Predição de Emissão de Gases de Exaustão de Turbinas em Termelétricas usando Redes Neuronais e Modelos Híbridos

Antonio Rocha Azevedo

Projeto de Final de Curso

Orientadores

Prof. Argimiro Resende Secchi

Prof. Bruno Didier Olivier Capron

Eng. Ataíde Souza Andrade Neto

Julho de 2022

PREDIÇÃO DE EMISSÃO DE GASES DE EXAUSTÃO DE TURBINAS EM TERMELÉTRICAS USANDO REDES NEURONAIS E MODELOS HÍBRIDOS

Antonio Rocha Azevedo

Projeto de Final de Curso submetido ao Corpo Docente da Escola de Química como parte dos requisitos necessários à obtenção do grau de Engenheiro Químico.

Aprovado por:

_____

Prof. Maurício Bezerra de Souza Jr., D.Sc.

_____

Prof. André Ferreira Young, D.Sc.

Orientado por:

_____

Prof. Argimiro Resende Secchi, D. Sc.

_____

Prof. Bruno Didier Olivier Capron, D. Sc.

_____

Eng. Ataíde Souza Andrade Neto, M. Sc.

Rio de Janeiro, RJ - Brasil
Julho de 2022

Azevedo, Antonio Rocha.
Predição de emissão de gases de exaustão de turbinas em termelétricas usando redes neuronais e modelos híbridos / Antonio Rocha Azevedo. Rio de Janeiro: UFRJ/EQ, 2022.
xi, 46 p.; il. (Monografia) - Universidade Federal do Rio de Janeiro, Escola de Química, 2022.
Orientadores: Argimiro Resende Secchi, Bruno Didier Olivier Capron e Ataíde Souza Andrade Neto.
1. Emissão de Gases. 2. Redes Neuronais. 3. Modelos Híbridos. 4. Monografia (Graduação - UFRJ/EQ). 5. Argimiro Resende Secchi, Bruno Didier Olivier Capron e Ataíde Souza Andrade Neto. I. Predição de emissão de gases usando redes neuronais e modelos híbridos.

Citação

Se eu vi mais longe, foi por estar sobre ombros de gigantes.

Isaac Newton

AGRADECIMENTOS

A meus pais e avós por todo o suporte, incentivo e apoio em minha vida pessoal e minha jornada profissional.

Aos professores André Ferreira Young e Heloisa Lajas Sanches Fernandes pelas maravilhosas aulas, experiência e apoio ao longo da minha graduação.

Aos professores Argimiro Resende Secchi e Bruno Didier Olivier Capron e Ataíde Souza Andrade Neto pela oportunidade de realizar este Projeto de Final de Curso em um tema que me interessa tanto.

Resumo do Projeto de Final de Curso apresentado à Escola de Química como parte dos requisitos necessários para obtenção do grau de engenheiro químico.

PREDIÇÃO DE EMISSÃO DE GASES DE EXAUSTÃO DE TURBINAS EM TERMELÉTRICAS USANDO REDES NEURONAIS E MODELOS HÍBRIDOS

Antonio Rocha Azevedo

Julho, 2022

Orientadores: Prof. Argimiro Resende Secchi, D. Sc.
Prof. Bruno Didier Olivier, D. Sc.
Eng. Ataíde Souza Andrade Neto, M. Sc.

O monitoramento de emissões é necessário em toda planta industrial que libere gases nocivos ao meio-ambiente, tendo em vista a intensificação das mudanças climáticas e o consequente recrudescimento da legislação vigente. Sistemas de Monitoramento Contínuo de Emissões (CEMS) são o padrão utilizado hoje em dia para essa quantificação, mas possuem alto custo de aquisição e manutenção, além de perderem sua sensibilidade ao longo do tempo. Assim, cresce o interesse no desenvolvimento de Sistemas Preditivos de Monitoramento de Emissões (PEMS), que se baseiam em modelos matemáticos para se prever a concentração dos gases de exaustão — trazendo benefícios econômicos e de operação. Neste trabalho, estuda-se o desenvolvimento de diferentes arquiteturas de redes neuronais para uso em PEMS: particularmente na predição da emissão de gases (NO_x, O_2 e CO) de uma usina termelétrica. Duas abordagens são estudadas: uma puramente baseada em dados — estimação direta da emissão dos gases pela rede neuronal — e outra híbrida — onde a rede é acoplada a um modelo fenomenológico, de maneira a prever seu erro de estimação. A segunda abordagem é relevante pois aumenta a aderência do modelo fenomenológico já existente aos dados da planta, além de facilitar e acelerar o processo de aprendizado dos modelos baseados em dados, devendo aprender mais rapidamente. Um conjunto de dados de 2015 para uma turbina a gás foram utilizados para o treino e validação dos modelos, utilizando-se uma separação de 70%/30% (respectivamente). Todas as redes obtiveram resultados similares na estimação de NO_x e O_2 em ambas as abordagens, embora os modelos híbridos tenham de fato aprendido mais facilmente. No entanto, nenhuma rede obteve êxito na estimação do CO. A dificuldade no aprendizado dos modelos nessa estimação mostra que não é possível traçar conclusões globais acerca da melhor arquitetura ou tipo de rede, que pode variar de acordo com a variável estimada. A proximidade da precisão dos modelos nos leva a crer que seu desempenho foi limitado pela quantidade de dados disponível para o aprendizado, dado à grande complexidade do problema. A recente obtenção de dados para uma faixa de 10 anos e 6 turbinas é um bom ponto de partida para análises mais robustas e conclusivas, embora deva-se atentar às faixas de operação usadas no treino assim como à qualidade dos dados.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

NOTAÇÃO

LETRAS MINÚSCULAS: escalares

Símbolo	**Descrição**
x	Variável de entrada
y	Variável de saída
$\hat{y}$	Estimação de uma variável de saída
w	Parâmetro (coeficiente/peso)
b	Parâmetro (viés)
f	Função de ativação
l	Função erro
η	Taxa de aprendizagem

LETRAS MINÚSCULAS EM NEGRITO: vetores

Símbolo	**Descrição**
$\mathbf{x}$	Vetor de entradas
$\mathbf{w}$	Vetor de coeficientes
$\boldsymbol{\theta}^{(i)}$	Vetor de parâmetros do modelo na iteração i

LETRAS MAIÚSCULAS: conjuntos

Símbolo	**Descrição**
D	Conjunto de dados de treino

LETRAS MAIÚSCULAS EM NEGRITO: matrizes

Símbolo	**Descrição**
$\mathbf{X}$	Matriz de dados de entrada
$\mathbf{Y}$	Matriz de saídas
$\mathbf{W}^{(l)}$	Matriz número l de parâmetros (coeficientes)

Capítulo 1

Introdução

1.1 Contexto

O presente Projeto de Final de Curso se insere no projeto de Pesquisa e Desenvolvimento de Modelos Matemáticos para Sistema Preditivo de Monitoramento de Emissões (PEMS) de Turbinas a Gás de Usinas Termelétricas, realizado em cooperação entre o Laboratório de Desenvolvimento de Software (LADES) da COPPE e a Petrobras.

Os objetivos dessa colaboração são de desenvolver, implementar e validar um modelo preditivo para monitoramento de emissões que seja capaz de descrever a composição dos gases de exaustão de turbinas em termelétricas ao longo da campanha de operação, como alternativa ao Sistema de Monitoramento Contínuo de Emissões.

1.2 Justificativa

O acompanhamento e a quantificação de emissões é uma tarefa essencial em qualquer atividade industrial que libere gases nocivos ao meio-ambiente. Dados os problemas e riscos causados pela poluição atmosférica, torna-se necessária a limitação da carga emitida em tais processos, ficando a cargo do poder público a definição dos valores permitidos e a punição adequada em caso de sua transgressão. Com a evolução do conhecimento e da conscientização em relação aos efeitos da poluição nos ecossistemas e nas mudanças climáticas, há a tendência de se restringir cada vez mais a concentração de gases que pode ser efetivamente liberada na atmosfera[1–4]. No Brasil, os padrões de qualidade do ar e limites máximos de emissões de poluentes por turbinas a gás são definidos pelas resoluções 419[2] e 436[5] do CONAMA, respectivamente

Estima-se que, no Brasil, cerca de 19% das emissões de gases de efeito estufa em 2019 tenham sido geradas pelo setor energético, atrás apenas das emissões relacionadas ao uso da terra (incluindo o desmatamento) e da agropecuária[6]. Esse valor está diretamente relacionado ao acionamento de termelétricas em situações de aumento de consumo de eletricidade (caso de 2019) ou períodos de baixa produção pelas outras fontes de energia[7]. Esse último caso é particularmente relevante no cenário atual, devido à crise energética causada pela escassez de chuvas e portanto pela baixa produção de energia por vias hidrelétricas, que correspondiam a 65,2% da matriz energética brasileira em 2021[8]. Finalmente, as usinas termelétricas ainda são

responsáveis por cerca de 20% da energia produzida no país[8,9].

Nesse contexto, por mais que o Brasil seja exemplo no uso de energias renováveis, ainda não é possível deixar as usinas termelétricas de lado. Assim, o investimento em novas técnicas e métodos de monitoramento se faz relevante, a fim de controlar as emissões desse tipo de fonte de maneira eficaz no quadro de uma legislação cada vez mais rigorosa.

O padrão de monitoramento e quantificação para fontes fixas utilizado hoje em dia é baseado em um Sistema de Monitoramento Contínuo de Emissões (CEMS), que caracteriza os gases de combustão continuamente[10]. No entanto, os equipamentos necessários para a implementação desse tipo de sistema possuem um alto custo de aquisição e manutenção, além de precisarem ser regularmente recalibrados e terem a tendência a perder a sua sensibilidade com o tempo. Nesse sentido, o desenvolvimento de um Sistema Preditivo de Monitoramento de Emissões (PEMS), que se baseia em modelos matemáticos para se descrever e prever as concentrações dos gases de exaustão a partir de variáveis do processo, traz uma grande facilidade de operação, implantação, manutenção e um custo irrisório[11]. Nos Estados Unidos, um PEMS pode substituir um CEMS pela legislação ambiental estadunidense, caso o mesmo cumpra alguns requerimentos de acurácia[12]. No Brasil, no entanto, ainda não há legislação que valide o licenciamento ambiental de PEMS como alternativa ao CEMS.

Os modelos utilizados em um PEMS podem ser fenomenológicos (utilizando-se de equações, correlações e restrições com interpretação física); baseados em dados (funções do tipo "caixa-preta", i.e. sem interpretação física, que visam somente uma estimação acurada da variável de interesse[13,14]); ou híbridos, integrando ambas as abordagens.

Os modelos físicos têm a vantagem de serem facilmente interpretáveis, terem uma quantidade menor de parâmetros e serem, portanto, mais simples e naturais. No entanto, eles nem sempre integram todas as interações possíveis entre variáveis, podendo ter um desempenho insatisfatório de acordo com a aplicação e condições de operação. Nesse sentido, modelos baseados em dados se destacam pela sua flexibilidade e acurácia, sendo capazes de extrair relações não triviais das informações que lhe são alimentadas. No entanto, costumam requerer uma quantidade de dados extremamente grande para poderem aprender bem. Além disso, sua capacidade de extrapolação é pior quando comparada à de seu equivalente fenomenológico.

Assim, cresce o interesse no uso e desenvolvimento de modelos híbridos, que procuram reunir o melhor dos dois métodos. A integração mais simples consiste em usar um modelo baseado em dados para corrigir a saída do modelo fenomenológico, melhorando sua acurácia. Um benefício desta abordagem é a redução da quantidade de pontos experimentais necessários para se treinar a parte baseada em dados[15]. Isso porque o modelo físico já haverá feito a modelagem mais complexa, restando ao modelo caixa-preta apenas reconhecer as pequenas nuances não levadas em conta pelo primeiro — uma tarefa mais simples.

1.3 Delimitação

Neste trabalho, pretende-se estudar o uso de redes neuronais artificiais para a modelagem baseada em dados. Trata-se de um tipo de modelo que se destaca pela sua grande capacidade de aprendizado e seu êxito em áreas diversas de aplicação como o reconhecimento de imagens e de voz, processamento de linguagem natural, regressão de dados temporais, entre outros[16–23]. Isso se deve, em parte, pela grande variedade de arquiteturas disponíveis, que lhes confere alta

flexibilidade. Assim, deseja-se abordar a diferença entre cada estrutura (explicadas em detalhes nas Seções 2.3 e 2.5) e os benefícios que elas podem trazer ao PEMS.

1.4 Objetivos

Os objetivos do presente trabalho são, portanto, de desenvolver, implementar, testar e comparar diferentes arquiteturas de redes neuronais artificiais (dos tipos *Feed Forward*, convolucionais e recorrentes) tanto para a estimação direta de emissões (de NO_x, O_2 e CO) quanto para a correção da predição do modelo termodinâmico (abordagem híbrida). Ao final, deseja-se discutir e responder: se essas abordagens são de fato capazes de fazer tais previsões com acurácia e precisão adequadas; qual classe (se alguma) de rede se sobressai na tarefa; e verificar se a abordagem híbrida traz de fato vantagens à modelagem e ao aprendizado dos modelos baseados em dados.

1.5 Estrutura

O presente trabalho se estrutura em 5 capítulos, incluindo esta introdução, como o primeiro. O Capítulo 2 trata da revisão bibliográfica sobre o PEMS e da fundamentação teórica atrelada às redes neuronais e aos modelos híbridos — explicando em detalhes o que é uma rede neuronal e os diferentes tipos explorados neste trabalho, assim como sua integração ao modelo fenomenológico. O capítulo também apresenta o algoritmo de aprendizado de máquina utilizado e os cuidados associados. Em seguida, o Capítulo 3 apresenta o modelo fenomenológico utilizado, a estrutura final das diferentes redes testadas, além de apresentar os dados utilizados e a metodologia seguida para o treino e avaliação das abordagens. O Capítulo 4 apresenta os resultados das melhores redes em cada abordagem e para cada variável, procurando compará-las entre si, levando em conta aspectos como a acurácia e a facilidade de aprendizado. Um curto estudo de reconciliação de dados também é apresentado, assim como uma curta análise com mais dados, obtidos ao final do trabalho. Por fim, o Capítulo 5 apresenta as conclusões deste trabalho e perspectivas futuras.

Capítulo 2

Revisão Bibliográfica e Fundamentação Teórica

2.1 Revisão Bibliográfica

PEMS desenvolvidos para turbinas a gás, tanto a partir de abordagens fenomenológicas quanto puramente baseadas em dados, são descritos na literatura. No entanto, houve dificuldade para se encontrar referências que descrevessem o uso de modelos híbridos como o descrito neste trabalho.

As abordagens físicas costumam se basear em modelos cinéticos e termodinâmicos, integrando balanços de massa e energia[1,24], ou então em modelos de redes de reatores químicos (*Chemical Reactor Networks* — CRN), gerados a partir de simulações de fluidodinâmica computacional (*Computational Fluid Dynamics* — CFD)[25–27].

Observa-se na literatura, no entanto, uma predominância de modelos baseados em dados. De acordo com Swanson e Lawrence[28,29], mais de 95% dos PEMS certificados nos Estados Unidos (regulação 40 CFR[12]) utilizam esse tipo de abordagem, e um grande foco é dado no uso de redes neuronais. Destaca-se que os uso de redes convolucionais e recorrentes é menos frequente na literatura para essa aplicação, embora essas arquiteturas sejam mais comumente encontradas na predição de emissões de motores automotivos[30,31]; de plantas a carvão[32,33]; e na detecção de falhas e prognóstico de desempenho de turbinas a gás[34–36].

Vanderhaegen *et al.*[37] usaram redes neuronais adaptativas para a predição de emissões de NO_x e CO. De acordo com os autores, o perfil de emissões de uma mesma turbina pode mudar durante sua vida útil, principalmente após consertos e substituição de equipamentos próximos, como válvulas. É um ponto que deve ser levado em conta após o desenvolvimento do PEMS, caso haja mudanças na planta. A solução apresentada no artigo é de recalibrar o PEMS regular e automaticamente com base nos dados históricos, de maneira a evitar que uma seleção manual de dados precise ser realizada, o que aumentaria muito o custo de manutenção do modelo.

O relatório de Nielsen[24] se aproxima bastante do escopo deste trabalho por aplicar uma abordagem física e baseada em dados para a predição da emissão de NO_x, O_2 e CO em plantas na Dinamarca e Oriente Médio, se diferenciando apenas por não aplicar uma abordagem híbrida. A parte fenomenológica modela a cinética das reações e balanços de massa e energia na câmara de combustão em 4 zonas, e utiliza redes neuronais artificiais do tipo *Feed Forward*

para a modelagem baseada em dados. O autor relata que, no caso de turbinas a gás, ambas as abordagens se saíram extremamente bem e têm resultados semelhantes na estimação das três variáveis, com resoluções de cerca de 1 ppm para o NO_x e o CO, e uma acurácia cerca de 10 vezes maior para o O_2. O autor também indica que redes treinadas para predição de uma turbina podem não funcionar para outras turbinas, mas que isso pode ser resolvido treinando-se os modelos com dados de todas as turbinas de interesse.

2.2 Inteligência Artificial e Aprendizado de Máquina

Primeiramente, é preciso delimitar o que se compreende por "Inteligência Artificial" (IA), uma definição complexa por causa da necessidade de se definir o que se entende por "inteligência". Para Poole[38], um agente inteligente é aquele que toma decisões apropriadas de acordo com seus objetivos e circunstâncias, sendo flexível a ambientes e objetivos que mudam, aprendendo pela experiência, de acordo com suas limitações perceptuais e de tempo de cálculo. Nesse contexto, a IA é a área que estuda máquinas que demonstram algum grau de comportamento inteligente, esse termo é utilizado, por extensão, para se referir às próprias máquinas (e assim foi feito ao longo deste trabalho). Atualmente, o termo costuma ser usado de maneira ampla para fazer referência a qualquer tipo de agente artificial que seja capaz de aprender por experiência e tomar decisões autonomamente, a fim de atingir objetivos que lhe foram programados.

A elaboração de IAs não é uma ideia nova, e remonta à origem dos computadores[39]. As IAs mais clássicas costumam ser algoritmos bem definidos e determinísticos, que dependem da correta implementação do problema pelo programador.

Uma parte dos algoritmos de IA, no entanto, enquadra-se na categoria de Aprendizado de Máquina (ML — de *Machine Learning*, em inglês). Nesses casos, o programa é desenvolvido de forma a ser capaz de aprender a resolver o problema definido, com pouca ou nenhuma interferência humana. Isso faz com que o programador não precise saber como resolver a tarefa em si, apenas como traduzi-la para a máquina de forma que ela seja capaz de interpretar as informações (os dados) disponíveis e tomar decisões por conta própria.

A idealização de máquinas capazes de aprender através da experiência é, mais uma vez, uma ideia tão antiga quanto a computação[40]. No entanto, sua ascensão e ubiquidade só foi possível graças aos avanços tecnológicos (em particular o aumento da velocidade e capacidade de computação) do século XXI.

O processo de aprendizado pode ser generalizado como um algoritmo através do qual o agente procura se adaptar, de maneira a maximizar a probabilidade de alcançar seus objetivos, de acordo com os dados que lhe são fornecidos. De um modo geral, os algoritmos modernos de aprendizagem são baseados em processos de otimização.

2.3 Redes Neuronais Artificiais

As Redes Neuronais Artificiais (ANNs — do inglês *Artificial Neural Networks*) são funções com parâmetros otimizáveis, ajustados através de algoritmos de ML — sendo assim consideradas IAs[41]. As ANNs realizam regressões (embora sejam também muito utilizadas em problemas de classificação[42]) entre variáveis de entrada e de saída. Elas fazem parte de uma

classe de modelos matemáticos ditos do tipo “caixa-preta”, caracterizados por um alto grau de complexidade e a falta de interpretabilidade física[13,14]. Por não integrarem nenhum conhecimento fenomenológico à sua estrutura, também costumam ser chamados de “modelos baseados em dados”. O interesse nesse tipo de regressor não está na *forma* da função, mas apenas em uma modelagem acurada da relação final entre as entradas e saídas.

Sua estrutura é inspirada no sistema nervoso central humano, de onde tiram seu nome[43,44]. Ela é composta por unidades fundamentais — neurônios — que podem ser interpretadas como funções. Por via de regra, o neurônio aplica uma função não linear (chamada “função de ativação”, f) à combinação linear das suas entradas ($\mathbf{x}$), ponderadas por um vetor $\mathbf{w}$, a fim de obter uma saída (y). Esta operação é mostrada na Equação 2.1, na qual f é a função de ativação e $\mathbf{w}$ e b são parâmetros. O parâmetro linear b costuma ser chamado de viés (*bias*), originário do conceito estatístico de mesmo nome. O resultado é então passado para outros neurônios e assim por diante.

$$y = f(w_1 \cdot x_1 + w_2 \cdot x_2 + \cdots + w_n \cdot x_n + b) = f(\mathbf{w} \cdot \mathbf{x} + b) \tag{2.1}$$

As ANNs são usualmente ilustradas por meio de grafos cujos nós representam os neurônios e cujas linhas representam as conexões entre os mesmos. Um exemplo é mostrado na Figura 2.1. O desenho desta rede, assim como os das Figuras 2.2, 3.2, 3.3 e 3.4, foram realizados com o auxílio da ferramenta de código aberto NN-SVG[45].

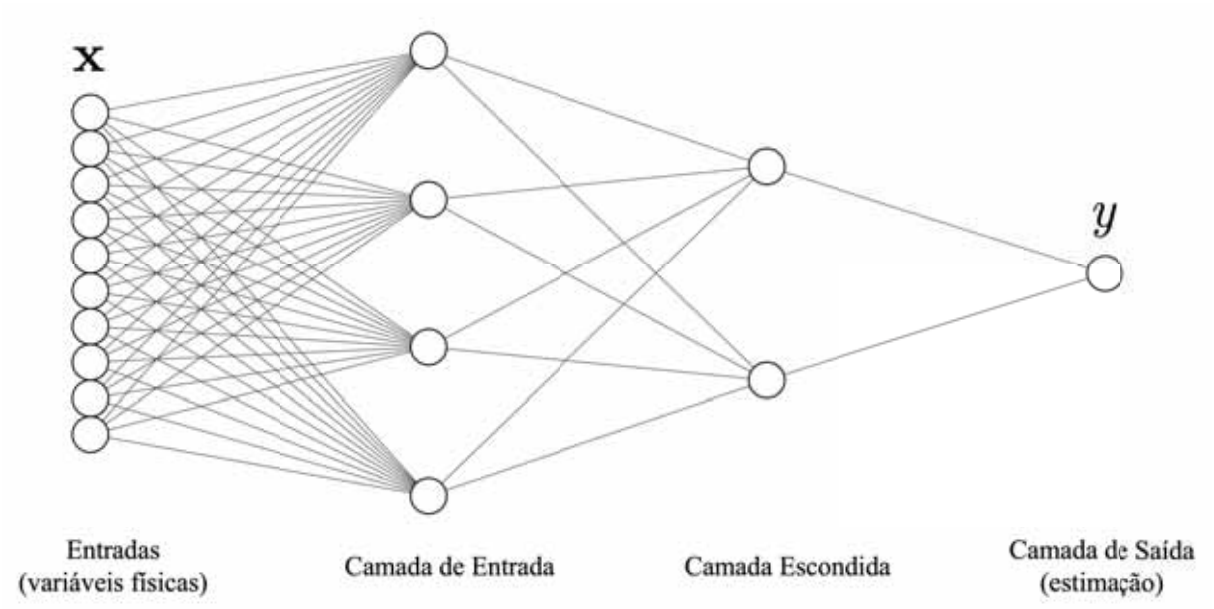

Figura 2.1: Ilustração de uma Rede Neuronal Artificial (ANN) clássica.

As linhas que chegam à esquerda de um neurônio indicam quais são suas entradas (*inputs*), e a linha que sai à direita sua saída (*output*).

Um conjunto de neurônios que recebe as mesmas variáveis é chamado de “camada”, que pode ser:

- De entrada: que recebe as entradas do modelo;

- De saída: a última camada, cujo resultado é a própria saída do modelo;
- Escondida: quaisquer camadas entre as de entrada e saída.

Quando uma ANN tem várias camadas escondidas, falamos em Aprendizado Profundo (DL — *Deep Learning*), uma vez que a interpretação física do modelo se torna ainda mais vaga e sua capacidade de aprendizado aumenta.

Essa arquitetura (conhecida como "Perceptron Multicamada") faz com que as ANNs sejam um aproximador universal, o que significa que qualquer função pode ser aproximada a uma precisão arbitrária por um número suficientemente grande de neurônios[46,47]. Deve-se destacar, no entanto, que a estrutura mostrada na Figura 2.1 e descrita no texto não é a única possível para as ANNs. Outros tipos de redes neuronais — que são citadas mais à frente — podem conter conexões temporais ou tratar as entradas e saídas de forma diferente. O uso de funções de ativação não lineares e múltiplas camadas escondidas é o que dá às ANNs a capacidade de generalização de dados e aprendizado de padrões, tornando-as significativamente mais eficazes que aproximações polinomiais, por exemplo[48].

É essa capacidade aproximativa que faz com que as ANNs sejam aplicadas com sucesso em tantas áreas diferentes. O uso de diferentes tipos e estruturas aumenta ainda mais a sua flexibilidade.

2.4 Algoritmo de aprendizado

Existem diversas abordagens para a estimação dos parâmetros de uma ANN, usualmente denominada de aprendizado (ou treino) de modelos — cada uma adequada para um tipo de problema diferente. O escopo deste trabalho se limita ao aprendizado supervisionado, onde o modelo aprende a partir de pares de dados de entrada e de saída conhecidos, e é otimizado de modo a minimizar seu erro de estimação — com algumas ressalvas, que serão feitas na Seção 2.4.3.

De maneira simplificada, informa-se um vetor de dados de entrada ($\mathbf{x}$) à rede, que nos retorna uma estimativa da variável de interesse ($\hat{y}$). A estimativa é comparada ao valor conhecido (y) e os parâmetros do modelo são então atualizados visando minimizar o erro do modelo ($e = y - \hat{y}$).

Como a aplicação aqui estudada é a regressão[1], utiliza-se a soma dos erros quadráticos do modelo como função objetivo. Esta função $l(D, \boldsymbol{\theta})$ é mostrada na Equação 2.3, na qual D representa o conjunto de dados de treino, constituído por pares de vetores de entrada $\mathbf{x}^i$ e valores de saída y^i conhecidos, como mostrado na Equação 2.2, enquanto $\boldsymbol{\theta}$ representa o vetor de parâmetros do modelo.

$$D = \{(\mathbf{x}^1, y^1), (\mathbf{x}^2, y^2), \cdots, (\mathbf{x}^{N-1}, y^{N-1}), (\mathbf{x}^N, y^N)\} \tag{2.2}$$

Assim, $\hat{y}(\mathbf{x}^i, \boldsymbol{\theta})$ é a estimativa da rede para o vetor de entrada $\mathbf{x}^i$, dado seu vetor de parâmetros.

[1] Outras funções erro podem ser mais relevantes em outras aplicações, como problemas de classificação[49].

$$l(D, \boldsymbol{\theta}) = \sum_{i=1}^{N} (e_i)^2 = \sum_{i=1}^{N} [y^i - \hat{y}(\mathbf{x}^i, \boldsymbol{\theta})]^2 \tag{2.3}$$

Diferentemente de uma regressão linear, não é possível calcular o valor ótimo dos parâmetros analiticamente. Isso se deve ao fato da função ser não linear em relação a seus parâmetros. Se faz assim necessário o uso de métodos de otimização não lineares. Neste trabalho, concentra-se em métodos baseados no gradiente, que são os mais comumente utilizados na área, e é explicado o princípio de sua retropropagação.

2.4.1 Método do gradiente

Os métodos de otimização não linear baseados no gradiente da função objetivo são recursivos, atualizando os valores dos parâmetros iterativamente. Lembra-se que o vetor gradiente de uma função aponta no sentido de maior crescimento da mesma. Assim, para minimizar a função erro $l(D, \boldsymbol{\theta})$, mudando o corpo dos parâmetros, é preciso atualizar o vetor de parâmetros $\boldsymbol{\theta}$ no sentido oposto do vetor $\nabla_{\boldsymbol{\theta}} l(D, \boldsymbol{\theta})$ — como mostrado na Equação 2.4 — na qual os sobrescritos entre parênteses indicam o número da iteração. O hiper-parâmetro η é o chamado *learning rate* ou "taxa de aprendizagem", e controla o peso que o vetor gradiente tem na atualização dos parâmetros.

$$\boldsymbol{\theta}^{(i+1)} \leftarrow \boldsymbol{\theta}^{(i)} - \eta \nabla_{\boldsymbol{\theta}} l(D, \boldsymbol{\theta}^{(i)}) \tag{2.4}$$

A Equação 2.4 é a mais básica e ilustrativa de como esse tipo de método funciona. No entanto, métodos mais complexos podem ser usados para se estimar o tamanho e sentido do passo a se dar a cada iteração. Pode-se citar o clássico método de Newton, onde em vez do gradiente temos um produto entre a inversa da matriz Hessiana da função erro e seu gradiente, e os métodos quasi-Newton, derivados e baseados no mesmo[50]. Esses métodos costumam estimar ou substituir a matriz Hessiana por uma aproximação, um exemplo é o BFGS[51]. Já o método de Levenberg-Marquardt[52] mistura as duas abordagens. Cita-se também o método do gradiente conjugado, em que a direção de cada novo passo é um vetor conjugado em relação aos passos anteriores (conjugação de Gram-Schmidt)[53].

A função erro de uma ANN em função de seus parâmetros é complexa por causa da sua não linearidade e dimensionalidade — apresentando alta não convexidade, mínimos locais e algum caráter estocástico herdado dos dados de entrada e saída. Por conta disso, métodos de otimização para função objetivo estocástica (i.e. que integram elementos de aleatoriedade inerentes dessa função e seus gradientes) e métodos não determinísticos como o treino em "mini-bateladas" (descrito mais a frente na seção 2.4.4) costumam ser usados[54].

2.4.2 Retropropagação do gradiente

Nos métodos baseados em gradiente, quando aplicados à ANN, a expressão analítica do gradiente não é desenvolvida. Em vez disso, ele é calculado numericamente através do método de

retropropagação, um caso particular do método de diferenciação automática, que se baseia na regra da cadeia. Cada termo é calculado e multiplicado camada por camada, de acordo com as operações e funções não lineares aplicadas por cada etapa e neurônio. Esse procedimento é extremamente eficiente e é uma das razões que tornam esse cálculo computacionalmente viável. A Figura 2.2 mostra uma ilustração desse processo para um parâmetro de um neurônio da camada escondida da estrutura ($w_{2,1}$). Os cálculos associados são explicitados nas Equações 2.5 a 2.10

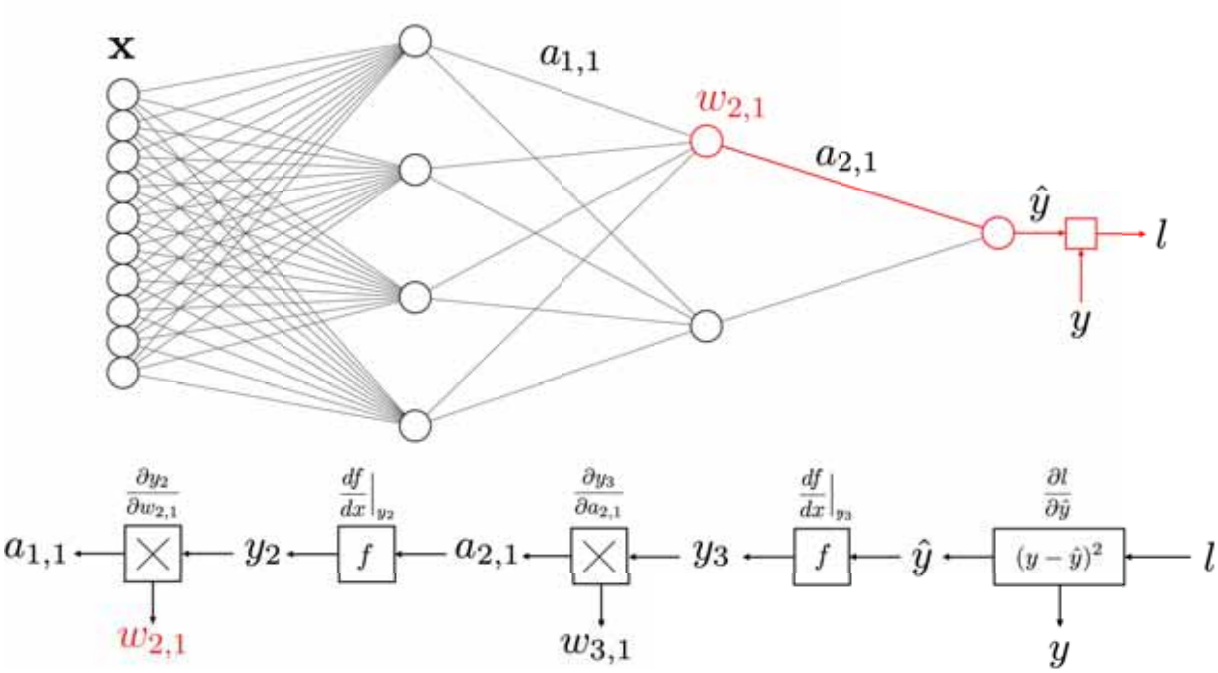

Figura 2.2: Exemplo do procedimento de retropropagação do gradiente.

$$l = (y - \hat{y})^2 \qquad \Rightarrow \frac{\partial l}{\partial \hat{y}} = -2(y - \hat{y}) \tag{2.5}$$

$$\hat{y} = f(y_3) \qquad \Rightarrow \frac{\partial \hat{y}}{\partial y_3} = \frac{df}{dx}\Big|_{y_3} \tag{2.6}$$

$$y_3 = w_{3,1}a_{2,1} + w_{3,2}a_{2,2} \qquad \Rightarrow \frac{\partial y_3}{\partial a_{2,1}} = w_{3,1} \tag{2.7}$$

$$a_{2,1} = f(y_2) \qquad \Rightarrow \frac{\partial a_{2,1}}{\partial y_2} = \frac{df}{dx}\Big|_{y_2} \tag{2.8}$$

$$y_2 = w_{2,1}a_{1,1} + w_{2,2}a_{1,2} + \cdots + w_{2,4}a_{1,4} \qquad \Rightarrow \frac{\partial y_2}{\partial w_{2,1}} = a_{1,1} \tag{2.9}$$

$$\frac{\partial l}{\partial w_{2,1}} = \frac{\partial l}{\partial \hat{y}}\frac{\partial \hat{y}}{\partial y_3}\frac{\partial y_3}{\partial a_{2,1}}\frac{\partial a_{2,1}}{\partial y_2}\frac{\partial y_2}{\partial w_{2,1}} = -2(y - \hat{y})\frac{df}{dx}\Big|_{y_3} w_{3,1}\frac{df}{dx}\Big|_{y_2} a_{1,1} \tag{2.10}$$

2.4.3 O problema do sobre-ajuste

Um problema comumente apresentado pelas redes neuronais é o chamado *sobre-ajuste* (em inglês *overfitting*), em que o modelo "memoriza" os pares de entrada e saída a ele apresentados durante o treino, obtendo resultados medíocres em quaisquer outros pontos. Em outras palavras, o modelo não aprende a **generalizar** a relação entre as variáveis de entrada e de saída. A Figura 2.3 mostra um exemplo claro de um modelo bem treinado e um modelo sobre-ajustado. Esse problema advém do fato de as redes neuronais geralmente possuírem uma quantidade muito grande de neurônios e portanto de parâmetros, o que pode dificultar o aprendizado.

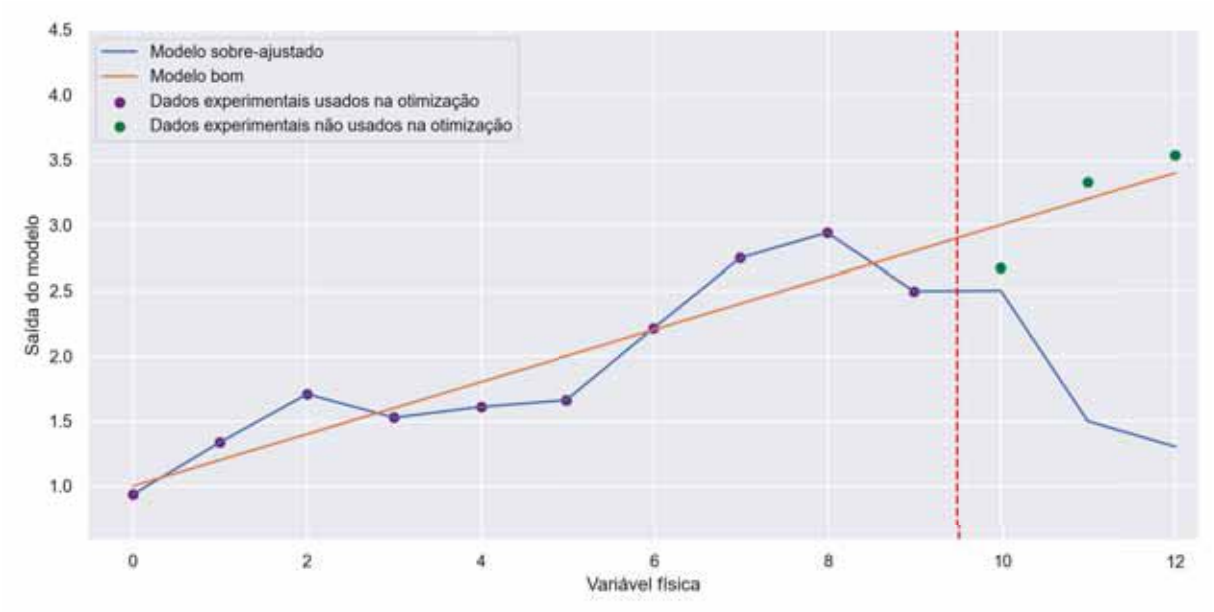

Figura 2.3: Diferença entre um bom modelo e outro sobre-ajustado.

Para se identificar facilmente quando um modelo deixa de aprender a generalizar e passa a sobre-ajustar, os dados disponíveis são separados em dois conjuntos: o de treino e o de validação. Os dados presentes no conjunto de treino são aqueles utilizados na estimação dos parâmetros do modelo. Os dados de validação, por sua vez, são apenas usados para medir o seu desempenho (capacidade de generalização). Durante o treino, o erro do modelo é concomitantemente medido nos dois conjuntos. Quando o mesmo deixa de aprender a generalizar e passa a sobre-ajustar os dados de treino, percebe-se que o erro de validação deixa de diminuir e passa a aumentar (enquanto o erro de treino diminui monotonicamente, por ser a função minimizada). Ao final, utiliza-se os parâmetros da iteração com menor erro de validação. A Figura 2.4 mostra a evolução típica do erro de treino e de validação ao longo das iterações.

É importante destacar que, por causa disso, os resultados das redes nos dados de treino nunca devem ser usados como parâmetro de sua acurácia. Assim, todos os resultados apresentados neste trabalho (salvo menção contrária) são relativos aos dados de validação. Destaca-se também que é costume utilizar entre 60 a 80% [55] dos dados disponíveis para o treino, o restante sendo separado para a validação.

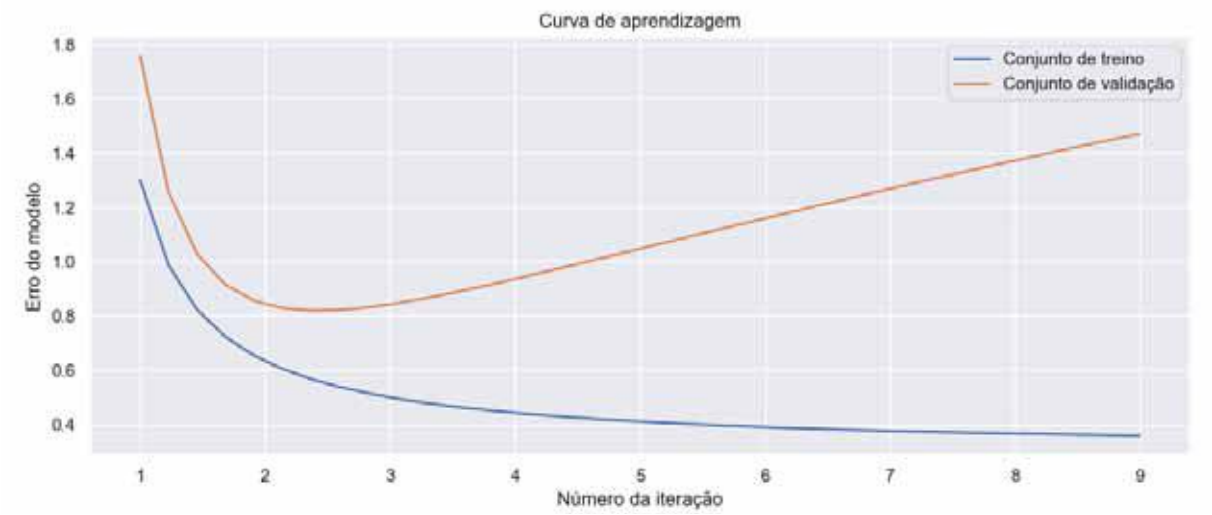

Figura 2.4: Típica curva de aprendizagem.

2.4.4 Considerações finais sobre o algoritmo de aprendizagem

Devido à grande dimensionalidade do problema de otimização e da dificuldade inerente do problema de regressão, a função objetivo costuma ter múltiplos pontos de mínimo locais. Isso dificulta a convergência do método para o mínimo global através do método do gradiente puro, que fica extremamente dependente da estimativa inicial dada para os parâmetros. Nesses casos, é comum integrar métodos não determinísticos ao processo de otimização de maneira a favorecer uma maior exploração do domínio. A maneira simples como isso costuma ser integrado com êxito em problemas de aprendizado de máquina é através de um treino em "mini-bateladas". Nesse caso, em vez de se dar um passo de otimização baseado no gradiente do erro de *todos* os dados de treino (processo em "batelada"), o conjunto de treino é dividido em subconjuntos (mini-bateladas) contendo um número menor de dados. Assim, múltiplos passos do método do gradiente são dados a cada passe pelo conjunto completo. Quanto menor o tamanho das mini-bateladas, melhor costuma ser o desempenho do modelo e acelera-se seu aprendizado. No entanto, esse método torna o processo computacionalmente mais lento[56], o que pode ser um fator limitante para treinos com uma grande quantidade de dados. Quando a quantidade de dados disponíveis é relativamente pequena, uma mini-batelada de tamanho unitário pode ser utilizada para todos os experimentos sem comprometer o tempo de cálculo, como é o caso neste trabalho.

Dado todo o discutido neste capítulo, é agora possível apresentar o algoritmo completo de treinamento e validação (Algoritmo 1, a seguir), levando também em consideração o cálculo do erro de validação. Ele é sempre composto de dois laços: o de iterações mais externas — usualmente chamadas de épocas (*epochs*) — e um laço interno que gira em torno dos dados disponíveis, realizando um passo da otimização após cada exemplo.

Os valores ótimos do número de épocas e da taxa de aprendizagem (η) variam de um tipo de modelo para o outro, de acordo com número, qualidade e variedade dos dados de entrada e com o tamanho da batelada usada na atualização dos parâmetros. De um modo geral, é ideal que a curva de aprendizagem seja o mais suave possível. De toda forma, a estimativa inicial para o valor dos parâmetros cria uma variabilidade que pode fazê-los tender para mínimos locais — ainda mais quando o modelo é bastante complexo e tem um número grande de parâmetros. Por causa disso, é recomendado que se treine o mesmo modelo diversas vezes (com uma inicialização

Algoritmo 1 Algoritmo de aprendizagem

para época $\leftarrow 1$ **até** número de épocas **faça**

$l_{\text{treino, época}} = 0$
$l_{\text{valid., época}} = 0$

para i $\leftarrow 1$ **até** número de dados de treino **faça** ▷ Laço de treino
$\hat{y}^i = \text{NN}(\mathbf{x}^i)$ ▷ Estimação
$l_{\text{treino, época}} = l_{\text{treino, época}} + \text{L}(y^i, \hat{y}^i)$ ▷ Cálculo da função objetivo
$\nabla_{\boldsymbol{\theta}} l^i = \nabla_{\boldsymbol{\theta}} \text{L}(\mathbf{x}^i, y^i, \boldsymbol{\theta})$ ▷ Cálculo do gradiente por retropropagação
$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \nabla_{\boldsymbol{\theta}} l^i$ ▷ Atualização dos parâmetros
fim do loop

para i $\leftarrow 1$ **até** número de dados de validação **faça** ▷ Laço de validação
$\hat{y}^i = \text{NN}(\mathbf{x}^i)$
$l_{\text{valid., época}} = l_{\text{valid., época}} + \text{L}(y^i, \hat{y}^i)$ ▷ Cálculo do erro de validação
fim do loop

fim do loop

diferente dos parâmetros a cada vez) para estimar seu desempenho médio e selecionar apenas aquele com o melhor resultado[57].

Finalmente, ao se comparar os resultados dos diferentes modelos, deve-se levar em conta a complexidade dos mesmos — o que pode ser feito comparando a quantidade de parâmetros que cada um possui. É esperado que modelos com mais parâmetros obtenham resultados melhores para problemas complexos, quando a quantidade de dados é suficiente.

2.5 Outros tipos de redes neuronais

Além da arquitetura de rede neuronal já descrita, dois outros tipos se destacam pela sua estrutura diferenciada, que lhes dá uma melhor capacidade de tratamento de dados sequenciais. Tratam-se das Redes Neuronais Convolucionais e as Redes Neuronais Recorrentes.

2.5.1 Redes Neuronais Convolucionais

As Redes Neuronais Convolucionais (CNNs — do inglês *Convolutional Neural Networks*) são redes desenvolvidas para o tratamento de dados sequenciais, ou que possuem uma relação de proximidade espacial. Elas são conhecidas por serem bem sucedidas em problemas de tratamento de imagens[16] e reconhecimento de padrões (tanto em imagens quanto em dados sequenciais como séries temporais)[17]. Elas já foram usadas com êxito no processamento de linguagem natural[18], na detecção e previsão de flutuações do mercado de ações[19], reconhecimento de sequências de DNA[20], entre outras.

As CNNs são compostas por filtros, que são matrizes de parâmetros, que se deslocam sobre a sequência de entradas aplicando transformações lineares sobre as mesmas. Os resultados

geram novas matrizes de dados que podem ser tratadas para o reconhecimento de características, reduzindo a dimensão do resultado. Essa etapa de tratamento (em inglês chamada de *pooling*) costuma aplicar uma operação de "máximo", separando apenas os maiores resultados de cada característica (cada linha da matriz). O vetor final pode então passar por uma última camada de neurônios (como a de uma ANN clássica) para adaptar a dimensão para a saída da rede. Pode-se também usar múltiplas camadas para esta última etapa.

Os filtros são caracterizados por três hiper-parâmetros: o tamanho de núcleo (*kernel size*), que define o comprimento da matriz, o passo (*stride*), que define o passo de deslocamento da matriz, e o número de filtros[2]. Devido à grande variedade de combinações possíveis para essas variáveis, e a possibilidade de se usar vários tipos de filtro em um mesmo modelo, as CNNs se tornam um tipo de rede muito flexível e capaz de reconhecer padrões. O encadeamento de várias camadas convolucionais também é comum em redes complexas. A Figura 2.5 mostra o princípio de aplicação de um filtro, a Equação 2.11 mostra a operação realizada em cada passo k: trata-se da soma de todos elementos resultantes de um produto ponto-a-ponto entre a matriz dos parâmetros $\mathbf{W}$ e a matriz de entrada $\mathbf{X}$, ignorando-se o possível termo de viés associado ao filtro. Cada filtro de mesmo tipo gera uma linha $\mathbf{Y}$ complementar — o índice l na Equação 2.11 indica o número do filtro. Filtros diferentes são tratados separadamente visto que os tamanhos finais das matrizes costumam ser incompatíveis.

$$Y_{l,k} = \sum_{i=1}^{N} \sum_{j=1}^{L} W_{i,j}^{(l)} \times X_{i,j+(k-1)passo} \tag{2.11}$$

A Figura 2.6 mostra um esquema de arquitetura clássica de uma CNN. Outros tipos de filtros podem ser aplicados e tratados paralelamente ao primeiro, e seus vetores de saída concatenados para a última etapa. A matriz Y também pode passar por outra camada de convolução em vez de ser tratada, por exemplo.

Um último detalhe acerca das redes convolucionais é a possibilidade delas receberem tamanhos variáveis de sequências de entrada. No entanto, há o risco de uma sequência ser menor que o tamanho do filtro, o que impossibilitaria o cálculo por incompatibilidade de tamanhos. Nesses casos é comum o uso de "*padding*", isto é, da concatenação de zeros ao começo e fim da sequência, aumentando artificialmente o tamanho da mesma e assegurando que não ocorrerão problemas de cálculo.

[2]Há também a "dilatação", cujo caso de aplicação foge do escopo deste trabalho.

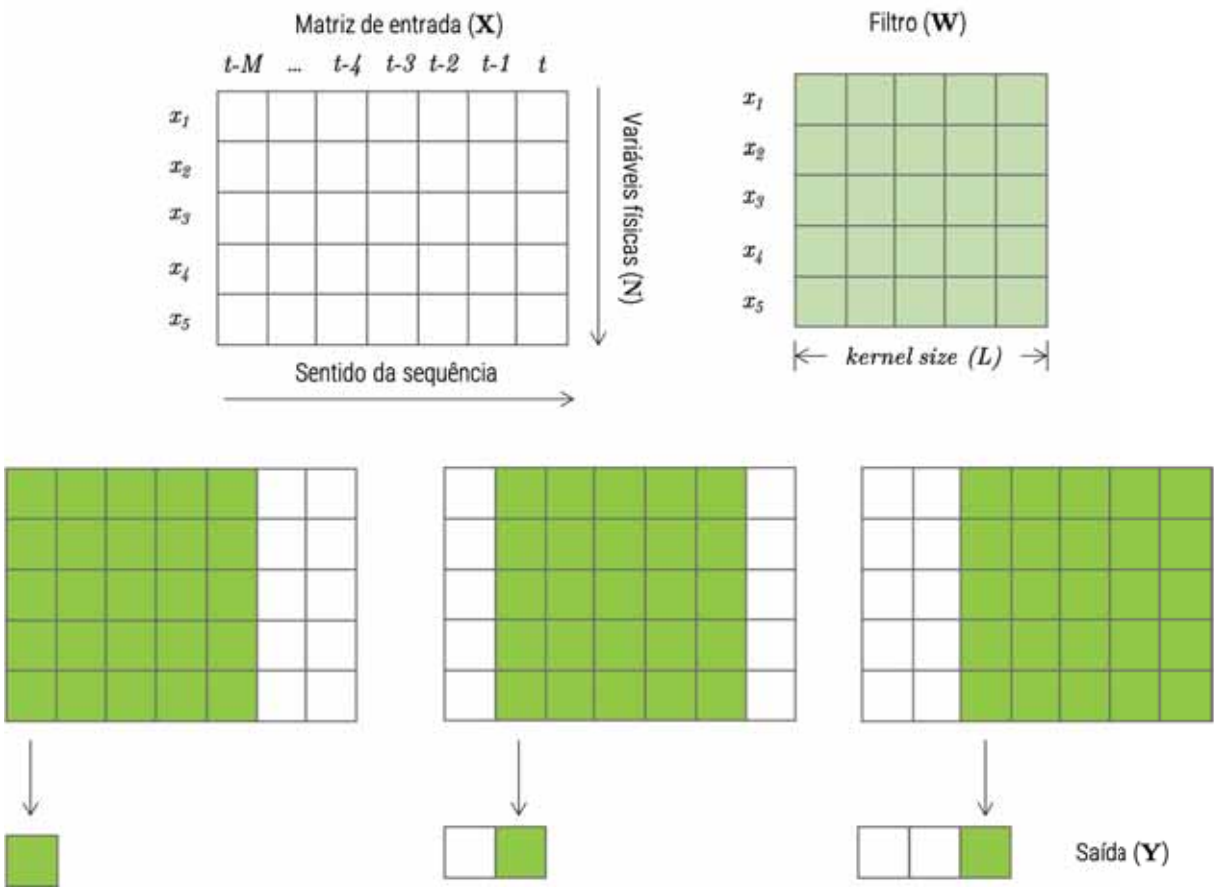

Figura 2.5: Ilustração do processo de convolução para um filtro de $passo = 1$.

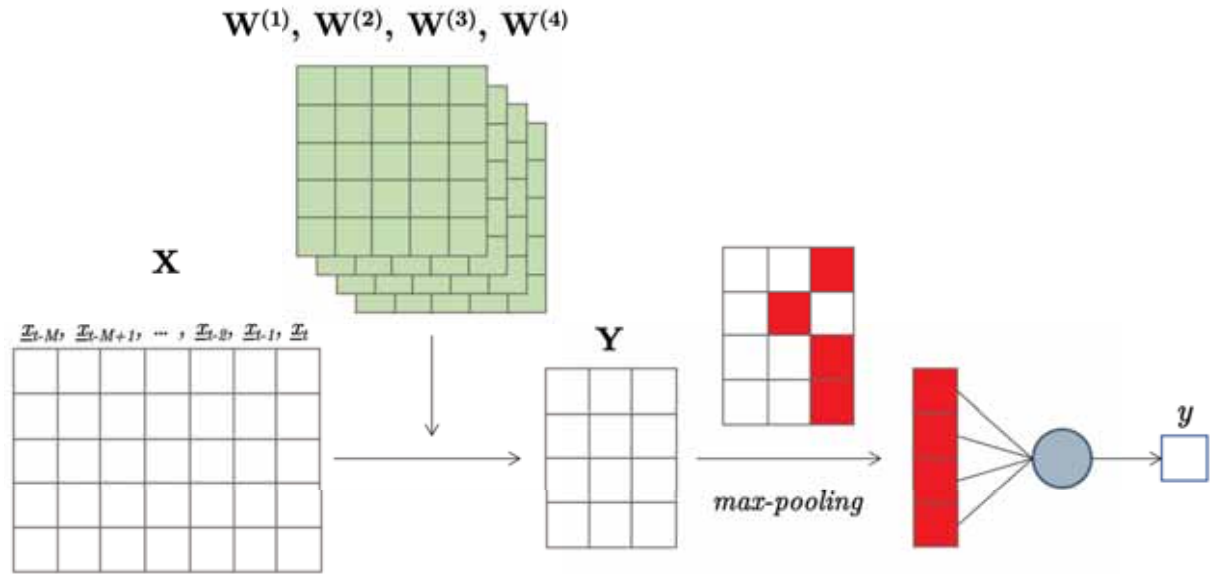

Figura 2.6: Ilustração de uma arquitetura de CNN clássica.

2.5.2 Redes Neuronais Recorrentes

As Redes Neuronais Recorrentes (RNNs) são redes particulares por possuírem uma conexão temporal, que conecta informações e estimativas obtidas para um instante de tempo t às entradas do instante $t+1$. Isto é, a rede traz informações de passos de tempo anteriores para a estimação do instante presente, a fim de melhorar as estimativas. Sua resposta em cada novo instante de tempo é dependente das entradas e dos estados anteriores. Elas são portanto adaptadas para sequências como as CNNs, embora tenham um funcionamento diferente. Enquanto as CNNs costumam extrair e tratar padrões adjacentes na sequência de dados, as RNNs costumam tirar informações ao longo da mesma de forma mais temporal. Dessa forma, esse tipo de rede se sobressai nos campos de predição e regressão de dados temporais[21], reconhecimento de voz e áudio[22], e processamento de linguagem natural[23].

A Figura 2.7 mostra o esquema básico de uma RNN, na qual $y(t)$ é a saída do modelo no instante de tempo t e $c(t)$ é um vetor de estado que serve apenas para agregar informações do instante de tempo anterior à estimação do instante de tempo seguinte. O bloco recorrente z^{-1} é uma forma de indicar a retenção dos valores para o instante de tempo seguinte. Para se treinar uma RNN pelo método de retropropagação do gradiente, sua estrutura precisa ser desdobrada temporalmente, como mostrado na Figura 2.8. Isso se deve à dependência da estimação em um instante de tempo t das estimações nos tempos anteriores. Um desenvolvimento matemático mais aprofundado é apresentado no apêndice A.

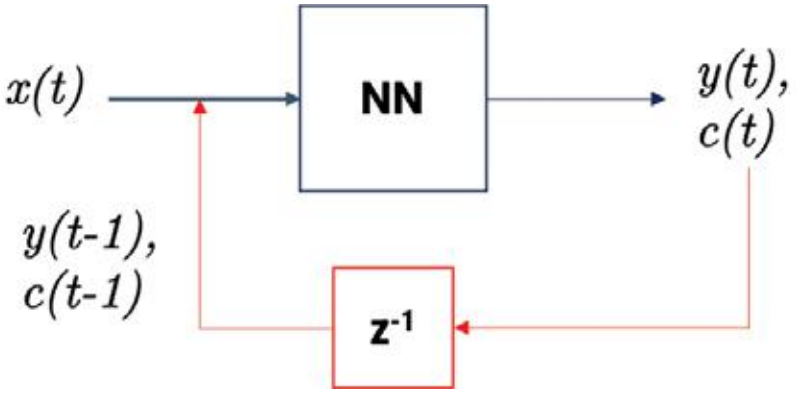

Figura 2.7: Esquema básico de um RNN.

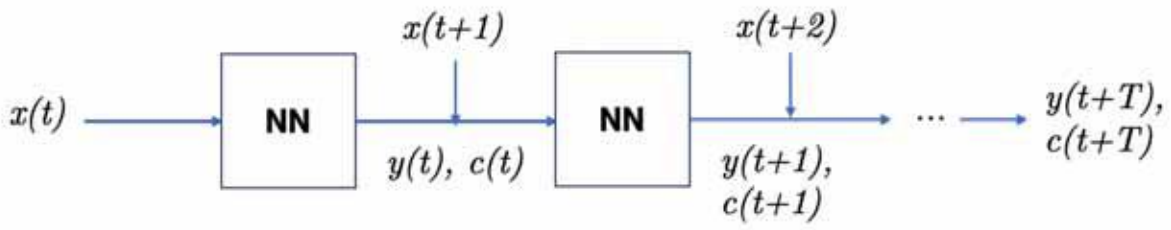

Figura 2.8: Desdobramento temporal de uma RNN.

Um problema conhecido das RNNs é a sua dificuldade de reter informações de pontos do começo da sequência[58]. Isso se deve ao método de retropropagação do gradiente usado no treino dos modelos. Conforme o tamanho da sequência aumenta, o gradiente dos primeiros pontos tende a diminuir a zero, ou (menos frequentemente) divergir a infinito[59] devido às várias multiplicações envolvidas nesse processo por conta da regra da cadeia.

Para resolver esse problema, Hochreiter e Schmidhuber[60] desenvolveram uma arquitetura apelidada de *Long Short-Term Memory* ("Longa Memória a Curto Prazo", em tradução livre), ou LSTM, composta por uma "célula" e três "portões": o de entrada (*input* — i), o de saída (*output* — o) e o de "esquecimento" (*forget* — f). Cada portão é uma camada de neurônios com função de ativação sigmoide. A célula (c) guarda valores de tempos arbitrários enquanto os portões regulam o fluxo de informação para a mesma. A Figura 2.9 mostra um esquema do funcionamento de uma rede LSTM e as Equações 2.12 — 2.17 descrevem as operações envolvidas. Nelas, mantemos a notação mais comumente encontrada na literatura. As variáveis minúsculas são vetores, e as maiúsculas são matrizes de parâmetros (W). Distingue-se o produto escalar "$\cdot$" do produto ponto-a-ponto "$\odot$". Finalmente, σ é a função sigmoide (Equação 2.18) e $\tanh$ é a função tangente hiperbólica. O vetor de saída, representado por h, também é usado como uma das entradas da conexão recorrente.

A rede LSTM revolucionou as RNNs, resolvendo o problema de atenuação do gradiente e permitindo a aplicação desse tipo de estrutura para sequências longas com alto grau de correlação entre diferentes instantes de tempo.

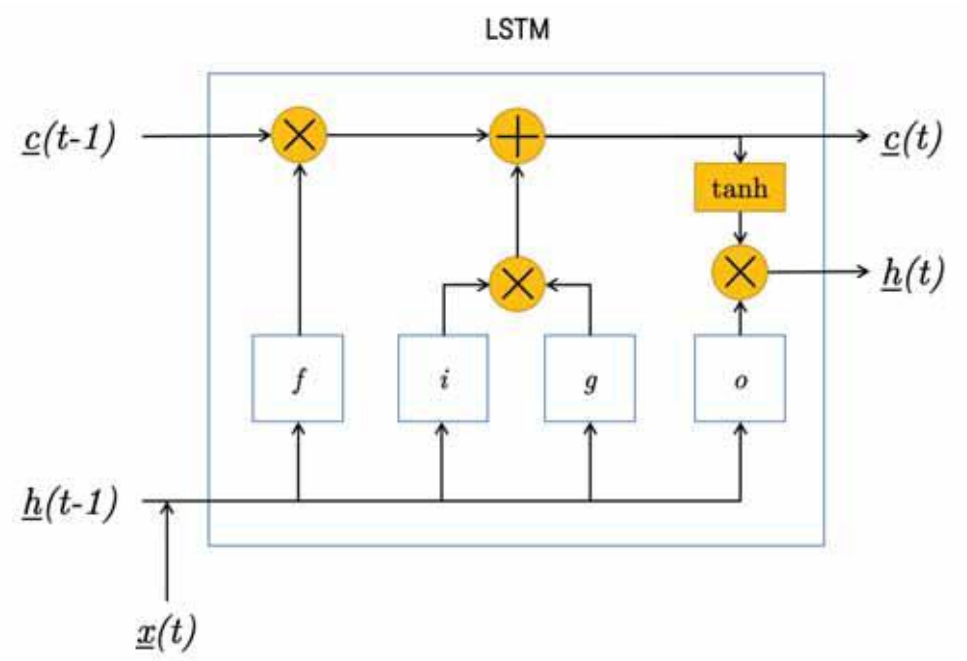

Figura 2.9: Estrutura da rede LSTM.

$$f(t) = \sigma(W_{fx} \cdot x(t) + W_{fh} \cdot h(t-1) + b_f) \tag{2.12}$$
$$i(t) = \sigma(W_{ix} \cdot x(t) + W_{ih} \cdot h(t-1) + b_i) \tag{2.13}$$
$$g(t) = \tanh(W_{gx} \cdot x(t) + W_{gh} \cdot h(t-1) + b_g) \tag{2.14}$$
$$o(t) = \sigma(W_{ox} \cdot x(t) + W_{oh} \cdot h(t-1) + b_o) \tag{2.15}$$
$$c(t) = f(t) \odot c(t-1) + i(t) \odot g(t) \tag{2.16}$$
$$h(t) = o(t) \odot \tanh(c(t)) \tag{2.17}$$
$$\sigma(x) = \frac{1}{1+e^{-x}} \tag{2.18}$$

2.6 Modelos Híbridos

Modelos Híbridos fazem parte de uma classe de modelos conhecidos como "caixas-cinza", que integram abordagens fenomenológicas a funções do tipo caixa-preta. Essa integração pode ser feita de diversas maneiras, como as formas em série, onde o modelo físico utiliza a saída do modelo baseado em dados como parte das variáveis de entrada[61]; em paralelo, onde o modelo baseado em dados serve como uma correção ao resultado do modelo físico[61]; os meta-modelos, que são modelos simplificados de outros modelos[62]; e o PIML (*Physics Informed Machine Learning* — Aprendizado de Máquina Informado pela Física, em tradução livre), no qual restrições físicas (usualmente descritas por equações diferenciais) são integradas ao processo de aprendizado supervisionado de um modelo tipo "caixa-preta"[63].

Esse tipo de modelagem é útil quando os modelos fenomenológicos não são suficientemente acurados para uma determinada aplicação e/ou quando sua expressão não compreende todas as variáveis ou interações relevantes no processo. Além disso, o aprendizado do modelo caixa-preta associado é facilitado[15], uma vez que o modelo físico fica encarregado da regressão mais complexa.

Neste trabalho, trabalhará-se com um modelo híbrido paralelo, onde o simulador de processos EMSO[64] realiza a parte fenomenológica (ver Seção 3.1.1) e as redes neuronais atuam como modelo baseado em dados, sendo o resultado final a soma das duas saídas, como mostrado na Figura 2.10. Assim, queremos que $\hat{y}_{EMSO} + \hat{y}_{Rede}$ tenda à saída medida y, de modo que devemos treinar as redes com o *erro* do simulador: $\hat{y}_{Rede} = y - \hat{y}_{EMSO}$.

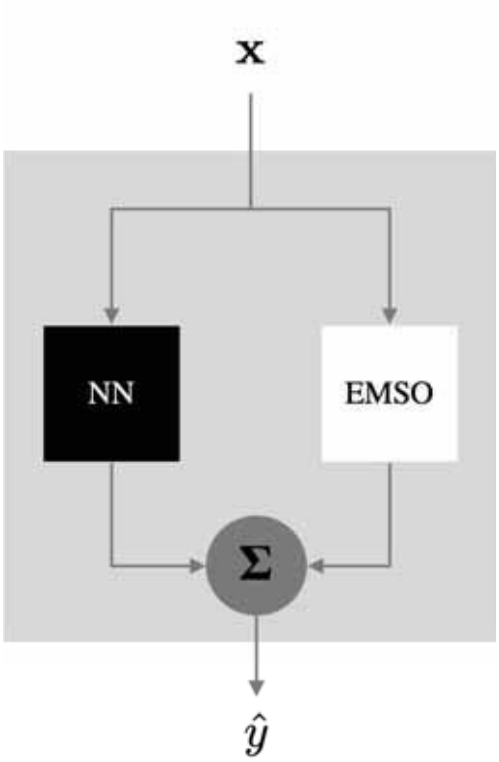

Figura 2.10: Estrutura do modelo híbrido empregado neste trabalho.

Capítulo 3

Modelagem e Metodologia

3.1 Modelagem

3.1.1 Modelo Fenomenológico

O modelo fenomenológico utilizado se baseia no princípio termodinâmico da minimização da energia de Gibbs (G) em um sistema reacional a pressão e temperatura constantes[65], sendo um método comumente utilizado no cálculo de equilíbrio de misturas de gases de combustão[65,66]. Essa função é representada na Equação 3.1, na qual T é a temperatura, P é a pressão e $\mathbf{n}$ é o vetor de número de mols de cada substância $j \in J$.

É preciso, no entanto, adicionar a condição de conservação da massa como restrição ao problema de otimização. Essa expressão é matematicamente descrita através do conjunto de Equações 3.2 (uma para cada elemento químico $i \in I$). Nela, $a_{i,j}$ é o número de átomos do elemento i na espécie j, n_j é o número de mols da espécie j no equilíbrio e $n_{f,j}$ o número de mols de entrada da mesma molécula no sistema.

$$\min_{\mathbf{n}} G(T, P, \mathbf{n}) \tag{3.1}$$

$$\sum_{j} a_{i,j}(n_j - n_{f,j}) = 0 \tag{3.2}$$

Esse problema pode ser resolvido através do método dos multiplicadores de Lagrange, que consiste em somar as restrições à função objetivo, ponderando-as pelos multiplicadores λ_i. A expressão final da função objetivo e do problema de otimização é representada pela Equação 3.3

$$\begin{aligned} &\min_{\mathbf{n},\boldsymbol{\lambda}} L(T, P, \mathbf{n}, \boldsymbol{\lambda}) \\ &= \min_{\mathbf{n},\boldsymbol{\lambda}} \left[G(T, P, \mathbf{n}) + \sum_{i} \sum_{j} \lambda_i a_{i,j}(n_j - n_{f,j}) \right] \end{aligned} \tag{3.3}$$

Em seu valor mínimo, as derivadas de L em relação a cada $\mathbf{n}$ e $\boldsymbol{\lambda}$ (variáveis de otimização) devem ser $\mathbf{0}$.[1] Assim, em vez de realizar um processo de otimização, podemos encontrar a solução do problema através da resolução do sistema de equações não lineares representado pelas Equações 3.4.

$$\begin{cases} \dfrac{\partial L}{\partial n_j} = \dfrac{\partial G}{\partial n_j} + \displaystyle\sum_i \lambda_i a_{i,j} = 0 & \forall j \in J \\ \dfrac{\partial L}{\partial \lambda_i} = \dfrac{\partial G}{\partial \lambda_i} + \displaystyle\sum_j a_{i,j}(n_j - n_{f,j}) = 0 & \forall i \in I \end{cases} \tag{3.4}$$

O termo $\frac{\partial G}{\partial n_j}$ é definido como o potencial químico μ_j da substância j, que pode ser definido segundo a Equação 3.5, em que $\Delta G^{\circ}_{f,j}(T)$ é a energia de Gibbs de formação da espécie j no estado padrão (gás ideal a 1 bar) e f_j a fugacidade da espécie na mistura. A energia de formação é calculada de acordo com a base de dados de McBride *et al.*[67], enquanto que o valor da fugacidade é calculado de acordo com uma equação de estado que represente o sistema. Neste trabalho, foi usada a modificação de Soave da equação cúbica de Redlich-Kwong[68].

$$\mu_j = \Delta G^{\circ}_{f,j}(T) + RT \ln\left(\frac{f_j(T, P, \mathbf{n})}{1 \text{ bar}}\right) \tag{3.5}$$

Finalmente, o termo $\frac{\partial G}{\partial \boldsymbol{\lambda}}$ é igual a zero, resultando nas Equações 3.6.

$$\begin{cases} \Delta G^{\circ}_{f,j}(T) + RT \ln(f_j) + \displaystyle\sum_i \lambda_i a_{i,j} = 0 & \forall j \in J \\ \displaystyle\sum_j a_{i,j}(n_j - n_{f,j}) = 0 & \forall i \in I \end{cases} \tag{3.6}$$

Esclarece-se que a modelagem fenomenológica aqui apresentada já havia sido implementada ao simulador de processos EMSO em projetos anteriores, para gerar estimativas das concentrações dos gases de saída da turbina. Neste trabalho, apenas os seus resultados foram diretamente utilizados para o desenvolvimento dos modelos híbridos.

3.1.2 Redes Neuronais Clássicas

Redes Neuronais Feed Forward

Foram desenvolvidas três redes neuronais do tipo *Feed Forward* — de estrutura derivada da clássica, apresentada na Figura 2.1. Duas versões foram criadas: uma com apenas uma camada escondida (para um total de três camadas), e uma versão de aprendizado profundo (variantes

[1] Note que a notação do problema foi encurtada ao se falar na derivada em relação a um vetor. Evidentemente, a condição real é de que as derivadas de L em relação a cada n_j e cada λ_i, para todo i e j, sejam zero. Define-se assim, por simplicidade, que $\frac{\partial L}{\partial \mathbf{z}} = [\frac{\partial L}{\partial z_1}, \frac{\partial L}{\partial z_2} \cdots \frac{\partial L}{\partial z_N}]^T$

chamadas de "*Deep*"), com quatro camadas escondidas (totalizando 6 camadas). As versões escondidas também contavam com duas camadas de *Dropout*, após a primeira e terceira camadas escondidas, a fim de melhorar seu aprendizado. Essa camada anula as saídas de uma certa proporção dos neurônios da camada anterior (escolhidos aleatoriamente de acordo com uma probabilidade) durante o treino. Foi demonstrado que essa prática diminui o sobre-ajuste de modelos complexos ao evitar que neurônios se co-adaptem em demasia (i.e. aprendam as mesmas características)[69]. Neste trabalho, uma probabilidade de 30% foi utilizada para ambas as camadas. Todas as versões utilizam como função de ativação a *Leaky ReLU*[70], mostrada na Figura 3.1 e descrita pela Equação 3.7. Trata-se de uma função de ativação que reduz problemas de atenuação de gradiente, visto que sua derivada tem um valor constante (diferente para $x \geq 0$ e $x < 0$).

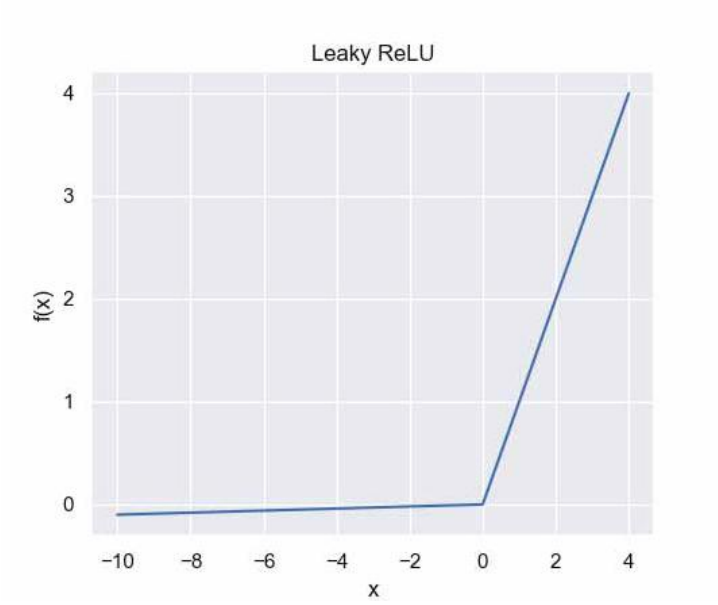

Figura 3.1: A função de ativação Leaky ReLU. Note que os eixos não estão na mesma escala.

$$f(x) = \begin{cases} x & x \geq 0 \\ 0{,}01x & x < 0 \end{cases} \tag{3.7}$$

As três redes são:

1. Rede Neuronal Clássica (FFNN): composta por camadas de 10 e 30 neurônios, nas versões simples e profunda, respectivamente. A estrutura é representada na Figura 3.2 e é a base para as seguintes.

2. Rede Neuronal Estruturada (SNN): semelhante à anterior, mas que tenta levar em conta a variável melhor correlacionada com a saída de maneira diferenciada. A ideia desta estrutura veio da realização de que a potência de turbina tinha uma correlação linear bastante importante com a concentração de NO_x(uma das variáveis a serem estimadas). Assim, essa variável de entrada foi integrada diretamente ao neurônio da camada de saída, que por sua vez não aplica uma função de ativação, realizando apenas uma combinação linear de suas entradas. A rede é representada na Figura 3.3. Essa variável será denominada por "variável especial" daqui para frente.

3. Rede Neuronal com Mínimos Quadrados (NN-MQ): Semelhante à anterior, mas cujos parâmetros da combinação linear da variável especial são fixos, estimados pelo método dos mínimos quadrados. Mais simplesmente, pode-se interpretar que a rede neuronal serve como um preditor do erro da regressão $NO_x = a_1 \cdot p + a_0$, em que p é a variável especial e os coeficientes a_0 e a_1 são estimados pelo método dos mínimos quadrados. Esta rede é representada pela Figura 3.4

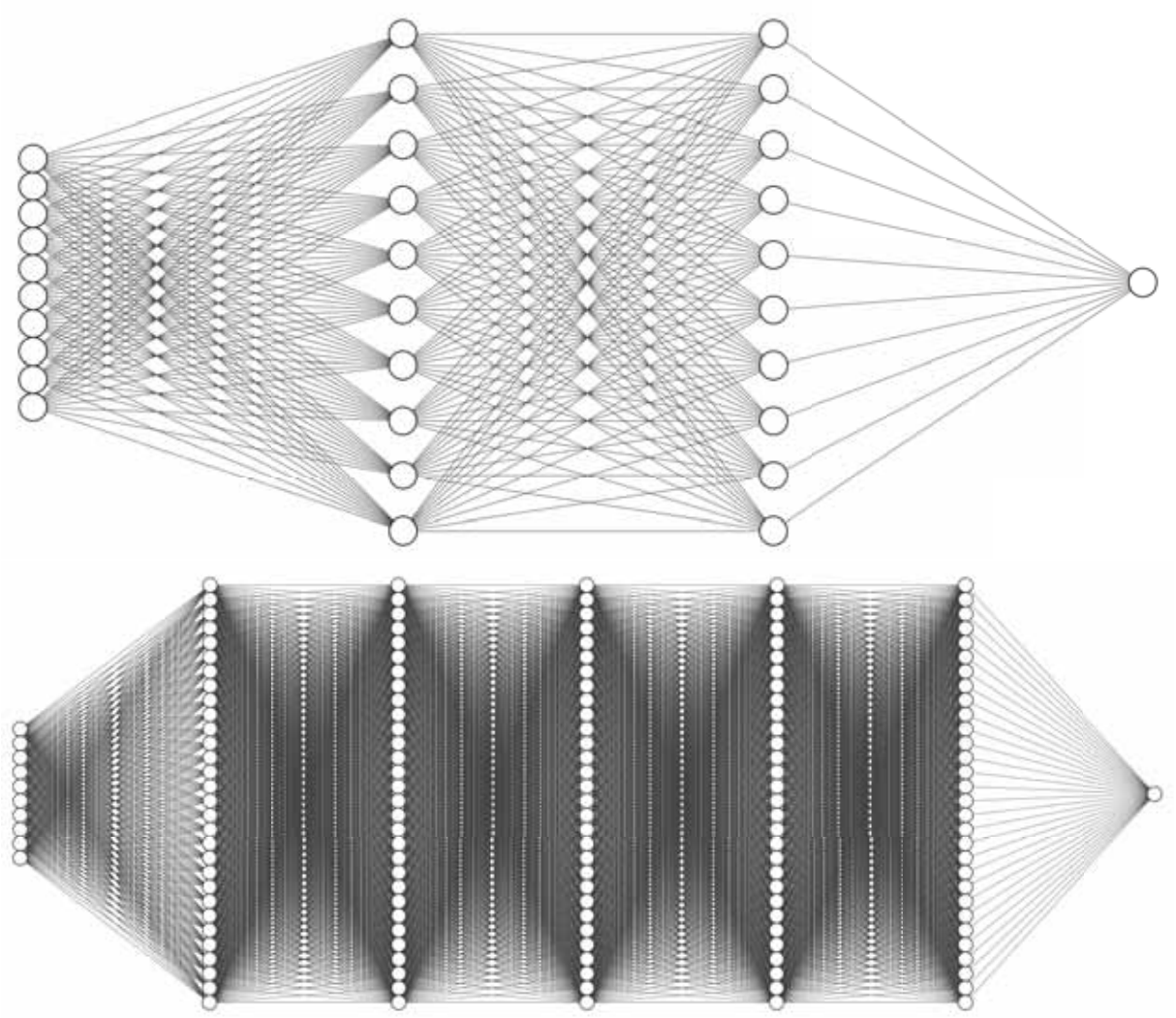

Figura 3.2: Estrutura da rede clássica (FFNN), nas versões simples e profunda, respectivamente.

Redes Neuronais Convolucionais

Como descrito na Seção 2.5.1, esse tipo de rede recebe uma sequência de entradas na forma de uma matriz. O tamanho dessa sequência (número de colunas) é um hiper-parâmetro. Neste trabalho, o valor escolhido foi de 11 (1 do ponto do instante de tempo da estimação, mais os 10 pontos que o precedem; esses pontos são referidos como a "memória" do sistema). Isso é possível visto que o passo de tempo entre os pontos é constante e igual a 1 hora na maior parte da massa de dados. Nos casos em que existe um intervalo de tempo maior entre um ponto e o anterior (espaços oriundos da remoção de dados durante a etapa de tratamento) usou-se

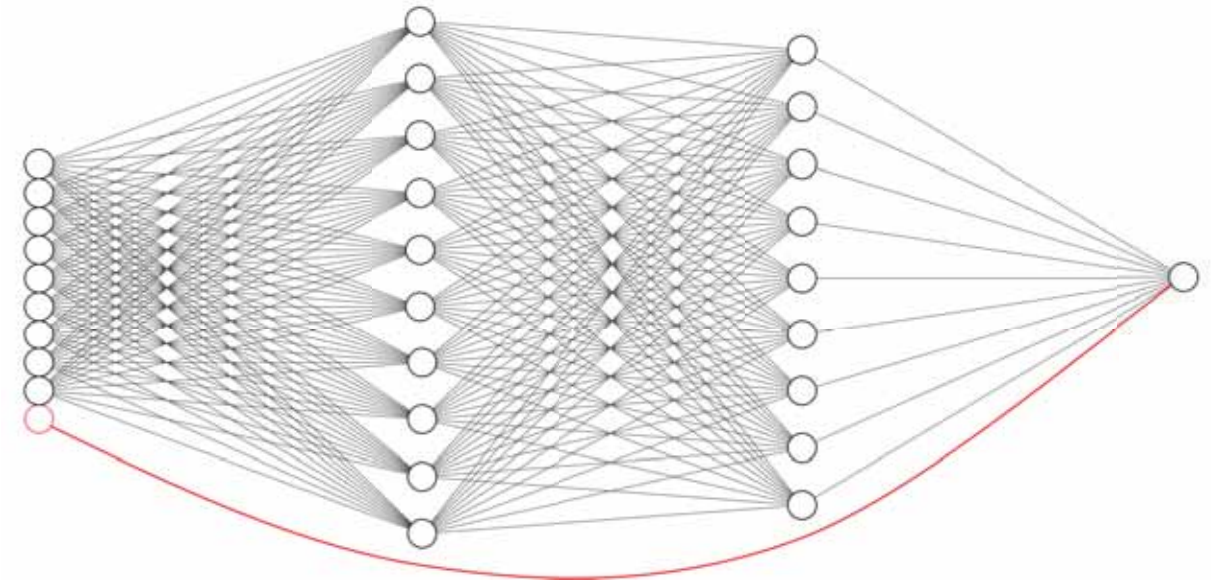

Figura 3.3: Estrutura da rede neuronal estruturada.

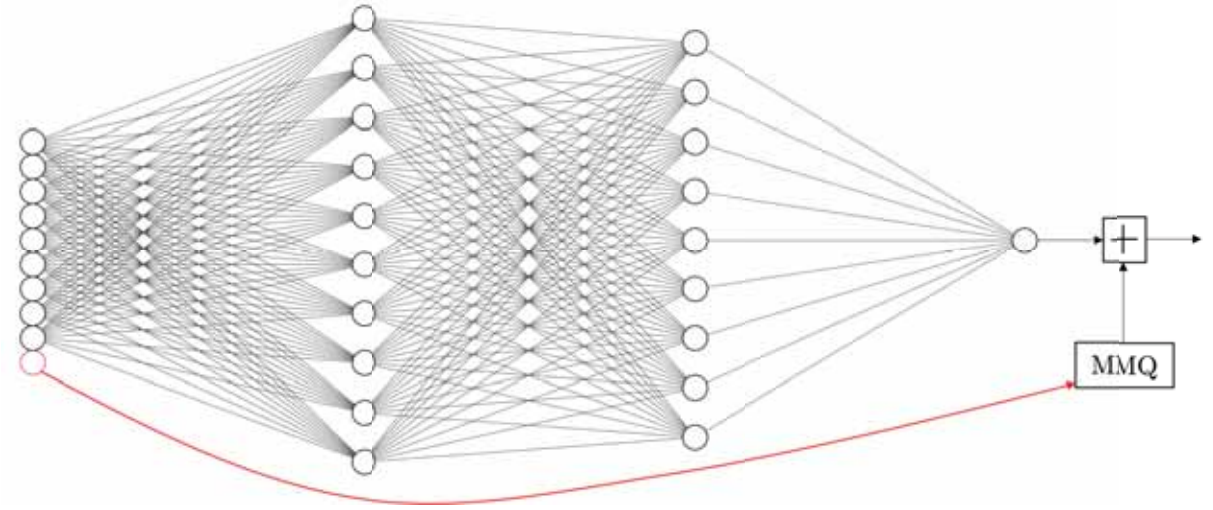

Figura 3.4: Estrutura da rede neuronal com mínimos quadrados.

apenas os pontos até esse salto de tempo. Assim, a rede foi treinada com tamanhos variáveis de sequência, o que é útil pois aumenta a flexibilidade da mesma para lidar com falta de pontos.

Foram testadas três arquiteturas de rede convolucional, com o intuito de se comparar um tratamento convolucional em paralelo; uma aplicação única com o mesmo número de filtros; e uma aplicação profunda (em série e com mais filtros). Essas estruturas são exploradas a seguir:

1. Versão 0: Quatro camadas paralelas de convolução, cada uma com três filtros. Os tamanhos de filtro (*kernel size*) são 2, 3, 4 e 11, todas com passo (*stride*) 1. As matrizes resultantes da convolução seguem para uma etapa de *max-pooling*, onde o valor máximo de cada linha é selecionado, resultando em um vetor de 3 entradas. Os resultados do *max-pooling* de cada tipo de filtro são concatenados em um vetor que passa então por uma camada para resultar na saída. Uma ilustração desse processo é representado na Figura 3.5.

2. Versão 1: Apenas uma camada de convolução com 12 filtros de tamanho 2 e passo 1. Aplica-se um *max-pooling* no resultado e o vetor final de 12 entradas passa por um neurônio para se chegar ao resultado final de estimação. Essa arquitetura é representada na Figura 3.6.

3. Versão 2: Uma versão mais complexa com duas camadas de convolução, a primeira com 30 filtros de tamanho 4 e passo 1, e a segunda com 50 filtros de tamanho 3 e passo 1. Uma etapa de *max-pooling* é realizada após cada convolução e uma camada comum final é aplicada ao resultado a fim de adaptá-lo à dimensão das saídas. A estrutura é mostrada na Figura 3.7.

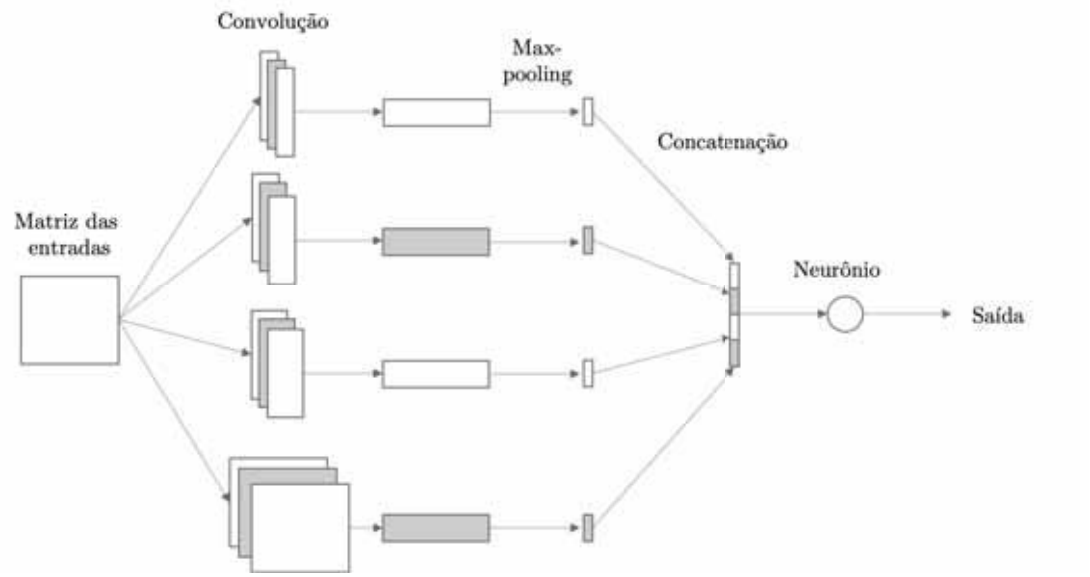

Figura 3.5: Estrutura da CNN versão 0.

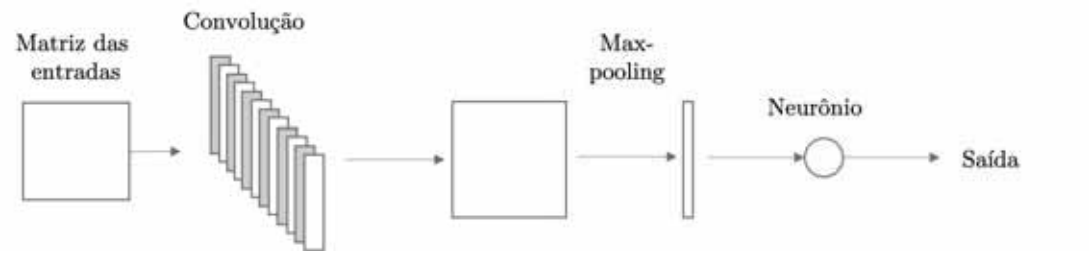

Figura 3.6: Estrutura da CNN versão 1.

Redes Neuronais Recorrentes

Assim como as CNNs, as RNNs recebem sequências variáveis de dados. O processo de adequação dos dados é rigorosamente igual ao das CNNs, exceto pelo *padding*. Como as RNNs não têm restrições a respeito do tamanho das sequências, o uso de *padding* não se faz necessário. Diferentemente das CNNs, as RNNs tratam os pontos na ordem da sequência até chegarem à entrada mais recente. É somente essa estimação final, resultado do tratamento cumulativo de todas as entradas anteriores, que é de fato usado como valor de saída da rede.

Foram implementadas duas RNNs:

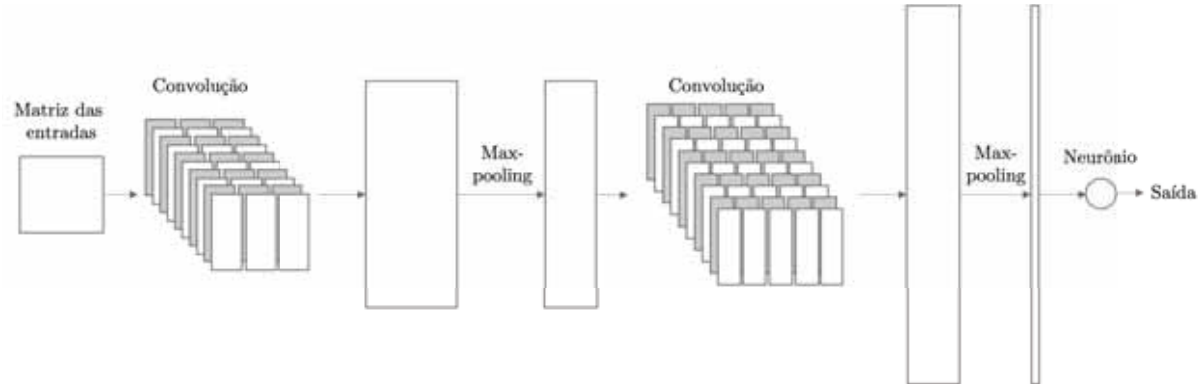

Figura 3.7: Estrutura da CNN versão 2.

1. Uma RNN simples: trata-se de uma rede de Elman[71] com vetor de saída (h) com 20 coordenadas. Esse vetor passa então por um neurônio para chegar ao resultado final. Essa rede é ilustrada na Figura 3.8.

2. Uma LSTM: descrita na Seção 2.5.2 e ilustrada na Figura 2.9. Como a RNN simples, o vetor de saída tem 20 dimensões e passa por um neurônio para se adaptar à saída 1D.

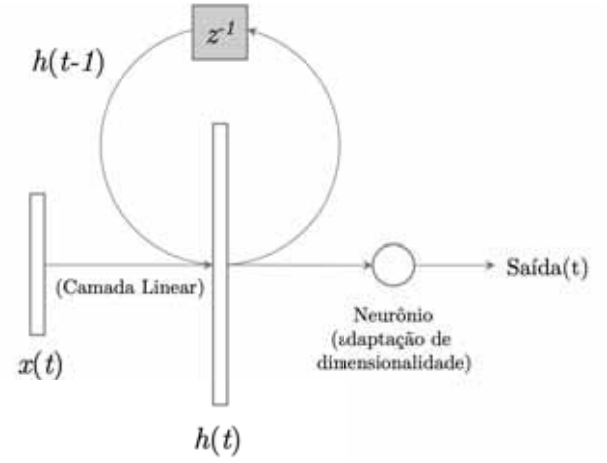

Figura 3.8: Estrutura da RNN simples usada.

3.2 Metodologia

Todas as redes neuronais e rotinas de treino foram desenvolvidas em Python com a ajuda da biblioteca PyTorch. O treino em si foi realizado em servidores disponibilizados pela plataforma Google Colab. Todos os modelos foram treinados múltiplas vezes (a fim de se levar em conta a variabilidade da inicialização de seus parâmetros) tanto para a estimação direta da concentração de gases de saída da turbina, quanto para a estimação do erro do simulador na estimação da mesma. O método de otimização escolhido foi o ADAM[72] (de *Adaptive Moment Estimation*), que tem sido cada vez mais comum em aplicações de aprendizado de máquina. Em todos os casos, o treino foi realizado com um tamanho de batelada igual a 1 (um passo do otimizador após cada dado de treino). O valor da taxa de aprendizagem e do número de épocas foi encontrado

manualmente, fazendo-os variar até se ver graficamente que a otimização era suficientemente suave sem ser demasiadamente lenta. Essa análise não foi automatizada devido à pequena quantidade de hiper-parâmetros estudados, além do interesse em se visualizar a evolução da forma das curvas de aprendizagem com o incremento dos mesmos. Novamente, os valores ótimos desses hiper-parâmetros podem ser diferentes de acordo com os modelos e os dados, o que é discutido no Capítulo 4.

3.2.1 Base de dados

Para o treinamento e validação dos modelos, foram utilizados dados entre 01/01/2015 e 02/01/2016 da turbina a gás número 1 do bloco 2, denominada TG21, de uma termelétrica. O conjunto de dados consiste de diversas variáveis medidas durante a operação da planta, disponibilizados em valores médios dentro de uma janela de operação a cada uma hora. Os dados brutos são primeiramente tratados, de maneira a se retirar valores espúrios e períodos em que a turbina se encontrava fora de operação. Ao final desse processo, totalizavam-se 6.513 instantes de medição, consequentemente repartidos de acordo com a data, de modo que os primeiros 70% (4.560) foram separados para o conjunto de treino, e os 30% restantes (1.953) para o de validação.

A Figura 3.9 mostra um esquema simplificado de um turbogerador, isto é, do sistema composto por um compressor, câmara de combustão e turbina utilizado para a geração de energia. Dentre as variáveis medidas nesse sistema, dez já haviam sido escolhidas como entradas no desenvolvimento de outros modelos em trabalhos prévios, sendo portanto utilizadas neste trabalho para a estimação:

1. Temperatura na câmara de combustão;
2. Pressão na câmara de combustão;
3. Fração de metano no combustível;
4. Fração de etano no combustível;
5. Fração de propano no combustível;
6. Fração de CO_2 no combustível;
7. Fração de N_2 no combustível;
8. Vazão mássica dos gases de exaustão;
9. Vazão mássica de combustível;
10. Potência ativa selecionada da turbina.

Quanto às variáveis estimadas, foram treinados modelos para estimar a emissão de NO_x, O_2 e CO — assim como para o erro do simulador na estimação dessas variáveis, no caso da abordagem híbrida. As 10 variáveis de entrada foram classificadas segundo sua correlação linear com a concentração dos gases estimadas no conjunto de treino, e aquelas com maior correlação foram escolhidas como as variáveis especiais usadas. Essas variáveis são identificadas na Tabela 3.1. Para o erro do simulador, as mesmas variáveis especiais foram utilizadas de acordo com a saída associada.

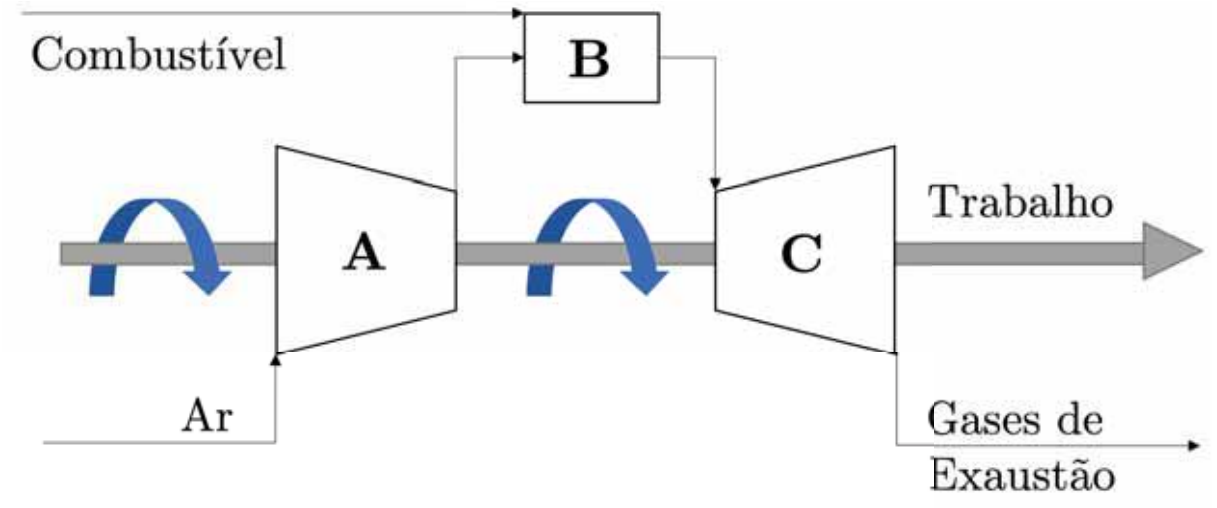

Figura 3.9: Esquema simplificado de um turbogerador, composto por: A) um compressor; B) uma câmara de combustão; e C) uma turbina.

Tabela 3.1: Variáveis de entrada melhor correlacionadas com cada uma das variáveis estimadas.

Variável de Saída	Variável especial associada
NO_x	Potência ativa selecionada da turbina
O_2	Temperatura na câmara de combustão
CO	Fração de metano no combustível

3.2.2 Critérios de avaliação

Como discutido na Seção 2.4.3, somente os resultados obtidos sobre o conjunto de dados de validação devem ser usados para avaliar as redes. Ao final do treino, várias métricas são medidas relativamente a esses pontos para fins comparativos, dentre elas cita-se: o erro absoluto médio (EAM); o erro relativo médio (ERM); e o coeficiente de correlação (ρ ou CC) — representados pelas Equações 3.8 a 3.10.

$$EAM = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{3.8}$$

$$ERM = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \tag{3.9}$$

$$\rho = \frac{\sum_{i=1}^{n}(\hat{y}_i - \bar{\hat{y}})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(\hat{y}_i - \bar{\hat{y}})^2}\cdot\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}} \tag{3.10}$$

Na prática, o erro absoluto médio e o coeficiente de correlação são os mais relevantes. Isso pois EAM tem uma interpretação física bem clara, enquanto que o erro relativo médio pode adquirir valores pouco representativos da qualidade do modelo quando os valores absolutos da variável de saída são próximos de zero ($y_i \to 0 \Rightarrow ERM \to \infty$). O coeficiente de correlação, por sua vez, ajuda a identificar o quão bem as redes reagem à evolução da variável de saída, e é

útil na interpretação de determinados resultados no Capítulo 4. Finalmente, recorda-se que, ao se comparar diferentes modelos, deve-se levar em conta sua complexidade (representada pelo número de parâmetros que possuem).

Capítulo 4

Resultados e Discussão

Todos os modelos (para ambas as abordagens) foram treinados múltiplas vezes em diversas condições de número de épocas — desde 10 até 1000 iterações — e taxas de aprendizado — de 10^{-4} até 5×10^{-7}. De um modo geral, é necessário aumentar o número de épocas ao se diminuir o valor da taxa de aprendizado, prática que costuma produzir uma curva de aprendizado mais lenta e suave visto que os passos de otimização são menores.

Os resultados dos melhores modelos de cada tipo de rede desenvolvida, para cada uma das abordagens e variáveis de saída, são apresentados nas seções a seguir. Destaca-se que os resultados apresentados para a modelagem híbrida já levam em conta a soma das saídas da rede e do modelo fenomenológico, completando a estimativa da variável de interesse, de modo que eles podem ser diretamente comparados.

4.1 Estimação de NO_x

A Tabela 4.1 apresenta os melhores resultados (com base no menor erro absoluto médio) de cada tipo de rede para a estimação de NO_x nas duas abordagens. Nela, EAM identifica o erro absoluto médio em ppm, ERM o erro relativo médio, CC o coeficiente de correlação e LR a taxa de aprendizado com que a rede foi treinada. Na graduação das cores, quanto mais verde melhor é o resultado entre os modelos e quanto mais vermelho pior. A graduação das cores foi realizada por coluna, incluindo ambas as abordagens.

Tabela 4.1: Resultados para a estimação de NO_x.

		Abordagem baseada em dados				Abordagem Híbrida			
Redes	**Parâmetros**	**EAM**	**ERM**	**CC**	**LR**	**EAM**	**ERM**	**CC**	**LR**
FFNN	231	1,022	4,67%	53%	2E-06	1,000	4,61%	54%	1E-06
Deep FFNN	5011	1,001	4,54%	61%	3E-06	0,948	4,34%	61%	9E-07
SNN	210	1,000	4,58%	55%	2E-06	1,052	4,86%	46%	1E-06
Deep SNN	4983	1,065	4,89%	48%	9E-07	1,011	4,67%	56%	9E-07
NN-MQ	209	1,084	4,91%	56%	2E-06	1,021	4,65%	61%	1E-06
Deep NN-MQ	4982	1,034	4,74%	59%	9E-07	1,013	4,64%	57%	9E-07
CNN V0	625	1,068	4,89%	46%	1E-06	1,056	4,82%	53%	1E-06
CNN V1	265	1,064	4,81%	53%	1E-06	1,084	4,94%	47%	1E-06
CNN V2	5831	1,129	5,13%	48%	9E-07	1,054	4,80%	52%	1E-06
RNN	661	1,188	5,41%	37%	1E-06	1,061	4,90%	49%	1E-06
LSTM	2581	1,117	5,08%	50%	1E-06	1,082	4,99%	46%	1E-06

Observa-se que, globalmente, a abordagem híbrida obteve resultados ligeiramente melhores que a puramente baseada em dados, embora a diferença seja relativamente pequena. O EAM da abordagem baseada em dados esteve contido entre 1,000 e 1,188, enquanto a da abordagem híbrida entre 0,948 e 1,084. Em termos de ERM, a diferença entre a melhor e a pior rede (independentemente da abordagem) foi de apenas 1,07 ponto percentual, mostrando que todas tiveram um resultado muito semelhante. Os modelos, de um modo geral, obtiveram um ERM inferior a 5%. O resultado da melhor rede híbrida (Deep FFNN) se destacou pela grande diferença em EAM quando comparada à segunda melhor rede (uma diferença de 0,052). É relevante dizer que este resultado se sobressaiu até mesmo em relação aos outros treinos desta mesma arquitetura e abordagem (que obtiveram EAMs flutuando em torno de 1,1). Esse desempenho é portanto atribuído a uma inicialização de parâmetros particularmente boa, o que é, claro, dado ao acaso. Isso mostra a complexidade do problema de otimização devido à grande quantidade de parâmetros e desta tarefa de estimação em particular, mostrando a dificuldade de se encontrar o mínimo global da função objetivo.

A tabela também mostra que a taxa de aprendizado usada no treino dos modelos ótimos foi relativamente parecida (a maior parte em 9×10^{-7} ou 1×10^{-6}). Tratam-se de valores mais próximos do limite inferior testado, indicando uma preferência por uma otimização suave, sem ser demasiadamente lenta. Como em vários casos seus valores são iguais entre abordagens, é possível fazer uma comparação entre o número da época ótima — aquela em que o erro de validação foi mínimo — a fim de comparar a velocidade de aprendizado das abordagens[1]. Observa-se que a abordagem híbrida aprendeu mais rápido, de um modo geral, nos casos onde a comparação é possível. Os valores aproximados da época ótima para cada modelo podem ser encontrados na Tabela 4.2, onde a abordagem mais rápida (quando a comparação é possível) é iluminada em amarelo — no caso das redes Deep SNN e CNN V0, há um empate. As redes em cinza não podem ser comparadas diretamente pois a abordagem de maior LR convergiu mais rapidamente (o que é esperado). A convergência excepcionalmente rápida da rede RNN na abordagem baseada em dados é vista como resultado de uma estimativa inicial dos parâmetros particularmente próxima de um ponto de mínimo. Esses dados não foram incluídos nas tabelas de resultados pois a mesma análise não é relevante para as variáveis seguintes, como ficará claro. O valor é aproximado pois o número da época ótima, embora calculado ao final do treino, não é salvo, e teve de ser recalculado com base nas imagens das curvas de aprendizagem.

Comparando arquiteturas, observa-se que as redes convolucionais e recorrentes se saíram pior na estimação nos dois casos. Isso mostra que o NO_x e o erro de estimação do simulador para essa variável não possuem uma tendência temporal que trouxesse vantagem para o uso desses tipos de rede. De fato, o espaço de tempo entre pontos de dados adjacentes era de no mínimo uma hora, o que não corresponde a uma quantidade de tempo representativa da dinâmica do processo. Assim, fenomenologicamente falando, faz sentido que essas redes não tenham trazido vantagem para a estimação do NO_x, e que tal característica possa talvez ter sido um empecilho ao seu aprendizado.

Dentre as redes do tipo *Feed Forward*, observa-se que a rede SNN rasa se saiu melhor que as redes FFNN rasa e profunda na estimação puramente baseada em dados. Isso é interessante pois, comparativamente à rede NN profunda, ela possui muito menos parâmetros. Surpreendentemente, a SNN profunda se saiu pior que a rasa — indicando que o aumento no número de parâmetros possivelmente dificultou o aprendizado da mesma, uma hipótese melhor discutida

[1] Isso não seria possível se as taxas de aprendizado fossem muito diferentes, pois para valores mais altos a época ótima seria atingida necessariamente mais rápido, embora muito possivelmente com um EAM pior.

Tabela 4.2: Época aproximada de menor erro de validação, obtido a partir das curvas de aprendizagem dos melhores modelos.

	Abordagem baseada em dados		Abordagem Híbrida	
Redes	**Época ótima**	**LR**	**Época ótima**	**LR**
FFNN	350	2E-06	250	1E-06
Deep FFNN	125	3E-06	500	9E-07
SNN	700	2E-06	350	1E-06
Deep SNN	600	9E-07	600	9E-07
NN-MQ	700	2E-06	500	1E-06
Deep NN-MQ	500	9E-07	450	9E-07
CNN V0	400	1E-06	400	1E-06
CNN V1	700	1E-06	350	1E-06
CNN V2	200	9E-07	120	1E-06
RNN	10	1E-06	350	1E-06
LSTM	280	1E-06	50	1E-06

na Seção 4.4. As redes que incluíam diretamente a estimação dos mínimos quadrados também se saíram pior, indicando que a fixação desses parâmetros de antemão diminuiu a flexibilidade da rede em relação ao uso da variável especial. Na abordagem híbrida, as redes NN simples se saíram melhor, indicando que os modelos SNN e NN-MQ apenas impediram que relações não lineares dessa variável fossem modeladas.

A Figura 4.1 mostra graficamente os resultados do melhor modelo baseado em dados e híbrido no conjunto de validação. É evidente que, de um modo geral, ambos os modelos foram capazes de modelar a evolução do NO_x relativamente bem.

Figura 4.1: Resultados para a estimação do NO_x do melhor modelo baseado em dados (SNN) e híbrido (Deep FFNN), comparados contra os dados experimentais.

4.2 Estimação de O_2

Assim como na estimação de NO_x, os resultados (apresentados na Tabela 4.3) foram relativamente semelhantes para as duas abordagens, embora algumas ressalvas devam ser feitas.

Tabela 4.3: Resultados para a estimação de O_2.

		Abordagem baseada em dados				Abordagem híbrida			
Rede	**Parâmetros**	**EAM**	**ERM**	**CC**	**LR**	**EAM**	**ERM**	**CC**	**LR**
FFNN	231	0,042	0,27%	52%	1E-06	0,040	0,26%	47%	8E-06
Deep FFNN	5011	0,041	0,27%	55%	9E-07	0,043	0,28%	41%	9E-07
SNN	210	0,041	0,27%	53%	5E-06	0,044	0,29%	36%	8E-06
Deep SNN	4983	0,036	0,24%	50%	9E-07	0,050	0,33%	26%	9E-07
NN-MQ	209	0,041	0,27%	41%	1E-06	0,035	0,23%	55%	9E-07
Deep NN-MQ	4982	0,044	0,29%	45%	9E-07	0,042	0,27%	36%	9E-07
CNN V0	625	0,040	0,26%	44%	3E-06	0,050	0,33%	41%	8E-06
CNN V1	265	0,039	0,26%	42%	3E-06	0,054	0,36%	39%	8E-06
CNN V2	5831	0,039	0,25%	50%	1E-06	0,042	0,28%	41%	1E-06
RNN	661	0,035	0,23%	61%	1E-06	0,038	0,25%	57%	8E-06
LSTM	2581	0,037	0,24%	52%	3E-06	0,041	0,27%	51%	8E-06

A primeira delas é que o EAM é reportado em concentração percentual, enquanto a de NO_x era em ppm. Em comparação à estimação do NO_x, ambas as abordagens parecem ter obtido resultados mais acurados na estimação do O_2: erros relativos médios de cerca de 0,25%. No entanto, essa melhoria está mais provavelmente associada ao fato desta variável estar contida numa faixa de valores muito mais restrita. No conjunto de validação, a concentração de O_2 se encontra sobretudo entre 15,15% e 15,45%, enquanto o NO_x varia desde 18 ppm até 26 ppm. Pode-se tentar comparar os EAM dessas duas variáveis dividindo seus valores pelos respectivos desvios padrões da variável estimada (no conjunto de validação), a fim de se encontrar uma espécie de acurácia relativa. Isso é útil pois as redes trabalham com valores de entrada e saída normalizados com a média e o desvio padrão dos dados de treino (por mais que as análises apresentadas sejam todas feitas já na escala real). Chega-se assim a:

$$\text{Para o } NO_x\text{: } \frac{0{,}948}{1{,}476313} = 0{,}642 \qquad \text{Para o } O_2\text{: } \frac{0{,}035}{0{,}051242} = 0{,}683 \tag{4.1}$$

Devido à proximidade dos valores, é possível julgar que os modelos obtiveram um desempenho semelhante para as duas variáveis.

A segunda ressalva é que, neste caso, os modelos híbridos obtiveram resultados ligeiramente piores que os baseados em dados. Ainda assim, os modelos híbridos parecem ter aprendido com mais facilidade. Isso é indicado pela taxa de aprendizado dos melhores modelos, que, de um modo geral, foi maior para os modelos híbridos que para seus equivalentes puramente baseados em dados — indicando que uma otimização menos suave seja necessária e que, portanto, chegue-se ao ponto ótimo com mais facilidade e num número menor de iterações. A razão por trás dessa piora dos modelos híbridos não é perfeitamente compreendida. Provavelmente, há alguma particularidade nos dados do erro provenientes do modelo fenomenológico que dificultem o aprendizado pelos modelos baseados em dados de alguma forma. Isso parece ser corroborado pelo fato das redes profundas obterem resultados piores que as rasas nessa abordagem (com

exceção da rede CNN V2, a com mais parâmetros dentre as CNNs), indicando que o número de parâmetros dificultou seu aprendizado. De toda forma, os resultados ainda são muito próximos dos modelos baseados em dados e, baseado no que já foi discutido para o NO_x, podem ser resultado de uma pior inicialização dos parâmetros. A Figura 4.2 mostra graficamente os resultados dos melhores modelos, em comparação com os dados experimentais.

Figura 4.2: Resultados para a estimação de O_2 do melhor modelo baseado em dados (RNN) e híbrido (NN-MQ), comparados contra os dados experimentais.

Em termos de arquitetura, observamos desta vez que as redes convolucionais e recorrentes se saíram melhor que as demais, ao menos para a estimação puramente baseada em dados. Esse resultado é interessante pois apresenta uma tendência exatamente contrária à vista para o NO_x, mostrando que a escolha da arquitetura é realmente dependente da variável de interesse. Isso parece indicar que há alguma dependência temporal importante nos dados do O_2, que favorece o aprendizado dessas redes. Essa tendência é menos clara para os modelos híbridos, já que as redes CNN v0 e v1 obtiveram os piores resultados. A melhor rede na abordagem baseada em dados foi a RNN, enquanto na abordagem híbrida foi a NN-MQ, com resultados praticamente iguais.

4.3 Estimação de CO

Os resultados dos melhores modelos de estimação de CO são apresentados na Tabela 4.4. Os valores de EAM são reportados em ppm.

Diferentemente dos dois últimos casos, nenhuma das redes foi capaz de aprender a evolução do CO, mesmo levando a taxa de aprendizado a valores abaixo de 1×10^{-7}, mostrando que não se trata de uma dificuldade de otimização. Alguns pontos tornam essa conclusão bem clara. O primeiro deles é o coeficiente de correlação negativo encontrado para alguns dos melhores modelos. Lembra-se que o coeficiente de correlação aqui explicitado é dado pela Equação 3.10,

Tabela 4.4: Resultados para a estimação de CO.

		Abordagem baseada em dados				Abordagem híbrida			
Rede	**Parâmetros**	**EAM**	**ERM**	**CC**	**LR**	**EAM**	**ERM**	**CC**	**LR**
FFNN	231	1,678	34,04%	44%	3E-06	1,637	33,10%	-11%	8E-08
Deep FFNN	5011	1,555	31,34%	35%	1E-06	1,531	30,83%	-45%	1E-06
SNN	210	1,108	23,40%	39%	1E-06	1,433	29,31%	43%	8E-06
Deep SNN	4983	1,204	24,70%	-42%	1E-06	0,938	20,51%	42%	1E-06
NN-MQ	209	1,414	28,29%	-23%	3E-06	1,457	29,45%	39%	8E-06
Deep NN-MQ	4982	1,638	33,05%	-17%	1E-06	1,605	32,16%	-42%	1E-06
CNN V0	625	1,843	37,45%	-33%	1E-06	1,674	33,68%	-39%	1E-06
CNN V1	265	1,783	36,34%	-6%	1E-06	1,407	28,89%	30%	1E-06
CNN V2	5831	1,464	29,60%	-22%	1E-06	1,579	31,98%	40%	1E-06
RNN	661	1,414	28,47%	29%	1E-06	1,719	34,88%	-9%	1E-06
LSTM	2581	1,532	30,79%	-14%	1E-06	1,712	34,66%	-14%	1E-06

que pode, matematicamente, ser negativo. Um CC negativo indica uma tendência inversa entre as variáveis estudadas. Não é cabível que a estimação $\hat{y}$ de um modelo apresente relação inversa à variável de interesse y. Para os modelos com CC positivo, esta conclusão não é tão evidente a partir da tabela, já que o ERM por si só também não é capaz de indicá-lo, pelo exposto no final da Seção 3.2.2. O resultado gráfico dos melhores modelos é mostrado na Figura 4.3.

Figura 4.3: Resultados para a estimação de CO do melhor modelo baseado em dados (SNN) e híbrido (Deep SNN), comparados contra os dados experimentais.

A maneira mais clara de se concluir que os modelos não foram capazes de aprender é através da curva de aprendizagem: exceto por quatro casos (dentre os 22 melhores), a curva do erro de validação cresce monotonicamente desde a primeira época, indicando um "sobre-ajuste" desde o começo. Além disso, observa-se na maior parte dos modelos que o erro de treino começa a convergir em poucas iterações, demonstrando que pouco aprendizado é feito sobre esse conjunto. É de se esperar, portanto, que pouca generalização seja alcançada. Destaca-se que esse fenômeno foi observado em ambas as abordagens. Uma comparação entre as curvas

de aprendizado típicas do treino para o CO e para o O_2 é mostrada na Figura 4.4. Ainda assim, algum grau de aprendizado é bem-sucedido na primeira iteração, já que os melhores modelos conseguem prever o crescimento da concentração de CO no final do período de dados, por mais que a acurácia seja baixa. Essa tendência é mostrada na Figura 4.3.

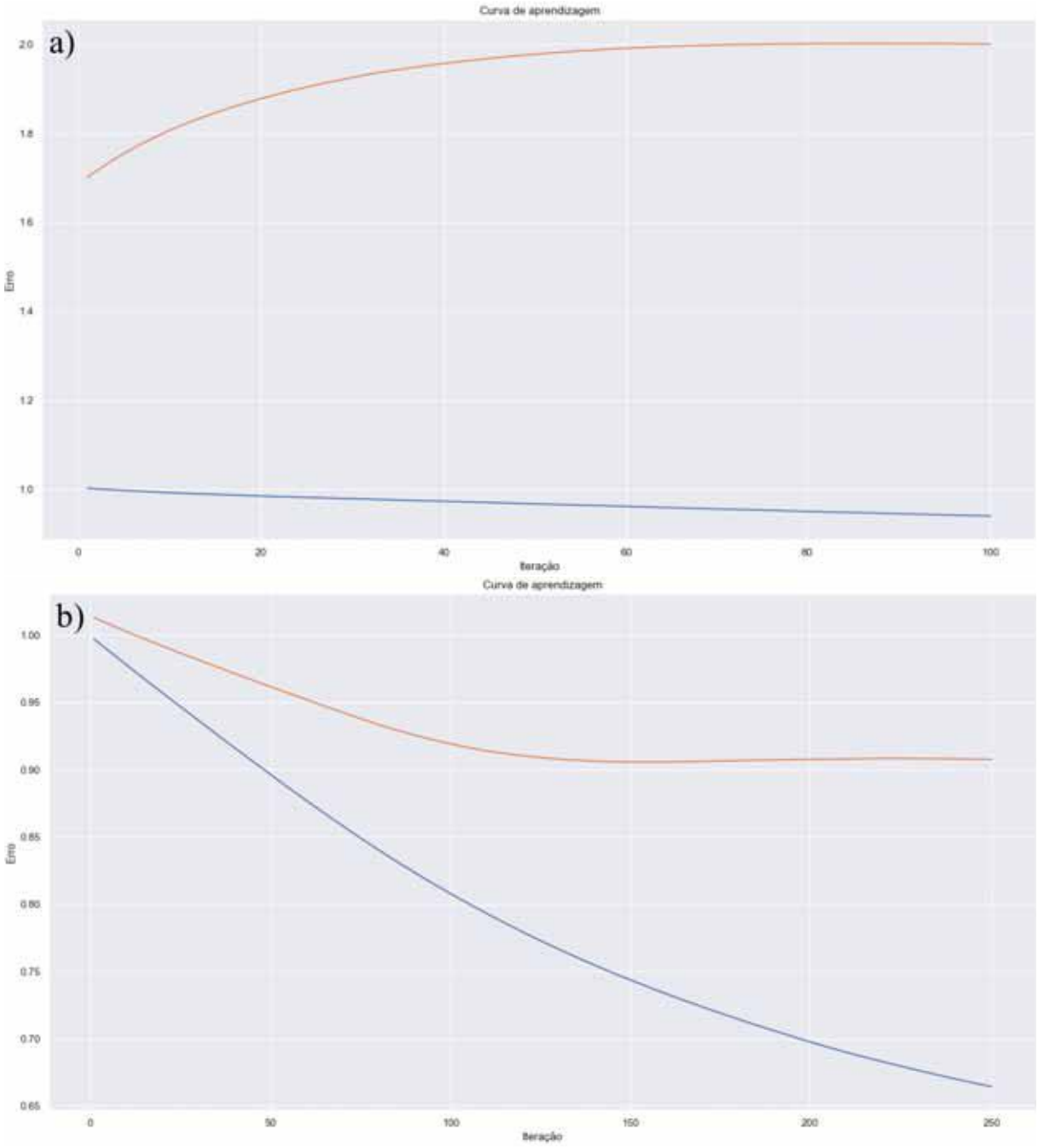

Figura 4.4: Comparação entre as curvas de aprendizagem no conjunto de validação (em laranja) e no conjunto de treino (em azul) para o melhor modelo FFNN na estimação de a) CO e; b) O_2. Esta rede foi selecionada por ilustrar bem a tendência observada no aprendizado do CO, a mesma arquitetura sendo usada para comparação com o O_2.

4.4 Discussão

A variedade dos resultados obtidos mostra como cada uma das variáveis de saída teve uma resposta diferente ao treino. Isso torna inviável, portanto, uma análise **global** sobre a melhor arquitetura de rede, apenas comparações individuais. Como descrito nas seções anteriores, foi observado que a abordagem híbrida aprendeu de fato mais rápido tanto para o NO_x quanto para o O_2, mesmo que, neste último caso, os resultados tenham sido ligeiramente piores quando comparados à abordagem puramente baseada em dados.

É bastante claro, no entanto, que todos os modelos de estimação de NO_x e de O_2 obtiveram um EAM muito próximo entre abordagens e arquiteturas. Acredita-se que a acurácia dos modelos esteja sendo limitada em primeiro lugar pela quantidade de dados disponíveis (cerca de 4500 para o treino, como descrito na Seção 3.2.1), de modo que as melhores redes se destacam apenas graças a uma inicialização favorável dos parâmetros da rede. Não há maneira de se definir um número mínimo necessário para um bom aprendizado dos modelos, visto que se trata de uma variável muito dependente do problema em si e de seu contexto[73–75]. Isso também pode explicar o fato das redes mais complexas não obterem sempre os melhores resultados, já que sua robustez não implica em um melhor desempenho que as demais em *datasets* limitados[76]. Clarifica-se que esses modelos estão de fato *aprendendo* e sendo capazes de *generalizar* informações, como é indicado pelos resultados gráficos e suas curvas de aprendizagem, que mostram um decrescimento do erro de validação, antes de ocorrer sobre-ajuste. Isso é diferente do que ocorre com os modelos de estimação de CO, que mostram muita dificuldade em generalizar, e até mesmo em minimizar o erro de treino.

4.5 Reconciliação dos dados

A reconciliação é uma técnica de tratamento de dados que visa corrigir as medições de forma a garantir que elas respeitem restrições físicas e matemáticas impostas ao sistema e ao conjunto de variáveis, como balanços de massa e energia[77]. Conjuntos de dados costumam apresentar desvios em relação a tais leis por causa de erros e imprecisões dos métodos de medição. A reconciliação se apresenta assim como uma forma de melhorar a consistência das medições e a interpretabilidade física das mesmas. Nesta seção, estuda-se a possibilidade de utilização deste método com os dados de 2015 e seu impacto no desempenho dos modelos.

A partir das variáveis disponíveis, foi apenas possível reconciliar os dados de fração molar no combustível (um total de 12 variáveis), de maneira a garantir que sua soma fosse igual a 100% em todos os instantes de medição. A reconciliação foi feita por simples renormalização da soma dos dados em cada instante de medição. Destaca-se que 5 das 10 entradas dos modelos fazem parte do conjunto de variáveis reconciliadas, de modo que esse tratamento poderia trazer uma diferença significativa ao aprendizado dos modelos.

Todas as arquiteturas de rede foram treinadas ao menos 5 vezes nas duas abordagens para a estimação de NO_x com os novos dados. Os resultados das melhores redes de cada caso são mostrados na Tabela 4.5, ao lado dos resultados originais de estimação para a mesma variável.

A princípio, a reconciliação dos dados pareceu trazer vantagem para a abordagem puramente baseada em dados e para algumas redes da abordagem híbrida — em particular as CNNs. A rede SNN híbrida com dados reconciliados obteve um resultado melhor que a melhor rede dentre as

Tabela 4.5: Comparação entre os melhores resultados de EAM obtidos com os dados originais e os dados reconciliados.

	Abordagem baseada em dados		Abordagem Híbrida	
Redes	**Original**	**Reconciliado**	**Original**	**Reconciliado**
FFNN	1,022	0,950	1,000	1,030
Deep FFNN	1,001	0,986	0,948	0,990
SNN	1,000	1,094	1,052	0,939
Deep SNN	1,065	0,993	1,011	1,042
NN-MQ	1,084	1,038	1,021	1,116
Deep NN-MQ	1,034	1,124	1,013	1,089
CNN V0	1,068	1,029	1,056	1,043
CNN V1	1,064	1,016	1,084	1,000
CNN V2	1,129	1,057	1,054	1,044
RNN	1,188	1,110	1,061	1,141
LSTM	1,117	1,052	1,082	1,056

originais. Para verificar se a reconciliação dos dados traz resultados consistentemente melhores — i.e. que eles não são apenas causados por uma melhor estimativa inicial dos parâmetros — a média do EAM obtido para cada tipo de rede e abordagem é comparada. Tais valores são apresentados na Tabela 4.6.

Observa-se mais uma vez que todos os valores são bastante semelhantes e que não parecem apresentar uma tendência clara. Assim, é possível concluir que, neste caso, o desempenho dos modelos está sendo limitado pela *quantidade* de dados, e não pela sua *qualidade*.

Tabela 4.6: Valores médios de EAM obtidos pelas redes treinadas com os dados originais e reconciliados.

	Abordagem baseada em dados		Abordagem Híbrida	
Redes	**Original**	**Reconciliado**	**Original**	**Reconciliado**
FFNN	1,112	1,093	1,071	1,103
Deep FFNN	1,187	1,062	1,113	1,103
SNN	1,151	1,163	1,136	1,086
Deep SNN	1,099	1,077	1,079	1,109
NN-MQ	1,168	1,138	1,189	1,163
Deep NN-MQ	1,097	1,170	1,116	1,131
CNN V0	1,112	1,139	1,102	1,135
CNN V1	1,153	1,091	1,137	1,079
CNN V2	1,189	1,145	1,144	1,097
RNN	1,174	1,182	1,165	1,215
LSTM	1,147	1,160	1,188	1,124

4.6 Análises com mais dados

No começo de 2022, foram adquiridos dados completos da mesma turbina para os anos de 2012 a 2022, assim como dados da mesma época para outras 5 turbinas. No momento da redação

deste trabalho, os resultados da estimação do modelo fenomenológico para esses dados ainda não estavam disponíveis para análise dos modelos híbridos, e somente análises preliminares com modelos puramente baseados em dados puderam ser feitas.

Foi observado que os modelos já desenvolvidos obtiveram resultados piores nos demais anos, o que era esperado pois os dados com que foram treinados eram limitados em termos de faixa de operação da turbina — um exemplo é mostrado na Figura 4.5. Um modelo só será capaz de estimar com acurácia valores em faixas de operação semelhantes às quais ele foi treinado. Assim, além de se precisar de mais dados, é necessário que o conjunto de treino seja escolhido de forma a abranger a maior variedade de condições das variáveis de entrada e de saída possíveis, a fim de se garantir uma maior versatilidade do modelo. O conjunto de validação, por sua vez, não deve conter faixas de operação não cobertas pelos dados de treino.

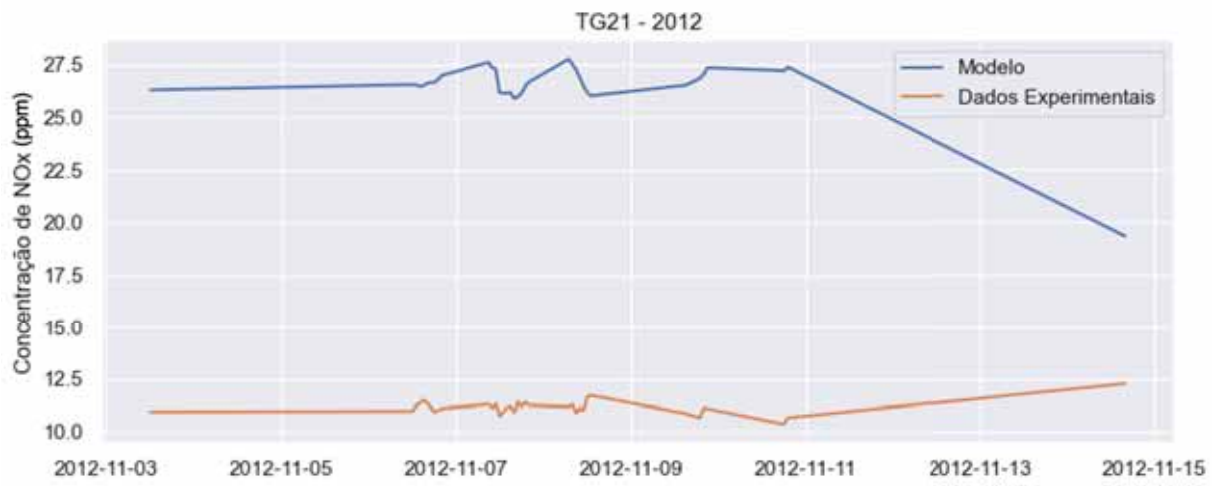

Figura 4.5: Resultados da melhor rede SNN (azul) para dados selecionados da turbina TG21 de 2012 (laranja). O EAM do modelo nesse conjunto foi de 15,4.

Também foi observado que os dados de algumas variáveis tinham uma qualidade pior nos demais anos (valores constantes, repetidos ao longo de vários instantes de medição), fazendo com que a qualidade da estimação também piorasse. Assim, caso o tratamento dos dados não seja capaz de resolver o problema, pode-se também testar o uso de outras variáveis de entrada para a estimação das saídas estudadas neste trabalho.

Capítulo 5

Considerações Finais

5.1 Conclusões

Neste trabalho, foi testado um total de onze arquiteturas de redes neuronais em dois tipos de abordagem: uma puramente baseada em dados e outra híbrida, acoplada a um modelo fenomenológico. Elas foram treinadas para a estimação de 3 variáveis de interesse: a concentração de NO_x, de O_2 e CO na saída da turbina de uma usina termelétrica.

Observou-se que a resposta de cada uma das variáveis às abordagens, arquiteturas e treino foram diferentes, embora a convergência da abordagem híbrida tenha sido mais rápida nos casos em que houve aprendizado significativo. Apesar disso, todos os modelos obtiveram resultados semelhantes entre abordagens e arquiteturas para uma mesma variável, e tudo indica que isso se deve às limitações impostas pelo baixo número de dados disponíveis.

O desenvolvimento de modelos com mais dados se faz necessário para que as redes possam ser treinadas em condições mais robustas e também validadas com mais pontos, o que facilita a comparação entre modelos e permite resultados mais conclusivos. Como discutido em seções anteriores, a quantidade necessária de pontos para se treinar e validar um modelo não é simples de se estimar, e depende muito da complexidade dos modelos e do problema.

5.2 Perspectivas

O desempenho dos modelos na estimação de NO_x e O_2 mostra que o uso de redes neuronais e modelos híbridos para a estimação da emissão de gases é uma tarefa possível, por mais que os resultados para o CO não tenham sido bons. Além disso, a maior facilidade de aprendizado dos modelos híbridos deve trazer vantagens nas análises com maior quantidade de dados. Espera-se que o tamanho das séries históricas recentemente obtidas seja suficiente para continuar as análises após seu tratamento. Atenção especial deve ser dada às condições de operação ao se desenvolver novos conjuntos de treino e validação. Uma seleção adequada das variáveis de entrada, levando em conta a qualidade das medidas, se faz necessária para garantir a robustez do modelo. Métodos de redução de dimensionalidade como a Análise de Componentes Principais (PCA)[78] podem ser utilizados a fim de reduzir múltiplas variáveis medidas a poucas variáveis de entrada.

Referências Bibliográficas

1 CHAKRAVARTHY, S., VOHRA, A., GILL, B. Predictive emission monitors (PEMS) for NOx generation in process heaters. **Computers & Chemical Engineering**, v. 23, n. 11, pp. 1649 – 1659, 2000.

2 BRASIL. Resolução CONAMA Nº 491, de 19/11/2018. **Diário Oficial da União**, Ministério do Meio Ambiente (MMA), Conselho Nacional do Meio Ambiente, 19 nov. 2018.

3 PAOLETTI, E., BYTNEROWICZ, A., ANDERSEN, C., AUGUSTAITIS, A., FERRETTI, M., GRULKE, N., GÜNTHARDT-GOERG, M. S., INNES, J., JOHNSON, D., KARNOSKY, D., LUANGJAME, J., MATYSSEK, R., MCNULTY, S., MÜLLER-STARCK, G., MUSSELMAN, R., PERCY, K. Impacts of Air Pollution and Climate Change on Forest Ecosystems — Emerging Research Needs. **The Scientific World JOURNAL**, v. 7, pp. 783864, Jan 1900.

4 LOVETT, G. M., TEAR, T. H., EVERS, D. C., FINDLAY, S. E. G., COSBY, B. J., DUNSCOMB, J. K., DRISCOLL, C. T., WEATHERS, K. C. Effects of Air Pollution on Ecosystems and Biological Diversity in the Eastern United States. **Annals of the New York Academy of Sciences**, v. 1162, n. 1, pp. 99–135, 2009.

5 BRASIL. Resolução CONAMA Nº 436, de 22/12/2011. **Diário Oficial da União**, Ministério do Meio Ambiente (MMA), Conselho Nacional do Meio Ambiente, 22 dez. 2011.

6 ALBUQUERQUE, I., ALENCAR, A., ANGELO, C., AZEVEDO, T., BARCELLOS, F., COLUNA, I., JUNIOR, C. C., CREMER, M., PIATTO, M., POTENZA, R., QUINTANA, G., SHIMBO, J., TSAI, D., ZIMBRES, B. Análise das Emissões Brasileiras de Gases de Efeito Estufa E Suas Implicações Para as Metas de Clima do Brasil. **Sistema de Estimativas de Emissões e Remocões de Gases de Efeito Estufa (SEEG)**, 2020.

7 MALAR, J. P. Crise energética deve aliviar em 2022, mas espaço para queda em contas é pequeno. **CNN Brasil**. Dec 2021. Disponível em: `https://www.cnnbrasil.com.br/business/crise-energetica-deve-aliviar-em-2022-mas-espaco-para-queda-em-contas-e-pequeno/#:~:text=O%20Brasil%20passou%20em%202021,risco%20de%20apag%C3%B5es%20ou%20racionamento`. Último acesso em 21 de junho de 2022.

8 MATOS, R. A. D. S. **Balanço Energético Nacional 2021**. Relatório, Empresa de Pesquisa Energética, 2021.

9 ANEEL. **BIG - Banco de Informações de Geração**. Disponível em: `http://www2.aneel.gov.br/aplicacoes/capacidadebrasil/capacidadebrasil.cfm`. Último acesso em 03 de Outubro de 2021.

10 JAHNKE, J. A. **Continuous emission monitoring**. New York, John Wiley & Sons, 2000.

11 HADJISKI, M., BOSHNAKOV, K., CHRISTOVA, N. Simulation-based Predictive Emission Monitoring System. *In*: 19th EUROPEAN CONFERENCE ON MODELLING AND SIMULATION, 2005. **Proceedings 19th European Conference on Modelling and Simulation**, ISBN 1-84233-112-4, 2005.

12 U. S. Environmental Protection Agency. **Appendix A-1 to Part 60 - Test Methods 1 through 2FS**. Environmental Protection Agency, Subchapter C- Air Programs, 2016.

13 LJUNG, L. Black-box models from input-output measurements". *In*: : IMTC 2001. **Proceedings of the 18th IEEE Instrumentation and Measurement Technology Conference. Rediscovering Measurement in the Age of Informatics**, v. 1, pp. 138–146 vol.1, 2001.

14 BUHRMESTER, V., MUNCH, D., ARENS, M. Analysis of Explainers of Black Box Deep Neural Networks for Computer Vision: A Survey, **ARXIV**, 2019. Disponível em: `https://arxiv.org/abs/1911.12116`. Último acesso em 21 de junho de 2022.

15 XIONG, Q., JUTAN, A. Grey-box modelling and control of chemical processes. **Chemical Engineering Science**, v. 57, n. 6, pp. 1027–1039, 2002.

16 LECUN, Y., HAFFNER, P., BOTTOU, L., BENGIO, Y. Object Recognition with Gradient-Based Learning, Berlin, Heidelberg, **Springer Berlin Heidelberg**, pp. 319–345, 1999.

17 LECUN, Y., BENGIO, Y. Convolutional networks for images, speech, and time series. **The handbook of brain theory and neural networks**, v. 3361, n. 10, pp. 1995, 1995.

18 COLLOBERT, R., WESTON, J. A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning. *In*: : ICML '08. **Proceedings of the 25th International Conference on Machine Learning**, p. 160–167, New York, NY, USA, 2008.

19 TSANTEKIDIS, A., PASSALIS, N., TEFAS, A., KANNIAINEN, J., GABBOUJ, M., IOSIFIDIS, A. Forecasting Stock Prices from the Limit Order Book Using Convolutional Neural Networks. *In*: : **2017 IEEE 19th Conference on Business Informatics (CBI)**, v. 01, pp. 7–12, 2017.

20 ALIPANAHI, B., DELONG, A., WEIRAUCH, M. T., FREY, B. J. Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning. **Nature Biotechnology**, v. 33, n. 8, pp. 831–838, Aug 2015.

21 MILJANOVIC, M. Comparative analysis of recurrent and finite impulse response neural networks in time series prediction. **Indian Journal of Computer Science and Engineering**, v. 3, n. 1, pp. 180–191, 2012.

22 SAK, H., SENIOR, A., BEAUFAYS, F. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. **Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH**, pp. 338–342, 01 2014.

23 YIN, W., KANN, K., YU, M., SCHÜTZE, H. Comparative Study of CNN and RNN for Natural Language Processing. **ARXIV**, 2017.

24 SANDVIG NIELSEN, J. **PEMS: Advanced predictive emission monitoring**, Jul 2010.

25 FICHET, V., KANNICHE, M., PLION, P., GICQUEL, O. A reactor network model for predicting NOx emissions in gas turbines. **Fuel**, v. 89, n. 9, pp. 2202–2210, 2010.

26 DE TONI, A., HAYASHI, T., SCHNEIDER, P. A reactor network model for predicting NOx emissions in an industrial natural gas burner. **Journal of the Brazilian Society of Mechanical Sciences and Engineering**, v. 35, 10 2013.

27 NOVOSSELOV, I., MALTE, P., YUAN, S., SRINIVASAN, R. Chemical Reactor Network Application to Emissions Prediction for Industial DLE Gas Turbine. **Proceedings of the ASME Turbo Expo**, v. 1, 01 2006.

28 ANTSON, O. K., PELLIKKA, T. **Predictive Emission Monitoring Systems , PEMS and their acceptance and use in Europe**. VTT Technical Research Centre of Finland. 2014.

29 SWANSON, B., LAWRENCE, P. An Alternative Approach to Continuous Compliance Monitoring and Turbine Plant Optimization using a PEMS (Predictive Emission Monitoring System). *In*: : **18th Symposium of the Industrial Application of Gas Turbines Comittee**. Banff, Alberta, Canada, 2009.

30 BELLONE, M., KARAYIANNIDIS, Y., FAGHANI, E. Comparison of CNN and LSTM for Modeling Virtual Sensors in an Engine. **SAE International Journal of Advances and Current** Practices in Mobility, v. 2, n. 5, pp. 2632–2639, 2020.

31 ARSIE, I., PIANESE, C., SORRENTINO, M. Development of recurrent neural networks for virtual sensing of NOx emissions in internal combustion engines. **SAE International Journal of Fuels and Lubricants**, v. 2, n. 2, pp. 354–361, 2010.

32 YANG, G., WANG, Y., LI, X. Prediction of the NOx emissions from thermal power plant using long-short term memory neural network. **Energy**, v. 192, pp. 116597, 2020.

33 WANG, X., LIU, W., WANG, Y., YANG, G. A hybrid NOx emission prediction model based on CEEMDAN and AM-LSTM. **Fuel**, v. 310, pp. 122486, 2022.

34 MUHAMMAD, M., MOHAMMADREZA, T. B., ABDUL KARIM, Z. A. Methodology for short-term performance prognostic of gas turbine using recurrent neural network. *In*: : **2015 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)**, pp. 787–791, 2015.

35 ASGARI, H., ORY, E., LAPPALAINEN, J. Recurrent Neural Network Based Simulation of a Single Shaft Gas Turbine. *In*: : **Proceedings of The 61st SIMS Conference on Simulation and Modelling SIMS 2020**, pp. 99–106, 2021.

36 BAI, M., YANG, X., LIU, J., LIU, J., YU, D. Convolutional neural network-based deep transfer learning for fault detection of gas turbine combustion chambers. **Applied Energy**, v. 302, pp. 117509, 2021.

37 VANDERHAEGEN, E., DENEVE, M., LAGET, H., FANIEL, N. Predictive Emissions Monitoring Using a Continuously Updating Neural Network. *In*: : **Proceedings of the Turbo Expo: Power for Land, Sea, and Air**, v. 2, 10 2010.

38 POOLE, D. Computational intelligence : a logical approach. **New York, Oxford University Press**, 1998.

39 TURING, A. M. COMPUTING MACHINERY AND INTELLIGENCE. **Mind**, v. LIX, n. 236, pp. 433–460, 10 1950.

40 SAMUEL, A. L. Some Studies in Machine Learning Using the Game of Checkers. **IBM Journal of Research and Development**, v. 3, n. 3, pp. 210–229, 1959.

41 KAVLAKOGLU, E. AI vs. Machine Learning vs. Deep Learning vs. neural networks: What's the difference?. **IBM**, 2020. Disponível em: `https://www.ibm.com/cloud/blog/ai-vs-machine-learning-vs-deep-learning-vs-neural-networks`. Último acesso em 23 de junho de 2022.

42 ZHANG, G. Neural networks for classification: a survey. **IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)**, v. 30, n. 4, pp. 451–462, 2000.

43 MCCULLOCH, W., PITTS, W. A logical calculus of the ideas immanent in nervous activity. **Bulletin of Mathematical Biology**, v. 52, pp. 99–115, 1990.

44 ROSENBLATT, F. The Perceptron: A Probabilistic Model for Information Storage and Organization in The Brain. **Psychological Review**, pp. 65–386, 1958.

45 LENAIL, A. NN-SVG: Publication-Ready Neural Network Architecture Schematics. **Journal of Open Source Software**, v. 4, n. 33, pp. 747, 2019.

46 HORNIK, K. Multilayer Feedforward Networks are Universal Approximators. **Neural Networks**, v. 2, n. 5, pp. 359–366, 1989.

47 LESHNO, M., LIN, V. Y., PINKUS, A., SCHOCKEN, S. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. **Neural Networks**, v. 6, n. 6, pp. 861–867, 1993.

48 ROLNICK, D., TEGMARK, M. The power of deeper networks for expressing natural functions. **ARXIV**, 2017. Disponível em: `https://arxiv.org/abs/1705.05502`. Último acesso em 23 de junho de 2022.

49 RIPLEY, B. D. Neural networks and related methods for classification. **Journal of the Royal Statistical Society: Series B (Methodological)**, v. 56, n. 3, pp. 409–437, 1994.

50 MARTÍNEZ, J. M. Practical quasi-Newton methods for solving nonlinear systems. **Journal of Computational and Applied Mathematics**, v. 124, n. 1, pp. 97–121, 2000.

51 BROYDEN, C. G. The Convergence of a Class of Double-rank Minimization Algorithms: 2. The New Algorithm. **IMA Journal of Applied Mathematics**, v. 6, n. 3, pp. 222– 231, 09 1970.

52 GAVIN, H. P. The Levenberg-Marquardt algorithm for nonlinear least squares curvefitting problems. **Department of Civil and Environmental Engineering, Duke University**, pp. 1–19, 2019.

53 SHEWCHUK, J. R. **An Introduction to the Conjugate Gradient Method Without the Agonizing Pain**, School of Computer Science, Carnegie Mellon University, Pittsburgh, 1994.

54 BOTTOU, L. Large-Scale Machine Learning with Stochastic Gradient Descent. *In*: : Lechevallier, Y., Saporta, G. (eds.), **Proceedings of COMPSTAT'2010**, pp. 177–186, Heidelberg, 2010.

55 JOSEPH, V. R. Optimal ratio for data splitting. **Statistical Analysis and Data Mining: The ASA Data Science Journal** (2022), 1– 8.

56 KESKAR, N. S., MUDIGERE, D., NOCEDAL, J., SMELYANSKIY, M., TANG, P. T. P. On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima. **ARXIV**, 2016. Disponível em: `https://arxiv.org/abs/1609.04836`. Último acesso em 23 de junho.

57 VIDAL, C., MALYSZ, P., KOLLMEYER, P., EMADI, A. Machine Learning Applied to Electrified Vehicle Battery State of Charge and State of Health Estimation: State-of-the-Art. **IEEE Access**, v. 8, pp. 52796–52814, 2020.

58 BENGIO, Y., SIMARD, P., FRASCONI, P. Learning long-term dependencies with gradient descent is difficult **IEEE Trans Neural Netw**, v. 5, n. 2, pp. 157–166, 1994.

59 HOCHREITER, S., BENGIO, Y., FRASCONI, P., SCHMIDHUBER, J. **Gradient flow in recurrent nets: the difficulty of learning long-term dependencies**, 2001.

60 HOCHREITER, S., SCHMIDHUBER, J. Long Short-Term Memory. **Neural Computation**, v. 9, n. 8, pp. 1735–1780, 11 1997.

61 BUI, L., JOSWIAK, M., CASTILLO, I., PHILLIPS, A., YANG, J., HICKMAN, D. A Hybrid Modeling Approach for Catalyst Monitoring and Lifetime Prediction. **ACS Engineering Au**, v. 2, n. 1, pp. 17–26, 2022.

62 ALLEMANG, D., HENDLER, J. Chapter 15 - Expert modeling in OWL. *In*: : Allemang, D., Hendler, J. (eds.), **Semantic Web for the Working Ontologist (Second Edition)**, second edition ed., Boston, Morgan Kaufmann, pp. 325–333, 2011.

63 RAISSI, M., PERDIKARIS, P., KARNIADAKIS, G. Data-driven Solutions of Nonlinear Partial Differential Equations. **Physics Informed Deep Learning (Part I)**, 2017.

64 SOARES, R. D. P., SECCHI, A. EMSO: A new environment for modelling, simulation and optimisation. *In*: : Kraslawski, A., Turunen, I. (eds.), **European Symposium on Computer Aided Process Engineering-13**, v. 14, **Computer Aided Chemical Engineering**, Elsevier, pp. 947–952, 2003.

65 TANG, Y., ZHANG, J., JIA, B., HE, Z., XIA, Y. Investigation on the solution of nitric oxide emission model for diesel engine using optimization algorithms. **Fuel**, v. 228, pp. 81 – 91, 2018.

66 LIU, Q., PROUST, C., GOMEZ, F., LUART, D., LEN, C. The prediction multi-phase, multi reactant equilibria by minimizing the Gibbs energy of the system: Review of available techniques and proposal of a new method based on a Monte Carlo technique. **Chemical Engineering Science**, v. 216, pp. 115433, 2020.

67 MCBRIDE, B. J., ZEHE, M. J., GORDON, S. NASA Glenn Coefficients for Calculating Thermodynamic Properties of Individual Species, Report, **NASA Glenn Research Center**, 2002.

68 SOAVE, G. Equilibrium constants from a modified Redlich-Kwong equation of state. **Chemical Engineering Science**, v. 27, n. 6, pp. 1197–1203, 1972.

69 SRIVASTAVA, N., HINTON, G., KRIZHEVSKY, A., SUTSKEVER, I., SALAKHUTDINOV, R. Dropout: a simple way to prevent neural networks from overfitting. **The journal of machine learning research**, v. 15, n. 1, pp. 1929–1958, 2014.

70 MAAS, A. L., HANNUN, A. Y., NG, A. Y. Rectifier nonlinearities improve neural network acoustic models. *In*: : **Proc. ICML**, v. 30, p. 3, Citeseer, 2013.

71 ELMAN, J. L. Finding structure in time. **Cognitive science**, v. 14, n. 2, pp. 179–211, 1990.

72 KINGMA, D. P., BA, J. Adam: A Method for Stochastic Optimization. **ARXIV**, 2014. Disponível em: `https://arxiv.org/abs/1412.6980`. Último acesso em 23 de junho de 2022.

73 AJIBOYE, A. R., ABDULLAH-ARSHAH, R., QIN, H., ISAH-KEBBE, H. Evaluating the effect of dataset size on predictive model using supervised learning technique. **International Journal of Software Engineering & Computer Sciences (IJSECS)**, v. 1, pp. 75–84, 2015.

74 BARBEDO, J. G. A. Impact of dataset size and variety on the effectiveness of deep learning and transfer learning for plant disease classification. **Computers and Electronics in Agriculture**, v. 153, pp. 46–53, 2018.

75 RAUDYS, S. J., JAIN, A. K. Small sample size effects in statistical pattern recognition: Recommendations for practitioners. **IEEE Transactions on pattern analysis and machine intelligence**, v. 13, n. 3, pp. 252–264, 1991.

76 ALTHNIAN, A., ALSAEED, D., AL-BAITY, H., SAMHA, A., DRIS, A. B., ALZAKARI, N., ELWAFA, A. A., KURDI, H. Impact of Dataset Size on Classification Performance: An Empirical Evaluation in the Medical Domain. **Applied Sciences**, v. 11, n. 2, 2021.

77 FRANÇA, R., SOUZA, D., OLIVEIRA JÚNIOR, A. A reconciliação de dados na resolução de problemas industriais. **Scientia Plena**, v. 12, 07 2016.

78 PEARSON, K. On lines and planes of closest fit to systems of points in space. **The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science**, v. 2, n. 11, pp. 559–572, 1901.

Apêndice A

Retropropagação através do tempo

Neste apêndice, é mostrado em mais detalhes como o método de retropropagação do gradiente é adaptado para redes recorrentes, visto que a estimação para cada instante de tempo é dependente das estimações que a precedem.

Por motivos didáticos, será utilizada a estrutura da Figura A.1, a seguir, para a RNN estudada. A expressão de $\hat{y}$ é dada pela Equação A.1, na qual f é uma função de ativação.

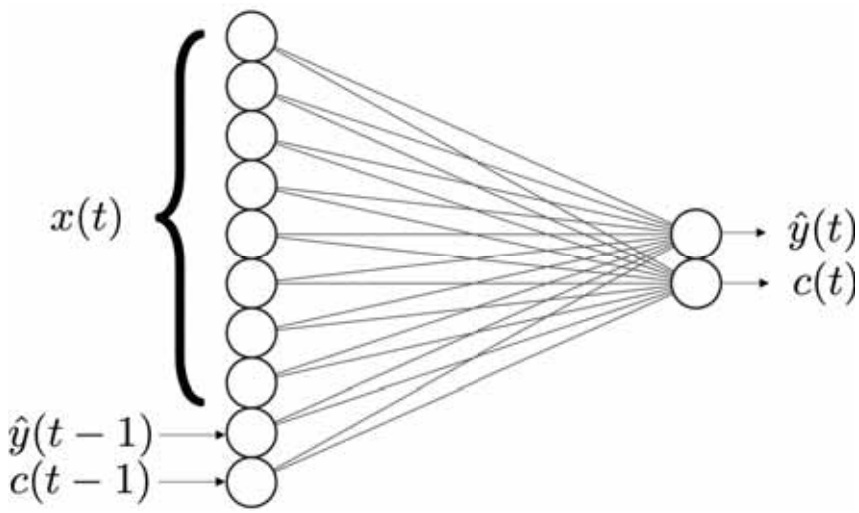

Figura A.1: Estrutura da RNN de exemplo.

$$\hat{y}(t) = f\left(w_y\hat{y}(t-1) + w_c c(t-1) + \sum_{i=1}^{8} w_i x_i\right) \tag{A.1}$$

Durante o treino, é necessário calcular a derivada da função erro l em relação ao parâmetro w_y. Para encontrá-la, utiliza-se a regra da cadeia. Como é possível notar, esse gradiente depende das estimações anteriores, que por sua vez também dependem de w_y. Essa dependência é mostrada no desdobramento temporal dessa rede, mostrado na Figura A.2.

$$l(t) = (y - \hat{y})^2 \qquad \Rightarrow \frac{\partial l}{\partial \hat{y}(t)} = -2(y - \hat{y}(t)) \tag{A.2}$$

$$\hat{y}(t) = f(z(t)) \qquad \Rightarrow \frac{\partial \hat{y}(t)}{\partial z(t)} = \frac{df}{dx}\Big|_{z(t)} \tag{A.3}$$

$$z(t) = w_y \hat{y}(t-1) + \cdots \qquad \Rightarrow \frac{\partial z(t)}{\partial w_y} = \hat{y}(t-1) + w_y \frac{\partial(\hat{y}(t-1))}{\partial w_y} \tag{A.4}$$

$$\hat{y}(t-1) = f(z(t-1)) \qquad \Rightarrow \frac{\partial \hat{y}(t-1)}{\partial z(t-1)} = \frac{df}{dx}\Big|_{z(t-1)} \tag{A.5}$$

$$\vdots \qquad\qquad \vdots \tag{A.6}$$

$$\frac{\partial l}{\partial w_y} = \frac{\partial l}{\partial \hat{y}(t)} \frac{\partial \hat{y}(t)}{\partial z(t)} \frac{\partial z(t)}{\partial w_y} = -2(y - \hat{y}(t)) \frac{df}{dx}\Big|_{z(t)} \left(\hat{y}(t-1) + w_y \frac{\partial(\hat{y}(t-1))}{\partial w_y} \right) \tag{A.7}$$

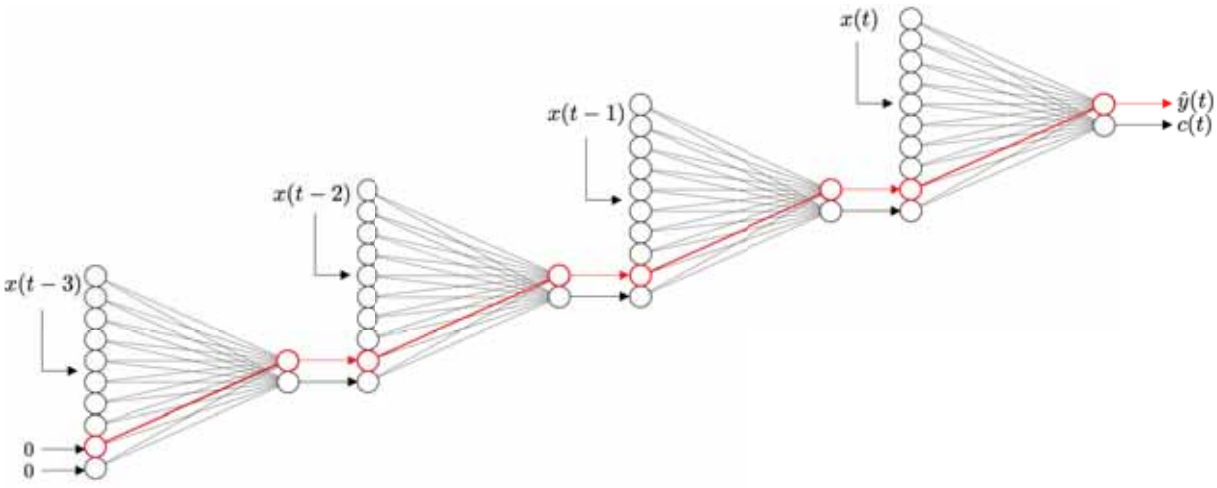

Figura A.2: Estrtura da RNN de exemplo desdobrada temporalmente. Neste caso, a sequência utilizada possui apenas 4 instantes de tempo.

Observe que a derivada da função de ativação precederá a derivada de cada instante de tempo anterior. Para as funções de ativação mais comuns, essa derivada costuma ter módulo menor que 1, de modo que as multiplicações sucessivas realizadas nesse cálculo tendem a reduzir a importância dos primeiros pontos da sequência a zero. Esse é um problema comum das RNNs, e que costuma dificultar o aprendizado de redes para aplicações que requerem um longo efeito de "memória". Esse problema tem sido contornado através de arquiteturas engenhosas com o uso de "portões", como é o caso da rede LSTM[60] citada na Seção 2.5.2.