INSTITUTO FEDERAL DE EDUCAÇÃO CIÊNCIA E TECNOLOGIA DE MINAS GERAIS – *CAMPUS* BAMBUÍ
BACHARELADO EM ENGENHARIA DE COMPUTAÇÃO

Lucas Batista dos Santos

DETECÇÃO DE *OUTLIERS* EM SÉRIES TEMPORAIS DE CARGA COM REDES NEURAIS RECORRENTES

Bambuí - MG
2023

LUCAS BATISTA DOS SANTOS

DETECÇÃO DE *OUTLIERS* EM SÉRIES TEMPORAIS DE CARGA COM REDES NEURAIS RECORRENTES

Trabalho de conclusão de curso apresentado ao Curso de Bacharelado em Engenharia de Computação do Instituto Federal de Educação Ciência e Tecnologia de Minas Gerais – *Campus* Bambuí para obtenção do grau de Bacharel em Engenharia de Computação.

Orientador: Prof. Dr. Ciniro Aparecido Leite Nametala

Bambuí - MG
2023

Catalogação na Fonte Biblioteca IFMG - Campus Bambuí

S237d Santos, Lucas Batista dos.
Detecção de outliers em séries temporais de carga com redes neurais recorrentes. / Lucas Batista dos Santos. – 2023.
47 f. : il. ; color.

Orientador: Dr. Ciniro Aparecido Leite Nametala.
Trabalho de Conclusão de Curso (graduação) - Instituto Federal de Educação, Ciência e Tecnologia de Minas Gerais – Campus Bambuí, MG, Curso Bacharelado em Engenharia de Computação, 2023.

1. Rede neural LSTM. 2. Detecção de outliers. 3. Subsistema Sudeste/Centro-Oeste. I. Nametala, Ciniro Aparecido Leite. II. Instituto Federal de Educação, Ciência e Tecnologia de Minas Gerais – Campus Bambuí, MG. III. Título.

CDD 006.320151

Elaborada por Douglas Bernardes de Castro- CRB-6/2802

Lucas Batista dos Santos

DETECÇÃO DE *OUTLIERS* EM SÉRIES TEMPORAIS DE CARGA COM REDES NEURAIS RECORRENTES

Trabalho de conclusão de curso apresentado ao Curso de Bacharelado em Engenharia de Computação do Instituto Federal de Educação Ciência e Tecnologia de Minas Gerais – *Campus* Bambuí para obtenção do grau de Bacharel em Engenharia de Computação.

Aprovado em 11 de Dezembro de 2023 pela banca examinadora:

Prof. Dr. Ciniro Aparecido Leite Nametala – IFMG – *Campus* Bambuí – (Orientador)
Prof. Dr. Marcos Roberto Ribeiro – IFMG - *Campus* Bambuí
Prof. Me. Gabriel da Silva – IFMG - *Campus* Bambuí

Documento assinado eletronicamente por **Marcos Roberto Ribeiro**, **Professor**, em 11/12/2023, às 14:36, conforme Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Gabriel da Silva**, **Professor**, em 11/12/2023, às 14:36, conforme Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Ciniro Aparecido Leite Nametala**, **Professor**, em 11/12/2023, às 14:37, conforme Decreto nº 10.543, de 13 de novembro de 2020.

A autenticidade do documento pode ser conferida no site https://sei.ifmg.edu.br/consultadocs informando o código verificador **1758488** e o código CRC **02FF35E6**.

Bambuí - MG
2023

Aos meus pais e à minha irmã.

AGRADECIMENTOS

Agradeço a todos que contribuíram para a realização deste trabalho. Em especial, meus sinceros agradecimentos à minha família, cujo apoio tem sido a base de todas as minhas escolhas e conquistas. Agradeço também ao meu orientador, cuja orientação e ensinamentos valiosos enriqueceram minha jornada acadêmica. Mais uma vez, obrigado a todos que fizeram parte desta jornada.

“A persistência é o menor caminho do êxito.“
(Charles Chaplin)

RESUMO

Um aspecto importante da curva de carga horária disponibilizada pelo Operador Nacional do Sistema Elétrico (ONS) é a capacidade de analisar a variação no consumo de energia ao longo do tempo. A demanda horária, influenciada pela sazonalidade, serve como um indicador do consumo esperado de energia e do adequado funcionamento do Sistema Interligado Nacional (SIN). Embora seja possível identificar visualmente *outliers* na série temporal de carga horária, essa abordagem torna-se trabalhosa e imprecisa. Portanto, é preferível detectar automaticamente os *outliers*, levando em consideração os padrões sazonais. Neste trabalho, um modelo de Rede Neural Artificial (RNA) com células *Long Short-Term Memory* (LSTM) foi empregado para identificar *outliers* com base na sazonalidade. Foram utilizados dados históricos de consumo de energia coletados ao longo do tempo no subsistema Sudeste/Centro-Oeste (SECO). Para rotular os dados, utilizou-se a técnica *Interquartile Range* (IQR). O treinamento da RNA foi conduzido com base nos dados rotulados para os anos de 2020 e 2021. Posteriormente, o modelo foi testado com dados do ano de 2022. Os resultados foram positivos, alcançando métricas de *precision* e *recall* de 98% e 96%, respectivamente.

Palavras-chave: Rede Neural LSTM. Detecção de *Outliers*. Subsistema Sudeste/Centro-Oeste

ABSTRACT

An important aspect of the load curve provided by the National Electric System Operator is the ability to analyze variations in energy consumption over time. Hourly demand, influenced by seasonality, serves as an indicator of expected energy consumption and the proper functioning of the National Interconnected System. While visually identifying outliers in the time series of hourly load is possible, this approach becomes laborious and imprecise. Therefore, it is preferable to automatically detect outliers, considering seasonal patterns. In this study, a model of Artificial Neural Network with Long Short-Term Memory cells was employed to identify outliers based on seasonality. Historical energy consumption data collected over time in the Southeast/Central-West subsystem were used. The Interquartile Range (IQR) technique was employed for data labeling. The ANN was trained based on labeled data for the years 2020 and 2021, and subsequently tested with 2022 data. The results were positive, achieving precision and recall metrics of 98% and 96%, respectively.

Keywords: LSTM Neural Network. Outlier Detection. Southeast/Central-West Subsystem.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

API	*Application Programming Interface*
DSR	*Design Science Research*
GRU	*Gated Recurrent Unit*
IA	Inteligência Artificial
IQR	*Interquartile Range*
LSTM	*Long Short-Term Memory*
MWh/h	Megawatts-hora por hora
MLP	*Multilayer Perceptron*
ONS	Operador Nacional do Sistema Elétrico
RBF	*Radial basis function*
RNA	Rede Neural Artificial
RNN	Rede Neural Recorrente
SECO	Sudeste/Centro-Oeste
SIN	Sistema Integrado Nacional
TLBO	*Teaching-Learning Based Optimization*
Wh	Watts-hora

SUMÁRIO

1 INTRODUÇÃO

O Sistema Integrado Nacional (SIN), responsável pela produção e transmissão de energia elétrica do Brasil, é um sistema hidro-termo-eólico de grande porte, com predominância de usinas hidrelétricas com múltiplos proprietários estatais e privados, sendo constituído por quatro subsistemas: Sul, SECO (Sudeste/Centro-Oeste), Nordeste e Norte (ONS, 2023). Vale ressaltar que o subsistema SECO é considerado o maior, visto que é o principal gerador de energia, com predominância em usinas hidrelétricas responsáveis por 70% dos reservatórios do sistema (ONS, 2022).

A coordenação e controle da operação das instalações de geração e transmissão de energia elétrica do SIN e sistemas isolados são atribuições do Operador Nacional do Sistema Elétrico (ONS). O ONS é uma pessoa jurídica de direito privado sob a forma de associação civil sem fins lucrativos, que desenvolve uma série de estudos para garantir a segurança do suprimento energético no país, com intenção de contribuir com a expansão do SIN, prover acesso à rede de transmissão de forma não discriminatória, visando sempre o menor custo e garantia de segurança do sistema (ONS, 2023a).

Segundo Mota (2021, p. 14), "as cargas elétricas são compostas considerando-se o consumo de energia elétrica de vários aparelhos e dispositivos elétricos [...]". Portanto, ao estudar a curva de carga horária disponibilizada pelo ONS, uma tarefa importante é analisar as variações dos valores ao longo do tempo. A demanda horária é influenciada pela sazonalidade, resultando em perfis e padrões específicos para diferentes dias da semana, fins de semana e dias especiais (CAMPOS, 2008; GUIRELLI, 2006). A detecção de *outliers* na série temporal da carga é, nesse contexto, uma técnica importante, pois permite identificar situações atípicas, como interrupções inesperadas, crises de abastecimento ou até grandes eventos, como visto na pandemia da COVID-19.

Como mencionado por Alla e Adari (2019), um *outlier* é um ponto atípico que pode ocorrer em um conjunto de dados, sendo geralmente causado por erros aleatórios ou variações naturais nos dados. *Outlier* também é um termo definido por Dash *et al.* (2023) como um valor que difere bastante do restante dos dados. Pode-se ver também, no trabalho de Wang, Bah e Hammad (2019), que a definição de *outlier* é um ponto de dados que difere significativamente dos demais ou que não está em conformidade com o padrão normal esperado do fenômeno que representa. Com base nessas definições, ao longo deste texto, o uso dos termos *outlier* e ponto atípico será feito de forma intercambiável, referindo-se sempre ao conceito no mesmo sentido dado por estas definições.

Em geral, não há uma teoria estatística que estabeleça uma separação clara entre *outliers* e pontos normais. Em vez disso, existem abordagens que definem critérios para determinar o quão distante uma observação deve estar para ser considerado *outlier* (BRUCE; BRUCE, 2019). A detecção de *outliers* abrange uma ampla variedade de técnicas, muitas das quais são essencialmente semelhantes, mas possuem diferentes nomes, como detecção de *outliers*, detecção de novidades e detecção de anomalias (HODGE;

AUSTIN, 2004). Além disso, de acordo com Alla e Adari (2019), as tarefas de detecção de *outliers* e detecção de novidades estão intimamente relacionadas à detecção de anomalias. Com base nessas definições, ao longo deste trabalho, especialmente durante a revisão do estado da arte, esses termos serão empregados para a análise da utilização das técnicas, independentemente de se tratarem de detecção de *outliers*, detecção de novidades ou detecção de anomalias.

1.1 Justificativa

A detecção de *outliers* em séries temporais é uma tarefa na qual podem ser empregadas diferentes técnicas. Estas, de forma geral, usando métodos de aprendizado de máquina e estatística (VISHWAKARMA; PAUL; ELSAWAH, 2020). Entre essas duas áreas, destaca-se no contexto da Engenharia de Computação, o aprendizado de máquina, pois, utilizando-se de algoritmos inteligentes, tem como vantagem principal a criação de modelos de forma automática. Esses modelos são soluções candidatas à resolução do problema quando são fornecidos na etapa de treinamento uma quantidade suficiente de informação, que permita a estimação probabilística das observações amostrais (OMAR; NGADI; JEBUR, 2013).

Diante disso, embora seja possível observar visualmente alguns *outliers* na série de carga horária, esse processo pode ser trabalhoso e impreciso se feito manualmente, especialmente quando considerados os contextos sazonais e o viés de classificação humano. Portanto, é oportuna a investigação de métodos de aprendizado automatizados, visando melhorar a precisão e eficiência na identificação desses momentos atípicos que não seguem necessariamente os padrões probabilísticos históricos.

1.2 Proposta

Neste trabalho, fez-se uso de uma Rede Neural Artificial (RNA), baseada em células *Long Short-Term Memory* LSTM, para realizar a detecção de *outliers*, visando identificar momentos que perturbaram a situação de sazonalidade da carga.

Com base em dados históricos de consumo coletados ao longo do tempo no subsistema SECO, foram analisados períodos históricos que contemplaram três anos de dados, isto é, 2020, 2021 e 2022. Destaca-se o ano de 2020, no qual ocorreu a pandemia da COVID-19 e o ano de 2021, quando ocorreu a maior crise hídrica dos últimos 91 anos no abastecimento dos reservatórios brasileiros (NAMETALA *et al.*, 2023).

Com esse tipo de monitor, pretende-se munir os usuários encarregados das tarefas de gestão energética com informações úteis acerca de períodos que possam afetar a sazonalidade da carga, identificando ocorrências como *outliers* no contexto temporal.

1.2.1 Objetivo Geral

Este trabalho tem como objetivo avaliar o desempenho de uma RNA em detectar *outliers* na série temporal de carga horária do subsistema SECO do SIN, com base em dados históricos de consumo de energia.

1.2.2 Objetivos Específicos

Como objetivos específicos, é pretensão deste trabalho:

- Elaborar um *dataset* (conjunto de dados) da série temporal de carga horária para, pelo menos, três anos, de modo que já existam previamente momentos de eventos atípicos que possam favorecer a ocorrência de *outliers*;
- Implementar uma RNA;
- Avaliar o desempenho da RNA em detectar os *outliers*, comparando com os resultados previamente conhecidos, rotulados pelo método estatístico *Interquartile Range* (IQR).

1.3 Estrutura do documento

Esse trabalho está estruturado em seis capítulos, incluindo as referências.

Na sequência dessa introdução, são apresentados os fundamentos teóricos necessários para compreender as principais técnicas utilizadas na metodologia. Tal capítulo capítulo aborda conceitos de séries temporais, fornecendo uma introdução à curva de carga horária, e explora conceitos relacionados a *outliers* e as RNAs. Além disso, é apresentado o estado da arte na detecção de *outliers*, abrangendo trabalhos que utilizaram tanto RNAs quanto métodos estatísticos para esse propósito.

No Capítulo 3 é apresentada a metodologia. O texto se organiza na mesma sequência que foi realizada a implementação da solução.

No Capítulo 4 são descritos os resultados e discussões dos experimentos realizados.

No Capítulo 5 é feita uma conclusão que abrange as principais contribuições, limitações da pesquisa e proposta de trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo estabelece a base teórica essencial para a compreensão da pesquisa desenvolvida. Serão abordados conceitos fundamentais relacionados a séries temporais, *outliers* e RNAs.

2.1 Séries Temporais

Segundo Wooldridge (2015), séries temporais são observações sobre dados que variam ao longo do tempo. São representadas por uma sequência de números coletados em intervalos regulares e estão fortemente relacionadas com suas histórias recentes, de maneira que a ordem cronológica de sua observação importa. Outra característica que pode ser encontrada em algumas das séries temporais é o padrão sazonal. O padrão sazonal consiste em observações que apresentam as mesmas propriedades estatísticas de forma cíclica em um dado período de tempo. Além disso, séries também podem apresentar tendências ou variações repetidas ao longo do tempo. Todos esses fatores podem ser utilizados para definir se uma série possui ou não boas propriedades preditivas (MORETTIN; TOLOI, 2018). São exemplos de séries temporais os preços de ações, histórico de vendas, oferta monetária e outras.

Assim, ao obter uma série temporal em instantes discretos, é possível realizar análises para investigar os fatores que levaram à geração da série, fazer previsões de valores no curto ou longo prazo, descrever o comportamento da série por meio de gráficos para verificar a existência de tendências, ciclos e variações sazonais, além de procurar por periodicidades relevantes nos dados (MORETTIN; TOLOI, 2018).

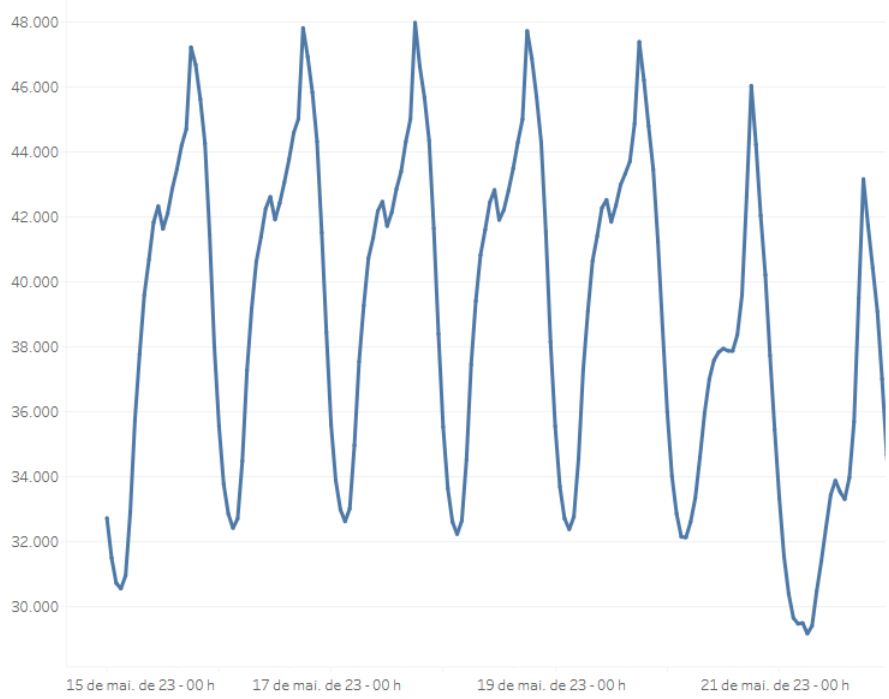
2.2 Curva de carga horária

A carga é uma medida de energia expressa em Watts-hora (Wh) e está relacionada com a quantidade de energia consumida ou fornecida por um sistema durante um determinado tempo. Em termos simples, a carga representa a quantidade de eletricidade utilizada por dispositivos elétricos ao permanecerem ligados por um determinado período. Portanto, a curva de carga horária representa os valores de energia consumidos hora a hora, sendo expresso na unidade de Megawatt-hora por hora (MWh/h) (BOYLESTAD, 2012).

A Figura 1 apresenta um exemplo de uma curva de carga horária para o subsistema SECO entre os dias 15/05/2023 a 22/05/2023. Essa representação permite visualizar o padrão de consumo ao longo do dia.

Nesse contexto, a série temporal da curva de carga horária é composta por uma variável do consumo de energia em MWh/h, coletado em intervalos regulares de tempo (t). Para uma análise de hora em hora, o número de observações é determinado

Figura 1 – Curva de carga horária do SIN.



Fonte: ONS, 2023b.

pela quantidade de horas. Considerando um período de n dias, em que cada observação começa a meia-noite e termina às 23 horas do mesmo dia, serão vistas, por exemplo, 24 observações diárias. Portanto, o número total de observações ao longo do tempo será igual a $(24 \times n)$.

2.3 *Outliers*

Em um *dataset*, os valores extremos que se distanciam da maioria das outras observações são chamados de *outliers*. Um *outlier* pode ser detectado por métodos estatísticos usando métricas de distâncias (ALLA; ADARI, 2019). Um exemplo que ilustra este conceito é o uso da amplitude interquartil para classificar uma observação como atípica quando esta se encontra a mais de 1,5 vezes distante da mediana da amostra (BRUCE; BRUCE, 2019). Portanto, a detecção de *outliers* é uma tarefa aplicável à análise estatística de dados em séries temporais, visto que essas séries podem ter seus valores ordenados para uma avaliação segundo seus percentis[1] (MINGOTI, 2007). Em uma série temporal, os *outliers* são observações que se desviam do padrão esperado ou do modelo usual, os quais não se enquadram nas expectativas dentro do contexto e do método de aplicação (ALLA; ADARI, 2019; VISHWAKARMA; PAUL; ELSAWAH, 2020).

[1] Um percentil é uma medida estatística que divide uma amostra ordenada (em ordem crescente) em 100 partes iguais. Define-se o percentil Q_k, para $k = 1, 2, ..., 99$, como o valor para o qual $k\%$ dos elementos da amostra são menores ou iguais a Q_k, e os restantes $100 - k\%$ elementos da amostra são maiores ou iguais a Q_k (MARTINS, 2014).

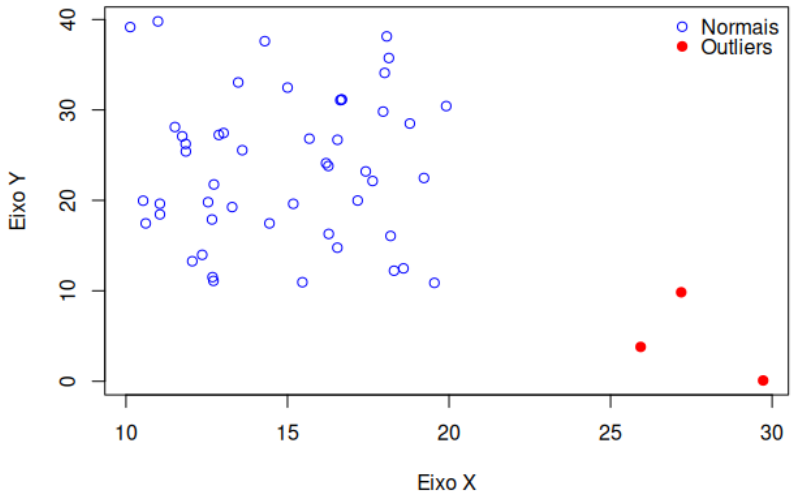
Segundo Alla e Adari (2019), os *outliers* podem ser generalizadas em três categorias:

1. *Outlier* baseado em pontos: São definidas como observações individuais que se encontram significativamente distantes do restante dos dados. A Figura 2 ilustra esse tipo de comportamento;

2. *Outlier* baseado no contexto: Refere-se a pontos de dados que podem parecer normais à primeira vista, mas quando o contexto é considerado, esses dados são considerados atípicos. Por exemplo, o aumento repentino de compras fora de feriados;

3. *Outlier* baseado em padrão: Para ser considerado um *outlier*, é necessário observar se o comportamento da distribuição se desvia dos padrões e tendências anteriores.

Figura 2 – Exemplo de *outlier* baseado em pontos.



Fonte: Elaborado pelo Autor, 2023.

A detecção de *outliers*, de acordo com Alla e Adari (2019), é o processo pelo qual um algoritmo identifica certos dados ou padrões como atípicos. Nesse sentido, uma variedade de algoritmos pode ser utilizada, incluindo RNAs, abordagens baseadas em lógica nebulosa, algoritmos de estatística clássica, entre outros. Nesse contexto, as técnicas de detecção de *outliers* que se baseiam na estimação de modelos podem ser classificadas segundo Alla e Adari (2019), como:

1. Supervisionada: O *dataset* precisa estar rotulado, geralmente envolve um método de classificação binária, no qual o modelo é treinado para classificar os dados como *outliers* e normais;

2. Semi-supervisionada: Apenas uma parte dos dados são rotulados. Supõe-se que somente os dados de treinamento são rotulados com a classe normal, enquanto os demais dados permanecem sem rótulos;

3. Não supervisionada: Considera que a ocorrência de dados normais é significativamente maior do que a ocorrência de *outliers*. Nesse caso, os dados não são rotulados com base nas suas classes. Em vez disso, o modelo é treinado para aprender a distinguir os pontos normais dos *outliers* com base nas características dos dados.

2.4 Redes Neurais Artificias

Segundo Chollet (2021), a Inteligência Artificial (IA) teve início por volta de 1950, quando cientistas da computação questionaram se um computador poderia pensar de maneira que conseguisse realizar tarefas intelectuais feitas por humanos. Devido à IA ser uma área abrangente, surgiram subáreas como aprendizado de máquina, *deep learning* e inteligência simbólica. O aprendizado de máquina surgiu a partir do questionamento se, de fato, um computador poderia aprender sozinho e executar além do programado, de maneira a permitir que humanos inserissem dados e regras para serem usadas em problemas. A diferença fundamental do aprendizado de máquina é que, quando o modelo é treinado com dados relevantes, ele se torna capaz de encontrar soluções estatísticas que podem ser generalizadas para novos problemas, automatizando assim tarefas.

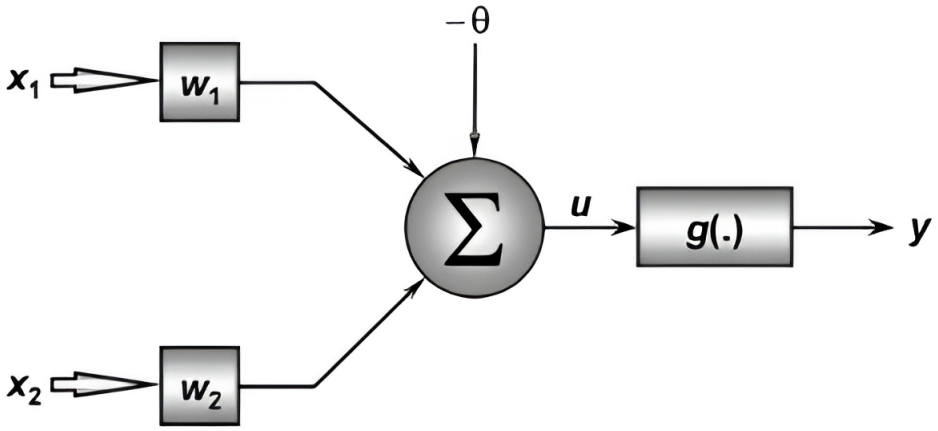
As RNAs são modelos de aprendizado de máquina que tiveram sua pesquisa inspirada na simulação do funcionamento do cérebro e das interconexões entre os neurônios. Em essência, uma RNA é um algoritmo projetado para imitar algumas das funções que o cérebro orgânico pode realizar. Sendo implementada em *hardware* ou *software*, uma parte importante das RNAs é seu processo de aprendizagem feito via algoritmo de otimização. Estes algoritmos têm por função principal encontrar um conjunto de pesos sinápticos. Uma RNA pode possuir diversas características e, no contexto das séries temporais, destaca-se a adaptabilidade, inclusive em situações de não estacionariedade (HAYKIN, 2001; GOODFELLOW; BENGIO; COURVILLE, 2016; BRAGA; LUDERMIR; CARVALHO, 2007; DA SILVA; SPATTI; FLAUZINO, 2016).

2.4.1 Perceptron

A forma mais simples de uma RNA é o *Perceptron*, uma arquitetura básica que consiste em apenas uma camada de entrada e um neurônio com uma saída (DA SILVA; SPATTI; FLAUZINO, 2016). Na Figura 3, é possível observar que múltiplos sinais de entrada podem ser mapeados e, após o processamento realizado pela rede, é gerada uma única saída. Geralmente, os sinais de saída são determinados por meio de uma função de ativação, que pode ser implementada utilizando funções, tais como degrau, linear ou sigmoide (HAYKIN, 2001).

A arquitetura da *Multilayer Perceptron* (MLP) é uma generalização do *perceptron* de camada única. Essa configuração de rede neural inclui uma camada de entrada, uma ou mais camadas intermediárias (também conhecidas como camadas escondidas)

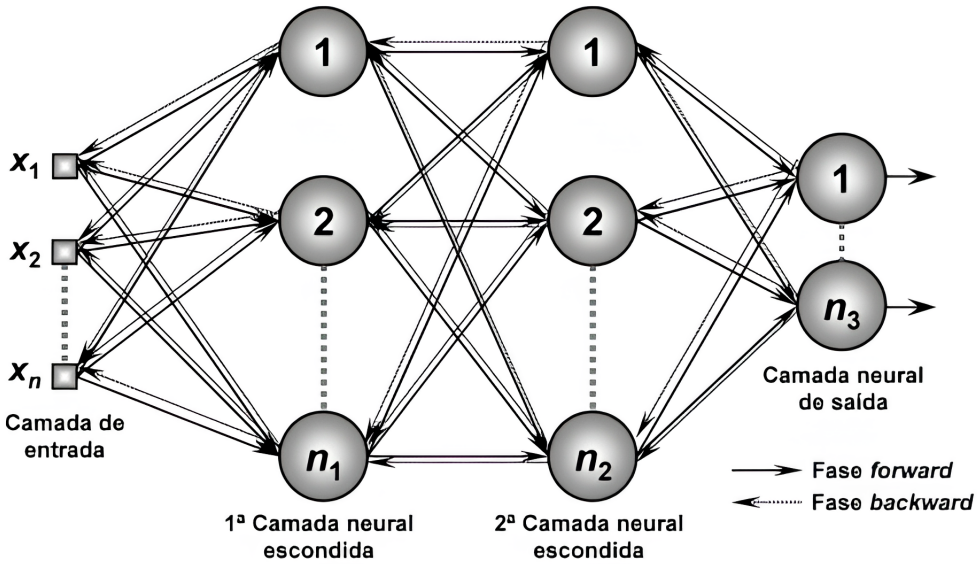
Figura 3 – *Perceptron* simples com duas entradas.



Fonte: DA SILVA; SPATTI; FLAUZINO, 2016.

e uma camada de saída, que pode conter vários neurônios. A popularidade da MLP aumentou no final dos anos 1980, principalmente devido ao algoritmo de aprendizado chamado *Backpropagation*. O *Backpropagation* treina a MLP em duas etapas: a etapa *forward*, em que os sinais de entrada são processados pela rede até a geração da saída, e a etapa *backward*, em que o erro entre a saída obtida e a resposta esperada é retropropagado pela rede para ajustar os pesos. A Figura 4 ilustra uma representação da rede MLP (DA SILVA; SPATTI; FLAUZINO, 2016; HAYKIN, 2001).

Figura 4 – MLP com processos *forward* e *backward*.



Fonte: DA SILVA; SPATTI; FLAUZINO, 2016.

As RNAs possuem diversas aplicações em setores como financeiro, elétrico, automação e outros. Uma das principais aplicações das RNAs é a classificação, que consiste em atribuir um padrão desconhecido a uma das classes conhecidas. Esse processo é frequentemente realizado por meio do aprendizado supervisionado, em que amostras são

rotuladas com suas respectivas classes. Durante o aprendizado, as amostras são fornecidas como entrada para a RNA e as saídas são comparadas com as classes correspondentes. A RNA ajusta seus pesos para estabelecer as relações entre os padrões de entrada e as classes de saída correspondentes. Após o treinamento, a rede pode-se tornar capaz de classificar padrões de entrada (BRAGA; LUDERMIR; CARVALHO, 2007).

2.4.2 Arquiteturas recorrentes

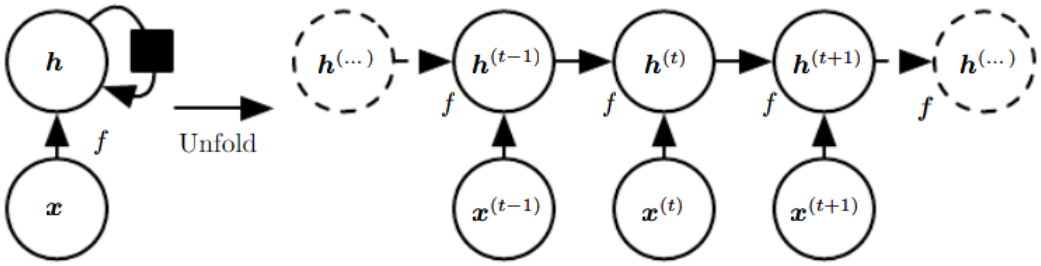
As redes neurais recorrentes (RNNs) são uma família de RNAs projetadas para lidar com dados sequenciais. Enquanto uma MLP tem como objetivo processar a relação entre pares de entrada e saída $[x(n), y(n)]$, ela se torna ineficiente ao lidar com valores passados $x(n-k)$, que têm influência no valor de saída $y(n)$. Isso ocorre porque uma MLP não compartilha uma matriz de pesos w para todos os instantes possíveis de n. Por outro lado, as RNNs são especializadas no processamento de sequências longas de dados, permitindo que os valores passados sejam considerados durante o processamento. Além disso, a maioria das RNNs é capaz de lidar com sequências de comprimento variável (GOODFELLOW; BENGIO; COURVILLE, 2016; NAMETALA, 2023).

As RNNs compartilham os mesmos parâmetros em todos os instantes de tempo. Essa abordagem é importante para permitir que o modelo generalize o aprendizado. Caso fossem utilizados parâmetros separados para cada instante, o modelo teria dificuldade em generalizar adequadamente para *datasets* novos, especialmente aqueles com comprimentos de sequência e posições no tempo diferentes.

O compartilhamento de parâmetros é particularmente relevante quando uma informação específica pode ocorrer em várias posições dentro da sequência. Por exemplo, em uma MLP, as frases “Eu fui no Japão em 2023” e “Em 2023, eu fui no Japão” seriam tratadas de forma diferente e a resposta para a pergunta “Em que ano viajei?” seria examinada de maneira distinta na estrutura da rede. Isso implicaria, na necessidade de construir uma MLP que aprendesse todas as regras da linguagem separadamente em cada posição da frase. Por outro lado, em uma RNN isso não ocorre, uma vez que compartilha os mesmos parâmetros ao longo do tempo (NAMETALA, 2023).

Uma RNN compartilha seus parâmetros, de modo que cada elemento da saída é uma função das saídas anteriores. Em outras palavras, a saída é realimentada para as entradas. Isso significa que cada elemento de saída é produzido utilizando a mesma regra aplicada às saídas anteriores. Essa propriedade de compartilhamento de parâmetros resulta em um grafo computacional profundo. Na Figura 5 pode ser visto um diagrama que ilustra uma RNN que processa informações da entrada `x`, incorporando-as ao estado `h` que é transmitido ao longo do tempo (GOODFELLOW; BENGIO; COURVILLE, 2016).

Figura 5 – Grafo recorrente desdobrado.

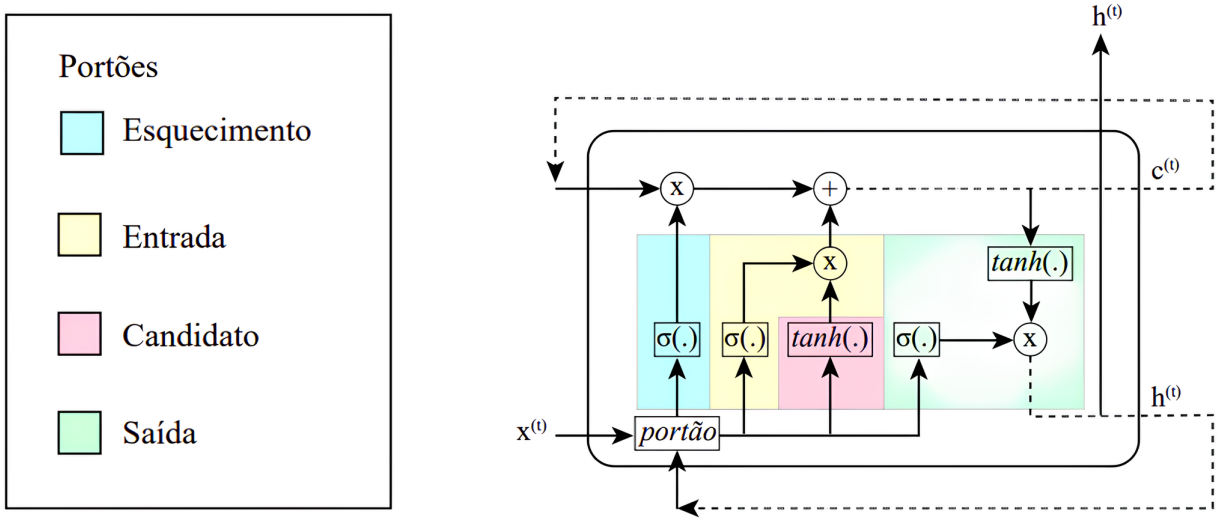


Fonte: GOODFELLOW; BENGIO; COURVILLE, 2016.

2.4.3 Redes Long Short-Term Memory (LSTM)

A rede LSTM, uma variante das RNNs, têm como característica a habilidade em capturar, otimizar e lidar com dependências de longo prazo em sequências. Sua arquitetura foi especialmente concebida para superar os problemas de explosão ou desvanecimento de gradientes que podem afetar o treinamento de redes recorrentes. Ao contrário das RNNs convencionais, uma célula LSTM não apenas modela a recorrência por meio de uma única unidade não linear, mas também emprega recorrência interna controlada por funções não lineares denominadas "portões". Essas funções atuam em conjunto com a entrada para combinar, modificar e propagar o estado oculto da rede (GOODFELLOW; BENGIO; COURVILLE, 2016). A Figura 6 ilustra a representação de uma célula LSTM (NAMETALA, 2023).

Figura 6 – Esquema de uma RNA recorrente LSTM.



Fonte: NAMETALA, 2023.

Uma célula LSTM possui duas saídas. A primeira saída é chamada de estado de célula $c^{(t)}$, enquanto a segunda saída é denominada estado oculto $h^{(t)}$. O estado de saída da célula $c^{(t)}$ tem como propósito armazenar e transmitir informação do instante $(t-1)$. Essas saídas são processadas por quatro portões que ajustam a entrada por meio

de funções de ativação e pesos. Considerando um vetor de entrada $x^{(t)}$ no tempo t, e $h^{(t-1)}$ como o vetor de estado oculto anterior, cada um dos portões possui suas próprias matrizes de pesos representadas por U^k e W^k, sendo que o índice k denota o portão específico (NAMETALA, 2023; GOODFELLOW; BENGIO; COURVILLE, 2016).

A primeira etapa da LSTM corresponde ao portão de esquecimento, que avalia o estado anterior $c^{(t-1)}$ por meio de uma função logística sigmoide, restringindo seus resultados no intervalo entre 0 e 1. Isso determina se o estado anterior será mantido ou descartado. A equação do portão de esquecimento é expressa na Equação (2.1).

$$f^{(t)} = \sigma\left(b^f + \sum U^f x^{(t)} + \sum W^f h^{(t-1)}\right) \tag{2.1}$$

A função sigmoide é representada conforme Equação (2.2).

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2.2}$$

Em seguida, ocorre a soma do estado anterior da célula $c^{(t-1)}$ ao valor de entrada modulado. Essa soma é gerada pelos portões de entrada e candidato. O portão de entrada, operando por meio da função sigmoide, gera uma saída que varia entre 0 e 1, determinando se o resultado do portão candidato será propagado ou descartado, conforme expresso na Equação (2.3).

$$p^{(t)} = \sigma\left(b^p + \sum U^p x^{(t)} + \sum W^p h^{(t-1)}\right) \tag{2.3}$$

O portão candidato utiliza a função tangente hiperbólica para considerar valores negativos e permitir a inibição de um sinal em vez de bloqueá-lo, conforme ilustrado na Equação (2.4).

$$d^{(t)} = tanh\left(b^d + \sum U^d x^{(t)} + \sum W^d h^{(t-1)}\right) \tag{2.4}$$

onde $tanh$ é a tangente hiperbólica definida como (2.5):

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{2.5}$$

Segundo a Equação (2.6), a combinação desses três portões resulta na atualização do estado da célula $c^{(t)}$, a qual é utilizada de forma recorrente no instante subsequente $(t + 1)$.

$$c^{(t)} = f^{(t)} c^{(t-1)} + p^{(t)} d^{(t)} \tag{2.6}$$

Finalmente, o cálculo da saída da LSTM é realizado. Inicialmente, o estado da célula $c^{(t)}$ passa por uma função tangente hiperbólica para linearizar os valores no intervalo de -1 a 1. Posteriormente, o resultado é multiplicado pelo valor obtido pelo portão de

saída, conforme Equação (2.7), o qual determina a proporção do estado atual presente na saída da LSTM, como indicado na Equação (2.8)[2].

$$o^{(t)} = \sigma\left(b^o + \sum U^o x^{(t)} + \sum W^o h^{(t-1)}\right) \tag{2.7}$$

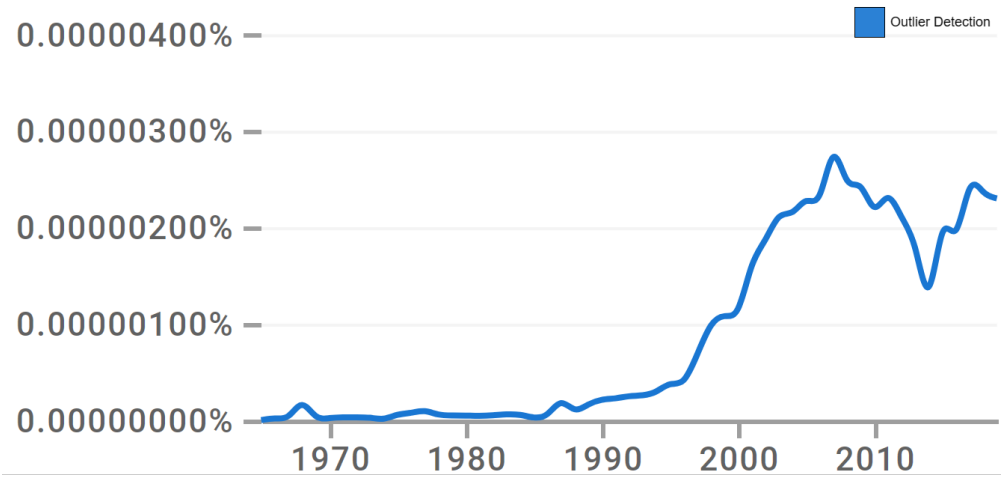
$$h^{(t)} = o^{(t)} tanh\left(c^{(t)}\right) \tag{2.8}$$

2.5 Estado da arte

Esta seção apresenta o estado da arte no campo de estudo deste trabalho. Uma análise foi conduzida em trabalhos científicos publicados em anais de conferências e revistas, abrangendo a detecção de *outliers* ou anomalias no consumo de energia elétrica no período de 2019 a 2023, destacando aqueles que empregaram RNAs e métodos estatísticos. Essa seção tem como propósito examinar os métodos que contribuíram para o desenvolvimento da área.

A detecção de *outlier* tem sido explorada na literatura científica desde 1970. Essa afirmação pode ser conferida pela Figura 7, em que é possível observar um aumento na frequência de citações após esse ano.

Figura 7 – Frequência relativa de citações ao nome *outlier detection* na história.



Fonte: BOOKS, 2023.

Wang, Bah e Hammad (2019) conduziram uma pesquisa relacionada ao progresso sobre a detecção de *outliers* em uma ampla gama de áreas de aplicação, incluindo detecção de fraudes, segurança de redes, análise de dados de séries temporais e outros domínios relevantes. Os autores fornecem uma análise crítica dos métodos existentes, destacando suas vantagens, desvantagens e limitações. Além disso, são discutidos os principais desafios enfrentados na detecção de *outliers*, como a escalabilidade para grandes volumes de dados, a complexidade computacional e a interpretação dos resultados. O

[2] Para uma descrição detalhada deste processo consultar (NAMETALA, 2023, p. 184).

artigo também enfatiza a importância de abordagens de aprendizado de máquina, como o aprendizado profundo, na detecção de *outliers*.

Em Himeur *et al.* (2021) é feita uma revisão das técnicas existentes para detecção de anomalias no consumo de energia elétrica. Os autores destacam a competência das RNNs na análise de séries temporais, permitindo a identificação de comportamentos dinâmicos e a previsão de anomalias durante o uso de energia, ao mesmo tempo em que as distinguem de desvios relacionados à sazonalidade, ao clima e feriados. O artigo também aborda técnicas estatísticas e abordagens de aprendizado supervisionado. Além disso, os autores apresentam um resumo das técnicas relevantes de detecção de anomalias baseadas em IA, incluindo seus pontos fortes e fracos.

No trabalho de Chahla *et al.* (2019), foi proposta uma abordagem que combina RNNs com arquitetura LSTM e o método de agrupamento *K-means* para identificar e prever anomalias no consumo de energia. O modelo foi configurado com três camadas ocultas e treinado utilizando o otimizador Adam para realizar previsões do consumo de energia da próxima hora, usando dados das últimas 24 horas. A abordagem obteve resultados consistentes na detecção de anomalias no consumo de energia.

Em Silva *et al.* (2019), uma RNN baseada em LSTM foi aplicada com a técnica de *Negative Selection* para realizar a previsão de anomalias no consumo de energia elétrica. Os dados utilizados consistiram no histórico de consumo de energia registrado em intervalos de 15 minutos ao longo de 20 semanas, especificamente nas segundas-feiras. A abordagem que obteve o melhor desempenho ao prever as anomalias obteve uma precisão média de 77%.

O estudo realizado por Nascimento *et al.* (2021) empregou um método híbrido que combinou técnicas de regressão e métodos estatísticos clássicos para a detecção de *outliers* em um *dataset* contendo medidas de energia elétrica. Dentre os métodos utilizados, destacam-se o *BoxPlot*, *Skewed Boxplot* e *Adjusted Boxplot*. Foi observado que os melhores métodos foram o *Skewed Boxplot*, o qual obteve uma precisão de 99% e o *Adjusted Boxplot* com 64% quando aplicado em dados reais.

Li *et al.* (2023) propuseram uma técnica híbrida que combina redes neurais convolucionais e um algoritmo de *RandomForest* otimizado para prever o consumo de energia e detectar *outliers* na série temporal. Os autores introduziram uma abordagem de detecção baseada em erros de previsão. Uma comparação entre os algoritmos *DecisionTree*, *AdaBoost*, *RandomForest* e *GridSearchCV* com *RandomForest* foi realizada para detectar *outliers*, resultando em taxas de acurácia de 90%, 32%, 90% e 96%, respectivamente.

Dash *et al.* (2023) realizaram um *framework* no qual o método estatístico IQR é usado para detectar *outliers* nos dados e lidar com eles pelo método de *Winsorização*, que realiza a normalização dos *outliers*. Em seguida, foi realizada a construção de um classificador baseado em RNAs do tipo *Radial Basis Function* (RBF), treinados com *Teaching-Learning Based Optimization* (TLBO). Foi encontrada a melhor topologia

possível da RBF e o TLBO foi escolhido para buscar o melhor conjunto de parâmetros possível. Após o treinamento, o classificador foi testado em todos os *datasets*. Os resultados experimentais revelaram que a abordagem proposta possui uma precisão melhor do que outras abordagens concorrentes testadas.

Nametala (2023) desenvolveu uma ferramenta para modelar e prever cenários futuros de preço no mercado elétrico brasileiro. Para isso, utilizou técnicas de aprendizado de máquina, em especial as Redes Neurais Atencionais, para mapear sequências em séries temporais. Essas técnicas foram combinadas em conjunto com métodos estatísticos. Um dos aspectos relevante do trabalho é a extensa revisão bibliográfica realizada, abordando o mercado elétrico, os avanços recentes na área de RNAs e suas tendências. A tese apresenta uma comparação entre o estado da arte de RNA e métodos provenientes de outras áreas, como a estatística. Além disso, no contexto da detecção e tratamento dos *outliers*, esse autor utilizou a técnica IQR para detectar e substituir valores atípicos. No total foram classificados 47,8 milhões de valores. Os resultados reportam um total de 71157 substituições.

Após essas breves considerações teóricas que serão aplicadas aos resultados, a seguir, apresenta-se a metodologia, capítulo no qual os procedimentos técnicos serão descritos.

3 METODOLOGIA

A classificação deste trabalho, do ponto de vista de sua natureza, é uma pesquisa aplicada, pois utiliza técnicas de IA para resolver um problema específico. Quanto aos objetivos, é uma pesquisa exploratória, portanto visa investigar e compreender as condições em que os *outliers* na série de carga podem ocorrer. Além disso, em termos de abordagem, é uma pesquisa quantitativa, na qual se estabelece uma relação de causa e efeito que pode ser mensurada por meio de uma função matemática, conforme definido por Severino (2013). Quanto aos procedimentos, foi utilizado o *Design Science Research* (DSR), com a utilização do artefato de instanciação, em decorrência da necessidade de implementação do sistema para a resolução do problema (PIMENTEL; FILIPPO; SANTORO, 2018). Ainda, quanto à classificação específica da área de estudo, segundo Wazlawick (2009), o estilo, "algo diferente", é o mais apropriado para o presente trabalho, uma vez que se trata de uma comparação entre técnicas.

Segundo Sousa e Marques (2020), na DSR é “necessária a realização de revisões literárias para que se caracterize conhecimento técnico e científico e a condução de avaliações para avaliar o problema encontrado e o artefato produzido a partir deste problema”. Assim, a abordagem proposta para o desenvolvimento deste trabalho seguirá os passos do processo do DSR definido por Peffers *et al.* (2007), os quais são descritos a seguir.

1. Identificação do problema: Para realizar a identificação do problema, foi feito o levantamento da revisão bibliográfica sobre os temas de detecção de *outliers*. Nas subseções 1.1 e 1.2, a identificação do problema é formalizada;

2. Definição dos objetivos: Nas subseções 1.2.1, 1.2.2, são apresentados os objetivos;

3. Projeto e desenvolvimento: Na subseção 3.1, é apresentada a metodologia utilizada para a criação do artefato;

4. Demonstração: Na seção 4, são apresentadas as demonstrações do artefato para resolver o problema;

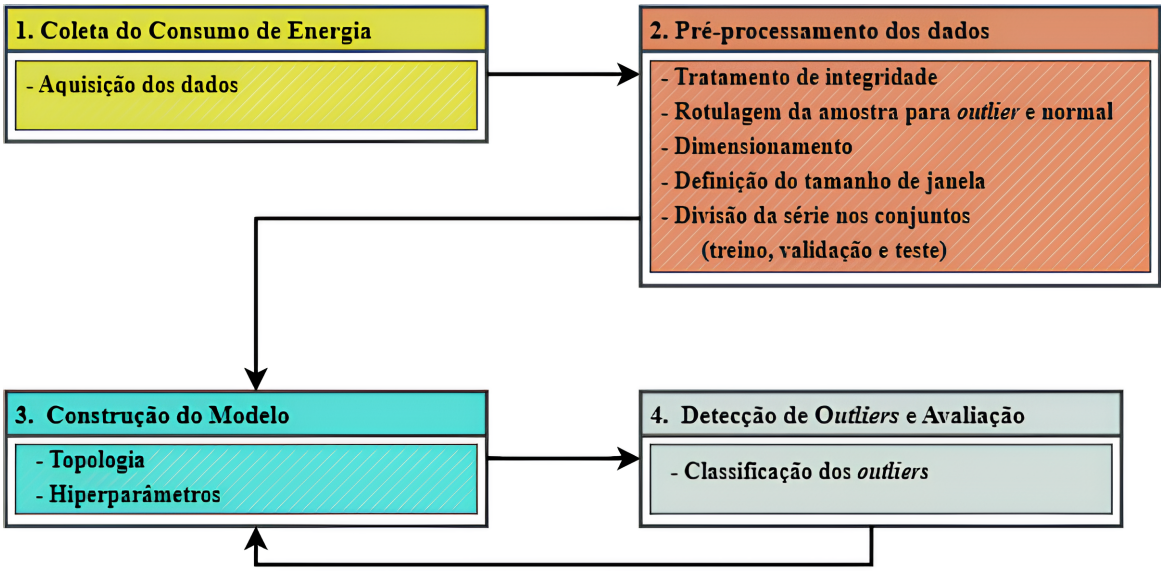
5. Avaliação: Nas seções 4 e 5, são realizadas as avaliações dos resultados alcançados;

6. Comunicação e Contribuição: É realizada ao longo do trabalho, ao comunicar a relevância e eficácia da solução.

3.1 Solução proposta

Na revisão realizada por Himeur *et al.* (2021), são propostas etapas principais para conduzir uma abordagem de detecção de *outliers* supervisionada. Neste trabalho, foram adotadas etapas semelhantes, conforme ilustrado na Figura 8. Cada uma dessas etapas é explicada nas próximas seções seguindo a ordem cronológica.

Figura 8 – Principais etapas utilizadas para realizar uma detecção de *outliers* com aprendizado supervisionado.



Fonte: Adaptado de: HIMEUR *et al.*, 2021.

3.2 Aquisição dos dados

Na extração dos dados, foi utilizada a ferramenta disponibilizada pelo portal da ONS, que permitiu obter o *dataset* no formato *comma separated value* (csv) (ONS, 2023b). Os dados coletados referem-se à curva de carga horária do subsistema SECO, que contém registros históricos do consumo de carga. Cada subsistema do SIN tem suas leituras realizadas separadamente, exceto em casos de intercâmbio de energia que podem conter dados entre as transferências do SIN e países vizinhos (NAMETALA, 2023). As leituras ocorrem em intervalos de 1 hora e a variável de interesse é a carga medida em MWh/h.

A Figura 9 apresenta uma parte do *dataset* coletado, que abrange o período de 01/01/2020 a 31/12/2022 (1096 dias, 3 anos e 26304 observações). A coluna *date* foi modificada para conter apenas a informação da data, enquanto a coluna *int_hour* foi convertida para valores inteiros, variando de 0 a 23, representando as horas no intervalo de 00:00 a 23:00, respectivamente. Além disso, a coluna de *load_mwh* (carga) foi transformada para inteiro e uma coluna de rótulo *is_outlier* foi incluída para indicar se cada registro é ou não um *outlier* com base no IQR.

3.3 Pré-processamento

Na etapa de pré-processamento, o objetivo foi realizar a preparação do *dataset* para sua utilização no modelo LSTM. Isso implica na remoção de informações irrelevantes que prejudicam o aprendizado da RNA, além da inclusão de informações relevantes que melhorem o desempenho do modelo. Nesta seção, são abordadas as principais etapas e

Figura 9 – Exemplo do *dataset* da curva de carga horária coletada.

	date	int_hour	subsistema	load_mwh	is_outlier
0	2020-01-01	0	Sudeste/Centro-Oeste	33108	False
1	2020-01-01	1	Sudeste/Centro-Oeste	33368	False
2	2020-01-01	2	Sudeste/Centro-Oeste	33040	False
3	2020-01-01	3	Sudeste/Centro-Oeste	32399	False
4	2020-01-01	4	Sudeste/Centro-Oeste	31776	False
...	...	...	...	...	...
26299	2022-12-31	19	Sudeste/Centro-Oeste	44647	True
26300	2022-12-31	20	Sudeste/Centro-Oeste	43516	True
26301	2022-12-31	21	Sudeste/Centro-Oeste	39411	True
26302	2022-12-31	22	Sudeste/Centro-Oeste	35797	False
26303	2022-12-31	23	Sudeste/Centro-Oeste	33689	False

26304 rows × 5 columns

Fonte: Elaborado pelo Autor, 2023.

técnicas empregadas para efetuar esse preparo no *dataset*, a fim de torná-lo adequado para ser utilizado como entrada na rede LSTM.

3.3.1 Tratamento de integridade

Os dados extraídos podem apresentar inconsistências. Nesse sentido, a primeira etapa do pré-processamento consistiu em realizar uma verificação de integridade para identificar a existência de possíveis dados faltantes em todas as datas, horas e variáveis do subsistema SECO. Além disso, foram buscados conteúdos inválidos computacionalmente, como *inf, null* ou *NaN*.

Os dados detectados como não íntegros foram substituídos da seguinte forma: caso fosse detectado apenas um valor faltante, a substituição é realizada utilizando a média entre os valores das duas horas existentes imediatamente anterior e posterior ao dado faltante. Para intervalos de 24 horas ou mais, a substituição é realizada com base nos valores dos mesmos dias e horas da semana anterior.

3.3.2 Rotulagem da amostra

Para tornar o *dataset* adequado a um problema supervisionado de classificação, é necessário aplicar o método IQR para identificar e classificar os *outliers*. A variável carga foi utilizada como referência para essa classificação, em que os valores que estavam fora do intervalo estabelecido pelo IQR foram considerados *outliers*. Como descrito por Bruce e Bruce (2019), o IQR utiliza o intervalo entre o primeiro quartil (25% dos dados) e o

terceiro quartil (75% dos dados) para estabelecer uma medida de dispersão e identificar observações que se encontram além dessa faixa como *outliers*.

Portanto, o cálculo do IQR pode ser realizado conforme demonstrado na Equação (3.1), em que $q3$ representa o quantil superior (75% dos dados) e $q1$ o quantil inferior (25% dos dados). Em seguida, são calculados os limites Ls (superior) e Li (inferior), como descrito nas Equações (3.2) e (3.3), de modo que a margem m é utilizada para estabelecer esses limites. Assim, qualquer valor que exceda os limites Ls e Li é considerado um *outlier* (DASH *et al.*, 2023; NAMETALA, 2023).

$$IQR = q3 - q1 \tag{3.1}$$

$$Ls = q3 + m * IQR \tag{3.2}$$

$$Li = q1 - m * IQR \tag{3.3}$$

Na Figura 10, é ilustrado um exemplo do método IQR, no qual os quartis (primeiro e terceiro) são utilizados com uma margem de 1,5. Nesse exemplo, valores que estejam dentro do intervalo de -2,698 a +2,698 desvios padrões a partir da mediana da distribuição normal não são considerados *outliers*. Isso significa que cerca de 99,3% das observações são consideradas normais e não são identificadas como *outliers*.

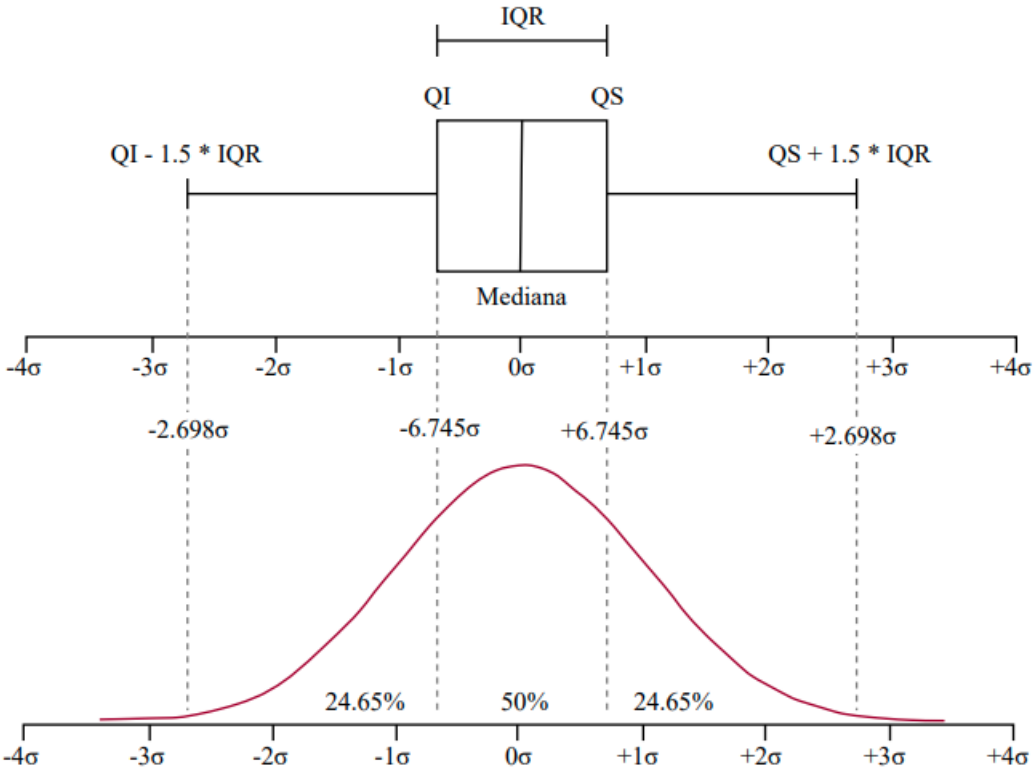
Conforme definido por Alla e Adari (2019), o método IQR é uma abordagem de detecção de *outlier* não supervisionada. Ou seja, não requer o uso de rótulos ou informações externas para identificar os *outliers*. Em vez disso, o foco está na análise da distribuição dos dados e na identificação de valores que se afastem significativamente do comportamento esperado.

O IQR é utilizado para classificar os dados da distribuição em duas categorias: *outlier* ou normal. Esses rótulos foram posteriormente utilizados como conjunto de treinamento, validação e teste para avaliar o desempenho da LSTM em classificar os *outliers*.

3.3.3 Escalonamento

O escalonamento ajusta os dados para uma escala de 0 a 1, conforme a Equação (3.4). Isso é importante para que a RNA possa lidar com a magnitude de todas as variáveis em uma mesma escala, evitando problemas relacionados à diferença de unidades de medida. Além disso, o escalonamento ajuda a escalonar os valores para a faixa de variação dinâmica das funções de ativação, prevenindo a saturação dos neurônios e garantindo um melhor desempenho da RNA na detecção de *outliers*. Os valores de treinamento, validação e teste passam por esse escalonamento, o qual é realizado considerando os valores mínimos

Figura 10 – IQR com primeiro e terceiro quartil em uma distribuição normal com margem de 1,5.



Fonte: NAMETALA, 2023.

e máximos históricos dos dados (DA SILVA; SPATTI; FLAUZINO, 2016; NAMETALA, 2023).

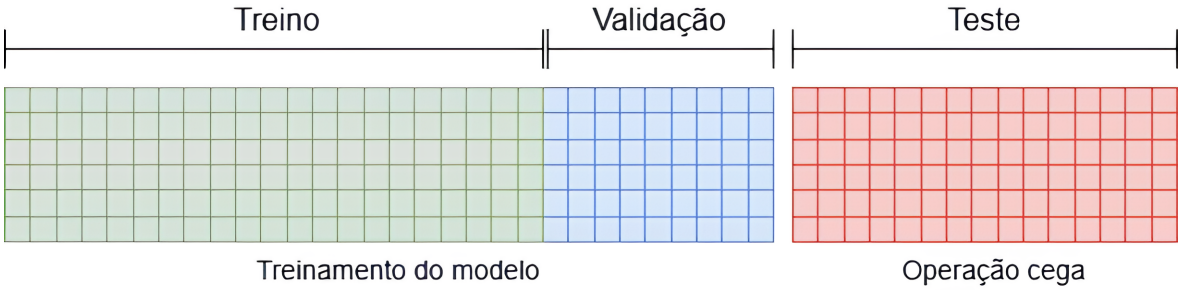
$$x_{\text{novo}} = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{3.4}$$

Na fórmula apresentada, x representa o valor original da variável que está sendo padronizada, $\min(x)$ é o valor mínimo do *dataset* e $\max(x)$ é o valor máximo. O resultado da fórmula, x_{novo} representa o valor dimensionado da variável, ajustado para uma escala de 0 a 1.

3.3.4 Separação dos conjuntos

A etapa de divisão é realizada considerando o período dos eventos conhecidos, com os dados de treinamento e validação selecionados para os anos de 2020 e 2021 e os dados de teste reservados para o ano de 2022. A Figura 11 ilustra essa divisão. Os dados de treino e validação são usados para treinar o modelo, permitindo que a RNA aprenda com esses dados e ajuste seus parâmetros. Por sua vez, os dados de teste são usados para avaliar o desempenho do modelo, sem que este modelo tenha tido conhecimento prévio deles durante o treinamento.

Figura 11 – Divisão dos conjuntos treino, validação e teste.



Fonte: Elaborado pelo Autor, 2023.

3.3.5 Tamanho de Janela Temporal

A última etapa do pré-processamento consiste em realizar o janelamento que representa o número de dados que o modelo utiliza para realizar a classificação. No contexto deste trabalho, o conjunto foi convertido para uma janela de tamanho 168. Isso significa que o modelo levará em consideração os dados das últimas 168 horas, ou seja, a última semana, para classificar o valor de carga mais recente como *outlier* ou não. Cabe ressaltar que o modelo não realiza a previsão da ocorrência de *outlier*, mas sim classifica a última observação, conforme Tabela 1. É importante notar que o janelamento foi realizado de maneira separada para os conjunto de treino, validação e teste.

Tabela 1 – Exemplo do *dataset* divido em janelas com tamanho de 168.

Janela de Carga	Rótulo *outlier*
$[c_0, c_1, \ldots, c_{167}]$	$is\_outlier_{167}$
$[c_1, c_2, \ldots, c_{168}]$	$is\_outlier_{168}$
$[c_2, c_3, \ldots, c_{169}]$	$is\_outlier_{169}$
$\vdots$	$\vdots$
$[c_{26135}, c_{26136}, \ldots, c_{26303}]$	$is\_outlier_{26303}$

Fonte: Elaborado pelo Autor, 2023.

3.4 Construção do Modelo

Após o pré-processamento dos dados, o modelo LSTM foi desenvolvido utilizando a linguagem de programação Python versão 3.10 (PYTHON, 2023), juntamente com as bibliotecas *Keras* 3.10 (KERAS, 2023) e *TensorFlow GPU* 2.10 (TENSORFLOW, 2023). *Keras* é uma *Application Programming Interface* (API), desenvolvida sobre o *TensorFlow* que facilita a implementação da RNA.

Essas bibliotecas permitem ajustar uma quantidade de hiperparâmetos que determinam tanto a arquitetura quanto a maneira que a RNA é treinada. Neste trabalho, a topologia e os valores para os hiperparâmetros foram determinados por meio de uma avaliação empírica, considerando a variação do número de camadas, número de neurônios,

tamanho do lote e função de ativação. Vale salientar que enquanto variava um parâmetro, os outros eram mantidos fixos, verificando-se, dessa forma, o efeito.

3.4.1 Topologia

A melhor topologia encontrada para o modelo baseado em LSTM é apresentada na Tabela 2. O modelo possui uma camada de entrada LSTM com 64 neurônios, seguida por duas camadas escondidas: uma LSTM com 32 neurônios e outra densa com oito neurônios. Por fim, a camada de saída possui um neurônio, responsável por classificar os *outliers*. As camadas densas são responsáveis por ajustar os formatos da saída. Por fim, foi utilizado o inicializador *LecunNormal*, sendo utilizado *seed* igual a 43.

Tabela 2 – Topologia do modelo baseado em LSTM.

N.º camada	Tipo camada	N.º neurônios	Função de ativação	N.º parâmetros
1	LSTM	64	Tangente hiperbólica	16896
2	LSTM	32	Tangente hiperbólica	12416
3	Densa	8	Sigmoid	264
4	Densa	1	Sigmoid	9

Fonte: Elaborado pelo Autor, 2023.

3.4.2 Hiperparâmetros

Os hiperparâmetros são valores utilizados para controlar o processo de aprendizado das RNAs. Eles diferem dos parâmetros de peso dos neurônios, os quais são ajustados durante o treinamento dos modelos. Os hiperparâmetros devem ser definidos manualmente e sua escolha pode ter um impacto significativo no desempenho do modelo. A Tabela 3 mostra os melhores hiperparâmetros encontrados na fase de parametrização.

Tabela 3 – Hiperparâmetros utilizados no modelo baseado em LSTM.

Hiperparâmetros	Valores
Tamanho *batch*	32
Número de épocas	200
Otimizador	Adam
Taxa de aprendizado	0,001

Fonte: Elaborado pelo Autor, 2023.

O otimizador Adam (TENSORFLOW, 2023b) é um otimizador de aprendizado profundo que foi escolhido devido à sua capacidade de ajustar a taxa de aprendizado individualmente para cada parâmetro da RNA (NAMETALA, 2023).

3.5 Classificação de *Outliers*

Na área da estatística, é frequente utilizar a matriz de confusão para quantificar a concordância entre os valores previstos pela RNA e os rótulos fornecidos. O Quadro

1 apresenta um exemplo de matriz de confusão. Essa matriz oferece uma visão geral dos resultados do modelo, de modo que permite a análise de quatro elementos principais (ALLA; ADARI, 2019):

- Verdadeiro Positivo (VP): quando o valor real é positivo, e o valor previsto é positivo;
- Verdadeiro Negativo (VN): quando o valor real é negativo, e o valor previsto é negativo;
- Falso Negativo (FN): Quando o valor real é positivo, e o valor previsto é negativo;
- Falso Positivo (FP): Quando o valor real é negativo, e o valor previsto é positivo.

Portanto, esse recurso é útil em problemas de classificação binária, como a detecção de *outliers*, por auxiliar na compreensão do desempenho do modelo em relação à classificação correta dos dados (SOKOLOVA; LAPALME, 2009; ALLA; ADARI, 2019).

Quadro 1 – Exemplo de uma Matriz de Confusão.

	Valores Previstos	
Valores Reais	Classe Positiva	Classe Negativa
Classe Positiva	VP	FN
Classe Negativa	FP	VN

Fonte: Elaborado pelo Autor, 2023.

As métricas de avaliação de classificação devem priorizar a classe mais relevante do problema, a qual frequentemente pode ser a classe minoritária (BRZEZINSKI *et al.*, 2018). Este trabalho trata de um problema supervisionado de classificação, com o objetivo de detectar observações classificadas como *outliers* em uma série temporal, no qual o número de *outliers* é significativamente inferior em relação às observações normais, resultando em dados desbalanceados (HE; MA, 2013; MIGLIATO, 2021). Portanto, para avaliar adequadamente o desempenho do modelo, é fundamental que as medidas de avaliação priorizem a classe minoritária. Nesse cenário, as métricas recomendadas que utilizam a matriz de confusão são *precision* e *recall*, segundo Migliato (2021) e Sokolova e Lapalme (2009).

A métrica *precision* avalia a proporção de observações corretamente classificadas como verdadeiras positivas em relação a todos os pontos de dados da classe positiva prevista. Portanto, *precision* responde à pergunta de quantos dos pontos de dados verdadeiros positivos foram previstos corretamente pelo modelo. Simultaneamente, *recall* descreve a quantidade de previsões verdadeiras identificadas corretamente em relação aos pontos de dados positivos reais, refletindo a capacidade do modelo de encontrar e identificar todos os casos positivos corretos (ALLA; ADARI, 2019; BRZEZINSKI *et al.*, 2018; MIGLIATO, 2021). Dessa forma, as fórmulas de *precision* e *recall* são derivadas da matriz de confusão, sendo representadas pelas Equações (3.5) e (3.6) (ALLA; ADARI, 2019).

$$Precision = \frac{\text{VP}}{\text{VP} + \text{FP}} \tag{3.5}$$

$$Recall = \frac{\text{VP}}{\text{VP} + \text{FN}} \tag{3.6}$$

3.6 Materiais e Tecnologias

Para a realização do experimento, é recomendado o uso de um computador com sistema operacional Linux ou Windows 10, contendo um processador de 6 núcleos da Intel ou AMD com velocidade de 3,5 GHz, com 8 GB de memória RAM disponível e uma placa de vídeo com pelo menos 4GB de memória compartilhada. Essa recomendação decorre do fato de que os resultados apresentados no capítulo a seguir foram gerados fazendo uso deste *setup* de configuração.

Quanto às tecnologias, é necessário ter o Python na versão 3.10 (PYTHON, 2023), instalado para a implementação dos algoritmos durante as etapas de pré-processamento e classificação de *outliers*. Na fase de pré-processamento, a biblioteca *scikit-learn* (LEARN, 2023) foi empregada, principalmente nas etapas de separação dos conjuntos de dados. Adicionalmente, para a construção do modelo, foram utilizadas as bibliotecas *Keras* na versão 3.10 (KERAS, 2023) e *TensorFlow GPU* na versão 2.10.

Após a apresentação dos procedimentos metodológicos a serem seguidos, os resultados obtidos neste trabalho são descritos.

4 RESULTADOS E DISCUSSÃO

Os resultados deste capítulo dizem respeito a curva de carga horária do subsistema SECO. Foram selecionados dados compreendidos entre o período de 01/01/2020 a 31/12/2022 (1096 dias e 26304 horas). Os anos de 2020 e 2021 foram utilizados para treino e validação no modelo baseado em LSTM. O ano de 2022 diz respeito ao conjunto de teste que nunca foi apresentado ou utilizado em nenhuma das fases de treino ou validação.

4.1 Tratamentos dos dados

Na etapa de tratamento dos dados são mostrados os resultados obtidos por meio da Etapa 2, a qual está detalhada na metodologia.

4.1.1 Tratamento de integridade e Rotulagem dos dados

Nenhuma lacuna ou inconsistência foi identificada durante a análise de integridade dos dados. Portanto, a etapa seguinte consistiu na aplicação da rotulagem. Para realizar os rótulos dos *outliers* no *dataset*, foi utilizado o método IQR com uma margem correspondente a 0,5 vezes para limite Ls (superior) e Li (inferior). Os limites obtidos para a margem utilizada são apresentados na Tabela 4, na qual também são sumarizados os valores máximos, mínimos e médios da carga registrada no período de interesse.

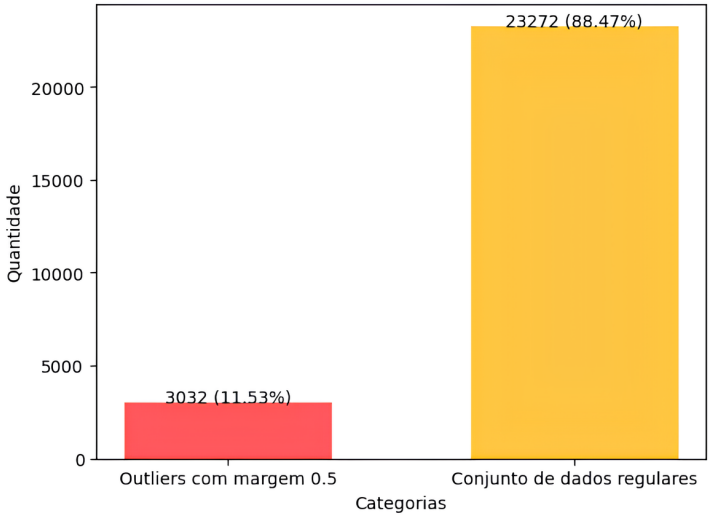
Tabela 4 – Sumarização do *dataset* com limites superior e inferior.

Variável	Máximo	Mínimo	Média	Ls	Li
Carga MWh	52571	21658	38389	47328	29675

Fonte: Elaborado pelo Autor, 2023.

A contagem dos dados rotulados são apresentados na Figura 12. Sob essa margem predefinida, foram observados 3032 registros identificados como *outliers*.

Figura 12 – Comparação entre a quantidade de valores atípicos e observações regulares.



Fonte: Elaborado pelo Autor, 2023.

Com os dados rotulados, as etapas subsequentes consistiram em melhorar a representação do conjunto de dados para ser utilizado no treinamento da RNN.

4.1.2 Escalonamento e Janelamento temporal

A Equação (3.4) foi utilizada para escalonar os dados no intervalo de 0 a 1. Para realizar o escalonamento, um valor máximo de 55.000 MWh e um valor mínimo de 19.000 MWh foram considerados. Este passo foi feito para estabelecer uma faixa específica de variação, garantindo que a RNA não fosse enviesada para a faixa dinâmica do conjunto de teste.

Por último, foi empregada a técnica de janelamento. Como resultado dessa abordagem, um total de 25969 janelas foram obtidas, considerando o escalonamento de 168 horas para o conjunto de treino e teste realizado de maneira separada. A Tabela 5 oferece uma descrição da divisão do *dataset*, utilizado como entrada para o modelo baseado em LSTM.

Tabela 5 – *Datasets* de treino, validação e teste. Os *datasets* X e Y possuem 3 dimensões, sendo elas: número total de amostras, número de elementos em cada observação e quantidade de característica por observação.

Dataset	X (entrada)	Y (saída)
$[X_{treino}, Y_{treino}]$	[12163, 168, 1]	[12163, 1, 1]
$[X_{validacao}, Y_{validacao}]$	[5213, 168, 1]	[5213, 1, 1]
$[X_{teste}, Y_{teste}]$	[8593, 168, 1]	[8593, 1, 1]

Fonte: Elaborado pelo Autor, 2023.

4.1.3 Divisão do dataset

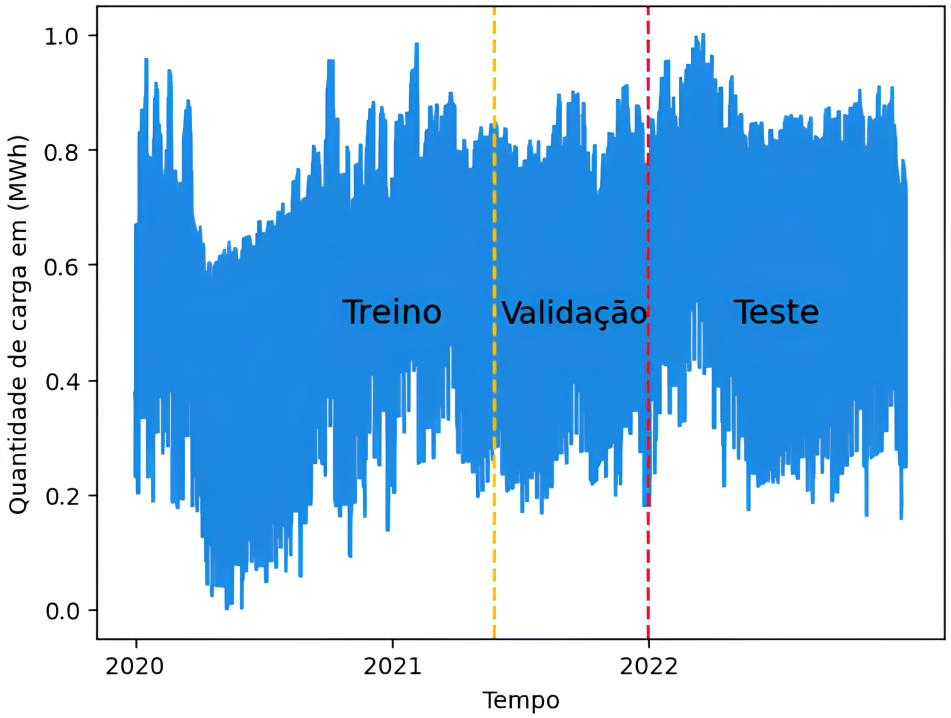
Os conjuntos de treino e validação agregaram 17377 horas (724 dias), sendo que a validação compreendeu 5213 horas (217 dias). O conjunto de teste possuiu 8593 horas (358 dias). Os conjuntos de treinamento, validação e teste corresponderam a parcelas de 46,66%, 20%, 33,33% do conjunto total de dados, conforme detalhado na Tabela 6. A Figura 13 mostra os dados divididos na série temporal de carga. As linhas pontilhadas ilustram a proporção dos dados para cada etapa.

Tabela 6 – Divisão do *dataset*.

Conjunto	início	Fim	Horas	Dias	Parcela
Treino e Validação	07/01/2020	31/12/2021	17377	724	66.66%
Treino	07/01/2020	30/05/2021	12497	509	46.66%
Validação	30/05/2021	02/01/2022	5213	217	20%
Teste	07/01/2022	31/12/2022	8593	358	33.33%

Fonte: Elaborado pelo Autor, 2023.

Figura 13 – Gráfico com as regiões de treino, validação e teste.

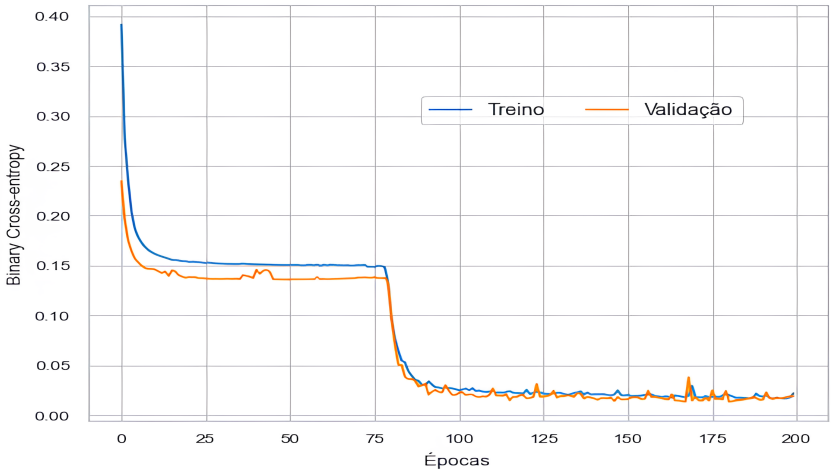


Fonte: Elaborado pelo Autor, 2023.

4.2 Treinamento do modelo

O modelo LSTM passou por 200 épocas de treinamento, tendo como critério de parada a técnica de *Early Stopping*, que determina que o treinamento deve ser interrompido quando não houver melhorias na função de custo por 100 épocas, conforme foi definido. A Figura 14 ilustra a evolução dos erros ao longo das épocas, comparando os conjuntos de treinamento e validação. No treinamento, o erro mínimo atingiu 16×10^{-3}. Na validação, o erro cai para 13×10^{-3}. Foi também utilizada a função *ModelCheckpoint* do *Keras* para salvar os pesos apenas do melhor modelo.

Figura 14 – Evolução dos erros baseado em *Binary Cross-entropy* ao longo de 200 épocas.



Fonte: Elaborado pelo Autor, 2023.

4.3 Avaliação do modelo

Por fim, a previsão foi realizada utilizando-se o modelo treinado. Os resultados estão apresentados no Quadro 2, que exibe os valores reais e previstos na matriz de confusão. A classe positiva refere-se aos *outliers*, enquanto a classe negativa corresponde aos valores de carga normais.

Quadro 2 – Resultado da previsão do modelo treinado.

	Valores Previstos	
Valores Reais	Classe Positiva	Classe Negativa
Classe Positiva	879	39
Classe Negativa	17	7658

Fonte: Elaborado pelo Autor, 2023.

Ao realizar a análise das métricas de *precision* e *recall*, os resultados obtidos foram de 98% e 96%, respectivamente, de acordo com a Tabela 7. Portanto, isso mostra que o modelo apresenta capacidade de classificar corretamente a maioria das observações como *outliers*, inclusive, de forma bastante semelhante ao método IQR.

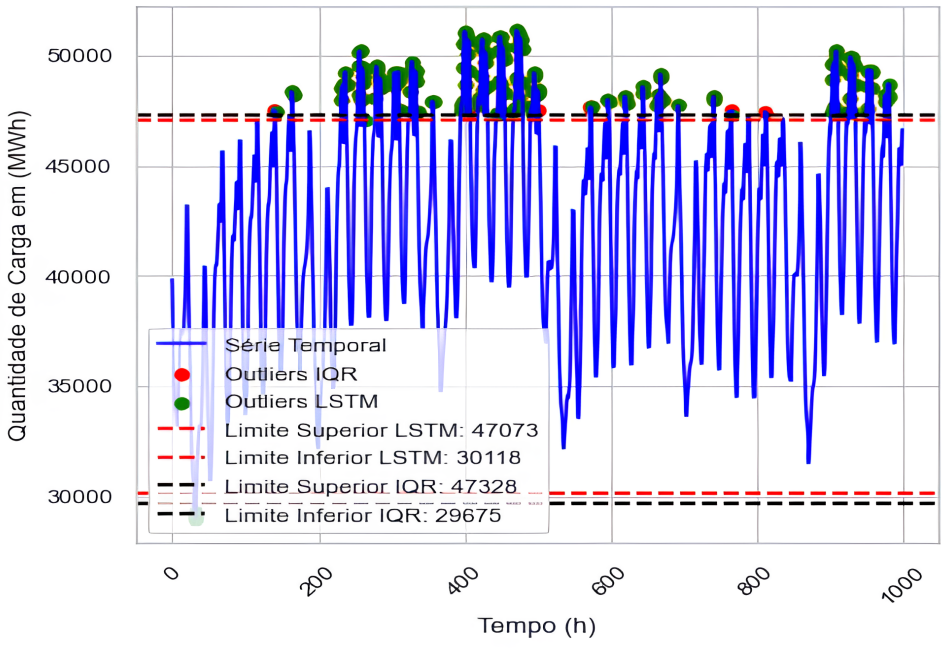
Tabela 7 – Sumarização do resultado de *precision* e *recall*.

Método	Resultado
Precision	98%
Recall	96%

Fonte: Elaborado pelo Autor, 2023.

A Figura 15 exibe a série temporal com marcadores indicando os *outliers* identificados tanto pelo método IQR quanto pela LSTM, nas primeiras 1000 horas.

Figura 15 – Série temporal com *outliers* identificados para o IQR e LSTM.



Fonte: Elaborado pelo Autor, 2023.

Pode-se perceber que a RNA acertou a maioria das observações, com alguns erros localizados na região limite entre valores atípicos e normais. De maneira distinta, a LSTM ampliou os limiares para a classificação de observações como *outliers* e normais.

Os resultados demonstram que o modelo LSTM apresenta, portanto, resultados positivos na detecção de *outliers* na série temporal de carga horária do subsistema SECO. Por meio da análise das métricas *precision* e *recall*, constata-se que o modelo consegue identificar a maioria das observações corretamente, revelando sua capacidade de discernir entre valores atípicos e normais. Embora algumas discrepâncias tenham sido notadas em comparação com o método IQR, essas diferenças podem ter ocorrido pelo fato de que a LSTM considera a sazonalidade na sua classificação, já o IQR, não.

5 CONCLUSÃO

Neste estudo foi utilizada uma RNA baseada em células LSTM para a detecção de *outliers* na série temporal de carga horária do subsistema SECO do SIN. O foco principal foi identificar momentos que poderiam impactar a sazonalidade da carga, com o intuito de fornecer informações úteis para a gestão de energia ao indicar observações como *outliers* no contexto temporal.

A estratégia adotada foi baseada nas etapas definidas nos objetivos específicos, que se inicia com a criação de um *dataset* abrangendo um período de pelo menos três anos, abarcando os anos de 2020 e 2021 os quais, respectivamente, foram marcados pela pandemia da COVID-19 e pela maior crise de abastecimento no SIN nos últimos 91 anos, seguido pelo ano de 2022 utilizado para teste. Em seguida, a RNA baseada em células LSTM foi implementada e treinada. Por fim, sua capacidade de detecção de *outliers* foi avaliada com as métricas *precision* e *recall*, sendo os resultados comparados com o método estatístico IQR, também utilizado para realizar a rotulagem de parte do *dataset*.

Os resultados obtidos foram positivos. A RNA conseguiu classificar a maioria dos *outliers*, com métricas de *precision* e *recall* atingindo 98% e 96%, respectivamente. Isso confirma a habilidade do modelo em distinguir entre valores atípicos e normais na série temporal de carga horária. Apesar de algumas discrepâncias em relação ao método IQR, essas diferenças podem ser explicadas pela consideração do contexto temporal na abordagem da LSTM. Do ponto de vista prático, não há como afirmar neste estudo os motivos da existência de um *outlier*. Nesse sentido, ao considerar que a LSTM leva em conta o contexto sazonal e o IQR não, pode-se especular que sua classificação é mais precisa.

Um aspecto importante do uso de uma LSTM nesse contexto reside na sua capacidade de classificar valores futuros sem a necessidade de processar todo o *dataset*. Isso significa que a LSTM tem a capacidade de avaliar e identificar *outliers* à medida que novos dados são apresentados, permitindo uma abordagem mais eficiente e dinâmica na detecção dos *outliers*.

Como possíveis continuações a este trabalho sugere-se:

- Conduzir uma análise das causas subjacentes às ocorrências de *outliers*, levando em conta os contextos temporais e eventos relevantes;
- Utilizar outros períodos de tempo para realizar o treinamento e teste do modelo;
- Utilizar outros valores de margem para o IQR;
- Expandir a classificação para outros subsistemas elétricos;
- Realizar uma comparação do desempenho com outros tipos de RNNs, como as *Gated Recurrent Unit* (GRU);

- Realizar a detecção dos *outliers* com outras técnicas como o método *RandomForest* e comparar a previsão com a LSTM.

Assim este trabalho cumpre com os objetivos propostos, alcançando resultados positivos na avaliação do modelo.

REFERÊNCIAS

ALLA, S.; ADARI, S. K. **Beginning anomaly detection using python-based deep learning**: With Keras and PyTorch. New Jersey: Apress, 2019.

BOOKS, G. **Google Ngram Viewer**. 2023. Disponível em: https://bit.ly/3tfjUPr. Acesso em: 15 mai. 2023.

BOYLESTAD, R. L. **Introdução à Análise de Circuitos**. 12. ed. São Paulo: Pearson Prentice Hall, 2012.

BRAGA, A. d. P.; LUDERMIR, T. B.; CARVALHO, A. C. P. d. L. F. **Redes Neurais Artificiais**: teoria e aplicações. LTC, 2007.

BRUCE, P.; BRUCE, A. **Estatística prática para cientistas de dados**. Rio de Janeiro: Alta Books, 2019.

BRZEZINSKI, D. *et al.* Visual-based analysis of classification measures and their properties for class imbalanced problems. **Information Sciences**, Elsevier BV, v. 462, p. 242–261, 2018. DOI: 10.1016/j.ins.2018.06.020.

CAMPOS, R. J. **Previsão de séries temporais com aplicações a séries de consumo de energia elétrica**. 2008. Dissertação (Mestrado em Engenharia Elétrica) – Universidade Federal de Minas Gerais (UFMG), Belo Horizonte, 2008. Disponível em: http://hdl.handle.net/1843/BUOS-8CTETD. Acesso em: 13 mai. 2023.

CHAHLA, C. *et al.* A Novel Approach for Anomaly Detection in Power Consumption Data. **ICPRAM**, p. 483–490, 2019.

CHOLLET, F. **Deep learning with Python**. Manning, 2021.

DA SILVA, I. N.; SPATTI, D. H.; FLAUZINO, R. A. **Redes neurais artificiais para engenharia e ciências aplicadas**. 2. ed. São Paulo: Artliber, 2016.

DASH, C. S. K. *et al.* An outliers detection and elimination framework in classification task of data mining. **Decision Analytics Journal**, p. 100164, 2023. DOI: 10.1016/j.dajour.2023.100164.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. MIT press, 2016. Disponível em: https://www.deeplearningbook.org/. Acesso em: 16 mai. 2023.

GUIRELLI, C. R. **Previsão da carga de curto prazo de áreas elétricas através de técnicas de inteligência artificial**. 2006. Tese (Doutorado em Engenharia) – Universidade de São Paulo (USP), São Paulo, 2006. DOI: 10.11606/T.3.2006.tde-19042007-142653.

HAYKIN, S. **Redes Neurais**: princípios e prática. 2. ed. Porto Alegre: Bookman, 2001.

HE, H.; MA, Y. **Imbalanced learning**: foundations, algorithms, and applications. 2013. v. 1.

HIMEUR, Y. *et al.* Artificial intelligence based anomaly detection of energy consumption in buildings: A review, current trends and new perspectives. **Applied Energy**, v. 287, p. 116601, 2021. ISSN 0306-2619. DOI: 10.1016/j.apenergy.2021.116601.

HODGE, V. J.; AUSTIN, J. A survey of outlier detection methodologies. **Artificial intelligence review**, Springer, v. 22, p. 85–126, 2004.

KERAS. **Keras Documentation**. 2023. Disponível em: https://keras.io/. Acesso em: 9 nov. 2023.

LEARN, S. **Scikit-learn Documentation**. 2023. Disponível em: https://scikit-learn.org/stable/. Acesso em: 18 dez. 2023.

LI, C. *et al.* Detection of Outliers in Time Series Power Data Based on Prediction Errors. **Energies**, v. 16, n. 2, 2023. ISSN 1996-1073. DOI: 10.3390/en16020582.

MARTINS, M. E. G. Percentis. **Revista de Ciência Elementar**, Casa das Ciências, v. 2, n. 3, 2014.

MIGLIATO, A. L. T. **Detecção de Outliers em Dados não Vistos de Séries Temporais por meio de Erros de Predição com SARIMA e Redes Neurais Recorrentes LSTM e GRU**. 2021. Dissertação (Mestrado em Matemática, Estatística e Computação) - Instituto de Ciências Matemáticas e de Computação – Universidade de São Paulo, São Carlos, 2021. DOI: 10.11606/D.55.2021.tde-21012022-175531.

MINGOTI, S. A. **Análise de dados através de métodos de estatística multivariada**: uma abordagem aplicada. 1. ed. Belo Horizonte: Editora UFMG, 2007.

MORETTIN, P. A.; TOLOI, C. M. **Análise de séries temporais**: modelos lineares univariados. 3. ed.: Editora Blucher, 2018.

MOTA, C. N. S. **Previsão de Cargas Elétricas usando Backpropagation Estocástico**. 2021. Dissertação (Mestrado em Engenharia Elétrica) – Universidade Estadual Paulista (UNESP), São Paulo, 2021. Disponível em: http://hdl.handle.net/11449/210841. Acesso em: 13 mai. 2023.

NAMETALA, C. A. L. **Redes Neurais Atencionais aplicadas a modelagem e previsão de preços no Mercado de Eletricidade Brasileiro**. 2023. Tese (Doutorado em Engenharia Elétrica) – Universidade de São Paulo, São Carlos, 2023. DOI: 10.11606/T.18.2023.tde-16032023-161345.

NAMETALA, C. A. L. *et al.* Analysis of hourly price granularity implementation in the Brazilian deregulated electricity contracting environment. **Utilities Policy**, v. 81, p. 101513, 2023. DOI: 10.1016/j.jup.2023.101513. Acesso em: 13 mai. 2023.

NASCIMENTO, G. F. M. *et al.* Outlier Detection in Buildings' Power Consumption Data Using Forecast Error. **Energies**, v. 14, n. 24, 2021. ISSN 1996-1073. DOI: 10.3390/en14248325.

OMAR, S.; NGADI, A.; JEBUR, H. H. Machine Learning Techniques for Anomaly Detection: An Overview. **International Journal of Computer Applications**, v. 79, 2013. DOI: 10.5120/13715-1478.

ONS. **Histórico da Operação**: Curva de Carga Horária. 2023b. Disponível em: https://www.ons.org.br/Paginas/resultados-da-operacao/historico-da-operacao/curva_carga_horaria.aspx. Acesso em: 13 mai. 2023.

_____. **O QUE É O SIN**. 2023. Disponível em: https://www.ons.org.br/paginas/sobre-o-sin/o-que-e-o-sin. Acesso em: 13 mai. 2023.

_____. **O QUE É ONS**. 2023a. Disponível em: https://www.ons.org.br/paginas/sobre-o-ons/o-que-e-ons. Acesso em: 13 mai. 2023.

_____. **ONS**: Afluência estimada no Sudeste/Centro-Oeste é de 116% da MLT na primeira previsão de 2023. 2022. Disponível em: https://www.ons.org.br/Paginas/Noticias/20221230-ONS-Aflu%C3%AAncia-estimada-no-SudesteCentro-Oeste-%C3%A9-de-116-da-MLT-na-primeira-previs%C3%A3o-de-2023.aspx. Acesso em: 13 mai. 2023.

PEFFERS, K. *et al.* A design science research methodology for information systems research. **Journal of Management Information Systems**, v. 24, p. 45–77, 2007.

PIMENTEL, M.; FILIPPO, D.; SANTORO, F. M. Design Science Research: fazendo pesquisas científicas rigorosas atreladas ao desenvolvimento de artefatos computacionais projetados para a educação. **Metodologia de Pesquisa em Informática na Educação: Concepção da Pesquisa**, SBC, Porto Alegre, 2018.

PYTHON. **Portal do Python**. 2023. Disponível em: https://www.python.org/. Acesso em: 22 jul. 2023.

SEVERINO, A. J. **Metodologia do trabalho científico**. 1. ed. São Paulo: Cortex Editora, 2013.

SILVA, A. da *et al.* A Method for Anomaly Prediction in Power Consumption using Long Short-Term Memory and Negative Selection. **2019 IEEE International Symposium on Circuits and Systems (ISCAS)**, p. 1–5, 2019. DOI: 10.1109/ISCAS.2019.8702152.

SOKOLOVA, M.; LAPALME, G. A systematic analysis of performance measures for classification tasks. **Information processing & management**, Elsevier, v. 45, n. 4, p. 427–437, 2009.

SOUSA, T. A.; MARQUES, A. B. LEARN Board Game: A game for teaching Software Architecture created through Design Science Research, p. 834–843, 2020.

TENSORFLOW. **Adam**. 2023b. Disponível em: https://www.tensorflow.org/api_docs/python/tf/keras/optimizers/Adam. Acesso em: 18 dez. 2023.

_____. **GPU support**. 2023. Disponível em: https://www.tensorflow.org/?hl=pt-br. Acesso em: 9 nov. 2023.

VISHWAKARMA, G. K.; PAUL, C.; ELSAWAH, A. An algorithm for outlier detection in a time series model using backpropagation neural network. **Journal of King Saud University - Science**, v. 32, p. 3328–3336, 2020. DOI: 10.1016/j.jksus.2020.09.018.

WANG, H.; BAH, M. J.; HAMMAD, M. Progress in Outlier Detection Techniques: A Survey. **IEEE Access**, v. 7, p. 107964–108000, 2019. DOI: 10.1109/ACCESS.2019.2932769.

WAZLAWICK, R. S. **Metodologia de pesquisa para ciência da computação**. Elsevier, 2009. v. 2.

WOOLDRIDGE, J. M. **Introductory econometrics**: a modern approach. 6. ed.: Cengage Learning, 2015.