CAPÍTULO 6

Avaliação do desempenho de algoritmos de treinamento de redes neurais artificiais e máquinas de vetores suporte para a previsão de geração fotovoltaica

6.1 INTRODUÇÃO

O Brasil é um país de clima predominantemente tropical com um grande potencial energético para a geração de energia elétrica por meio da energia solar. Avaliando somente as regiões Centro-Oeste e Sudeste, em média, o período de insolação é de 7 a 8 horas diárias, com uma irradiação solar anual média de 16 a 18 (MJ/m².dia) (Aneel, 2005).

A geração de energia elétrica via placas fotovoltaicas em determinados casos se torna uma alternativa viável quando se analisam comunidades distantes e de difícil acesso a linhas e redes de distribuição de energia elétrica (Shang; Srinivasan; Reindl, 2016).

Em 2012, a Aneel aprovou a Resolução Normativa – RN 482/2012 que estabeleceu as condições gerais para o acesso a micro e minigeração aos sistemas de distribuição de energia elétrica em baixa tensão, alimentados por fontes renováveis de energia (Junior *et al.*, 2015). Tem se visto, dessa maneira, um grande aumento na geração de energia elétrica via placas fotovoltaicas e o aumento na procura por essa tecnologia.

Dessa forma, o setor elétrico nacional tem experimentado uma expressiva elevação futura de geração de energia elétrica via placas fotovoltaicas.

Em determinadas épocas do ano, por exemplo, o período seco no Brasil, há uma elevação na demanda de energia elétrica, em virtude da intensa utilização de refrigeradores de ar. Uma alternativa técnica para restabelecer o equilíbrio carga-geração para as concessionárias e permissionárias de energia elétrica seria utilizar a energia fornecida pela micro e minigeração. Contudo, o perfil de carga dos consumidores residenciais se comporta de uma maneira bastante específica, conforme discutido no Capítulo 1. Durante o dia, período em que a geração de energia elétrica por PV é maior devido à insolação, os consumidores estão fora de suas residências. Assim, toda a energia elétrica produzida é injetada na rede, gerando um aumento considerável de potência ativa disponível na rede de distribuição originando, assim, uma variação no nível de tensão da rede, nesse caso um aumento de tensão (Descheemaeker *et al.*, 2014). O mesmo fenômeno ocorre de forma inversa ao anoitecer, ou seja, com a redução da produção e, consequentemente, da injeção de potência ativa na rede, pode-se ter um afundamento de tensão em pontos específicos do alimentador.

Assim, uma sobretensão ou uma subtensão ocasionada por micro e minigeradoras inseridas na rede de distribuição de energia elétrica pode acabar danificando e comprometendo o funcionamento de motores, equipamentos eletrônicos, eletrodomésticos e a própria estrutura da rede no que tange às capacidades de distribuição dos condutores elétricos presentes. Nesse sentido, estudos vêm sendo feitos para caracterizar (Salazar; Carrión, 2015), modelar (Conceição; Silva, 2015), simular (Netto, 2015) e planejar o comportamento e a operação de sistemas de distribuição com geração distribuída (Barin *et al.*, 2012).

Dessa maneira torna-se imprescindível o estudo desses fatores, de forma a desenvolver meios que possam prever e eventualmente evitar esses tipos de efeitos.

Introduzido por Vapnik *et al.*, o *support vector machine* (SVM) é um tipo de aprendizagem computacional derivada da teoria estatística de aprendizagem e da teoria da dimensão Vapnik-Chervonenkis (VC). Baseado no princípio da minimização estrutural de risco, o SVM tem um bom desempenho em generalização e tem sido utilizado em larga escala em diversas aplicações, como para a resolução de classificação, reconhecimento de padrões, caracterização de textos, regressão etc. (Qizhong, 2007; Han; Dang; Ren, 2012; Tsang; Yeung; Chan, 2003).

Neste capítulo, é apresentada uma análise de desempenho de algoritmos de treinamento utilizados em redes neurais artificiais, que considera a avaliação de impactos técnicos como o desempenho, o tempo de processamento e respostas desses algoritmos quando utilizados para previsão temporal utilizando-se grande quantidade de entrada de dados, com dados de temperatura da placa fotovoltaica e irradiação

solar variáveis no tempo e ao longo de um horizonte determinado. O objetivo é encontrar uma ferramenta que leve ao melhor desempenho da rede, minimizando ou maximizando cada aspecto técnico segundo o interesse de se obterem respostas de processamento mais rápidas e com menores erros, conduzindo a soluções mais reais e diversificadas para a tomada de decisões, conhecidas como soluções otimizadas. Dada a natureza combinatória desse problema, que requer uma ferramenta de otimização capaz de manipular múltiplos objetivos, os impactos técnicos serão avaliados simultaneamente utilizando uma metodologia baseada no conceito do NARX (do inglês *nonlinear autoregressive with external input*). São comparados 7 algoritmos de treinamento disponibilizados na *toolbox* do MatLab®: BFGS Quasi-Newton (BFG), Bayesian Regularization (BR), Conjugate Gradient with Powell/Beale Restarts (CGB), Polak-Ribiére Conjugate Gradient (CGP), Resilient Backpropagation (RP), Scaled Conjugate Gradient (SCG) e Levenberg-Marquardt (LM).

Os diversos algoritmos de treinamento da rede neural são subsequentemente testados e o desempenho de cada um deles é testado e discutido e, posteriormente, o que obteve melhor desempenho é comparado com o desempenho da técnica denominada Máquinas de Vetores Suporte (SVM, do inglês *support vector machines*).

6.2 O FUNCIONAMENTO DA FOTOCÉLULA

Para que se compreenda e justifique a escolha das variáveis de entrada utilizadas para treinar os classificadores, esta seção discute sobre quais os fatores externos que afetam o desempenho do sistema de geração fotovoltaico.

6.2.1 O efeito da irradiação na fotocélula

A corrente emitida por uma fotocélula é diretamente relacionada à irradiação solar em sua superfície. A corrente de curto-circuito é linearmente proporcional a essa irradiação.

Já a tensão de circuito aberto se refere à tensão através do diodo interno da junção *p-n* quando a fotocorrente gerada passa por ele. A dependência da tensão de circuito aberto à irradiação corresponde à dependência que a tensão da junção *p-n* tem com a fotocorrente, de forma que quando a irradiação é baixa – sendo também a fotocorrente gerada – a tensão de circuito aberto é baixa.

A Figura 6.1 demonstra como é o formato da curva de corrente por tensão para diversas irradiações.

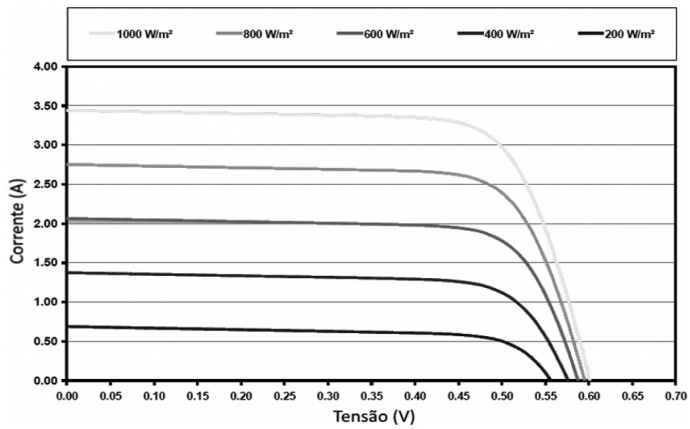


Figura 6.1 Curva corrente por tensão de uma fotocélula com diferentes irradiações.
Fonte: elaborada pelo autor.

6.2.2 O efeito da temperatura na fotocélula

Em semicondutores, a largura de banda de energia diminui com o aumento da temperatura. Em uma fotocélula, com a temperatura mais alta, mais fótons têm energia suficiente para criar pares *p-n*.

A tensão de circuito aberto é fortemente dependente da temperatura decrescendo substancialmente e fazendo com que a corrente de curto-circuito cresça levemente (Skoplaki; Boudouvis; Palyvos, 2008). A Figura 6.2 demonstra o efeito da temperatura em uma fotocélula.

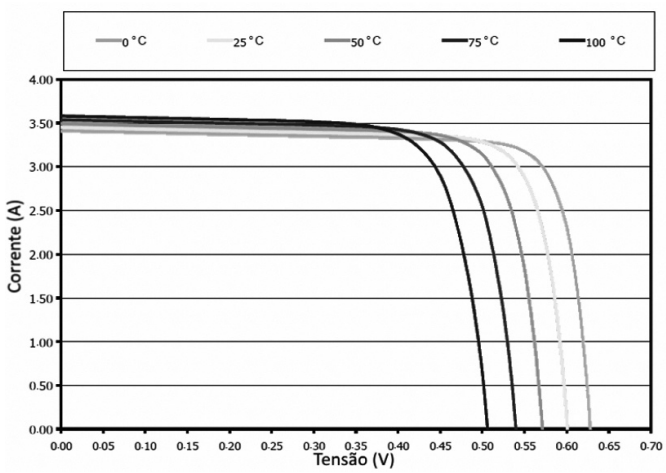


Figura 6.2 Curva de corrente por tensão de uma fotocélula com diferentes temperaturas.
Fonte: elaborada pelo autor.

6.3 REDES NEURAIS ARTIFICIAIS

O funcionamento do cérebro humano vem sendo alvo de estudos científicos durante muitos anos e ainda fascina pesquisadores, pelo fato de ser um órgão extremante complexo em seu funcionamento. A forma como o cérebro reconhece e identifica uma fisionomia ou um som é extremamente complexa. Porém, a compreensão do funcionamento de cérebro pode ser de grande valia para estudos computacionais quando estes são traduzidos em linguagens de programação.

Segundo Simon (1999), uma rede neural é um processador maciço paralelamente distribuído, constituído de unidades de processamento simples que têm a propensão natural para armazenar conhecimento experimental e torná-lo disponível para o uso. Ela se assemelha ao cérebro em dois aspectos:

- O conhecimento é adquirido pela rede a partir de seu ambiente por meio de um processo de aprendizagem.
- Forças de conexão entre neurônios, conhecidas como pesos sinápticos, são utilizadas para armazenar o conhecimento adquirido.

O processo de aprendizagem é realizado por meio de um algoritmo de aprendizagem cujo objetivo é mudar os pesos sinápticos de forma que a rede neural artificial se adapte e alcance o objetivo desejado (Simon, 2007).

Devido à sua disseminação e a seu desempenho para a resolução de problemas com o auxílio da computação, as redes neurais artificiais têm sido amplamente utilizadas, como em Alegre (2007), o qual utiliza uma rede neural para identificar uma pessoa por meio de sua voz. No que tange ao setor elétrico, as redes neurais artificiais foram aplicadas à previsão de carga em curto prazo em redes de distribuição (Ferro *et al.*, 2009), bem como para a previsão de eficiência de geração de energia elétrica por meio de placas fotovoltaicas (Rampinelli *et al.*, 2010) e como ferramenta auxiliar para a quantificação de variações de tensão de curta duração (Machado *et al.*, 2009).

6.4 DADOS SOLARES DA REGIÃO DE COLETA DOS DADOS

Foram coletados dados climáticos de irradiância (W/m^2), temperatura ambiental (°C), potência medida (W), temperatura da placa solar (°C) e as horas dos dias (h) para o mês de outubro de 2015 na cidade de Uberlândia-MG (Brasil). Esses dados serviram como dados de entrada para a preparação da rede.

Uberlândia se localiza no Estado de Minas Gerais na região conhecida como Triângulo Mineiro, próxima ao Estado de Goiás, caracterizando-se por ser um município com clima tropical com temperatura média anual de 21,5°C e radiação média anual de 6 kWh/m^2/dia. Nesse conjunto, a vegetação característica é o cerrado e suas

variáveis (UFU, 2016). A Figura 6.3 ilustra a radiação solar média diária anual na cidade de Uberlândia.

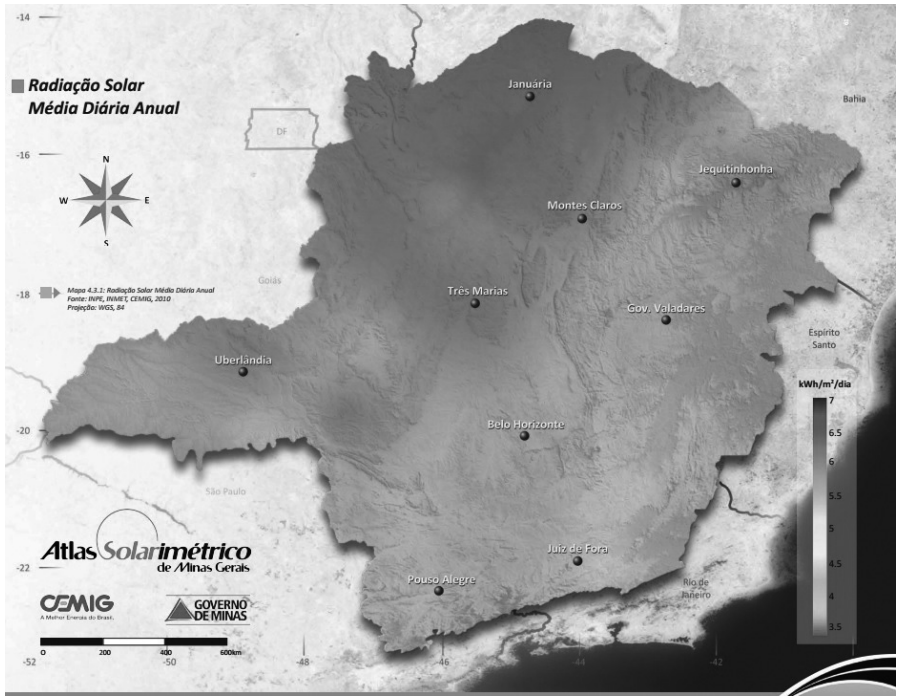


Figura 6.3 Atlas solarimétrico de Minas Gerais: radiação média diária anual.
Fonte: Cemig, 2012.

A Figura 6.4 mostra a potência gerada por uma placa fotovoltaica de acordo com os dados coletados em um período de 30 dias. Os dados de potência ativa gerada, de acordo com os dados de entrada citados anteriormente, serviram como dados de objetivo para a preparação da rede neural artificial.

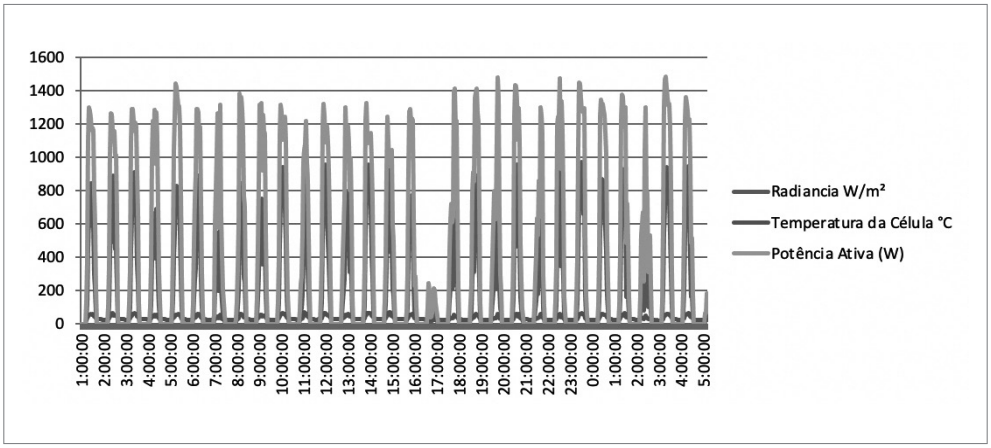


Figura 6.4 Irradiância, temperatura da célula e potência gerada.
Fonte: elaborada pelo autor.

Foram utilizadas 7 placas solares conectadas em série do tipo JT235PC de silício policristalino, de fabricação da empresa Jetion Solar (Jetion Solar, 2015), e suas características são mostradas na Tabela 6.1.

Tabela 6.1 Características elétricas da placa solar

Desempenho elétrico nas condições padrões de irradiância de 1.000 W/m², AM=1,5 e temperatura da célula de 25°C (STC)	
Potência máxima: $P_{máx}$ (W)	235 W (0/+5%)
Tensão de potência máxima: V_{mpp} (V)	30,5 V
Corrente de potência máxima: I_{mpp} (A)	7,71 A
Tensão de circuito aberto: V_{OC} (V)	37 V
Corrente de curto-circuito: I_{CC} (A)	8,4 A
Tensão máxima (V)	1000 V
Coeficiente de temperatura de V_{OC}	- 0,4049%/°C
Coeficiente de temperatura de I_{CC}	0,0825%/°C
Desempenho elétrico a 800 W/m², NOCT 20°C, AM=1,5, velocidade do vento 1 m/s	
Potência máxima: $P_{máx}$ (W)	172 W
Tensão de potência máxima: V_{mpp} (V)	27,7 V
Corrente de potência máxima: I_{mpp} (A)	6,2 A
Tensão de circuito aberto: V_{OC} (V)	33,9 V
Corrente de curto-circuito: I_{CC} (A)	6,8 A

Fonte: Jetion Solar, 2015.

6.5 ESTRUTURA DA REDE NEURAL ARTIFICIAL

Para a preparação da rede neural artificial, foi utilizado o software MatLab. Esse software disponibiliza em sua *toolbox* modelos de redes neurais artificiais, das quais podemos citar: *fitting tools*; *pattern-recognition tool*; *clustering tool*; *time series tool*.

Para este capítulo, foi utilizada a ferramenta para a previsão temporal de dados *time series tool*.

A ferramenta escolhida ainda dá a opção de 3 tipos de soluções para problemas: a NARX; a NAR (do inglês *nonlinear auto-regressive*); e a *nonlinear input-output*. Foi escolhida a solução NARX.

A rede neural NARX pode ser expressa por (6.1):

$$y_{t+1} = f\left(x_t, x_{t-1}, \ldots, x_{t-n}, y_t, y_{t-1}, \ldots y_{t-n,}\right) \tag{6.1}$$

onde o próximo valor do sinal de saída, y_{t+1}, é regredido utilizando os valores anteriores medidos y_t e y_{t-1}, e sinais de entradas u_t e u_{t-1}, (isto é, temperatura ambiental, temperatura da célula, tempo e potência medida). A função *f* representa a rede neural, em que os pesos de cada conexão na rede são treinados pelos algoritmos de treinamentos.

A estrutura da rede neural NARX é ilustrada na Figura 6.5.

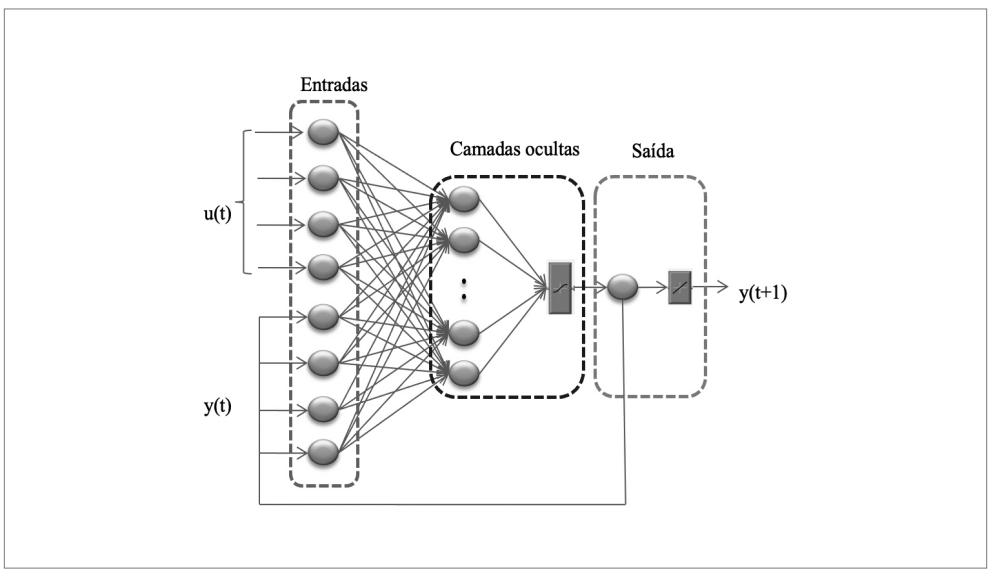


Figura 6.5 Estrutura da rede NARX.
Fonte: elaborada pelo autor.

Para cada sinal de entrada, p, está associado um peso, w, para fortalecer ou empobrecer o sinal de entrada. A RNA calcula o sinal de entrada de rede e usa um ativador de função (função de transferência) para calcular o sinal de saída, "o", dada a entrada da rede. A força do sinal de saída é ainda influenciada por um valor limiar, b, também referida como *bias* (Andries, 2007; Shayeghi; Shayanfar; Malik, 2007). Essa relação pode ser expressa por (6.2) (Tapakis; Michaelides; Charlambides, 2016):

$$x_j = \sum_{i=1}^{n} w_{ij} p_i + b_j \tag{6.2}$$

Foram testadas as funções de transferência: Linear, Log-Sigmoid e Tang-Sigmoid. A que obteve o melhor desempenho para o problema apresentado foi a Tangent-Sigmoid. Devido à limitação e ao foco deste capítulo, essa análise não é apresentada.

O número de neurônios utilizados na camada oculta foi de 10 neurônios. Pode-se aumentar o número de neurônios acima de 20, uma vez que grandes quantidades de neurônios ocultos dão maior flexibilidade a rede neural, pois assim a rede terá mais parâmetros para otimizar. Porém, isso resulta em um tempo de execução maior da simulação da rede.

6.6 ALGORITMOS DE TREINAMENTO

A preparação da rede necessita de um algoritmo de treinamento para a aprendizagem da rede neural artificial. O aprendizado consiste no ajuste dos pesos e limares da rede neural até que um certo critério seja satisfeito.

Neste trabalho, foram comparados 7 algoritmos de treinamento disponibilizados na *toolbox* do MatLab, os quais são: BFGS Quasi-Newton (BFG), Bayesian Regularization (BR), Conjugate Gradient with Powell/Beale Restarts (CGB), Polak-Ribiére Conjugate Gradient (CGP), Resilient Backpropagation (RP), Scaled Conjugate Gradient (SCG) e Levenberg-Marquardt (LM). As descrições desses algoritmos podem ser encontradas em Forouzanfar *et al.*, 2010; Er; Liu, 2009; Li; Wang, 2009; Jang; Sun; Mizutani, 1997; Riedmiller; Braun, 1993; Lera; Pinzolas, 1998; e Mellit; Pavan, 2010, respectivamente.

6.7 SVM

Considere o problema da separação de uma série de vetores de treinamento pertencentes a duas diferentes classes $(x_i, x_j), \ldots, (x_l, y_l)$, onde $x_i \in R^n$ é um vetor recurso e $y_i \in \{+1, -1\}$ é um rótulo de classe. De acordo com a teoria do SVM para classificação não linear, os dados originais são projetados em um determinado espaço de

alta funcionalidade dimensional H por um mapa não linear $\varnothing : R^n \to H$, então o problema da classificação não linear é transformado em uma classificação linear no espaço H (Qizhong, 2007; Han; Dang; Ren, 2012; Tsang; Yeung; Chan, 2003). Introduzindo a função de Kernel $K(x_i, x_j) = \left\langle \varnothing(x_i), \varnothing(x_j) \right\rangle$, não é necessário saber explicitamente a expressão de $\varnothing(\cdot)$. O correspondente problema de otimização de classificação não linear é dado por (6.3) e (6.4):

$$Min \frac{1}{2} \| w \|^2 + C \sum_{1}^{l} \xi_i \tag{6.3}$$

Sujeito a $y_i \left[w \cdot \varnothing(x) + b \right] \geq 1 - \xi_i$ (6.4)

$$\xi_i \geq 0, \quad i = 1, 2, \ldots l$$

onde C é o parâmetro de regularização que controla a troca entre a maximização da margem e a minimização do termo de erro de treinamento, e ξ_i é o fator de folga que é utilizado para relaxar as restrições de margem rígidas e a constante de regularização C > 0 (Lei; Gao; Ding, 2010). De acordo com o método de otimização de Lagrange e do princípio dual, as Equações (6.5) e (6.6) podem ser escritas como:

$$Max\, w(\alpha) = \sum_{i=i}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=i}^{l} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \tag{6.5}$$

Sujeito a $\sum_{i=i}^{l} \alpha_i y_i = 0$ (6.6)

$$\alpha_i \in [0, C], \quad i = 1, 2, \ldots, l$$

De (6.5), pode-se obter a otimização do hiperplano com a margem máxima (6.7):

$$f(x) = \sum_{av} \alpha_i y_i K(x_{i,} x) + b \tag{6.7}$$

Portanto, a função decisão baseada no SVM para classificação não linear no espaço interno é (6.8):

$$d(x) = sign\left(\sum_{av} \alpha_i y_i K\left(x_{i,} x\right) + b\right) \tag{6.8}$$

Para este trabalho, foi utilizada a função radial de Kernel (6.9):

$$\exp(-\left\|x - x_i\right\|^2 / \sigma^2 \tag{6.9}$$

6.8 PREPARAÇÃO DA REDE E ÍNDICES DE DESEMPENHO

Tratados os dados de entrada e de saída da rede neural escolhida, foi feito o treinamento dessa rede por meio dos 7 algoritmos citados anteriormente. A capacidade da rede utilizada para o treinamento foi 75%. Para validação e teste, ambos, 15%. Ao todo, entre dados de entrada e de objetivo, foram utilizados 2.820 dados para a preparação e o treinamento da rede. Os testes foram feitos em um PC com 3.40 GHz Intel Core i7A Pentium® e 8 GB de memória RAM.

Com relação ao treinamento da rede neural, uma das alternativas para resolver o problema da parada do treinamento é a técnica da validação cruzada. Em vez de definir o número exato de iterações de ajuste de pesos no treinamento, divide-se aleatoriamente o conjunto de dados em 3 subconjuntos: treinamento, validação e testes. Com isso, a cada iteração a rede é treinada, já com os pesos ajustados, é testada com o subconjunto de validação e o erro da predição é calculado ao final da iteração. A motivação é ajustar os pesos com os dados do subconjunto de treinamento e calcular o erro com os dados do subconjunto de validação, fornecendo, assim, dados diferentes à rede. Assim, o erro da validação cruzada começa alto, decresce até certo ponto e depois aumenta. Enquanto o erro da validação está diminuindo, a rede está generalizando, quando o erro começa a aumentar, ao mesmo tempo que o erro do treinamento continua a diminuir, a rede começa a decorar as entradas, perdendo a capacidade de generalização. Nesse momento, a rede deve parar o treinamento. Um exemplo de validação cruzada é ilustrado na Figura 6.6.

Como ferramenta utilizada para a validação dos dados, utilizou-se a regressão linear. Trata-se de um processo estatístico que ajuda a deduzir a relação entre um determinado número de variáveis dependentes e variáveis independentes. Essa análise é útil em estudos de dependência funcional entre fatores de entrada e saída, implicando que cada variável de entrada $(\mathrm{x}_1, \mathrm{x}_2, \mathrm{x}_3, ...)$ parcialmente determina o nível da variável de saída (y). Cada valor da variável independente x é associado com o valor da variável independente *y.* Uma explicação mais detalhada dessa ferramenta pode ser encontrada em Pulipaka; Mani; Kumar, 2016 e em İzgi *et al.*, 2012.

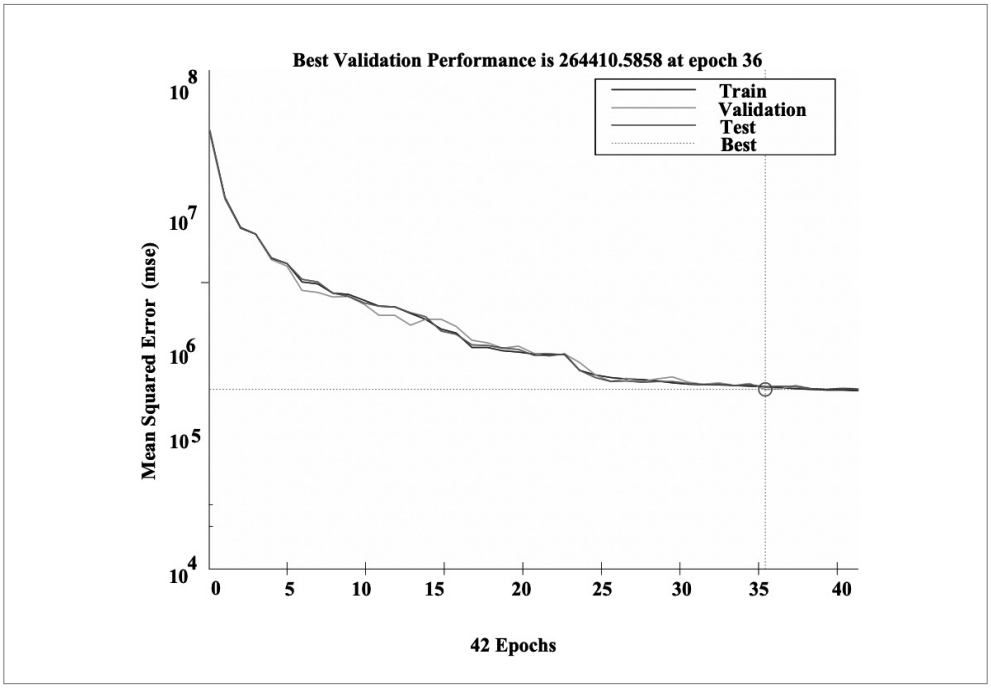


Figura 6.6 Exemplo de validação cruzada da rede neural para o algoritmo de treinamento SCG.
Fonte: elaborada pelo autor.

Outra ferramenta estatística utilizada para se obter o desempenho da rede neural em estudo é o erro médio percentual absoluto (Mape, do inglês *mean absolute percentage error*). O Mape é uma medida estatística da precisão de séries temporais, representando essa precisão em forma de porcentagem (Celik, 2011; Olatomiwa *et al.*, 2015; Bhardwaj *et al.*, 2013) e pode ser definida por (6.10):

$$MAPE(\%) = \left(\frac{1}{n} \sum_{i=1}^{n} \frac{y - x}{x} \right) \times 100 \tag{6.10}$$

sendo x os dados de entrada e y os dados previstos pela rede neural.

Por fim, também como ferramenta estatística a ser utilizada para a análise do desempenho dos algoritmos de treinamento, é o erro médio quadrático (RMSE, do inglês *root mean square error).* O RMSE dá a informação do desempenho em curto prazo da correlação "r", comparando a extensão do desvio do valor previsto a partir do valor real medido (Mentaschi *et al.*, 2013; Kaushika; Tomar; Kaushik, 2014; Bernecker *et al.*, 2014). A correlação "r" e RMSE são definidos por (6.11) e (6.12), respectivamente:

$$r = \frac{\sum_{i=1}^{n}\left(y_i - \overline{y}_i\right).\left(x_i - \overline{x}_i\right)}{\sqrt{\sum_{i=1}^{n}\left(y_i - \overline{y}_i\right).\sum_{i=1}^{n}\left(x - \overline{x}_i\right)}} \tag{6.11}$$

$$RMSE = \sqrt{\frac{\sum_{I=1}^{n}\left(y - x\right)^2}{n}} \tag{6.12}$$

sendo x e y os dados de entrada e previstos, respectivamente, $\overline{x}_i$ e $\overline{y}_i$ são os valores médios de x e y e, n é o número total de valores.

Quanto menores os valores do Mape e do RMSE, melhor é o desempenho da rede neural.

6.9 RESULTADOS E DISCUSSÕES

Após inseridos os dados de entrada (irradiação solar, temperatura ambiental e temperatura das placas solares) e os dados de objetivo (potência ativa medida), foi feito o treinamento, e o primeiro resultado analisado foi o número de iterações necessárias para que a rede fosse treinada e validada de acordo com o algoritmo de treinamento que estava sendo utilizado. Um maior número de iterações para validação, treinamento e teste resulta em mais tempo que a rede gasta para ser preparada. O tempo de treinamento e preparação da rede está diretamente ligado aos algoritmos de treinamento, uma vez que são as reduções dos erros promovidas por eles que servem de critério para a validação cruzada.

A Figura 6.7 mostra que o desempenho de alguns algoritmos pode ser afetado de acordo com a precisão requerida da aproximação desejada. A figura ilustra o número de iterações (*epochs*) versus o erro médio quadrático. Pode-se ver que o erro do algoritmo LM decai muito mais com as iterações do que os outros algoritmos analisados.

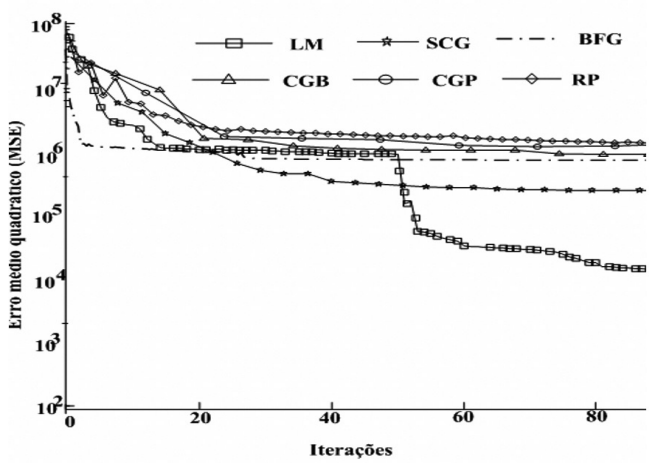
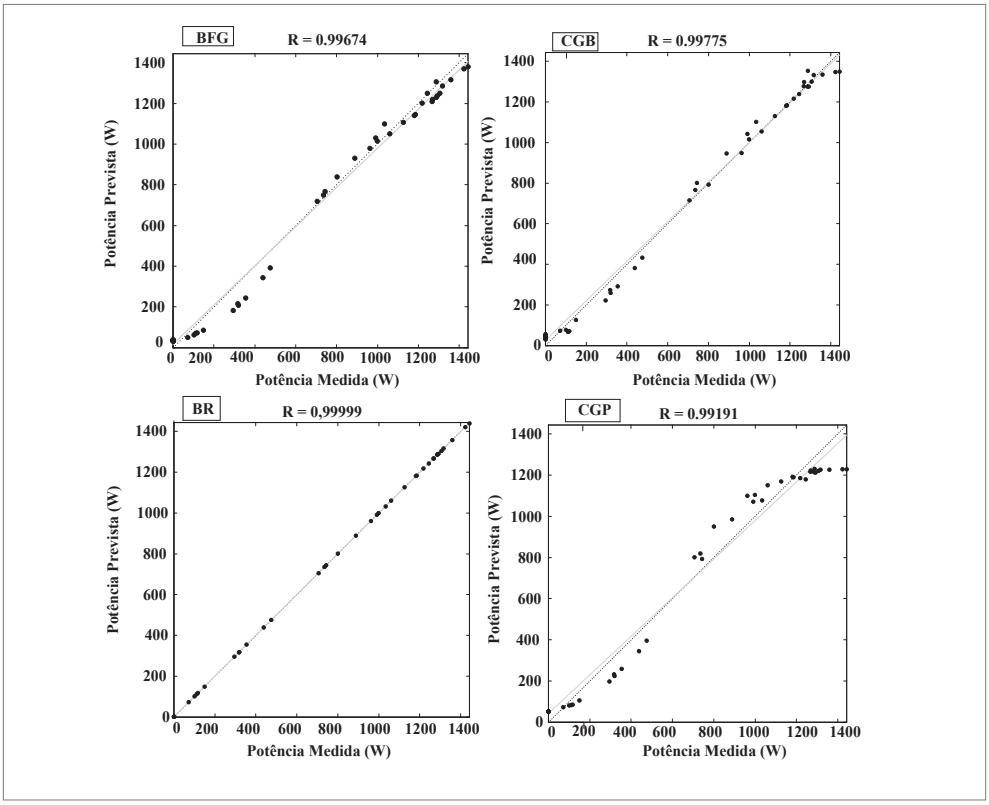


Figura 6.7 Desempenho dos algoritmos de treinamento.
Fonte: elaborada pelo autor.

O algoritmo BR, após 168 iterações, não foi validado, isopor isso, esse algoritmo não foi incluído na Figura 6.7. Por ordem de melhor desempenho para a validação cruzada dos algoritmos, o algoritmo que obteve o treino mais rápido foi o CGP, levando um total de 6 iterações para o treino, validação e teste; seguido do CGB com 21 iterações; SCG com 36 iterações, em terceiro lugar; o algoritmo RP necessitou de 75 iterações para o processo de treinamento; para o algoritmo LM, foram necessárias 134 iterações; o algoritmo BR necessitou de 168 iterações; e por último, com o tempo mais demorado, vem o algoritmo BFG, com 354 iterações para o processo.

O segundo índice a ser analisado é a regressão linear resultante entre os dados previstos e os medidos, de acordo com cada algoritmo de treinamento. Para isso, foi feita a previsão de potência gerada em um intervalo de 72 horas para valores de dados futuros, correspondentes aos dados de entrada utilizados para a preparação e o treinamento da rede. A Figura 6.8 ilustra as regressões obtidas.


BFG
R = 0.99674
CGB
R = 0.99775
BR
R = 0,99999
CGP
R = 0.99191
Potência Prevista (W)
Potência Medida (W)
0
200
400
600
800
1000
1200
1400

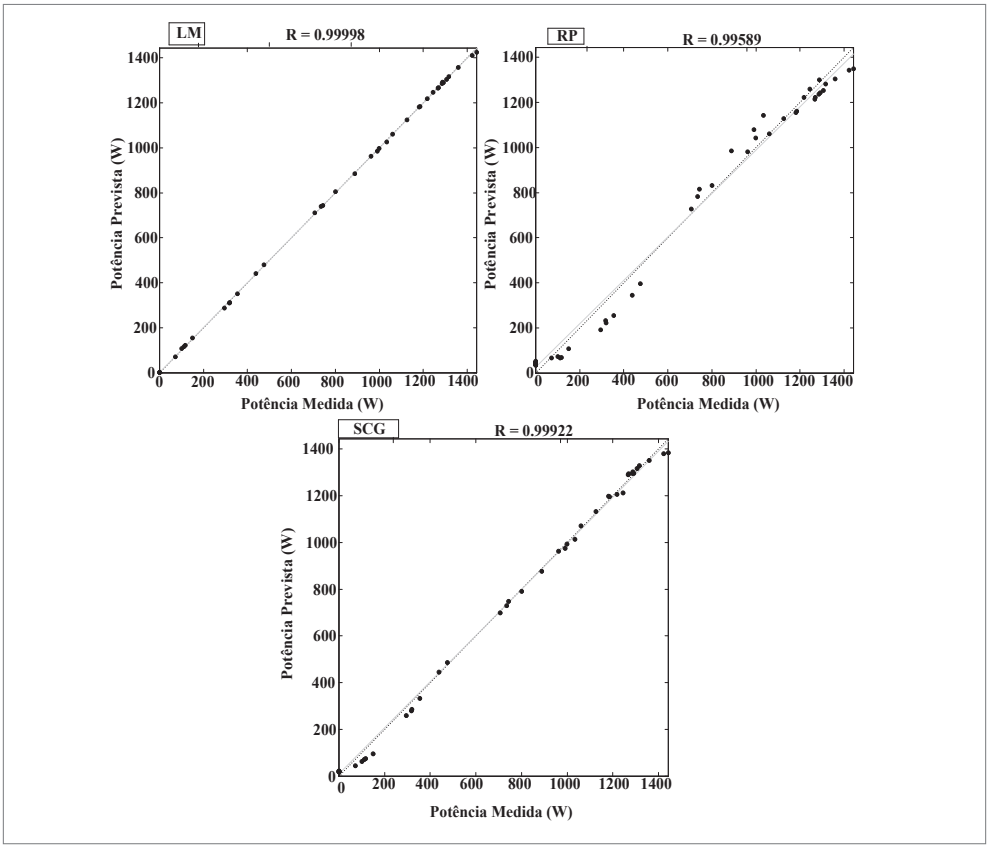


Figura 6.8 Regressão linear para os algoritmos de treinamento.
Fonte: elaborada pelo autor.

O algoritmo BR, apesar de não ter sido validado, apresentou a melhor regressão linear dentre todos os outros, com um valor de r = 0,9999; seguido do LM que apresentou uma regressão de r = 0,9998; o algoritmo SCG resultou em uma regressão de r = 0,9992; com uma regressão de 0,9977, vem o algoritmo CGB; com a quinta melhor regressão, tem-se o algoritmo 0,9967; em seguida vem o RP com uma regressão de r = 0,9958; e finalmente com a pior regressão linear dentre todos os algoritmos de treinamento avaliados, vem o algoritmo CGP com r = 0,9919.

Corroborando com o exposto anteriormente, as Figuras 6.9 a 6.15 ilustram a comparação entre os dados medidos e os dados previstos para os algoritmos BFG, BR, CGB, CGP, RP, SCG e LM, respectivamente.

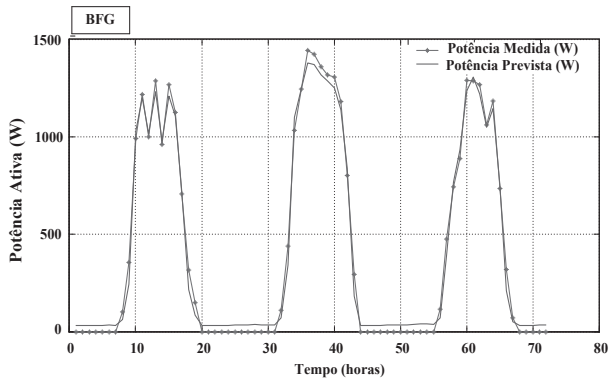


Figura 6.9 Potência medida (W) e potência prevista (W) em um intervalo de 72 horas para o algoritmo de treinamento BFG.
Fonte: elaborada pelo autor.

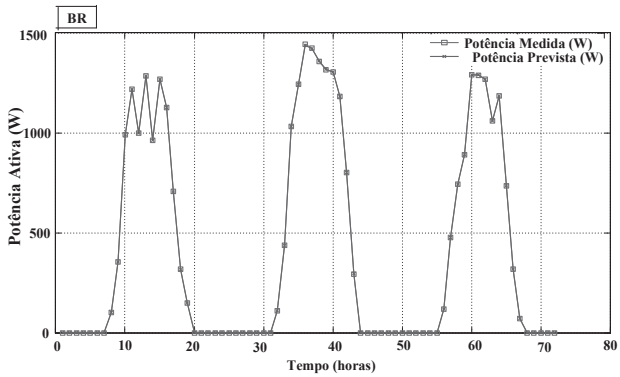


Figura 6.10 Potência medida (W) e potência prevista (W) em um intervalo de 72 horas para o algoritmo de treinamento BR.
Fonte: elaborada pelo autor.

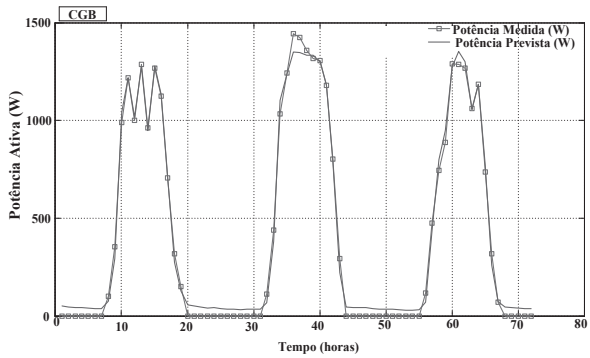


Figura 6.11 Potência medida (W) e potência prevista (W) em um intervalo de 72 horas para o algoritmo de treinamento CGB.
Fonte: elaborada pelo autor.

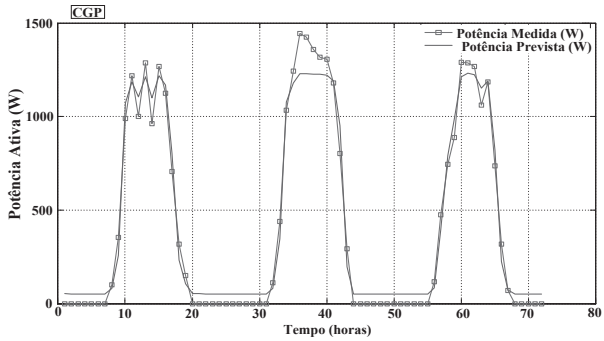


Figura 6.12 Potência medida (W) e potência prevista (W) em um intervalo de 72 horas para o algoritmo de treinamento CGP.
Fonte: elaborada pelo autor.

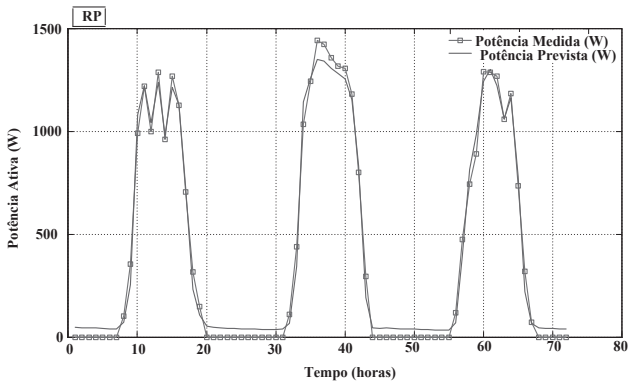


Figura 6.13 Potência medida (W) e potência prevista (W) em um intervalo de 72 horas para o algoritmo de treinamento RP.
Fonte: elaborada pelo autor.

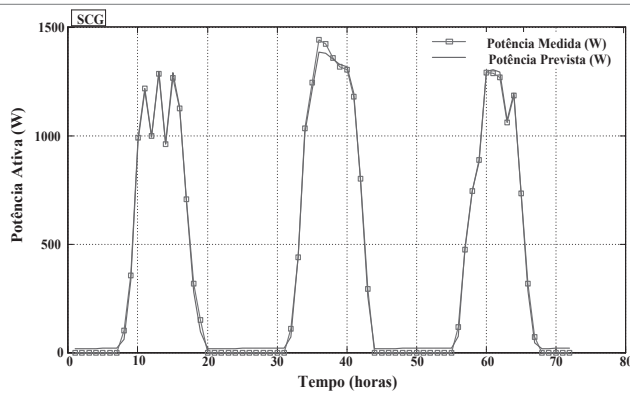


Figura 6.14 Potência medida (W) e potência prevista (W) em um intervalo de 72 horas para o algoritmo de treinamento SCG.
Fonte: elaborada pelo autor.

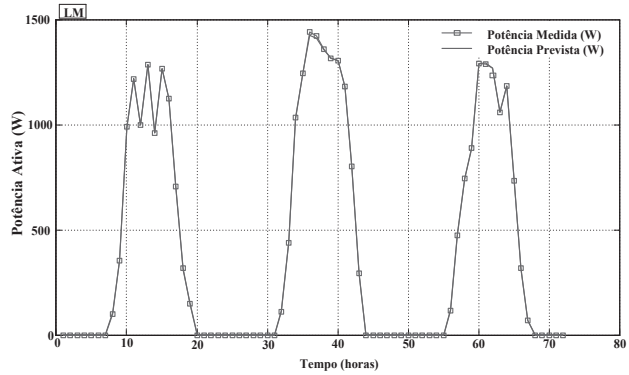


Figura 6.15 Potência medida (W) e potência prevista (W) em um intervalo de 72 horas para o algoritmo de treinamento LM
Fonte: elaborada pelo autor.

Como os últimos índices de desempenho a serem analisados, tem-se o Mape e o RMSE. O algoritmo de treinamento BR obteve os menores índices analisados, com um Mape de 0,02% e um RMSE de 0,11%. O LM apresentou um Mape de 0,31% e um RMSE de 0,74. Com um Mape de 5,28% e um RMSE de 6,13%, vem o CGB. Em seguida, com um Mape de 6,27% e um RMSE de 10,41%, encontra-se o algoritmo CGP. Para o algoritmo SCG, o Mape foi de 6,33% e o RMSE de 5,15%. O algoritmo de treinamento RP resultou um Mape de 7,85% e um RMSE de 8,76%. Analisando o algoritmo BFG, encontrou-se um Mape de 10,37% e um RMSE de 9,06%. É natural que os RMSE de alguns algoritmos sejam maiores do que os seus Mape, uma vez que no RMSE se tem uma soma quadrática, o que não subtrai os índices negativos da soma, ao contrário do que ocorre no Mape.

A Figura 6.16 ilustra os resultados dos Mape e RMSE obtidos.

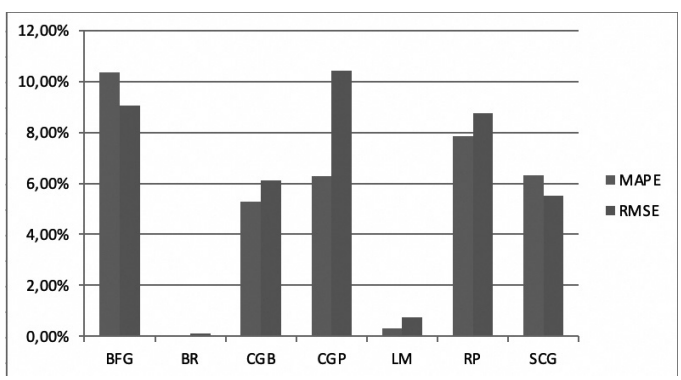


Figura 6.16 Mape e RMSE obtidos por algoritmo de treinamento.
Fonte: elaborada pelo autor.

A Tabela 6.2 traz um resumo dos dados que foram utilizados nesta análise.

Tabela 6.2 Resumo dos dados analisados de acordo com os algoritmos de treinamento

Algoritmos de treinamento	Iterações para validação da RNA	Regressão linear	Mape (%)	RMSE (%)
BFG	354	0,9967	10,37	9,06
BR	168	0,9999	0,02	0,11
CGB	21	0,9977	5,28	6,13
CGP	6	0,9919	6,27	10,41
LM	134	0,9999	0,31	0,74
RP	75	0,9958	7,85	8,76
SCG	36	0,9992	6,33	5,15

Fonte: elaborada pelo autor.

A Figura 6.17 ilustra o ranking dos algoritmos de treinamento analisados, de acordo com os desempenhos obtidos nas análises precedentes, com o objetivo de dar uma visão geral ao leitor.

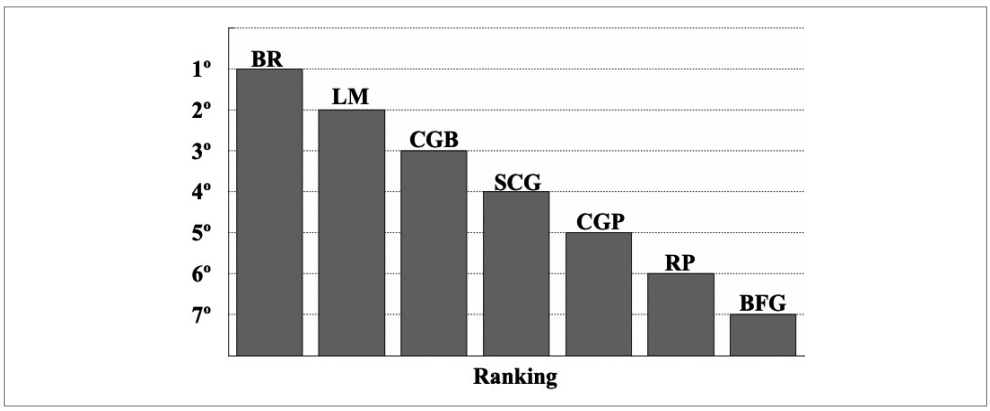


Figura 6.17 Ranking do desempenho dos algoritmos de treinamento.
Fonte: elaborada pelo autor.

De acordo com a Figura 6.17, o algoritmo que obteve, na média de todos os índices, o melhor desempenho para o problema exposto foi o BR, em segundo lugar o LM, em terceiro o CGB, em quarto o SCG, em quinto o CGP, em sexto o RP e em sétimo o BFG.

Realizadas as análises dos algoritmos de treinamento, apresentam-se agora os resultados da previsão temporal com o SVM. A Figura 6.18 ilustra a potência medida em (W) e a potência prevista em (W) pelo SVM.

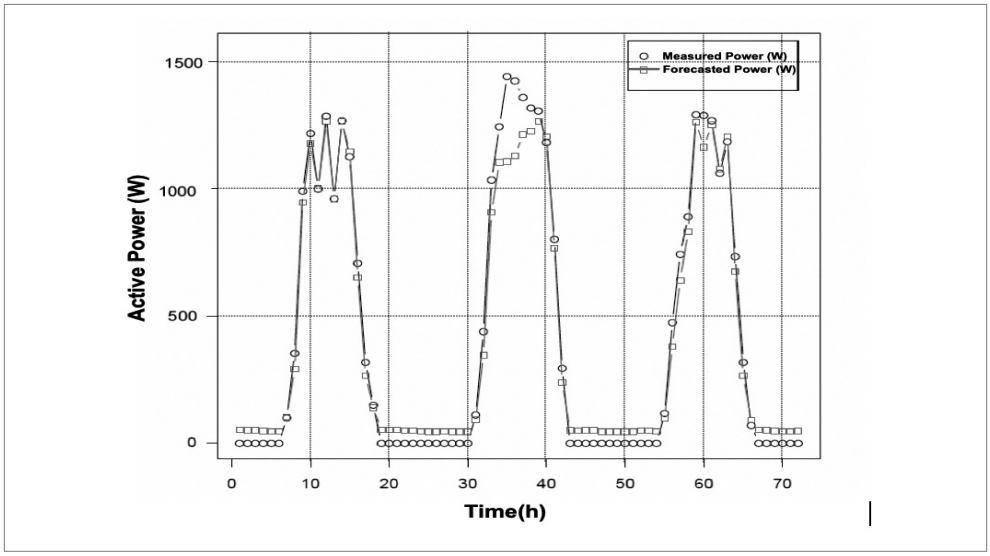


Figura 6.18 Potência medida (W) e potência prevista (W) em um intervalo de 72 horas para o SVM.
Fonte: elaborada pelo autor.

Os dados do SVM utilizado e o resultado de sua regressão linear são apresentados na Tabela 6.3.

Tabela 6.3 Dados do SVM utilizado para previsão temporal de dados

Tipo SVM	γ	ε	Coeficiente de regressão
Kernel radial	0,3333	0,1	0,9966

Fonte: elaborada pelo autor.

Para o SVM utilizado, obteve-se uma regressão linear de 0,9966, ou seja, uma regressão um pouco inferior às encontradas para os algoritmos de treinamento BR e LM. Pode-se ver pela Figura 6.18, comparativamente às Figuras 6.10 e 6.15, que a rede neural com os algoritmos supracitados obteve um melhor desempenho que o SVM.

Na Figura 6.19, pode-se ver o erro entre os dados previstos (em vermelho) e os dados medidos (em preto) para uma amostra de 700 dados, mesmo número de amostra de dados utilizados para o treinamento da RNA.

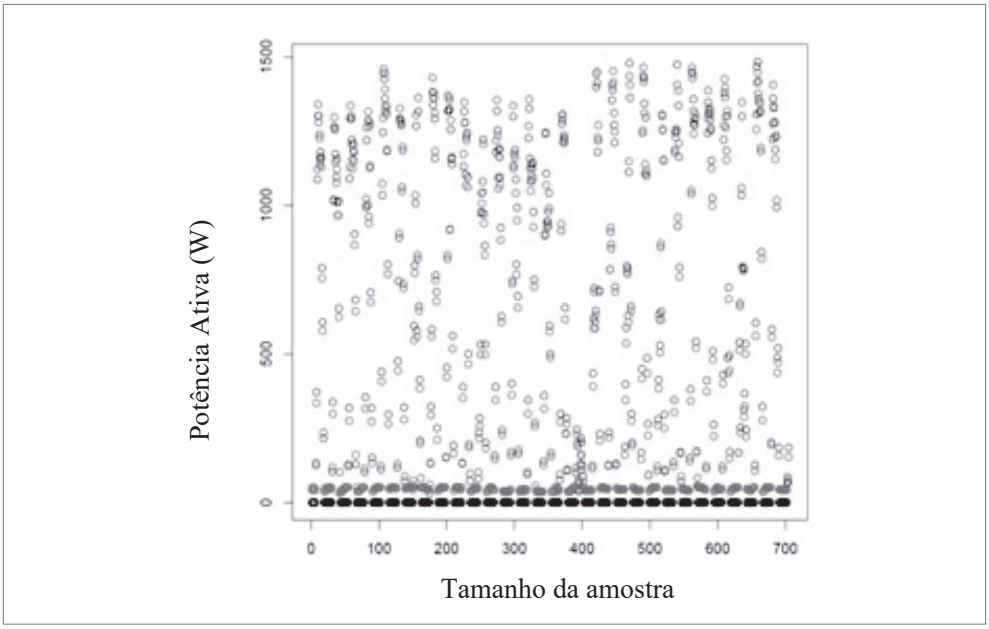


Figura 6.19 Dados previstos versus dados medidos.
Fonte: elaborada pelo autor.

6.10 CONCLUSÃO

Este estudo buscou comparar os algoritmos de treinamentos mais difundidos e utilizados para a previsão temporal de dados utilizando o software MatLab e, posteriormente, fazer uma comparação com um método de previsão de dados que vem sendo difundido.

Para uma quantidade de dados relativamente grande utilizadas na rede neural analisada neste capítulo, ao contrário do que se encontram em muitas publicações, o algoritmo de treinamento que obteve o melhor desempenho dentre todos os algoritmos analisados foi o Bayesian Regularization. Porém, apesar de ter obtido o segundo melhor desempenho, o algoritmo Levenberg-Marquardt necessitou de um tempo menor para treinar e preparar a rede neural. Os algoritmos derivados do algoritmo de treinamento Conjugate Gradient (CGB e SCG) também obtiveram resultados satisfatórios.

Comparando o desempenho dos dois melhores algoritmos de treinamento do RNA com o desempenho do SVM, obteve-se que, para a previsão de dados diversificados, a RNA respondeu melhor do que o SVM. Isso se deve ao fato de que a função Kernel tem uma limitação intrínseca quando amostras de dados com fortes ruídos ou não lineares são incluídos no conjunto de dados, e isso pode acarretar um mau desempenho do SVM.

Portanto, para o tipo de análise feita neste capítulo, recomenda-se o uso do algoritmo de treinamento denominado Bayesian Regularization ou o Levenberg-Marquardt.