º, 9º, 10º, 15º, 18º, 19º, 20º, 21º, 23º, 24º e 27º	15	0,15381
DE-AM	9	1º, 3º, 4º, 7º, 9º, 11º, 13º, 19º e 20º	13	0,15605

Fonte: Os autores (2024).

Observa-se, pelas Tabelas 3 e 4, que o menor valor de *RMSE* encontrado foi de 0,14884, que representa o *RMSE* do melhor modelo encontrado em todo o estudo. Desta forma, verifica-se que o GA, além de ter obtido um melhor comportamento médio em termos de convergência, encontrou um conjunto de parâmetros que gerou o melhor modelo dentre todos aqueles testados nas simulações. Vale ressaltar ainda que o conjunto de parâmetros indicados pelo GA, resulta na rede de menor complexidade, em comparação com os demais parâmetros indicados pelas outras técnicas nos seus melhores cenários.

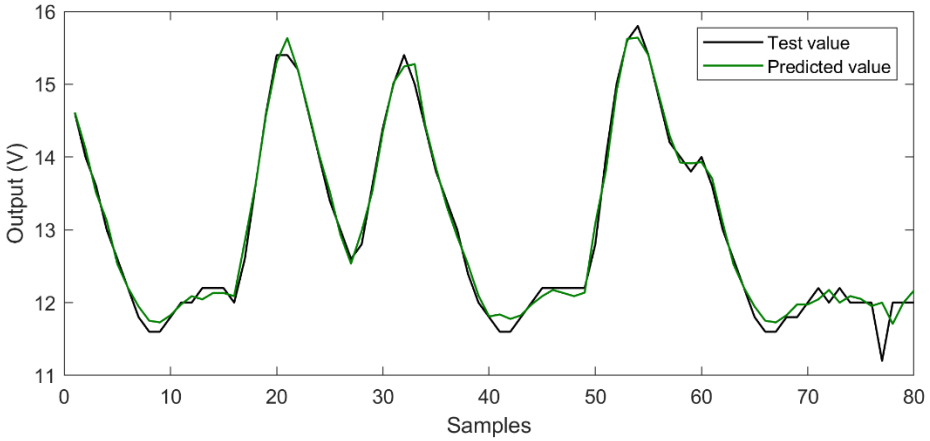
A rede neural parametrizada pelo GA apresenta apenas 8 neurônios na camada oculta e 5 entradas, que recebem 3 termos de ordem 1 e 2 termos de ordem 2. Esta rede neural está apresentada na Figura 11 e a simulação do modelo obtido com a mesma está apresentado na Figura 12.

Figura 11 – Rede neural configurada com os parâmetros selecionados pelo GA para realizar a modelagem do conversor buck



Fonte: Os autores (2024).

Figura 12 – Predição de um passo à frente do modelo obtido com a rede neural da Figura 11



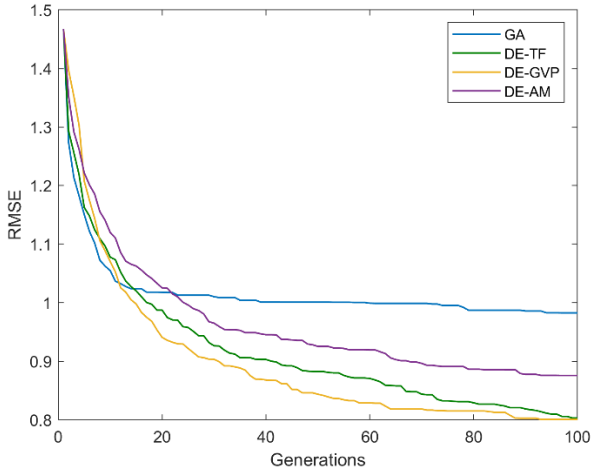
Fonte: Os autores (2024).

Verifica-se, pela Figura 12 que o modelo obtido com a rede neural apresentada na Figura 11 representa bem o comportamento dinâmico do sistema real, ou seja, além de apresentar o menor *RMSE* dentre todos os modelos obtidos, sua validação por meio da simulação também reforça a qualidade do modelo.

5.3 RESULTADOS PARA A PREVISÃO DA RESISTÊNCIA À COMPRESSÃO DE SCC COM ADIÇÃO DE FIBRAS

A estratégia proposta foi também aplicada no problema de previsão da resistência à compressão de SCC com adição de fibras, utilizando as configurações apresentadas na seção 5.1, e para cada uma das 4 opções de técnicas listadas na Tabela 1, sendo realizado um total de 30 simulações para cada técnica. As curvas de convergência média, resultantes das 30 simulações de cada técnica, podem ser observadas na Figura 13. Já os resultados estatísticos do *RMSE* para todas as técnicas estão apresentados na Tabela 5, sendo que o valor médio indicado corresponde aos apresentados na Figura 13.

Figura 13 – Curvas de convergência média das 4 técnicas aplicadas ao problema de previsão da resistência à compressão de SCC com adição de fibras.



Fonte: Os autores (2024).

Tabela 5 – Valores de RMSE alcançados pelas técnicas aplicadas ao problema de previsão da resistência à compressão de SCC com adição de fibras.

Técnica	Mínimo	Médio	Máximo	Desvio Padrão
GA	0,76828	0,98267	1,14150	0,092832
DE-TF	0,73032	0,80330	0,87576	0,039255
DE-GVP	**0,73866**	**0,80071**	**0,88993**	**0,034929**
DE-AM	0,73840	0,87581	0,97690	0,053924

Fonte: Os autores (2024).

Analisando a Figura 13 e a Tabela 5, é possível observar que a DE-GVP foi a técnica que alcançou o melhor desempenho médio (menor valor médio de *RMSE*) para o problema de previsão da resistência à compressão de SCC, com o valor de 0,80071. Em segundo lugar ficou a técnica DE-TF, com um *RMSE* médio apenas 0,82% maior que o alcançado pela DE-GVP. As técnicas com piores desempenhos médios foram DE-AM e GA, com *RMSE* médio 9,38% e 22,72% maiores que o da DE-GVP, respectivamente. Em relação às velocidades de convergência, as técnicas baseadas em DE apresentaram desempenhos semelhantes, sem que nenhuma parasse em mínimos locais. O GA, por sua vez apresentou maior velocidade de convergência no início das simulações, mas ficou preso em muitos mínimos locais a partir da 10ª geração.

A partir destas considerações, é possível verificar que, embora a maior parte dos parâmetros determinados para a rede estejam em formato binário (entradas), neste problema específico a técnica que apresentou melhores resultados foi a DE com codificação binária. Foi possível observar também que a escolha da codificação binária influenciou no desempenho da DE, sendo a codificação GVP aquela que proporcionou à DE o seu melhor desempenho médio.

Observando, agora, a coluna de "Mínimo" da Tabela 5, é possível observar os valores de *RMSE* do melhor modelo encontrado por cada técnica. Estes modelos são consequência do melhor conjunto de parâmetros (entradas e neurônios) que cada técnica selecionou para a rede neural. Estes parâmetros estão apresentados na Tabela 6.

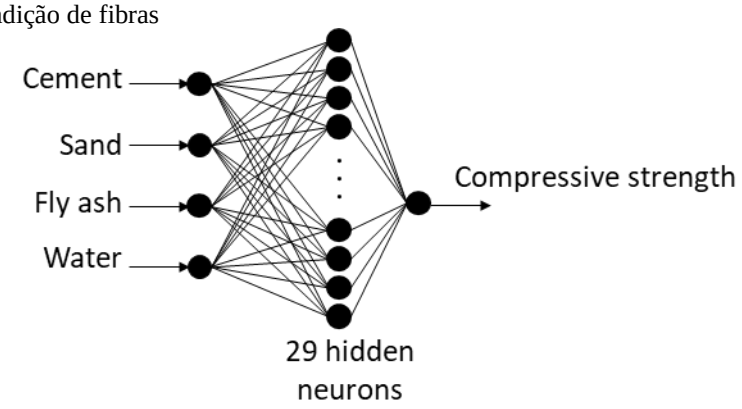
Tabela 6 – Parâmetros selecionados pelas técnicas no problema de previsão da resistência à compressão de SCC com adição de fibras

Técnica	Nº de entradas	Entradas escolhidas	Nº de neurônios	RMSE do modelo
GA	4	cimento, areia, cinzas volantes, água	30	0,76828
DE-TF	**4**	**cimento, areia, cinzas volantes, água**	**29**	**0,73032**
DE-GVP	5	cimento, areia, cinzas volantes, água, VMA	30	0,73866
DE-AM	4	cimento, areia, cinzas volantes, água	30	0,73840

Fonte: Os autores (2024).

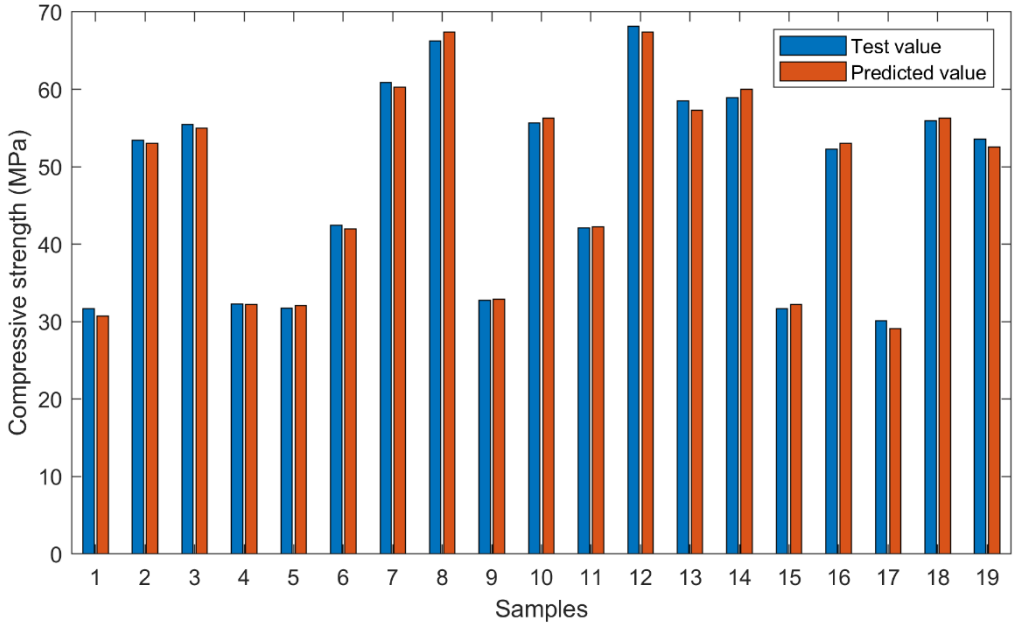
Observa-se, pelas Tabelas 5 e 6, que o menor valor de *RMSE* encontrado foi de 0,73032, que representa o *RMSE* do melhor modelo encontrado em todo o estudo. Este modelo possui como entradas cimento, areia, cinzas volantes e água e 29 neurônios na camada oculta, sendo estes parâmetros selecionados pela técnica DE-TF. Apesar deste modelo apresentar o menor valor de RMSE, verifica-se que as outras técnicas também obtiveram modelos com parâmetros semelhantes a este, tanto nas entradas quanto no número de neurônios da camada oculta. A rede neural que representa o melhor modelo obtido está apresentada na Figura 14 e os valores previstos para a saída estão apresentados na Figura 15.

Figura 14 – Rede neural configurada com os parâmetros selecionados pela DE-TF para realizar a previsão da resistência à compressão de SCC com adição de fibras



Fonte: Os autores (2024).

Figura 15 – Previsão da resistência à compressão do modelo obtido com a rede neural da Figura 14



Fonte: Os autores (2024).

Verifica-se, pela Figura 15, que o modelo obtido com a rede neural apresentada na Figura 14 representa bem o comportamento do compósito desenvolvido, ou seja, além de apresentar o menor *RMSE* dentre todos os modelos obtidos, a previsão dos valores de resistência à compressão em valores próximos aos reais também reforça a qualidade do modelo.

6 CONCLUSÃO

Este trabalho apresentou um estudo comparativo entre algoritmos evolutivos aplicados ao problema de parametrização automática de redes neurais utilizadas em identificação de sistemas e previsão de resistência à compressão de SCC com adição de fibras. Dois tipos de algoritmos foram utilizados: os que originalmente manipulam soluções binárias e aqueles projetados para trabalhar com valores contínuos e, por isso, precisam de uma codificação para manipular soluções no espaço de busca binário.

O algoritmo evolutivo de soluções binárias aplicado foi o Algoritmo Genético (GA) e o algoritmo evolutivo de soluções contínuas utilizado foi a Evolução Diferencial (DE). Para implementar as versões binárias da DE, foram aplicadas as codificações por Função de Transferência (TF), Prioridade de Maior Valor (GVP) e Modulação em Ângulo (AM).

Foram conduzidos dois estudos de caso utilizando redes neurais de camada oculta única, treinadas com Máquina de Aprendizado Extremo, sendo aplicadas na modelagem de um conversor *buck* e na previsão de resistência à compressão de SCC com adição de fibras, ambos os sistemas apresentando dados disponíveis na literatura.

No estudo de caso que abordou a modelagem do conversor buck, os resultados obtidos indicaram que o GA apresentou o melhor desempenho, localizando o melhor conjunto de parâmetros (entradas e neurônios) para a rede utilizada, gerando um modelo com menor complexidade e com menor RMSE. Outra percepção obtida dos resultados foi que a escolha da codificação binária influenciou significativamente no desempenho da DE, que obteve seu melhor desempenho médio quando combinada com a codificação TF. Foi possível verificar também que o melhor modelo obtido apresentou bons resultados quando submetido à validação do tipo simulação da predição de um passo à frente, adequando-se bem aos dados dinâmicos do sistema.

Já no outro estudo de caso, que abordou a previsão de resistência à compressão de SCC com adição de fibras, os resultados apontaram que os melhores parâmetros para a rede neural foram obtidos pela técnica DE-TF, gerando o modelo que apresentou o menor RMSE. Assim como no estudo de caso anterior, foi possível observar também que o melhor modelo obtido apresentou bons resultados quando submetido à validação através da comparação dos valores previstos de resistência à compressão com os valores reais, indicando que o modelo representou de forma satisfatória o comportamento do compósito apresentado.

Vale ressaltar que o estudo apresentado foi focado selecionar as entradas e a quantidade de neurônios de uma rede neural de camada oculta única, não priorizando, por exemplo, a escolha da função de ativação, pesos e bias iniciais, algoritmo de treinamento, dentre outros parâmetros possíveis. Além disso, o objetivo era obter modelos com o maior ajuste possível aos dados de teste dos sistemas, sem considerar outros tipos de características como complexidade do modelo e custo computacional.

6.1 TRABALHOS FUTUROS

As possibilidades de continuidade deste trabalho envolvem os seguintes tópicos, mas não limitadas a estes:

- Realizar a parametrização considerando outros parâmetros além das entradas e quantidade de neurônios;
- Testar a abordagem desenvolvida com outras topologias de redes neurais;
- Aplicar os métodos apresentados para obtenção de modelos de outros sistemas;
- Utilizar outros algoritmos evolutivos para reforçar as conclusões obtidas;
- Explorar outros métodos de codificação binária.

REFERÊNCIAS

AGUIRRE, L. A.; DONOSO-GARCIA, P. F.; SANTOS-FILHO, R. Use of a priori information in the identification of global nonlinear models-a case study using a buck converter. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, v. 47, n. 7, p. 1081–1085, jul. 2000.

BALASUBRAMANIAM, N; PADMANABAN, I. Prediction of the strength of self-compacting cementitious mix with glass fibre using machine learning. *JOURNAL OF CERAMIC PROCESSING RESEARCH*, v. 23, n. 6, p. 806–816, 2022.

CRAWFORD, Broderick *et al.* Putting continuous metaheuristics to work in binary search spaces. *Complexity*, v. 2017, 2017.

DAHI, Zakaria Abd El Moiz; MEZIOUD, Chaker; DRAA, Amer. Binary bat algorithm: on the efficiency of mapping functions when handling binary problems using continuous-variable-based metaheuristics. 2015, [S.l.]: Springer, 2015. p. 3–14.

DE CASTRO, Leandro Nunes; TIMMIS, Jonathan. *Artificial immune systems: a new computational intelligence approach*. [S.l.]: Springer Science & Business Media, 2002.

DORIGO, Marco. Optimization, learning and natural algorithms. *PhD Thesis, Politecnico di Milano*, 1992.

GASPAR-CUNHA, António; TAKAHASHI, Ricardo; ANTUNES, Carlos Henggeler. *Manual de Computação Evolutiva e Metaheurística*. Belo Horizonte: Editora UFMG, 2013.

GHOLAMZADEH-CHITGAR, Atefeh; BERENJIAN, Javad. Elman ANNs along with two different sets of inputs for predicting the properties of SCCs. *COMPUTERS AND CONCRETE*, v. 24, n. 5, p. 399–412, 2019.

HAYKIN, Simon. *Redes Neurais: Princípios e prática*. 2. ed. Porto Alegre: Bookman, 2001.

HOLLAND, John. Adaptation in natural and artificial systems: an introductory analysis with application to biology, control and artificial intelligence. *Control and artificial intelligence*, 1975.

HUANG, Guang-Bin; ZHU, Qin-Yu; SIEW, Chee-Kheong. Extreme learning machine: a new learning scheme of feedforward neural networks. 2004, [S.l.]: IEEE, 2004. p. 985–990.

KOZA, John R. *Genetic programming: on the programming of computers by means of natural selection*. London: MIT press, 1992. v. 1.

LIMA, Isaías; PINHEIRO, Carlos A M; SANTOS, Flávia A Oliveira. *Inteligência artificial*. [S.l.]: Elsevier Brasil, 2016. v. 1.

NAJM, Omar F; MOHAMED, Osama A; ALZARD, Mohammed H. The efficiency of statistical and artificial neural network techniques on evaluating and predicting compressive strength of sustainable SCC with basalt fibers. *Materials Today: Proceedings*, 2023.

OLIVEIRA, Ádamo Henrique Rocha De; LEANDRO, Gideon Villar. Algoritmos de Otimização Combinados Aplicados à Identificação de Sistemas. *XIV Conferência Brasileira de Dinâmica, Controle e Aplicações*, 2019.

PEÑARANDA, Juan Ramon Camarillo; SAAVEDRA-MONTES, Andrés Julián. Dynamic model

validation via error indexes. 2012, [S.l.]: IEEE, 2012. p. 1–6.

RECHENBERG, Ingo. Cybernetic solution path of an experimental problem. *Royal Aircraft Establishment Library Translation 1122*, 1965.

SAHA, P; PRASAD, M L V; KUMAR, P Rathish. Predicting strength of SCC using artificial neural network and multivariable regression analysis. *Computers and Concrete*, v. 20, n. 1, p. 31–38, 2017.

SCHWEFEL, H-P. Kybernetische Evolution als Strategie der experimentellen Forschung in der Stromungstechnik. *Diploma thesis, Technical Univ. of Berlin*, 1965.

SERRE, Denis. *Matrices: Theory and applications*. New York: [s.n.], 2002.

SILVA, Ivan Nunes Da; SPATTI, Danilo Hernane; FLAUZINO, Rogério Andrade. *Redes Neurais Artificiais Para Engenharia e Ciências Aplicadas*. 2. ed. São Paulo: Artliber, 2010.

STORN, Rainer; PRICE, Kenneth. Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces. *Journal of Global Optimization*, v. 11, n. 4, p. 341–359, 1997.

TAVAKOLI, H R *et al.* Prediction of combined effects of fibers and nanosilica on the mechanical properties of self-compacting concrete using artificial neural network. *Latin American Journal of Solids and Structures*, v. 11, n. 11, p. 1906–1923, 2014.

WILAMOWSKI, Bogdan M; IRWIN, J David. *Intelligent systems*. [S.l.]: CRC press, 2018.