**MINISTÉRIO DA EDUCAÇÃO**
**Universidade Federal do Triângulo Mineiro**
**Programa de Pós-graduação em Ciência e Tecnologia de Materiais**

LUCAS DE OLIVEIRA DAMANTE

# MAXIMIZAÇÃO DA EFICIÊNCIA ENERGÉTICA DE SUPERCAPACITORES DE GRAFENO USANDO RNA (REDES NEURAIS ARTIFICIAIS)

UBERABA, MG
2023

LUCAS DE OLIVEIRA DAMANTE

# MAXIMIZAÇÃO DA EFICIÊNCIA ENERGÉTICA DE SUPERCAPACITORES DE GRAFENO USANDO RNA (REDES NEURAIS ARTIFICIAIS)

Dissertação apresentada ao Curso de Mestrado em Ciências e Tecnologia de Materiais (PPGCTM), na área de Concentração: Materiais Não-Metálicos, da Universidade Federal do Triângulo Mineiro, como requisito parcial para obtenção do título de mestre em Ciências e Tecnologias de Materiais.

Orientador: Prof. Dr. Anderson Barbosa Lima

UBERABA, MG
2023

**Catalogação na fonte: Biblioteca da Universidade Federal do Triângulo Mineiro**

D16m

Damante, Lucas de Oliveira
Maximização da eficiência energética de supercapacitores de grafeno usando redes neurais artificias (RNA) / Lucas de Oliveira Damante. -- 2023.
104 p. : il., graf., tab.

Dissertação (Mestrado em Ciência e Tecnologia de Materiais) -- Universidade Federal do Triângulo Mineiro, Uberaba, MG, 2023
Orientador: Prof. Dr. Anderson Barbosa Lima

1. Redes neurais (Computação). 2. Capacitores. 3. Grafeno. 4. Inteligência artificial. I. Lima, Anderson Barbosa. II. Universidade Federal do Triângulo Mineiro. III. Título.

CDU 004.032.26:621.319.4

Leila Aparecida Anastácio CRB-6/2513

LUCAS DE OLIVEIRA DAMANTE

**Maximização da Eficiência Energética de Supercapacitores de Grafeno Usando RNA (redes neurais artificiais)**

Dissertação apresentada ao Programa de Pós-Graduação em Ciência e Tecnologia de Materiais, área de concentração - Materiais Aplicados em Biociências, da Universidade Federal do Triângulo Mineiro como requisito para obtenção do título de mestre.

Uberaba-MG, 16 de janeiro de 2023

**Banca Examinadora:**

Prof. Dr. Anderson Barbosa Lima – Orientador
Universidade Federal do Triângulo Mineiro

Prof. Dr. Leandro Cruvinel Mendes
Universidade Federal do Triângulo Mineiro

Prof. Dr. José Ricardo Gonçalves Manzan
Instituto Federal do Triângulo Mineiro

Documento assinado eletronicamente por **LEANDRO CRUVINEL LEMES, Professor do Magistério Superior**, em 25/01/2023, às 18:11, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020 e no art. 34 da Portaria Reitoria/UFTM nº 87, de 17 de agosto de 2021.

Documento assinado eletronicamente por **José Ricardo Gonçalves Manzan, Usuário Externo**, em 25/01/2023, às 21:42, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020 e no art. 34 da Portaria Reitoria/UFTM nº 87, de 17 de agosto de 2021.

A autenticidade deste documento pode ser conferida no site http://sei.uftm.edu.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **0912156** e o código CRC **1E9443A7**.

Dedico este trabalho a Deus!

# AGRADECIMENTOS

Agradeço primeiramente a DEUS que sempre está comigo em todas as fases da minha vida. Agradeço a minha Mãe Joana D'arc e a minha Madrinha Neusa, pessoas que são a base. Agradeço por tudo que me proporcionaram nesta vida, incluindo o carinho, amor e todas as condições necessárias para realizar este trabalho. Agradeço ao meu companheiro Antônio pelo apoio incondicional e principalmente por estar sempre por perto orientando com seus elogios e principalmente pelas suas críticas.

Agradeço a meu orientador Prof. Dr. Anderson Barbosa Lima por cada mínimo detalhe em suas aulas e por todas as suas orientações, pelo seu exemplo de profissionalismo e pela oportunidade e confiança durante o decorrer do Mestrado, me proporcionando experiências científicas, além de obter novos conhecimentos me ensinando muito sobre o que é pesquisar.

Agradeço aos professores do curso de Mestrado, dedicando tempo para reflexões e recomendações. Agradeço aos colegas de curso e em especial ao meu colega de curso, Guilherme que tanto contribuiu nos aprendizados sobre a linguagem de programação. Agradeço ao grupo de pesquisa do Prof. Dr. Rogério Gelamo, e pelo seu aluno Msc. Gabriel de Sousa Augusto, por fornecer os dados experimentais na fabricação de supercapacitores, imprescindíveis para este trabalho. Agradeço a Universidade Federal do Triângulo Mineiro por me proporcionar a oportunidade de realizar o mestrado e de contribuir para o meio acadêmico como cientista.

“Ciência é conhecimento organizado. Sabedoria é vida organizada.”

Immanuel Kant

## RESUMO

Nesta dissertação apresentamos que os supercapacitores podem ser otimizados usando simulações computacionais, desde que seguido uma série de procedimentos técnicos. Eletrônicos em geral cada vez mais usam em suas configurações armazenadores de energia: capacitores, baterias e os supercapacitores, por isso, este tema é muito relevante pesquisar. O trabalho em si tem como objetivo estudar o método de RNA (Redes Neurais Artificiais) para obtenção de capacitores e supercapacitores eletroquímicos, bem como calcular capacitâncias e testar as principais funções de ativação de uma rede neural: *ReLu, Sigmoid, Softmax, Softplus, Tanh*. Contudo, este trabalho tem um enfoque principal em (IA) Inteligência Artificial, bem como a obtenção da maximização da eficiência de supercapacitores eletroquímicos de grafeno, usando o método de RNA (*Artificial Neural Networks - ANN*). Além disso, foi desenvolvida uma caracterização da configuração das redes neurais de *Machine Learning*, enfatizando estratégias de como obter configurações máximas de supercapacitores eletroquímicos otimizados.



**Palavras-chave:** rede neural artificial, supercapacitores, grafeno, inteligência artificial

## ABSTRACT

In this dissertation we present that supercapacitors can be prepared using simply computer simulations, provided that a series of procedures is followed. Electronics in general increasingly use energy storage prototypes in their configurations: capacitors and supercapacitors, so this topic is very relevant to research. The research itself aims to study the ANN (Artificial Neural Networks) method to obtain capacitors and supercapacitors, as well as calculate capacitances and test the main activation functions of a neural network: ReLu, Sigmoid, Softmax, Softplus, Tanh . However, this research has a main focus on (AI) Artificial Intelligence, as well as the achievement of maximizing the efficiency of graphene supercapacitors, using the RNA (Artificial Neural Networks - ANN) method. In addition, a characterization of the configuration of machine learning neural networks was developed, emphasizing strategies on how to obtain supercapacitors.



**Keywords:** artificial neural network, supercapacitor, graphene, artificial intelligence

## LISTA DE ILUSTRAÇÕES

**Figura 1** - Esquema de montagem de um capacitor de placas paralelas.....24
**Figura 2** - Capacitores: dispositivo capaz de acumular cargas elétricas.....26
**Figura 3** - Ilustração das etapas de montagem de um supercapacitor.....27
**Figura 4** - Representação dos íons adsorvidos na interface do eletrodo.....28
**Figura 5** - Gráfico de Ragone .....29
**Figura 6** – Diagrama sobre como são as estruturas do Nanotubo de Carbono e Grafeno.....30
**Figura 7** – Esquema Redes neurais artificiais.....31
**Figura 8** - Modelo Perceptron.....33
**Figura** 9 - Ambientes virtuais do (Anaconda Navigator).....37
**Figura** 10 - Spyder (Ambiente de desenvolvimento científico em Python).....38
**Figura** 11 - Colab (Ambiente de desenvolvimento científico gratuito).....38
**Figura** 12 – Interface Colab (Ambiente de desenvolvimento científico gratuito).....39
**Figura** 13 - Arquitetura da rede neural artificial principal.....41
**Figura** 14 - Estrutura de código (script) da rede neural artificial.....41
**Figura** 15 – Gráfico de Ragone.....42
**Figura** 16 - Gráfico da Função de Ativação ReLu.....45
**Figura** 17 - Gráfico da Função de Ativação Sigmoid.....46
**Figura** 18 - Derivada da Função de Ativação ReLu.....47
**Figura** 19 - Derivada da Função de Ativação Softplus.....48
**Figura** 20 - Derivada da Função de Ativação Softplus.....48
**Figura** 21 - Derivada da Função de Ativação Tanh.....49
**Figura** 22 – Software de predição em cada intervalo.....52
**Figura** 23 - Gráficos do aprendizado em cada configuração ReLu.....55
**Figura** 24 - Gráficos do aprendizado em cada configuração Sigmoid.....57
**Figura** 25 - Gráficos do aprendizado em cada configuração Softplus.....58
**Figura** 26 - Rede neural para os mapas de desempenho de supercapacitores.....60
**Figura** 27 - Arquitetura da rede neural artificial .....63
**Figura** 28 - Gráficos do aprendizado em cada configuração ReLu.....64
**Figura** 29 - Gráficos do aprendizado em cada configuração Sigmoid.....66
**Figura** 30 - Gráficos do aprendizado em cada configuração Softmax.....67
**Figura** 31 - Gráficos do aprendizado em cada configuração Softplus.....69

**Figura** 32 - Gráficos do aprendizado em cada configuração Tanh.....70
**Figura** 33 - Gráficos do aprendizado em cada configuração ReLu (3D).....71
**Figura** 34 - Gráficos do aprendizado em cada configuração Sigmoid (3D).....73
**Figura** 35 - Gráficos do aprendizado em cada configuração Softmax (3D).....75
**Figura** 36 - Gráficos do aprendizado em cada configuração Softplus (3D).....76
**Figura** 37 – Rede neural e gráficos de aprendizado em diferentes números de épocas: ReLu.....79
**Figura** 38 – Rede neural e gráficos de aprendizado em diferentes números de épocas: Softplus.....81
**Figura** 39 – Rede neural e gráficos de aprendizado em diferentes números de épocas.....82
**Figura** 40 - Erros da (IA) Métricas de Machine Learning.....84

## LISTA DE TABELAS

**Tabela 1** - Processo de aprendizagem para treinamento teste via algoritmo: *ReLu*.....55
**Tabela 2** - Processo de aprendizagem para treinamento teste via algoritmo: *sigmoid*.....56
**Tabela 3** - Processo de aprendizagem para treinamento teste via algoritmo: *Softplus*.....58
**Tabela 4** - Cálculo de capacitância com amostras de 20 g.....62
**Tabela 5** - Processo de aprendizagem para treinamento teste via algoritmo: *ReLu*.....64
**Tabela 6** - Processo de aprendizagem para treinamento teste via algoritmo: *Sigmoid*.....65
**Tabela 7** - Processo de aprendizagem para treinamento teste via algoritmo: *Softmax*.....67
**Tabela 8** - Processo de aprendizagem para treinamento teste via algoritmo: *Sofplus*.....68
**Tabela 9** - Processo de aprendizagem para treinamento teste via algoritmo: *Tanh*.....*70*
**Tabela 10** - Processo de aprendizagem para treinamento teste via algoritmo: *ReLu*.....*71*
**Tabela 11** - Processo de aprendizagem para treinamento teste via algoritmo: *Sigmoid*.....*73*
**Tabela 12** - Processo de aprendizagem para treinamento teste via algoritmo: *Softmax*.....*74*
**Tabela 13** - Processo de aprendizagem para treinamento teste via algoritmo: *Sofplus*.....*76*
**Tabela 14** - Processo de aprendizagem para treinamento longos via algoritmo: *Softplus*.....*77*
**Tabela 15** - Configuração das previsões de fabricação de supercapacitores: menor massa...85
**Tabela 16** - Configuração das previsões fabricação de supercapacitores: massa intermediária.....86
**Tabela 17** - Configuração das previsões de fabricação de supercapacitores: maior massa....87
**Tabela 18** - (SOE) Supercapacitor Otimizado Energeticamente.....88
**Tabela 19** - Desvio Padrão na fabricação de supercapacitores: SOE .....89

## LISTAS DE ABREVIATURAS E SIGLAS

**IA –** Inteligência Artificial

**AG** – Algoritmo Genético

**RNA** – Redes Neurais Artificiais

**ANN –** Artificial Neural Networks

***ReLu*** – Rectified Linear Unit

***Tanh*** – Tangente hiperbólica

**CNT** – Nanotubo de carbono

**CTM** – Ciência e Tecnologia de Materiais

**EDLC** – Electric Double Layer Capacitor

**PIH** – Plano Interior de Helmholtz

**PEH** – Plano Exterior de Helmholtz

***Step Function*** - Função Degrau

**ML -** *Machine Learning*

**TF -** Tensor Flow

**API** – *Aplication Programming Interface*"

**CMD** – *Prompt* de comando da máquina

**GUI** – *Graphical User Interface*

**IDE** - Integrated Development Environment

**PYPI** – Python Package Index

***NumPy*** - Numerical Python é uma biblioteca

***Scipy*** - *Python* Científico

**Spyder** - Scientific PYthon Development EnviRonment

***Matplotlib*** - Biblioteca para criar visualizações

**MLG** – Modelos Lineares Generalizados

**MSE** - Erro Quadrado Médio *(Root Mean Squared Error)*

**RMSE** - Raiz Quadrada do Erro Médio *(Root Mean Squared Error)*

**MAE** – Erro Médio Absoluto (*Mean Absolute Error)*

**$R^2$** - Erro quadrático

**CV** – Gráfico de (corrente x potencial)

**Atm** – Pressão exercida pela atmosfera

**RAM -** Random Access Memory

**SOE** - Supercapacitor Otimizado Energeticamente

**Δm1 –** Categoria “Menor Massa”

**Δm2 –** Categoria “Massa Intermediária”

**Δm3 –** Categoria “Maior Massa”

**CPU -** Central Processing Unit

**1D** – Uma dimensão

**2D** – Duas dimensões

**3D** – Três dimensões

## LISTA DE SÍMBOLOS

$Q$ - carga

$I$ - corrente elétrica

$C$ - capacitância

$V$ - voltagem

$\varepsilon$ - constante de permissividade

$\int$ - Integral simples

$\oint$ - Integral num ciclo fechado

A - Área de secção da placa

l - Comprimento

d – distância

σ - Sigma

E - Campo Elétrico

F - Faraday

g - grama

k - constante dielétrica

En - Energia armazenada

W - Trabalho

dV - Infinitesimal da tensão

dW - Infinitesimal do trabalho

dQ - Infinitesimal da carga

dl - Infinitesimal do comprimento;

dI - Infinitesimal da corrente elétrica

ε0 - permissividade do vácuo (vácuo ou espaço livre);

εr - constante dielétrica relativa do material utilizado;

ΔV - diferença de potencial;

$f(x)$ - função genérica

$\phi$ - phi (letra grega)

$e^x$ - função neperiana

$\sum$ - Função somatória

$ln$ - logaritmo natural

$tanh$ - função tangente hiperbólica

$Softplus$ - função softplus

$\infty$ - infinito

$+\infty$ - mais infinito (apenas positivo)

$\Delta$m - intervalo de massas

$\Delta$p - intervalo de pressões

$\Delta$scan - intervalo de scanrate

kgf/cm² - kilograma-força por centímetros quadrado

**SUMÁRIO**

**Capítulo 1 – INTRODUÇÃO .....18**
1.1 CONTEXTUALIZAÇÃO .....18
1.2 OBJETIVO GERAL .....21
1.2.1 OBJETIVO ESPECÍFICO .....21
1.3 MOTIVAÇÃO E JUSTIFICATIVA .....21
1.4 HIPÓTESES DO TRABALHO .....22
1.5 POR QUE ESSE TRABALHO É RELEVANTE? .....22
**Capítulo 2 – REVISÃO TEÓRICA .....23**
2.1 INTELIGÊNCIA ARTIFICIAL .....23
2.2 CAPACITORES .....23
2.3 SUPERCAPACITORES OU CAPACITORES ELETROQUÍMICOS .....27
2.4 O GRAFENO .....30
2.5 REDES NEURAIS ARTIFICIAIS (RNA) .....30
2.6 MODELO PERCEPTRON E SIMULAÇÕES COMPUTACIONAIS .....33
2.7 ALGORITMOS E LINGUAGEM EM PYTHON.....34
2.7.1 APLICAÇÕES DE PYTHON NO CENÁRIO MUNDIAL .....34
2.7.2 MACHINE LEARNING.....35
2.7.3 *TENSORFLOW* .....36
2.7.4 DEEP LEARNING .....37
2.7.5 AMBIENTES DE PROGRAMAÇÃO EM PYTHON .....37
**Capítulo 3 – MÉTODOS E MATERIAIS .....40**
3.1 – O MÉTODO E A ARQUITETURA DA REDE NEURAL ARTIFICIAL .....40
3.2 – CÁLCULOS DE CAPACITÂNCIA.....42
3.3 – FUNÇÕES DE ATIVAÇÃO.....43
3.4 – EXPERIMENTOS COM FUNÇÕES DE ATIVAÇÃO.....45
3.4.1 – RELU FUNCTION ACTIVATION .....45
3.4.2 – SIGMOID FUNCTION ACTIVATION .....46
3.4.3 – SOFTMAX FUNCTION ACTIVATION .....47
3.4.4 – SOFTPLUS FUNCTION ACTIVATION .....47
3.4.5 – TANH FUNCTION ACTIVATION .....48
3.5 – MÉTRICAS DE AVALIAÇÃO .....49
3.6 – SEPARAÇÃO DE TREINOS E TESTES E NORMALIZAÇÃO DE DADOS.....51

3.7 – PREDIÇÃO DOS MODELOS DE (IA) .....52
**Capítulo 4 – RESULTADOS E DISCUSSÃO .....53**
4.1 – PROCESSO EXPERIMENTAL .....53
4.2 – CONJUNTO DE DADOS E NORMALIZAÇÃO DE DADOS .....54
4.3 – CONFIGURAÇÃO EXPERIMENTAL (TESTES DA PRIMEIRA ETAPA) .....54
4.3.1 – RELU ACTIVATION FUNCTION .....55
4.3.2 – SIGMOID ACTIVATION FUNCTION .....56
4.3.3 – SOFTPLUS ACTIVATION FUNCTION .....57
4.4 – COMPARAÇÃO COM O ESTADO DA ARTE .....59
4.5 – ANÁLISES DE CURVAS E CÁLCULOS DE CAPACITÂNCIAS .....61
4.6 – RESULTADO DO MÉTODO DE RNA's.....62
4.7 – RESULTADOS DOS TESTES DAS FUNÇÕES DE ATIVAÇÃO (1ªETAPA).....64
4.7.1 – CONFIGURAÇÃO A - ACTIVATION FUNCTION Relu .....64
4.7.2 – CONFIGURAÇÃO B - ACTIVATION FUNCTION Sigmoid .....65
4.7.3 – CONFIGURAÇÃO C - ACTIVATION FUNCTION Softmax .....67
4.7.4 – CONFIGURAÇÃO D - ACTIVATION FUNCTION Softplus .....68
4.7.5 – CONFIGURAÇÃO E - ACTIVATION FUNCTION tanh .....70
4.8 – TESTES FINAIS (REDES NEURAIS ARTIFICIAIS) *Machine Learning*.....71
4.8.1 – CONFIGURAÇÃO F (3D) - ACTIVATION FUNCTION "Relu" .....71
4.8.2 – CONFIGURAÇÃO G (3D) - ACTIVATION FUNCTION "Sigmoid" …….....73
4.8.3 – CONFIGURAÇÃO H (3D) - Function Activation "Softmax" .....74
4.8.4 – CONFIGURAÇÃO I (3D) - Function Activation "Softplus" .....76
4.9 – TESTES COM EXECUÇÃO EM LONGO PERÍODO (PRIMEIRA ETAPA) .....77
4.10 – SEGUNDA ETAPA (TREINAMENTOS E TESTES USANDO O COLAB).....78
4.11 – PREDIÇÃO E MAXIMIZAÇÃO.....85
4.12 – DESVIO PADRÃO E VARIÂNCIA DA MELHOR EFICIÊNCIA ENERGÉTICA...88
4.13 – DISCUSSÃO DOS RESULTADOS .....89
4.13.1 – DISCUSSÃO DOS RESULTADOS DA PRIMEIRA ETAPA (*Spyder*) .....90
4.13.2 – DISCUSSÃO DOS RESULTADOS DA SEGUNDA ETAPA (*Colab*) .....91
**5 – CONCLUSÕES .....93**
**6 – REFERÊNCIAS .....95**

# *Capítulo 1*

## 1 INTRODUÇÃO

### 1.1 CONTEXTUALIZAÇÃO

A evolução tecnológica na era da cultura digital do final do século XX e início do século XXI foi incisiva na instrumentalização e produção de novas máquinas inteligentes e dispositivos a partir de novos materiais [1]. Com isso, diversas biodiversidades da terra foram sistematizadas e desvendados alguns dos mistérios do universo, e, isto se deve ao fato essencialmente pelas descobertas dos novos materiais na natureza, principalmente pela construção de novos dispositivos eletrônicos, capazes de fazer medições e observações cada vez mais precisas.

Neste sentido, tendo em vista o extraordinário avanço tecnológico da era da informação,

rapidamente simples vídeos e artigos podem se difundir mundialmente com a implantação de sistemas informatizados inteligentes que obedecem aos algoritmos [2]. Então, áreas promissoras como as IA's (Inteligências Artificiais), nanotecnologias e a eletrônica [3, 4, 5, 6] em geral sempre chamam muita atenção dos pesquisadores e da sociedade em geral, por isso recebem assim um papel de destaque na ciência [1, 2, 7]. Suas aplicabilidades estão no contexto de produção de novos dispositivos eletrônicos a partir de materiais, comunicação e design, compilação de grandes quantidades de dados, finanças, economia, previsões climáticas, etc.

Deste modo, o problema é que hoje, somente grandes empresas como *Google* e *Facebook* detém este tipo de conhecimento mais refinado, principalmente porque exigem um grande poder computacional e os supercomputadores mais potentes. [2, 5, 6, 8, 9, 10]. Neste contexto pandêmico atual, causado pelo COVID-19-SARS-CoV-2, e devido à alta taxa de fluxo de dados e informações em rede ainda maior, acentuou-se a demanda e a procura por novos dispositivos modernos ligados em baterias de cargas. De modo que essa geração de novos jovens nativos digitais e com o acesso a informatização, foi necessário um grande aporte de sistemas elétricos e eletrônicos, bem como dispositivos mais modernos e cada vez mais eficientes energeticamente [9, 10]. Portanto, para acompanhar essa realidade foi necessário a criação de sistemas mais inteligentes e capazes de resolver problemas simples e de forma automatizada. Estamos falando das novas gerações de redes inteligentes capazes de automatizar processos que eventualmente seriam humanos, comumente chamada pelo termo

"*Data Science*" e "*Big Data*", que é um sistema capaz de analisar e obter informações a partir de grandes conjuntos de dados, algo que especificamente foi feito neste trabalho.

As aplicações diretas são no ramo da eletrônica, física de materiais, e no desenvolvimento em grande escala de protótipos, dispositivos eletrônicos, carros elétricos e etc. Ficando ainda mais evidente que hoje em dia, as máquinas e os eletrônicos em geral, cada vez mais utilizam em sua configuração protótipos armazenadores de energia, os capacitores e supercapacitores [2, 3, 4, 5, 6, 7, 8]. Portanto essa área e temática tem aplicabilidades em infinitos universos, pois existem diferentes tipos de capacitores capazes de armazenarem e fornecerem uma elevada densidade de energia num curto intervalo de tempo, os supercapacitores. Por exemplo, nos estudos de Yu Bin Tan e Jong-Min Lee (2013) [3], uma alta densidade de energia para um carro que pode ser alimentado por um supercapacitor, significaria que o tempo de carregamento do carro seria muito mais curto do que para uma bateria convencional. A densidade de energia é um parâmetro importante para supercapacitores, pois determina a rapidez com que a energia pode ser descarregada e carregada [11, 12, 13]. Sendo assim, o carro também seria capaz de acelerar em velocidades mais rápidas em comparação com a energia fornecida pela bateria [14].

Este trabalho teve como objetivo criar algoritmos baseados no conceito de *machine learning*, para o uso da técnica de RNA e maximizar a eficiência energética de supercapacitores. Além de calcular as capacitâncias específicas das amostras, visamos desenvolver um código em *Python*, usando conjunto de dados e medidas de capacitância de supercapacitores de um trabalho anterior que consistiu na fabricação e medições de potencial, densidades de energia, corrente elétrica dos dispositivos supercapacitores de grafeno [15].

Deste modo, introduzimos que este trabalho consistiu em desenvolver algoritmos, simular computacionalmente a fabricação de supercapacitores, prever capacitâncias otimizadas, calcular capacitâncias específicas (densidade de energia) e testar experimentalmente as principais funções de ativação de acordo com a literatura usando técnicas de (ML) "*machine learning*", que no português significa (aprendizado de máquina), em especial a técnica de RNA's "Redes Neurais Artificiais".

Hoje em dia, textos em código são representados como um vetor, e, em cada posição do vetor tem uma palavra. As palavras são representadas por vetores no espaço, e o texto é uma lista ordenada destes vetores. Existem diversos trabalhos que consideram estas novas configurações, sendo a maior parte delas infinitamente capaz de resolver qualquer problema computacional. E a maioria delas, estão usando redes neurais artificiais recorrentes. Porém,

elas são muito difíceis de serem treinadas, pois tem dependências longas, que é o caso de textos que geram muitos problemas no gradiente [16].

Com base na literatura da área, a comunidade científica tem desenvolvido trabalhos pelo mundo todo acerca da maximização e eficiência de supercapacitores usando o método de RNA [3, 4, 5, 6, 7, 8, 17, 18]. Os procedimentos metodológicos para a realização deste trabalho e os métodos de abordagem na problematização foi o "estado da arte", de forma quantitativa e teórica. Portanto, este trabalho ficou pautada em estudar sistematicamente as melhores configurações e parâmetros de supercapacitores, especialmente os baseados em eletrodos de grafeno, usando o método de RNA (Redes Neurais Artificiais) como a principal técnica de simulação computacional [19, 20, 21, 22].

O trabalho investigou como prever as melhores configurações e parâmetros físicos, a fim de atingir uma configuração eficiente na produção de supercapacitores de grafeno otimizados. Deste modo, esse trabalho subdividiu-se em duas (02) etapas de treinamentos e testes:

(i) Na primeira etapa foi utilizado um computador comum com 8GB de RAM e também uma máquina com especificações desconhecidas localizada na universidade (UFTM) com acesso remoto.

(ii) Na segunda etapa de trabalho foi usado o computador do *Google* com 12 GB de RAM e condições muito melhores, no ambiente do *Google Colab*. O objetivo em especial foi atingir uma melhor eficiência de armazenamento de energia nas duas situações, ou seja, maximizar a energia e potencializar a corrente elétrica a ser liberada pelo supercapacitor, em mais ciclos de carga.

Dessa maneira, métodos alternativos e essenciais foram usados neste trabalho, como: linguagem de programação em *Python, Colab do Google, TensorFlow, Keras, Numpy, Pandas, Scikit-learn, Matplotlib* e o *Pycaret* [128 -134] realização de cálculos de integrais, físicos e matemáticos. Além de levantamentos estatísticos das possíveis aplicações dentro do âmbito da comodidade da sociedade, aporte econômico e a era digital.

Todas as etapas do trabalho foram previamente idealizadas, possibilitando seu correto desenvolvimento e execução em tempo hábil. Os resultados nos mostraram que obtivemos a máxima capacitância perante as amostras e configurações de fabricação propostas. Além disso, mostramos com precisão os erros e as métricas em que usamos em nossos algoritmos, indicando a eficácia do método proposto para regressão de um valor maximizado.

## 1.2 - Objetivo Geral

O objetivo geral é utilizar os avanços recentes da IA usando redes neurais artificiais, e explorar a ideia de otimizar parâmetros para a construção de supercapacitores de grafeno. Portanto é objetivo deste trabalho, identificar e caracterizar perspectivas de programação e simular condições e parâmetros para atingirmos as melhores configurações testando funções de ativação do algoritmo. Especificamente procura-se estimar um valor numérico e não uma classificação de imagens, ou seja, é um trabalho que visa resolver um problema de regressão. Alcançando assim uma melhor eficiência nos resultados e consequentemente a maximização do armazenamento de energia usando método de Redes Neurais Artificiais (RNAs), e simulação computacional.

### 1.2.1 - Objetivos específicos

- Estudar os parâmetros físicos e tipos de capacitores, bem como o método de RNA;
- Prever a eficiência de armazenamento de supercapacitores, ou seja, a capacitância;
- Caracterizar as configurações das redes neurais e o *machine learning*;
- Contribuir para a formação de recursos na área de Ciência e Tecnologia de Materiais.

## 1.3 – Motivação e Justificativa

Todos os estudos sistemáticos de *machine learning* e simulações computacionais de novos materiais são de enorme interesse para entendermos todos os processos e otimizá-los. Deste modo, há uma necessidade de estudar as (IA’s) novas formas de refinamento e otimização dos dados experimentais usando a inteligência artificial e as redes neurais.

O trabalho se torna ainda mais relevante porque estamos falamos de um contexto em que a necessidade de baterias mais eficientes energeticamente é muito real. A energia solar, eólica e até mesmo a biomassa dependem de fatores externos para sua disponibilidade e de fatores dificilmente previstos. O vento, o sol e a matéria prima da biomassa possuem seu próprio tempo. Ou seja, é necessário que o armazenamento contorne tal sazonalidade para dispormos da energia renovável nas mais diversas situações. Como os cientistas e empresas têm interesse no potencial social e econômico das baterias, elas vêm avançando muito rapidamente.

Sendo assim, é possível desenvolver por programação e ciências da computação. E por conta de possíveis aplicações diretas na sociedade e por trazerem comodidade e conforto ao ser humano, há um grande interesse por essa área de pesquisa. Portanto, a justificativa para

sustentar o presente trabalho, consiste na altíssima importância que os estudos possuem para a área de materiais, ciências da computação e toda sociedade em geral.

Deste modo, este trabalho enfatiza estratégias de como contribuir com a solução do problema no âmbito técnico na forma de como obter supercapacitores com maior eficiência. Portanto, tendo em vista os grandes avanços tecnológicos deste tempo, trabalhar com redes neurais e inteligência artificial são de grande valia na área de materiais e para a comunidade científica.

### 1.4 – Hipóteses do trabalho

- Os supercapacitores têm excelente densidade de energia, taxa de carregamento rápida e ciclo de vida estendido e suas propriedades podem ser otimizadas com técnicas de RNA;
- A alta densidade de energia de um supercapacitor, representa que ele possui alto poder de armazenamento de energia;
- É possível otimizar e produzir supercapacitores com uma maximização da energia controlando parâmetros físicos, usando RNA [4].

### 1.5 – Por que esse trabalho é relevante?

Porque esse trabalho tem como objetivo identificar e simular os métodos de teste ideais para o desempenho do eletrodo para produção de supercapacitores com mais eficiência. Também porque esse trabalho enfatiza a necessidade crescente de padronizar os testes para garantir que os resultados dos testes sejam tão relevantes quanto possível para as aplicações da vida real.

A relevância deste trabalho também leva em conta as últimas décadas do século XX e início do século XXI, pois com o progresso da construção das válvulas eletrônicas e principalmente pelas descobertas de novos materiais semicondutores, surgiram então outros aparelhos, como *smartphones* e os "*wearable devices*". Todos esses aparelhos necessitam armazenar energia e os supercapacitores surgem como uma alternativa a esse armazenamento.

# *Capítulo 2*

## 2 – REVISÃO TEÓRICA

A revisão teórica deste trabalho de forma geral, abrangeu a revisão de artigos científicos, participação em palestras científicas, livros e *e-books*, sites de pesquisas científicas, vídeos na internet e discussões de outros autores falando sobre o tema em questão. Para ser possível a realização deste trabalho, foram feitos levantamentos bibliográficos acerca do tema, e um estudo mais profundo do estado da arte. Portanto, a fundamentação e referencial teórico deste trabalho divide-se em seções e subseções envolvendo algoritmos, programação, supercapacitores, inteligência artificial, *machine learning* e redes neurais.

### 2.1 - INTELIGÊNCIA ARTIFICIAL

Uma máquina tem como principal função seguir regras ou comandos oferecidos pelo usuário [11]. Deste modo, uma sequência de algoritmos programados num sistema de aprendizado de qualquer máquina, tenta simular capacidades humanas ligadas à inteligência [22]. Por isso, o raciocínio, a percepção de ambiente e a habilidade de análise para a tomada de decisão, pode ser adquirida por uma máquina por meio da inteligência artificial [11, 15, 18, 23, 24, 25].

A ideia é desenvolver gradativamente um amplo conhecimento acerca das mais variadas formas de inteligências artificiais, as quais são utilizadas na área da computação. Sendo assim, pode-se dizer, de uma maneira sintetizada, que o conceito de IA (Inteligência Artificial) está relacionado diretamente à capacidade de soluções tecnológicas. Isto é, realizarem atividades de um modo mais eficaz e eficiente. A Inteligência Artificial (IA) também é um campo da ciência, cujo propósito é estudar, desenvolver e empregar máquinas para realizarem atividades humanas de maneira autônoma [26, 27, 28, 29].

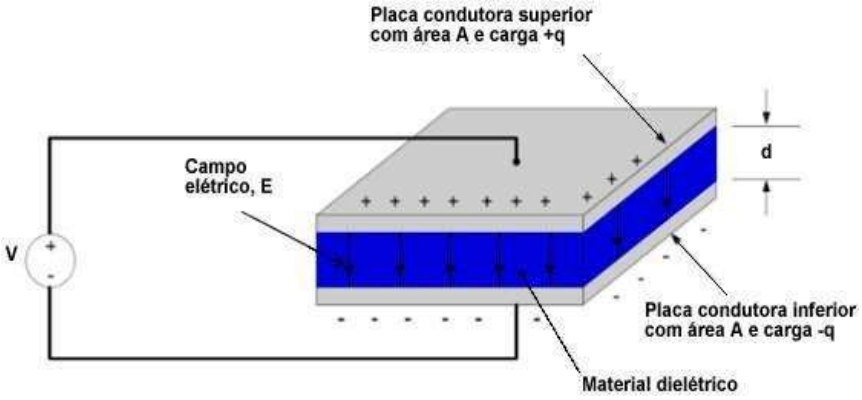
### 2.2 - CAPACITORES

Capacitores ou condensadores, como também são chamados, têm como sua principal característica armazenar cargas elétricas e, consequentemente, como tais cargas geram um campo elétrico, o trabalho para que este campo seja produzido é interpretado como a energia por ele armazenada [31, 32, 33, 34]. O modo mais simples de visualizar um capacitor é

considerá-lo como composto por 02 placas condutoras paralelas separadas por uma distância *d*.

Este espaço entre as placas é preenchido pelo material dielétrico, que pode ser o ar, o vácuo ou algum tipo de material isolante que impeça que ocorra movimentação de cargas entre as placas condutoras, um esquema representativo desse capacitor é apresentado na Figura 1 [32, 33 34].

Figura 1 - Esquema de montagem de um capacitor de placas paralelas.




Fonte: Modelo adaptado pelo autor [7].

Para carregar eletricamente o capacitor é necessário que se aplique uma diferença de potencial entre as placas, onde uma placa terá polaridade positiva (carregada com cargas positivas) e a outra polaridade negativa (carregada com cargas negativas), esta polaridade das placas se dá pela movimentação dos elétrons de uma placa para a outra [35, 36, 37, 38, 39]. Ao se conectar o capacitor a uma fonte de energia externa, a placa condutora que recebe os elétrons fica carregada negativamente e a placa a qual os elétrons saíram carregada positivamente. Com as placas carregadas surge um campo elétrico (E) perpendicular a elas, no sentido da placa positiva para a placa negativa, demandando um trabalho e assim armazenando energia [40].

A capacidade de armazenar energia de um capacitor, que recebe o nome de capacitância (C), é diretamente proporcional a quantidade de carga armazenada em função do potencial aplicado como apresentado na Equação 1, e pode também ser descrita pela Equação 2 para um capacitor de placas paralelas, em função da área das placas (A), permissividade elétrica do meio (ε) e a distância (d) entre as placas [35–38, 41].

$$C = \frac{Q}{V}$$

$$Q = CV \qquad \textit{Equação 1}$$

Sabendo que através da Lei de Gauss e da diferença de potencial entre as placas temos:

$$Q = \varepsilon \oint E\, dA = \varepsilon EA \qquad \textit{Equação 2}$$

$$V = \int_{+}^{-} E\, dl = Ed$$

Obtemos:

$$C = \frac{\varepsilon A}{d} \qquad \textit{Equação 3}$$

Assim, quanto maior a carga (Q) armazenada para dado potencial (V) maior será a capacitância deste dispositivo. Esta capacidade de armazenamento ainda está diretamente ligada ao quão o material entre as placas dificulta a movimentação das cargas entre elas uma vez que estas estejam carregadas (o quão dielétrico é o material) (Equação 3) [36, 38].

$$C = k\, C_o \qquad \textit{Equação 4}$$

Esta capacidade é descrita como $C_0$ sendo a capacitância do vácuo, k a constante dielétrica do meio. Esta constante k está diretamente ligada a polarizabilidade do material, ou seja, quanto maior for esta constante, mais fácil e mais polarizado o material ficará para uma mesma densidade de carga a ele aplicada, pois as cargas polarizadas no material criam um campo elétrico que se contrapõe ao campo gerado pelas cargas das placas condutoras, fazendo assim que o campo resultante seja menor, campo este entre duas placas paralelas dado pela Equação 4 [35, 36, 38].

$$E = \frac{V}{d} \qquad \textit{Equação 5}$$

Quando seu módulo diminui, consequentemente a diferença de potencial entre as placas diminui, pois à distância *d* entre as placas permanece inalterada, e assim o valor da capacitância aumenta. Para se determinar a energia armazenada (En) nos capacitores considera-se que a carga está uniformemente distribuída, assim essa energia é dada pelo trabalho (W) realizado para carregá-lo de 0 a uma carga Q, através do trabalho (dW) para mover uma parcela infinitesimal da carga dQ e da Equação 1, chegamos na forma simplificada do cálculo, que é dado pela Equação 5 [36]. Assim, para uma mesma faixa de potencial aplicado quanto maior a capacitância, maior será a energia armazenada no dispositivo.

$$dW = V\,dQ = \frac{Q}{C}\,dQ$$

$$W = \int dW \frac{1}{C}\int_0^Q Q'dQ' = \frac{Q^2}{2C}$$

$$En = \frac{Q^2}{2C} = \frac{1}{2}CV^2 \qquad \textit{Equação 6}$$

Figura 2 - Capacitores: dispositivo capaz de acumular cargas elétricas.

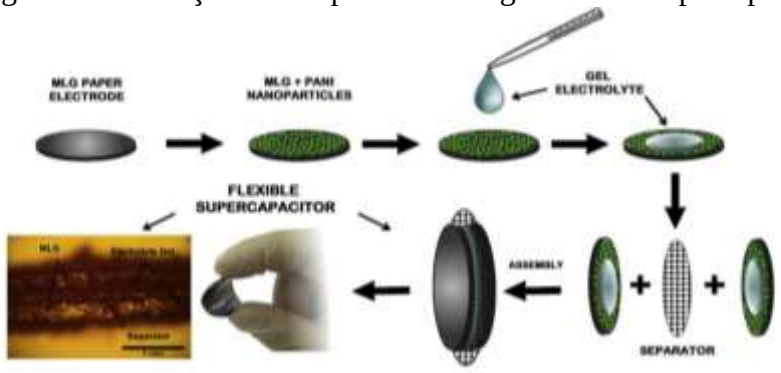
Fonte: InfoEscola, Acesso em: 03 de Agosto de 2021, Link disponível em: ***https://www.infoescola.com/eletricidade/associacao-de-capacitores/***

## 2.3 - SUPERCAPACITORES E CAPACITORES ELETROQUÍMICOS

Os supercapacitores são dispositivos capazes de fornecer e armazenar uma elevada densidade de potência ou densidade de energia em um intervalo de tempo curto [37, 38, 39]. Estes dispositivos apresentam capacitâncias elevadas, quando comparado aos capacitores comerciais comuns. Alguns supercapacitores podem armazenar de 10 a 100 vezes mais energia, quando em sua produção, alguns parâmetros físicos, tais como a massa, volume e pressão são monitorados e controlados constantemente. Outra vantagem, é que estes dispositivos aguentam muito mais ciclos de carga e descarga, além de carregarem muito mais rapidamente se comparado a baterias.

Os capacitores eletroquímicos são formados por dois eletrodos metálicos, ou de materiais condutores e um eletrólito, em estado líquido ou sólido. O armazenamento de energia se dá principalmente na dupla camada, que é a interface entre o eletrodo e o eletrólito (*electric double layer capacitor* – EDLC), por meio de forças eletrostáticas sem mudanças de fase [41], diferente das baterias, que possuem como princípio básico de armazenamento de energia reações faradaicas de oxidação e redução. Onde devido ao fluxo de elétrons ocorre a mudança de fase dos materiais envolvidos, o ânodo sofre oxidação, o que faz com que seu material, em alguns tipos de baterias, passe da fase sólida para aquosa, e o catodo sofre redução, o que gera uma mudança de fase de aquoso para sólido, um exemplo é a pilha de Daniell) [40, 41, 42].

Figura 3 - Ilustração das etapas de montagem de um supercapacitor.

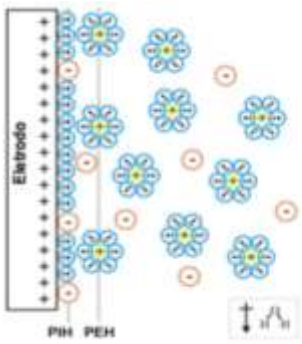



Fonte: [13]

Para que seja armazenada energia na dupla camada, os eletrodos têm de ser submetidos a uma diferença de potencial, onde um dos eletrodos estará carregado positivamente (com falta de elétrons) e o outro negativamente (com excesso de elétrons). Assim na interface do eletrodo com o eletrólito acontece a atração entre carga presente no eletrodo e a carga de polaridade oposta presente no eletrólito, onde na camada mais interna

temos íons especificamente adsorvidos (chamado de Plano Interior de Helmholtz - PIH) e na camada mais externa íons solvatados (chamado de Plano Exterior de Helmholtz - PEH), a região fora do plano exterior de Helmholtz recebe o nome de plano difuso, de acordo com o modelo de Bockris da dupla camada [40, 41 42, 46, 48], região onde a força de atração eletrostática começa a ser insuficiente para atrair os íons livres do eletrólito, como exemplificado na Figura 4.

Figura 4 - Representação dos íons adsorvidos na interface do eletrodo.

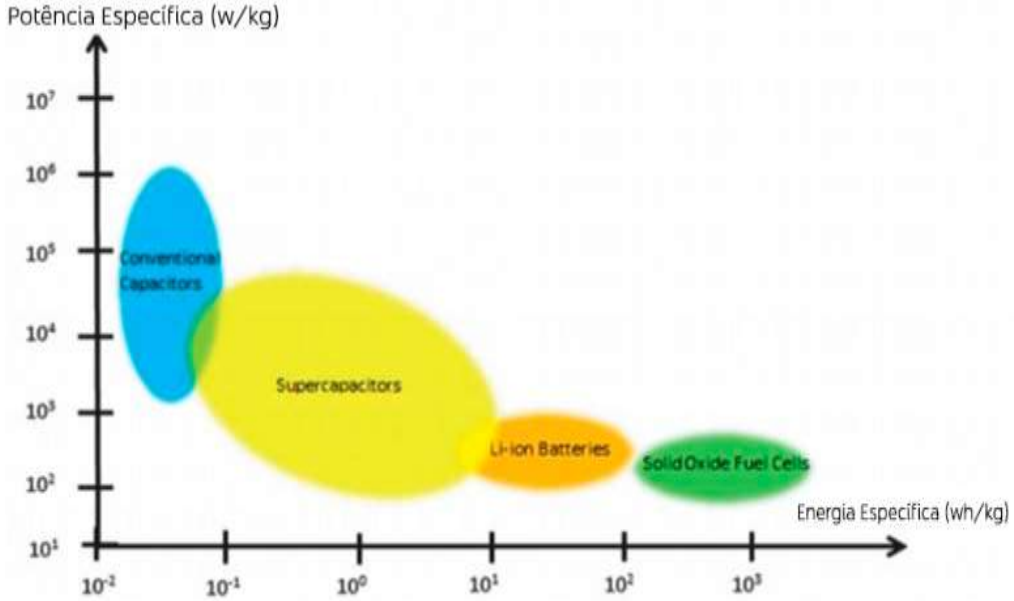



Fonte: [40]

Como a interface eletrodo-eletrólito é o local onde a carga se armazena, a área superficial é um parâmetro muito importante para determinar a capacidade de armazenamento de carga, por isso alótropos do carbono bidimensionais se tornam importantes materiais a serem utilizados nos eletrodos, como o grafeno, por possuir área superficial teórica de 2675 $m^2/g$ [43].

Além do armazenamento de carga da dupla camada, outra forma tem sido empregada em supercapacitores para aumentar seu desempenho, que é o emprego de óxidos metálicos ou polímeros condutores, que por meio de reações faradaicas rápidas reversíveis na superfície do eletrodo (reações de oxidação e redução – transferência de carga) faz com que energia seja armazenada (pseudocapacitância) [43, 44, 45, 46].

Figura 5 - Gráfico de Ragone.




Fonte: Yu Bin Tan and Jong-Min Lee cited in J. Mater. Chem. A, 2013 [47].

As propriedades dos supercapacitores são particularmente adaptadas para cada aplicação. Isto é, cada aplicação requer diferentes pulsos de energia durante curtos ou longos períodos, como segundos, milissegundos ou frações muito pequenas de segundo. Tais dispositivos são muito importantes e têm ganhado relevância para diversos usos, como telefones celulares, computadores portáteis, câmeras digitais, terminais de dados, sistemas automotivos de partida e veículos híbridos etc. Cada uma destas aplicações exige diferentes otimizações em tais dispositivos [43, 44, 47, 49].

Para a partida de um automóvel é necessária uma alta capacidade de fornecer corrente por um período de alguns segundos, enquanto para aplicações em dispositivos eletrônicos em geral eles devem fornecer pulsos de potência de duração de 0,5 a 5 ms [43, 45, 47, 48, 49]. As características dos supercapacitores estão entre aquelas dos capacitores convencionais e das baterias. Estes dispositivos fornecem capacitâncias da ordem de dezenas, podendo atingir até mesmo centenas de *Farad*, ou seja, várias ordens de grandeza mais elevadas que os capacitores convencionais [50, 51, 52].

Quando comparados às baterias, estes capacitores são capazes de fornecer altas densidades de potência (500 - 10000 W/Kg) e elevados ciclos de carga e descarga (> 100.000), embora ainda armazenem menos densidade de energia. Com a crescente demanda por energia, juntamente com a escassez de energia e os altos preços no mundo globalizado de hoje, tem havido um impulso revigorado para trabalhar com dispositivos avançados de armazenamento e gerenciamento de energia. Neste cenário, os supercapacitores têm atraído a

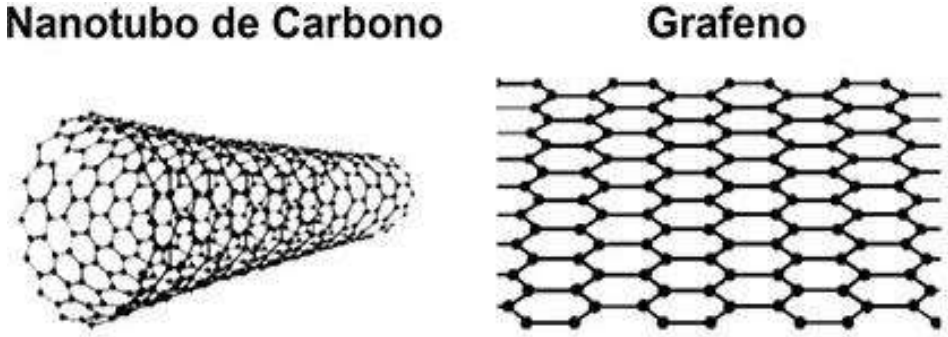
atenção especial devido à sua excelente densidade de energia, taxa de carregamento rápida e ciclo de vida estendido [52, 53, 54, 55].

## 2.4 – O GRAFENO

O grafeno atraiu amplo interesse no campo de trabalho de supercapacitores devido à sua estrutura 2D que lhe confere propriedades excepcionais, [53, 54, 55] como condutividade elétrica, propriedades mecânicas superiores, bem como uma área de superfície extensa melhor do que a dos nanotubos de carbono (CNTs), como pode ser observado na figura 6.

Além disso, ao contrário de outros materiais de carbono, o grafeno é particularmente ideal para aplicações de supercapacitores, pois sua área de superfície não varia com a distribuição do tamanho dos poros e concede acesso de eletrólito a ambas as superfícies [55]. Com isso, a área efetiva do eletrodo é muito maior que a área aparente.

Figura 6 – Diagrama sobre como são as estruturas do Nanotubo de Carbono e Grafeno.




Fonte: J. P. C. Trigueiro, UFMG, 2014) [11]

## 2.5 - REDES NEURAIS ARTIFICIAIS (RNA)

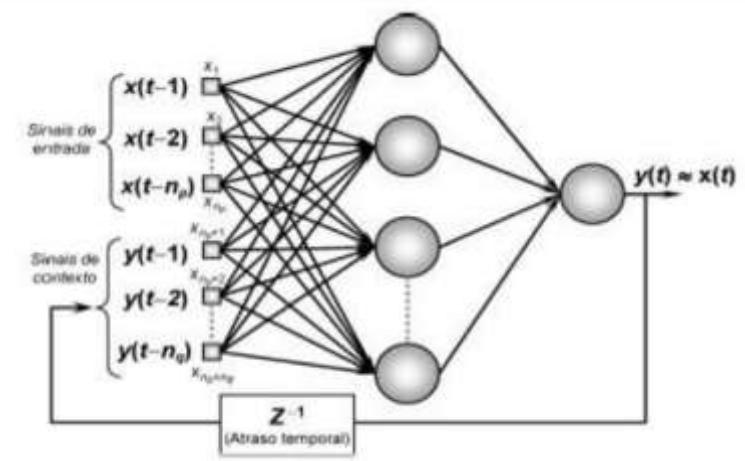
No ritmo atual dos trabalhos sobre RNAs (Redes Neurais Artificiais), segundo a literatura, [3, 7, 10, 33, 35, 37, 40, 46, 48] podemos definir brevemente, sendo sistemas paralelos distribuídos compostos por unidades de processamento simples (nodos) que calculam determinadas funções matemáticas (normalmente não-lineares). Tais unidades são dispostas em uma ou mais camadas e interligadas por um número de conexões, geralmente unidirecionais. Na maioria dos modelos estas conexões estão associadas a pesos, os quais armazenam o conhecimento representado no modelo e servem para ponderar a entrada recebida por cada neurônio da rede. O funcionamento destas redes é inspirado em uma estrutura física concebida pela natureza: o cérebro humano [56].

Nesta e o método de RNA’s, foi utilizado para estimar as melhores condições e parâmetros físicos na produção de supercapacitores, realizando diversos testes, alterando a

quantidade de neurônios, épocas e diversificando as entradas de novas variáveis. Com objetivo de satisfazer a crescente demanda por novos meios de produção e estocagem de energia, mais eficientes e com custo aceitável para aplicações em grande escala. Por isso, o desempenho de supercapacitores deve ser melhorado e novas soluções devem ser propostas e estudadas para o aperfeiçoamento de eletrodos e eletrólitos, com propriedades superiores às atuais.

Deste modo, as redes neurais artificiais surgem como um método para solucionar problemas através da simulação do cérebro humano, inclusive em seu comportamento, ou seja, aprendendo, errando e fazendo descobertas. Sendo assim, foi usado as redes neurais como a principal técnica de simulação deste trabalho.

Figura 7 – Esquema Redes neurais artificiais.




Fonte: A.P. Braga [2]

Nosso sistema nervoso, a nível celular, é composto por neurônios e suas conexões chamadas sinapses [56, 57, 58]. Basicamente um neurônio é uma estrutura que a partir de reações bioquímicas que o estimular geram um potencial elétrico para ser transmitido para suas conexões, de forma a criar padrões de conexão entre os mesmos para ativar algum processo de outro subsistema ou tecido ou simplesmente guardar padrões de sinapses no que convencionalmente entendemos como nossa memória.

A origem do impulso nervoso que se propaga através do neurônio é elétrica. O impulso acontece devido a alterações nas cargas elétricas das superfícies externa e interna da membrana celular. Quando um estímulo químico, mecânico ou elétrico chega ao neurônio, pode ocorrer a alteração da permeabilidade da membrana, o que permite uma grande entrada de sódio na célula e pequena saída de potássio dela. Com isso, ocorre uma inversão das cargas

ao redor dessa membrana, que fica despolarizada, gerando um potencial de ação [67]. O estímulo que gera o impulso nervoso deve ser forte o suficiente, acima de determinado valor crítico. Dessa forma, a intensidade das sensações vai depender do número de neurônios despolarizados e da frequência de impulsos [68, 69].

Na eletrônica usamos de um princípio parecido quando temos componentes eletrônicos que se comunicam em circuitos por meio de cargas de pulso elétrico gerado, convertido e propagado de forma controlada e com diferentes intensidades a fim de desencadear algum processo ou ação [59, 60].

Na computação digital propriamente dita temos modelos que a nível de linguagem de máquina (ou próxima a ela) realizam o chaveamento traduzindo informações de código binário para pulso ou ausência de pulso elétrico sobre portas lógicas para ativação das mesmas dentro de um circuito, independentemente de sua função e robustez [52 - 59].

Sendo assim, não diferentemente de outras áreas, aqui criamos modelos computacionais apenas como uma forma de abstrair e simplificar a codificação dos mesmos, uma vez que foi criado estruturas puramente lógicas onde a partir delas teremos condições de interpretar dados de entrada e saídas de ativação, além de realizar aprendizagem de máquina por meio de reconhecimento de padrões e até mesmo desenvolver mecanismos de interação homem-máquina ou de automação [60].

Em redes neurais artificiais, uma função de ativação pode definir a saída de um determinado conjunto de entradas [3, 4, 7, 10, 33, 35, 37, 56 - 60]. Ela é responsável pela camada de ativação e suas funções de ativação são aplicadas sobre os neurônios artificiais, cada um por sua vez com seus respectivos valores, pesos, funções de ativação etc.

Existe uma série de funções de ativações que podem ser utilizadas, dependendo do tipo de conjunto de entradas. Em alguns modelos haverá o que são chamadas de camadas ocultas, que são intermediárias, aplicando funções de ativação ou funcionando como uma espécie de filtros sobre os dados processados. Uma função de ativação trabalhará como modelo lógico para interpretar conjunto de dados de entrada, com pesos aplicados sobre os mesmos e ao final do processo, assim como em um neurônio real, podemos saber se ele é ativado ou não em meio a um processo. E assim, eventualmente podemos presumir se tal função de ativação ativou melhoria na eficiência do algoritmo [59, 60, 62].

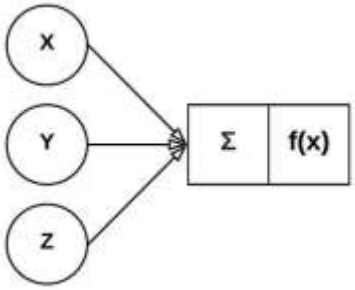
O processamento da camada é executado numa última função, chamada de função de ativação, onde o valor final poderá definir uma tomada de decisão, uma ativação (ou desativação) ou regressão [60, 61, 62, 64, 65].

Também é bastante comum se usar a *Step Function* (Função Degrau) onde é definido um valor como parâmetro e se o valor de X for igual ou maior a esse valor é feita a ativação do neurônio, caso contrário, não. Não Linear - *Sigmoid* - Probabilidade de ativação ou de classificação de acordo com a proximidade do valor de X a 0 ou a 1. Apresenta e considera os valores intermediários entre 0 e 1 de forma probabilística. Além de outras funções de processamento em treinamentos supervisionados de redes neurais artificiais como *ReLu, Sigmoid, Softmax, Softplus e Tanh.*

## 2.6 – MODELO PERCEPTRON E SIMULAÇÕES COMPUTACIONAIS

O modelo mais básico citado na literatura para fins didáticos é o modelo *perceptron*, onde temos a representação de 3 espaços alocados para entradas (X, Y e Z), note que estas 3 figuras se conectam com uma estrutura central com o símbolo sigma Σ, normalmente atribuído a uma função soma, operação aritmética bastante comum e, por fim, esta se comunica com uma última estrutura lógica onde há o símbolo de uma função de ativação *f(x)*. Em suma, estaremos sempre trabalhando com no mínimo essas três estruturas lógicas, haverá modelos muito mais complexos ou realmente diferentes em sua estrutura, mas por hora o que deve ficar bem entendido é que todo *perceptron* terá entradas, operações realizadas sobre estas e uma função que resultará na ativação ou não deste neurônio em seu contexto [70].

Figura 8 - Modelo *Perceptron*.




Fonte: A.P. Braga [2]

A partir deste modelo podemos resolver de forma computacional os chamados problemas linearmente separáveis. De forma bastante geral podemos entender tal conceito como o tipo de problema computacional onde se resulta apenas um valor a ser usado para uma tomada de decisão. Imagine que a partir desse modelo podemos criar uma pequena rede neural que pode processar dados para que se ative ou não um neurônio, podemos abstrair essa condição também com operadores lógicos, onde uma tomada de decisão resultava em 0 ou 1,

*True* ou *False*, *return* x ou *return* y etc. Tomadas de decisão onde o que importa é uma opção ou outra. Partindo para prática, agora atribuindo valores e funções para essa estrutura, podemos finalmente começar a entender de forma objetiva o processamento da mesma [70].

## 2.7 – ALGORITMOS E LINGUAGEM EM PYTHON

Algoritmos nada mais do que uma sequência de passos a serem executados por uma máquina programada a seguir exatamente as instruções feitas pelo programador. Por exemplo, ao viajarmos para um lugar que não se sabe ao certo como chegar, mas tendo em mãos a rota da estrada e uma sequência de vias e marginais a serem seguidas, a ordem de acesso às vias e o passo a passo a ser seguido importam. Neste caso, se alguma etapa foi pulada ou feita de modo incorreto, certamente ao final da viagem pode não se chegar ao local esperado. Portanto, é fundamental seguir a sequência da rota de maneira correta, para chegar no local desejado. Com os algoritmos a lógica é exatamente a mesma, tendo uma situação-problema, devemos criar um algoritmo que resolva o problema etapa por etapa, passando as instruções corretas para que a máquina possa interpretar [71].

Muitas vezes os programas criados para resolver problemas computacionais podem ou não ser executados conforme o esperado, pois, tudo vai depender de como as entradas de códigos são feitas para resolver o problema. E assim, se estiver seguindo as etapas corretas, certamente teremos saídas (resultados) que resolvam de fato o problema computacional [72].

A linguagem em *Python* é considerada a mais moderna e generalista no momento atual, pois a partir dela é possível criar configurações para a criação de qualquer sistema, independente do propósito, assim como também é possível em outras linguagens de programação. Ou seja, é possível configurar com Python e com outras linguagens, qualquer sistema operacional, *machine learning*, *data Science, web*, dispositivos móveis etc.

### 2.7.1 – APLICAÇÕES DE PYTHON NO CENÁRIO MUNDIAL

*Python* é uma das linguagens que mais teve crescimento quando comparado às demais linguagens de programação numa mesma época [72]. Isto se deve principalmente ao fato após grandes empresas espalhadas pelo mundo todo declararem publicamente que adotaram a linguagem *Python* em seus sistemas e bases de dados [72, 73, 74, 75 - 77]. Acarretando assim uma formação de comunidades gigantescas pelo mundo todo para explorar o seu potencial e suas aplicabilidades. Como exemplo de empresas que utilizam totalmente ou parcialmente *Python* em seus sistemas, podemos citar: *Google, YouTube, Instagram, Pinterest, Spotify, Dropbox, BitTorrent etc.*

A Educação básica do Brasil e em alguns países mais desenvolvidos pelo mundo, já estão adotando o ensino de programação em *Python* nos anos iniciais até o ensino médio, e nos cursos tecnólogos [77]. Boa parte desses jovens nativos digitais de hoje estão aprendendo em sua grade de disciplinas, tais como, programação e robótica, principalmente nos países norte-americanos e asiáticos [78]. No Brasil isso ainda é muito incipiente no ensino público [79]. No entanto, algumas instituições de ensino privadas e ensino superior oferecem regularmente.

Hoje em dia com a criação e atualizações constantes de novos *softwares*, novos jogos, mais do que nunca será preciso profissionais que saibam programação e *machine learning*. Podemos esperar então que futuramente *Python* crescerá ainda mais, pois novas áreas como *data science* e *machine learning*, se popularizaram ainda mais com a demanda de novas vagas no mercado em empresas que demandam cada vez mais por profissionais da área das tecnologias e área da programação. Ou seja, um perfil profissional de um programador com domínio em *Python* e que consiga trabalhar de forma natural com a linguagem que é desenvolvida o suficiente para criar qualquer solução computacional [77 - 79].

### 2.7.2 – MACHINE LEARNING

Ao longo da última década, os métodos de *machine learning* revolucionaram campos inteiros, incluindo reconhecimento de falas [4, 7, 11, 33, 52, 78], processamento de linguagem natural. Para entender de forma simples como se dá o mecanismo de *machine learning*, vamos raciocinar que temos um problema computacional a ser resolvido, onde temos várias amostras de materiais e objetos, e que precisamos classificá-los conforme cada um do seu tipo. Temos então um problema computacional a ser resolvido e precisamos usar o pensamento computacional para resolvermos este problema. Deste modo, temos que criar um *script*, que se trata de um programa usando uma sequência de códigos em linguagem de programação em *Python*. Depois o executamos num ambiente virtual, sendo que a cada execução do programa e a cada cálculo realizado pela máquina, trata-se de um aprendizado para a próxima etapa de aprendizado, sempre considerando a última resposta obtida [79 - 80, 81].

O *Machine Learning* (ML) tem se concentrado diretamente na conciliação de dados com modelos teóricos que descrevem padrões de variação produzidos [82]. Essa interação entre empirismo e teoria significa que muitos avanços no campo vieram da introdução de novos modelos computacionais, muitas vezes de complexidade crescente e que descrevem a natureza e a produção de padrões do qual observamos a partir de um conjunto de dados e

parâmetros específicos, por exemplo, parâmetros físicos e conjunto de dados na produção de supercapacitores [83, 84, 85].

Nesta revisão, consideramos a utilização de um poderoso método de análise que surgiu recentemente a: “cultura de modelagem algorítmica”, ou o que agora é comumente chamado de *machine Learning* (ML). Ao longo da última década, os métodos de ML têm revolucionado campos inteiros, incluindo reconhecimento de fala [83]. Existe literalmente uma infinidade de algoritmos que poderiam realizar essa tarefa, mas uma alternativa que chama muito a atenção são as redes neurais artificiais RNA’s. Por meio de um modelo de rede neural, podemos treinar a mesma para que reconheça quais são os valores, objetos, quais são suas características relevantes e assim treinar e testar para que com algumas configurações consiga realizar essa tarefa de forma rápida e precisa, além de que uma vez aprendida essa classificação, essa rede neural pode realizar novas classificações com base nos seus padrões aprendidos previamente [86].

Para esse processo é preciso criar uma estrutura de código onde a rede terá dados de entrada, oriundos das amostras a serem classificadas, processamento via camadas de neurônios artificiais encontrando padrões e aplicando funções matemáticas sobre os dados, para ao final do processo separar tais dados das amostras conforme o seu tipo. Neste processo existe o que é chamado de *machine Learning* supervisionado, onde literalmente temos que treinar a rede e executar os comando do programa, mostrar para a máquina o que é o resultado final correto para que ela aprenda os padrões intermediários que levaram até aquela resolução. Assim como também teremos modelos de aprendizado não supervisionado, onde a rede com base em alguns algoritmos conseguirá sozinha identificar e reconhecer os padrões necessários [82 - 87].

### 2.7.3 - TENSORFLOW: *www.tensorflow.org*

*Tensor Flow* (TF) é uma biblioteca de código aberto para *machine learning* aplicável a uma ampla variedade de tarefas. É um sistema para criação e treinamento de redes neurais para detectar e decifrar padrões e correlações, análogo à forma como humanos aprendem e raciocinam. Neste trabalho o TF será utilizado através da “*Aplication Programming Interface*” (API) *Keras*. Essa API permite uma interface otimizada para as aplicações utilizando RNAs [88].

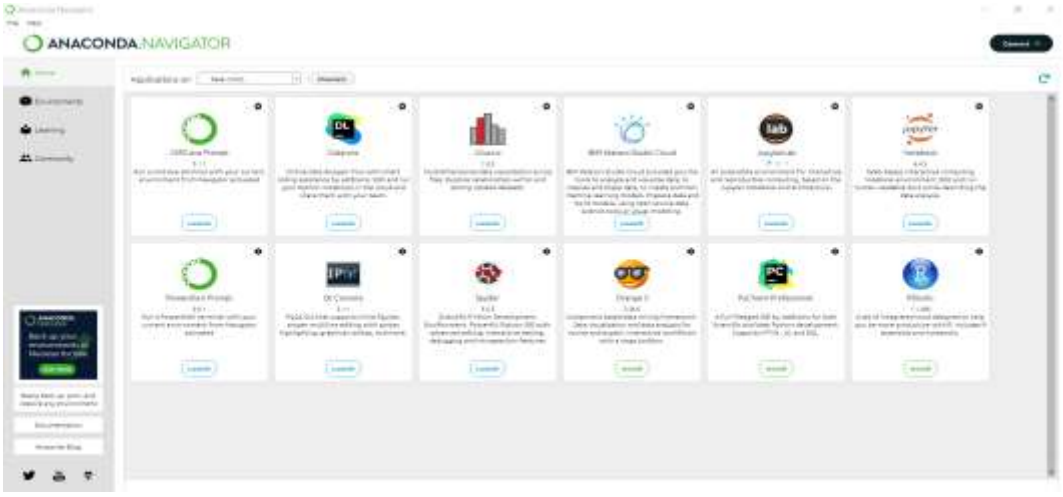
### 2.7.4 - DEEP LEARNING

*Deep Learning*, ou aprendizagem profunda, é um tipo de machine learning em meio a neurônios artificiais usando RNA (Redes Neurais Artificiais) e suas conexões estão separadas virtualmente por camadas de processamento, dependendo da aplicação da rede essa pode ter apenas uma camada, mais de uma camada ou até mesmo camadas sobre camadas (*deep learning*). Em suma, um modelo básico de rede neural conterá uma camada de entrada, que pode ser alimentada manualmente pelo usuário ou a partir de uma base de dados [89]**.**

### 2.7.5 - AMBIENTES DE PROGRAMAÇÃO EM PYTHON

O Anaconda é uma ferramenta que oferece todos os pacotes necessários em um lugar só [89, 90, 91]. Ele é muito utilizado pelos programadores e cientistas de dados justamente por causa dessa facilidade, por isso é muito importante o uso de *Python.* Para programadores iniciantes, talvez o uso do ambiente virtual acaba sendo muito importante, mas também pode ser utilizado o próprio controle de comando, prompt de comando da máquina (CMD) [91].

Figura 9 - Ambientes virtuais do (Anaconda Navigator).

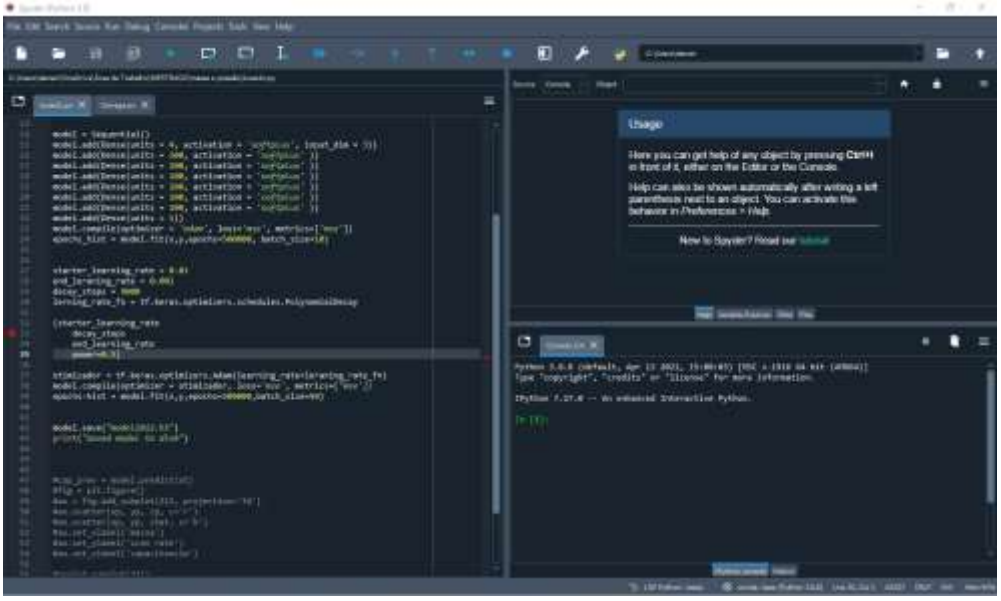



Fonte: Elaborado pelo software Anaconda.

O Anaconda inclui sua própria interface gráfica de usuário (GUI) e a Anaconda Navigator, que é uma alternativa visual para a tradicional interface por linhas de comando. Além disso, sua instalação é muito fácil e acaba sendo preferida, até mesmo, por cientistas de dados que não são programadores. Podemos instalar Anaconda IDE sem privilégios de administrador, o que possibilita sua utilização em qualquer tipo de computador como, por exemplo, os de laboratórios das universidades [91].

O *Spyder* também pode ser usado como biblioteca que fornece *widgets* poderosos relacionados ao console para aplicativos. Ele pode ser usado para integrar um console de depuração diretamente em seu design de interface gráfica com o usuário [91 - 93].

Figura 10 - Interface *Spyder* (Ambiente de desenvolvimento científico em *Python).*



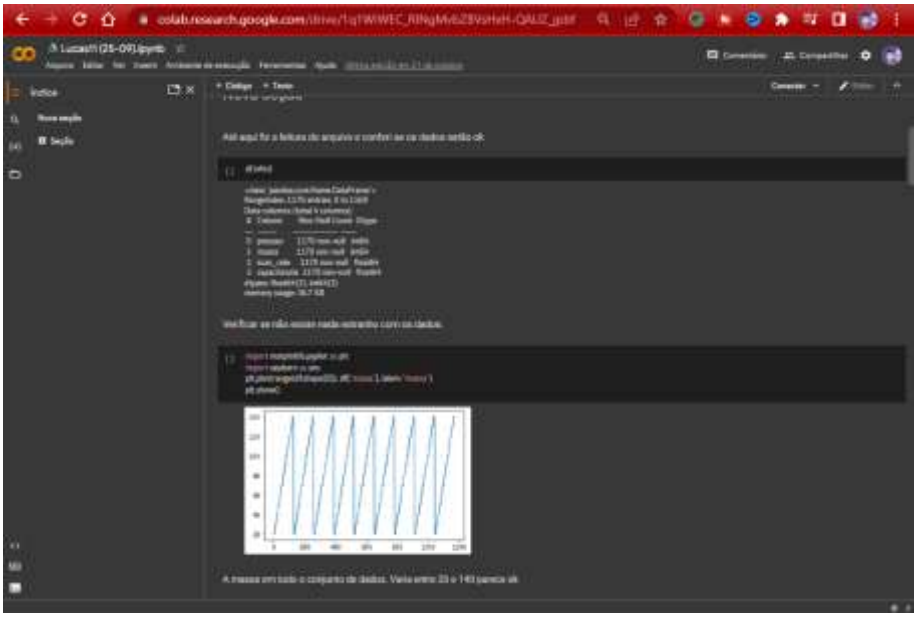
Fonte: Elaborado pelo software Anaconda.

Neste trabalho, em sua segunda parte foi utilizado a ferramenta em que a *Google* criou o *Google Colaboratory* ou simplesmente *Colab* para facilitar e difundir o ensino e a pesquisa em *machine learning* pelo mundo. O *Colab* é um ambiente virtual na nuvem da *Google* onde o programador tem acesso gratuito a um *Jupyter* notebook.

Figura 11 - *Colab* (Ambiente de desenvolvimento científico gratuito*).*


Fonte: Elaborado pelo *Google.*

Figura 12 – Interface *Colab* (Ambiente de desenvolvimento científico gratuito*).*



Fonte: Elaborado pelo software do *Colab*.

No *google Colab*, existem recursos avançados na criação de códigos, *IPython* (melhores interpretadores *Python* interativo) e bibliotecas *Python* populares como *NumPy, SciPy ou matplotlib* (Plotagem interativa 2D / 3D). O projeto tem o nome Jupyter pois foi originalmente criado para o ensino das linguagens *JUlia, PYThon* e R [128 – 134].

Este ambiente é executado em um navegador e oferece vários recursos interessantes para ensino e pesquisa. Ao criar uma sessão no *Colab*, o usuário tem acesso a um processador com dois núcleos, 12 GB de memória RAM e cache L3 de 40-50 Mb. Além disso, o usuário pode ter acesso a uma TPU ou uma GPU. Esta pesquisa essencialmente utilizou o *Colab* em toda segunda etapa deste trabalho.

# *Capítulo 3*

## 3 – MATERIAIS E MÉTODOS

A fim de investigar a crescente demanda por novos meios de produção e estocagem de energia com maior eficiência e baixo custo para aplicações em grande escala, o desempenho de supercapacitores deve ser otimizado e novas soluções computacionais devem ser estudadas e propostas para o aperfeiçoamento de propriedades superiores às atuais. Os procedimentos metodológicos para a realização deste trabalho e os métodos de abordagem na problematização foi o "estado da arte", de forma quantitativa e teórica [94].

As bibliografias citadas foram analisadas e pautadas em utilizar-se das técnicas e métricas que envolveram números, estatística e conjuntos de dados numéricos. Com base na literatura, foram analisados e comparados com outros trabalhos, para confrontos de ideias e análises gerais com base nos livros, artigos científicos, vídeos, palestras e teses. Além disso, foram acessados sites de pesquisas e divulgação científica e foi lido uma série de artigos e comparado o trabalho com embasamento de diferentes autores.

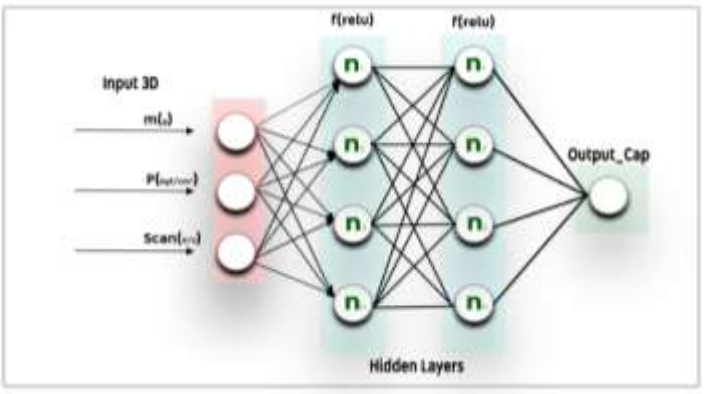
Os conjuntos de dados experimentais foram extraídos dos cálculos das capacitâncias e das curvas e gráficos obtidos nas voltametrias realizadas no trabalho citado (i) o primeiro conjunto de dados foram as massas, que variaram de 20 g até 140 g; (ii) o segundo conjunto de dados as pressões, nos quais variaram de 15 kgf /cm² até 250 kgf/cm²; (iii) o terceiro conjunto de dados os *scanrate,* que são as velocidades de varredura de altas frequências com que o scanner varreu os pontos em busca dos dados citados anteriormente no qual variaram de 0,005 V/s até 0,3 V/s; e (iv) o quarto conjunto de dados foram as capacitâncias que foram calculadas e obtidas como resultado deste trabalho, um processo que decorreu a partir da execução de um código resolvendo uma integral simples criada em *Python*, que será descrito nos resultados.

### 3.1 – O MÉTODO E A ARQUITETURA DA REDE NEURAL ARTIFICIAL

As RNA’s (redes neurais artificiais) surgem como um método alternativo para solucionar problemas computacionais através da simulação do cérebro humano, inclusive em seu comportamento de aprendizado, ou seja, aprendendo, errando e fazendo novas descobertas [3, 7, 10, 33, 35, 37, 40, 46, 48, 95]. Deste modo, foi usado as redes neurais como

a principal técnica ,de simulação computacional de materiais desse trabalho e pode ser observado a arquitetura da rede neural na figura (14).

Figura 13 - Arquitetura da rede neural artificial




Fonte: Elaborado pelo autor.

Primeiramente foram realizados testes programáticos de funções de ativação e predições usando o ambiente de programação *Spyder,* onde inserimos na rede neural todos os conjuntos de dados experimentais extraídos do artigo principal de fabricação dos supercapacitores [13], como citado nas referências. Nessa primeira etapa do trabalho, realizamos a montagem do código, vide figura (15) o qual serão apresentados a seguir, a realização dos treinamentos usando diferentes configurações, funções de ativação, número de épocas e neurônios para uma rede neural artificial. Com isso, também envolve analisar as predições dos parâmetros físicos na fabricação de supercapacitores.

Figura 14 - Estrutura de código da rede neural artificial

```
from keras.models import Sequential
from keras.layers import Dense, Dropout, GaussianDropout

def simple_model():
    model = Sequential()

    model.add(Dense(200, input_dim=3, kernel_initializer='normal', activation='relu'))
    model.add(Dense(100, kernel_initializer='normal', activation='relu'))
    model.add(Dense(50, kernel_initializer='normal', activation='relu'))
    model.add(Dense(1, kernel_initializer='normal'))

    model.compile(loss='mean_squared_error', optimizer='adam')
    return model

model = simple_model()
```

Fonte: Interface do *spyder*.

Para treinarmos a rede neural, separamos dados para treinamentos e testes, para evitar *overfitting*, um ajuste excessivo e um comportamento indesejável de aprendizado. Além disso os conjuntos de dados foram normalizados. Nosso código em si, todos os nossos dados experimentais foram divididos em 70% de dados para treinamento e 30% para os testes [96].

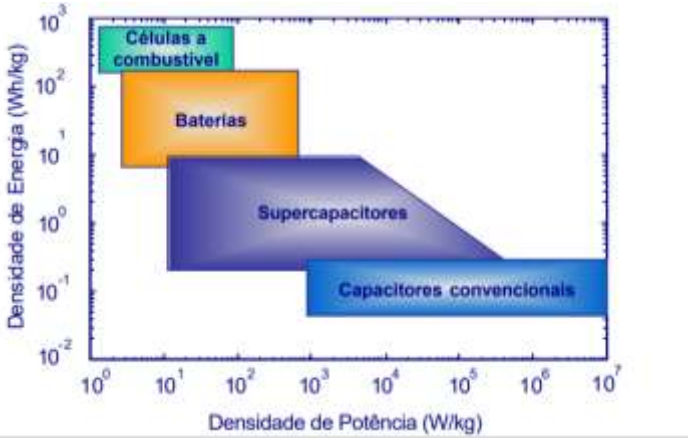
A técnica de normalização em que usamos neste trabalho foi a *MinMaxScaler*, cuja biblioteca é a *Scikit-learn*, no qual inserimos em nosso código em *Python*. Deste modo, foram apresentados na forma de tabelas e gráficos. Ainda foram analisados minuciosamente as principais configurações de parâmetros físicos e os níveis de desempenho das curvas de aprendizado e o valor de predição da inteligência artificial.

Sendo assim, a metodologia para o desenvolvimento deste trabalho foi essencialmente usar a técnica de RNA dentro do *script* de códigos. Mas, também investigar particularmente o estudo da arte da área de (IA) inteligência artificial, desenvolvendo uma possível solução computacional no âmbito técnico da produção de supercapacitores de grafeno com mais eficiência energética. Sendo assim, as investigações e metodologia foram sobre como tem sido o uso dos materiais de eletrodo em supercapacitores na literatura, visando essencialmente neste trabalho prever possíveis configurações de fabricação de supercapacitores de alta capacitância quântica específica e de altas densidades de potência.

## 3.2 - CÁLCULOS DE CAPACITÂNCIA

Com base no gráfico de *Ragone* da figura 16 [97], fica descrito a proporção da diferença nos valores da potência específica e da energia específica nos dispositivos de armazenamento.

Figura 15 – Gráfico de Ragone.




Fonte: Vaz & Pomilio [73]

Para capacitores eletroquímicos o armazenamento ocorre através da aplicação de um campo elétrico de uma fonte externa nas placas ou eletrodos [98], a retenção dos elétrons

ocorre na estrutura física da interface eletrodo-eletrólito e o descarregamento ocorre pela despolarização. De acordo com a equação a capacitância pode ser calculada se considerarmos que as placas paralelas constituídas de dois eletrodos planos idênticos de área A, separados pela distância (constante d), será aproximadamente igual a:

$$C = \varepsilon_o . \varepsilon_r \frac{A}{d} \qquad \textit{Equação 7}$$

onde:

C é a capacitância em farad (F);

ε0 é a permissividade do vácuo (vácuo ou espaço livre);

εr é a constante dielétrica do eletrólito ou permissividade relativa do material utilizado;

A é a área de cada placa (eletrodo, em $m^2$);

d é a distância entre os eletrodos (m).

Com isso, a energia elétrica armazenada em um capacitor é diretamente proporcional a área dos eletrodos (A), pela permissividade do eletrólito (εr) e inversamente proporcional à distância (d) entre as placas (eletrodos). Podemos calcular em um determinado intervalo de tensão, ou (janela de potencial) parâmetros importantes de um supercapacitor como a densidade de energia que determina a quantidade de energia armazenada e a densidade de potência numa taxa de entrega ou fornecimento dessa energia [98]. A capacitância específica pode ser calculada de acordo com a equação abaixo:

$$C = 2 \int \frac{I dv}{\Delta V . v} \qquad \textit{Equação 8}$$

onde:

C é a capacitância em farad (F);

*I* é a corrente elétrica (mA);

$dv$ é o infinitesimal da tensão;

ΔV é a diferença de potencial;

### 3.3 - FUNÇÕES DE ATIVAÇÃO

Modelos de redes neurais artificiais (RNAs) são alternativos ao uso de modelos de regressão [101 - 103]. Para resolver problemas de classificação binária, geralmente, modelos de regressão binomial são utilizados. Esse tipo de modelo é um caso especial dos Modelos Lineares Generalizados (MLG) [104].

A estrutura geral de um MLG é formada por uma componente aleatória, uma componente sistemática e uma função monotônica diferenciável, conhecida como função de ligação. No modelo de regressão binomial, as funções de ligação mais conhecidas são a logit (inversa da *sigmóid* logística) [103, 104], a probit e a função complemento log-log. Porém, a mais utilizada é a função logit, devido ao fato deste modelo fornecer o valor da razão de chances, o que não acontece quando as outras funções de ligação são utilizadas. Em RNA's, uma das limitações é justamente a dificuldade de fornecer o valor da razão de chances de cada variável estudada. Por isso, na maioria das vezes os pesquisadores optam pela regressão com o modelo logístico binomial.

O uso de RNAs fornece uma alternativa a essas análises, particularmente nas situações em que as variáveis dependentes e independentes exibem determinados relacionamentos não-lineares complexos, devido ao fato de que as RNAs são ferramentas úteis para o reconhecimento de padrões ou regressão de dados mesmo na presença de ruídos ou dados incompletos. Estes e outros fatores fazem com que o uso de modelos com redes neurais aumente a cada ano.

No início de 2009 já registrava mais de 21.924 trabalhos com redes neurais, 55% a mais em relação ao ano anterior, contra 69.815 trabalhos com regressão logística, 12% a mais em relação ao ano anterior [105]. O modelo neural pode identificar três elementos básicos tais como um conjunto de sinapses, um somatório e uma função de ativação. As funções de ativação mais utilizadas na prática são a função relu, sigmoid, logística e a função tangente hiperbólica, dependendo das características dos dados e aplicação. Além disso, alguns estudos têm mostrado a importância das funções de ativação para o aprendizado da rede neural. Hornik [106, 107] utilizou funções de ativação não-polinomiais. Singh e Chandra propuseram uma classe de funções sigmoids. Skoundrianos e Tzafestas [108] propuseram uma nova função de ativação sigmoidal com bons resultados para modelagem de sistemas dinâmicos de tempo discreto. Ma e Khorasani [109] usaram a função polinomial Hermite com resultados satisfatórios. Gomes e Ludermir [110] usaram as funções complemento log-log e probit para mostrar que quando os dados seguem uma distribuição binomial com características complemento log-log e probit o uso da função sigmoid logística na modelagem de redes neurais é inadequado [110].

Gomes e Ludermir [110] usaram as funções complemento log log e probit em redes neurais para aproximação e regressão. Estes e outros artigos citados mostram que a escolha da função de ativação é considerada, por muitos especialistas, tão importante quanto a

arquitetura e o algoritmo de aprendizagem da rede neural. A principal característica da função logit é que esta função assume um intervalo contínuo de valores entre 0 e 1.

## 3.4 – EXPERIMENTOS COM FUNÇÕES DE ATIVAÇÃO

Para este trabalho, os estudos das funções de ativação nos trouxeram diversas informações coletadas sobre todas as funções citadas para análise dos resultados de nossa rede neural. As análises aplicadas foram para regressão e identificação da melhor função de ativação.

Foram utilizadas 05 funções de ativação para comparação, sendo elas:

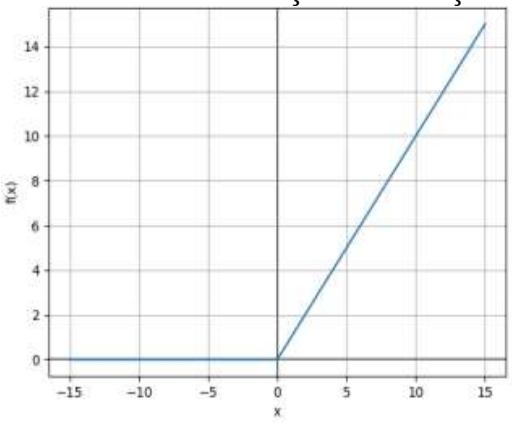
1. • *Rectified Linear Unit* ReLU;
2. • Sigmoid;
3. • Softmax;
4. • Softplus;
5. • *Tanh* Tangente Hiperbólica;

Para os treinamentos e testes, foi adotada uma rede neural com o algoritmo. Para análise, após várias simulações, foi concluído que a função que melhor se adaptou ao nosso problema inicialmente na primeira etapa foi a *Softplus*, por apresentar o menor erro entre a diferença entre o valor real e o valor predito, sendo o mais indicado para problemas de identificação de problemas como o nosso.

### 3.4.1 - ReLu Function Activation

A *Rectified Linear Unit* (ReLU) é uma função que, assim como a logística, também é definida como não-linear, e logo pode ser utilizada como função de ativação entre as redes neurais para aumentar seu poder de processamento (Cybenko, 2010) [112].

Figura 16 - Gráfico da Função de Ativação ReLu.




Fonte: [114]

$$f'(x) = 1 \qquad se\ x > 0$$
$$f'(x) = 0 \qquad se\ x \leqq 0$$
$$f(x) = max\,(0, x) \qquad \text{Equação 9}$$

Outro ponto, como pode ser visto nas Equações a *ReLU* ganha em relação às suas concorrentes por sua função e derivada ser de fácil cálculo, fazendo com que a rede neural se processe mais rapidamente. Uma das grandes áreas que a *ReLU* está sendo mais aplicada atualmente é a de aprendizagem profunda, e isso se deve, principalmente, pelo seu desempenho (Maas et al, 2013) [113, 114]. Uma de suas vantagens é conseguir resolver o problema do desaparecimento do gradiente que a funções *sigmóid* e *softplus* também encontram, porém outro problema que acontece é que para valores negativos, a saída é sempre 0, fazendo com que neurônios que atinjam este valor, possivelmente não conseguirão se recuperar. Um ponto importante que vale ressaltar, é que, por definição, é adotado que a derivada da *ReLU* para x = 0, é igual a 0. Logo, tem a continuidade que pode ser vista na figura.

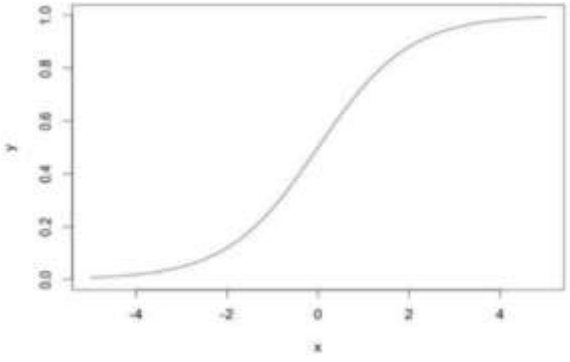
### 3.4.2 - Function Activation Sigmoid

A função de ativação *sigmoid* é comumente utilizada por redes neurais com propagação positiva (*Feedforward)* que precisam ter como saída apenas números positivos, em redes neurais multicamadas e em outras redes com sinais contínuos.

Apesar de seu grande uso, a função de ativação tangente hiperbólica é geralmente uma escolha mais adequada, possui a seguinte equação:

$$\phi(x) = \frac{1}{1 + e^{-x}} \qquad \text{Equação 10}$$

Figura 17 - Gráfico da Função de Ativação Sigmoid.




Fonte: [113]

### 3.4.3 - Function Activation Softmax

A função de ativação *softmax* é usada em redes neurais de regressão. Ela força a saída de uma rede neural a representar a probabilidade dos dados serem de uma das classes definidas. Sem ela as saídas dos neurônios são simplesmente valores numéricos onde o maior indica a classe vencedora [113, 114].

Equação:

$$\phi\, i = \frac{e^{zi}}{\sum_{j \epsilon group}\ e^{zj}} \qquad \textit{Equação 11}$$

Nessa equação, representa o índice do neurônio de saída (o) sendo calculado e j representa os índices de todos os neurônios de um nível. A variável z designa o vetor de neurônios de saída. Vale notar que a função de ativação *softmax* é calculada de forma diferente das demais apresentadas, uma vez que a saída de um neurônio depende dos outros neurônios de saída [113, 114].

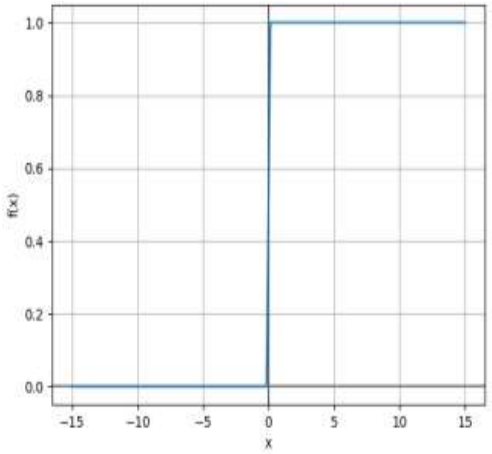
### 3.4.4 - Function Activation Softplus

A função Softplus foi introduzida no início dos anos 2000 por (Dugas et al, 2001) [114] e é definida pela Equação:

$$f(x) = ln(1 + e^{x}) \qquad \textit{Equação 12}$$

$$f'(x) = \frac{1}{1 + e^{-x}} \qquad \textit{Equação 13}$$

Figura 18 - Derivada da Função de Ativação *ReLu*.

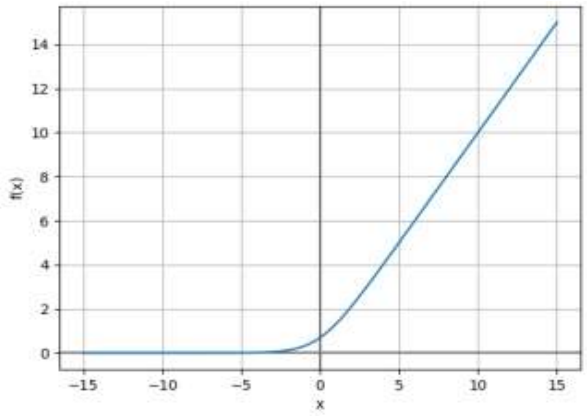



Fonte: [113]

Com um comportamento de função bem similar com a *ReLu*, a função *softplus* tende a ter um funcionamento parecido dentro da rede neural (Glorot et al, 2011) [114]. Entretanto, em comparação ao processamento, a *ReLU* se mostra superior à *Softplus*. Todavia, uma

característica a qual essa função de ativação se destaca é em ser diferenciável em todo o domínio, com derivada correspondente à função logística com valor de α = 1. [112 - 114].

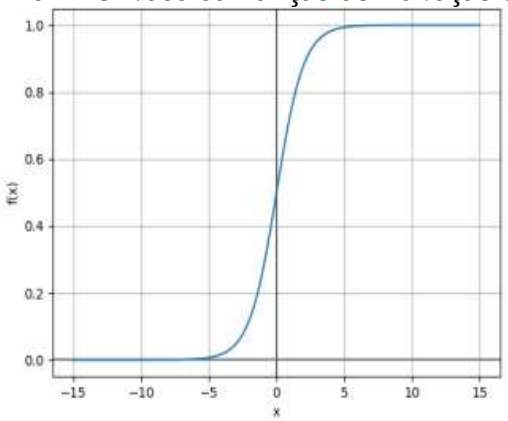
Figura 19 - Derivada da Função de Ativação *Softplus.*




Fonte: [113]

Ao se comparar a curva da função ReLU com a função softplus pode ser considerado que o comportamento assemelha-se, exceto com valores próximo a 0, onde a softplus tem vantagem por ser mais suave. Outra vantagem é que f(x) possui um pequeno intervalo onde seus valores são maiores que 0 para x < 0, logo, possivelmente não terá a característica negativa da ReLU em não conseguir recuperar alguns neurônios [114].

Figura 20 - Derivada da Função de Ativação Softplus.



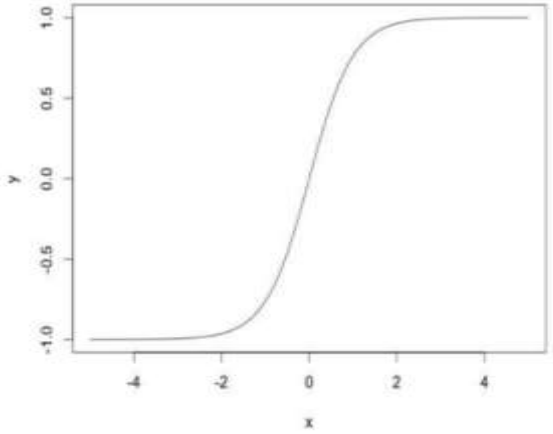

Fonte: [113]

### 3.4.5 - Function Activation tanh

A função de ativação tangente hiperbólica possui uso muito comum em redes neurais cujas saídas devem ser entre -1 e 1.

$$\phi(x) = tanh(x) \qquad \textit{Equação 14}$$

Figura 21 - Derivada da Função de Ativação Tanh.




Fonte: [114]

A vantagem é que as entradas negativas serão mapeadas fortemente negativas e as entradas zero serão mapeadas perto de zero no gráfico *tanh*. A função é diferenciável, é monotônica enquanto sua derivada não é monotônica. A função *tanh* é usada principalmente na regressão entre duas classes. As funções de ativação *tanh, relu e sigmoid* são as mais usadas em redes

## 3.5 - MÉTRICAS DE AVALIAÇÃO

As métricas escolhidas de forma incorreta para avaliação dos modelos de *machine learning* podem afetar diretamente a tomada de decisão. É um ponto de atenção para nossa aplicação, porém algumas decisões incorretas podem ser tão críticas de forma a gerarem consideráveis prejuízos financeiros e/ou comprometerem a saúde ou vida das pessoas [117]. Apresentamos as principais métricas utilizadas em problemas de regressão e no desenvolvimento de projetos de *Machine Learning e Data Science*.

A métrica *Mean Squared Error* - MSE, vide equação abaixo talvez seja a mais utilizada, esta função calcula a média dos erros do modelo ao quadrado. Ou seja, diferenças menores têm menos importância, enquanto diferenças maiores recebem mais peso. Um caso em que é crucial a utilização de métricas apropriadas para cada problema. O valor de tais métricas reflete a qualidade de um modelo, portanto se forem mal escolhidas, será impossível

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \qquad \textit{Equação } 15$$

$\mathrm{MSE}$ = mean squared error
$n$ = number of data points
$Y_i$ = observed values
$\hat{Y}_i$ = predicted values

avaliar se o modelo de fato está atendendo os requisitos necessários [117].

Por exemplo, existem casos em que erros diferentes possuem custos diferentes, e o cálculo da métrica deve refletir esta diferença. Outra situação é quando modelos de qualidades diferentes têm o mesmo valor para a métrica, que reflete uma escolha ruim ou a necessidade de utilizar mais de uma [117].

A métrica MAE – Mean Absolute Error, vide equação abaixo considera apenas valores absolutos (não negativos). De modo que neste erro estamos calculando a diferença entre o real e o previsto, podemos ter resultados negativos e essa diferença negativa é aplicada nas métricas anteriores, já nesta métrica temos que transformar a diferença em valores positivos e posteriormente tirar a média com base no número de elementos.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \qquad Equação\ 16$$

A métrica RMSE: Root Mean Squared Error, vide equação abaixo calcula a raiz quadrada dos erros médios do modelo ao quadrado da diferença entre o valor original a e o valor de previsão obtido pelo modelo. RMSE é a raiz do MSE; Como as métricas RMSE e MSE são elevadas ao quadrado, ambas são muito influenciadas por outliers.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \qquad Equação\ 17$$

A métrica de acurácia é uma das métricas mais populares para avaliar modelos de *machine learning*. Esta é a métrica mais simples. É basicamente o número de acertos (positivos) dividido pelo número total. Ela deve ser usada em *datasets* com a mesma proporção de exemplos para cada classe, e quando as penalidades de acerto e erro para cada classe forem as mesmas. Algumas definições e interpretações que aparecem em alguns textos relacionados com ajustamento de observações e suas aplicações.

Mikhail e Ackermann (1976) apresentam acurácia como sendo o grau de proximidade de uma estimativa com seu parâmetro (ou valor verdadeiro), enquanto precisão expressa o grau de consistência da grandeza medida com sua média [118]. Esses autores acrescentam que a acurácia reflete a proximidade de uma grandeza estatística ao valor do parâmetro para o qual ela foi estimada e que a precisão está diretamente ligada com a dispersão da distribuição das observações. Ainda os mesmos autores, eles afirmam que a precisão pode ser definida como o

grau de conformidade entre as séries de observações da mesma variável aleatória, e que a dispersão da distribuição de probabilidade é um indicador da precisão [119].

Também usamos neste trabalho a métrica $R^2$, o qual na literatura também pode ser conhecida coeficiente de determinação ou como R-dois. Esta métrica representa pra gente o percentual da variância dos dados pelo modelo. Seus resultados variam de 0 a 1, geralmente são expressos em termos percentuais, ou seja, variando entre 0% e 100%. Quanto maior é o valor de $R^2$, mais explicativo é o modelo em relação aos dados previstos, assim como pode ser observado na equação (18).

$$R^2(y, \hat{y}) = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \qquad Equação\ 18$$

Em termos de acurácia ratificam que ela pode ser definida como o grau de proximidade que uma estimativa tem de seu parâmetro, ou seja, proximidade do valor verdadeiro. Trata-se, no entanto, de uma contradição fácil de ser solucionada. Problema maior ocorre com ilustrações, extraídas de uma página da internet, muito popular, podendo levar os seus leitores a interpretações errôneas [120].

## 3.6 - SEPARAÇÃO DE TREINOS E TESTES E NORMALIZAÇÃO DE DADOS

Um dos métodos mais comuns de avaliação do modelo é chamado "Método *Hold Out*" [120], no qual separam-se os dados originais entre treino e teste. Os dados de treino são submetidos a RNA's e produzem o modelo. Após isso, os dados de teste são submetidos ao mesmo para que seja feita a previsão. Normalmente, esse modelo usa 70% dos dados para treino e 30% para testes. Se o desempenho foi satisfatório o modelo pode ser colocado em produção para classificar novos dados. Uma vantagem do método de *hold out* é que os dados são totalmente independentes. Além disso, possui custos computacionais mais baixos e uma desvantagem é que a avaliação do desempenho está sujeita a uma maior variância, dado o menor tamanho dos dados [120].

As funções métricas são usadas para julgar a performance do modelo, essa função é semelhante a função de perda, só que o resultado da avaliação não é usado para treinar o modelo, a métrica de acurácia calcula em qual frequência a previsão é parecida com o modelo do resultado esperado (KERAS, 2020). O método começa o treinamento da rede neural que recebe como parâmetros o banco de dados para o treinamento e suas respectivas classes, a quantidade de épocas, tamanho do lote, e a divisão para validação. Foram treinados na rede, usando o Keras pois pode separar parte dos dados de treinamento em um conjunto de dados para validação e avaliar o desempenho do modelo baseado nesse conjunto a cada época,

(BROWNLEE, 2016). As épocas são usadas para separar o treinamento em fases distintas, o que é útil para o registro e avaliações periódica, ou seja, após cada época serão salvos os valores das métricas e reutilizados nas próximas épocas, (KERAS, 2022).

A técnica de normalização em que usamos neste trabalho foi a *MinMaxScaler*, como pode ser observado na equação (18), cuja biblioteca é a *Scikit-learn.* Assim todos os nossos conjuntos de dados foram normalizados quando implementado em nosso código. Uma vez que também foi possível analisarmos visualmente as normalizações plotadas pelos gráficos das curvas específicas de massas, pressão, *scanrate* e capacitância.

$$MinMaxScaler = \frac{x_i - \min(x)}{\max(x) - \min(x)} \qquad \textit{Equação } 19$$

## 3.7 - PREDIÇÃO DOS MODELOS DE (IA)

Neste trabalho, especificamente procura-se estimar um valores numéricos e não uma classificação de imagens, ou seja, é um trabalho que visa resolver um problema de regressão. Logo abaixo, pode ser observado o modelo de predição na figura (23). De acordo com Turing (1950) [127] podemos esperar que as máquinas acabam por competir com os humanos em todos os aspectos puramente intelectuais, contudo o treinamento da IA em qualquer área do conhecimento requer uma coleção de informações digitais armazenadas e disponibilizadas em bancos de dados com o endosso de profissionais especializados. Quanto melhor a qualidade dos dados utilizados para o treinamento, melhores serão os resultados preditivos sobre uma determinada questão (BRUIN, 2014) [128].

Figura 22 – Software de predição em cada intervalo.


Fonte: Interface do *Spyder*

# *Capítulo 4*

## 4 - RESULTADOS E DISCUSSÃO

Serão apresentados neste capítulo os resultados obtidos acerca dos estudos teóricos e experimentais de *Machine Learning* usando as ferramentas e ambientes de programação como *Google Colab* e o *Spyder* do Anaconda, em especial os treinamentos e testes das funções de ativação usando a técnica de RNA (Redes Neurais Artificiais) objetivando a maximização da eficiência energética de supercapacitores de grafeno. Além dos cálculos de capacitâncias específicas para as amostras e as predições das capacitâncias máximas.

Basicamente, serão apresentados os resultados do processo metodológico utilizado sobre o estado da arte e os testes experimentais computacionais no qual foi dividido em:

**1ª ETAPA** – usando o ambiente Anaconda e *Spyder* em um computador comum com 8GB de RAM*;*

**2ª ETAPA** – usando o ambiente *Colab* e os supercomputadores do *google* com *12*GB de RAM e com GPU*;*

Serão analisados cada função e conterá uma breve descrição de seus contras e prós nas discussões desses resultados. De modo que isso nos permitiu formular diretrizes para escolher a melhor função de ativação, calcular as capacitâncias específicas e conseguir prever as melhores configurações de fabricação de supercapacitores de grafeno. Concluímos que conseguimos chegar em um resultado extremamente positivo e espera-se que na próxima etapa do Doutorado, seja aplicado a técnica experimental de fabricação de supercapacitores.

### 4.1 - PROCESSO EXPERIMENTAL

Avaliamos os resultados deste trabalho os conjuntos de dados experimentais e parâmetros físicos das amostras, entre eles: massa, pressão, capacitância e *scanrate*, extraídos do trabalho experimental de desenvolvimento de supercapacitores de grafeno realizado no Laboratório de Filmes Finos e Processos de Plasma na UFTM [13]. Também analisamos outros trabalhos usando uma abordagem semelhante à este trabalho em si, onde basicamente a tarefa principal é prever uma configuração correspondente que gere um determinado resultado energeticamente otimizado. Para isso ser possível, foi implementado códigos em *Python* usando *Keras* e *TensorFlow*.

A técnica de RNA aplicada a tal conjunto de dados, foi utilizada em vários trabalhos na literatura, tanto na automação de um sistema simples, quanto em processos e sistemas mais complexos. Por isso, comparamos e avaliamos este trabalho de mestrado com os mais diversos pesquisadores da área, para obter novas visões e discussões. Para execução deste

trabalho, foram criados códigos para que finalmente fossem usados para prever as configurações otimizadas na fabricação de supercapacitores.

Nas próximas seções, descrevemos os conjuntos de dados usados, os cálculos de capacitâncias, testes de funções de ativação usando diferentes configurações na execução do programa em nossos experimentos computacionais.

## 4.2 - CONJUNTOS DE DADOS E NORMALIZAÇÃO DE DADOS

Todos os dados supracitados nos materiais e métodos, exceto as capacitâncias calculadas, foram coletados a partir da função exportar do *Origin*, criando assim uma série de *workbooks* (arquivos de texto) em formato (*.dat*) para cada uma das amostras referidas nesta dissertação. Foram inicialmente totalizados 240 *workbooks* (.dat) somente dos dados extraídos das voltametrias. No entanto, com o decorrer do trabalho ao serem encontrados novos conjuntos de dados, criou-se assim *workbooks*, totalizando aproximadamente 350 arquivos de textos, *workbooks* (*.dat*). E, finalmente ao mesclarmos com os novos dados, resultou para nós em um único arquivo completo contendo todos os dados necessários para o treinamento de máquina, contendo 04 colunas de dados, sendo elas: pressão, massa, *scanrate e* capacitância.

Na segunda etapa deste trabalho, todos os nossos conjuntos de dados foram normalizados e a melhoria foi significativa, uma vez que além de analisarmos graficamente as curvas específicas de massas, pressão, *scanrate* e capacitância, também foi otimizado a forma de treinamento e teste da nossa rede neural artificial. A técnica de normalização em que usamos neste trabalho foi a *MinMaxScaler*, cuja biblioteca é a *Scikit-learn.*

## 4.3 - CONFIGURAÇÃO EXPERIMENTAL (TESTES DA PRIMEIRA ETAPA)

Nesta seção, descreveremos os primeiros resultados de cada configuração experimental usada para a execução de cada função de ativação no *Machin*e Learning aplicando a técnica.

Por isso, foi projetado avaliar dois cenários diferentes:

1. Análises das configurações adotadas experimentalmente para os testes das funções de ativação a fim de minimizarmos o erro e aumentarmos o aprendizado;
2. Avaliação dos gráficos obtidos experimentalmente pela inteligência artificial.

As tabelas e os gráficos a seguir descrevem as configurações e os resultados dos treinamentos de cada função de ativação, e, a seguir, os resultados aplicando a técnica de redes neurais usando o arquivo completo do conjunto de dados. A fim de aumentarmos o aprendizado da máquina e minimizar o erro o quanto fosse necessário, para que finalmente a

máquina pudesse assim, prever eventualmente valores maximizados energeticamente de capacitâncias, bem como os parâmetros físicos ideais e otimizados.

### 4.3.1 - *ReLu Activation Function*

*ReLU* é uma abreviação para *rectified linear unit*, ou unidade linear retificada. E ela reproduziu resultados no intervalo [0, ∞[.

Tabela 1 - Processo de aprendizagem para treinamento teste via algoritmo: *ReLu.*

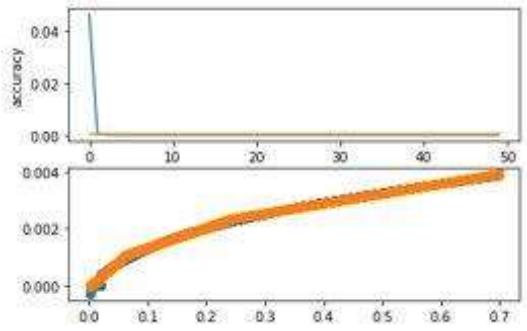
| TEST | N° layer | Função de Ativação | Epoch | Neurônios | Loss |
|---|---|---|---|---|---|
| 1ª configuração | 02 + saída | Relu | 50 | 8 | 8.3099e-05 |
| 2ª configuração | 03 + saída | Relu | 100 | 8 | 2.4348e-05 |
| 3ª configuração | 03 + saída | Relu | 200 | 8 | 3.5088e-04 |
| 4ª configuração | 03 + saída | Relu | 400 | 8 | 9.9157e-06 |
| 5ª configuração | 03 + saída | Relu | 800 | 8 / 800 | 1.1424e-05 |

Fonte: Elaborado pelo Autor

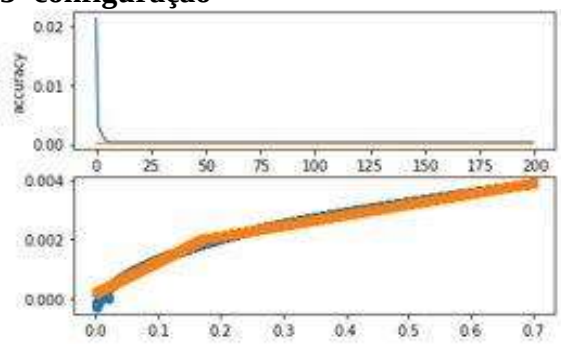
Demonstramos a seguir a figura dos gráficos que representa cada configuração de aprendizado e treinamento usando diferentes número de épocas.

Figura 23 - Gráficos do aprendizado em cada configuração ReLu.

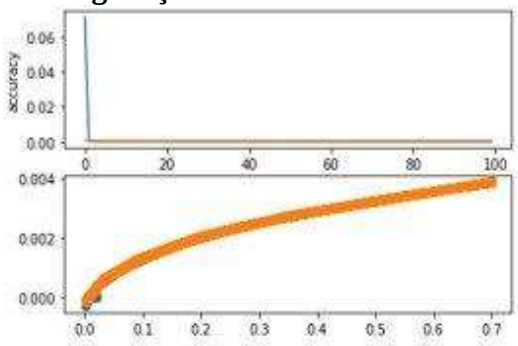
**1ª configuração**



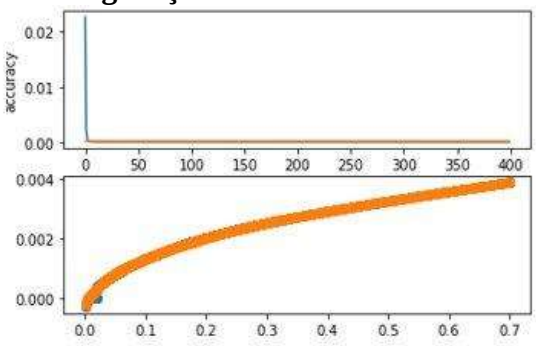

**3ª configuração**



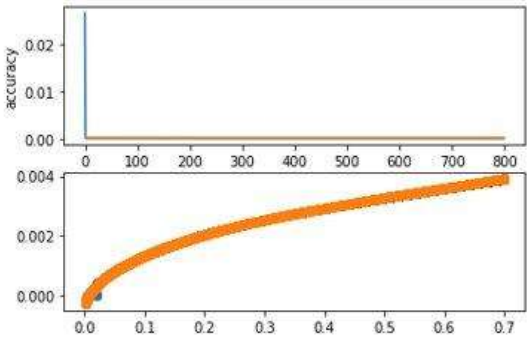

**2º configuração**




**4ª configuração**



**5ª configuração**




Fonte: Gráficos gerados pela inteligência artificial.

A função ReLU retorna 0 para todos os valores negativos, e o próprio valor para valores positivos. É uma função computacionalmente leve, entretanto não é centrada em zero. Como seu resultado é zero para valores negativos, ela tende a "apagar" alguns neurônios durante um passo *forward*, o que aumenta a velocidade do treinamento, mas por outro pode fazer com que esses neurônios "morram" e não aprendam nada se eles só receberem valores negativos [114]. Além disso, ela pode produzir ativações explodidas, já que não possui um limite positivo. Portanto, para o treinamento em uma única dimensão usando ReLu, foi considerado um resultado satisfatório, porém não o melhor para essa aplicação.

### 4.3.2 - *Sigmoid Activation Function*

A principal razão pela qual usamos a função *sigmoid* é porque ela existe entre (0 a 1). Deste modo, ao executarmos essa função, o objetivo era testarmos e obter os resultados para serem comparados. Portanto, segundo a literatura esta função é especialmente usada para modelos onde temos que prever a probabilidade como uma saída.

Tabela 2 - Processo de aprendizagem para treinamento teste via algoritmo: *sigmoid.*

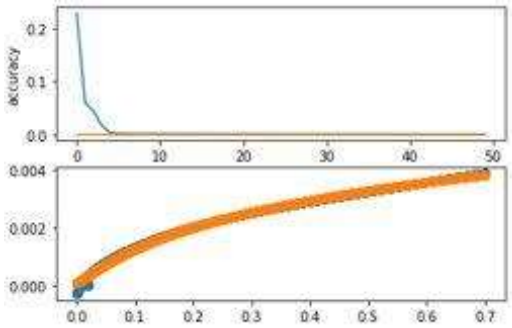
| TEST | N° layer | Função de Ativação | Epoch | Neurônios | Loss |
|---|---|---|---|---|---|
| 1ª configuração | 02 + saída | sigmoid | 50 | 8 | 1.4039e-04 |
| 2ª configuração | 03 + saída | sigmoid | 100 | 8 | 2.7488e-05 |
| 3ª configuração | 03 + saída | sigmoid | 200 | 8 | 1.2874e-05 |
| 4ª configuração | 03 + saída | sigmoid | 400 | 8 | 1.0854e-05 |
| 5ª configuração | 03 + saída | sigmoid | 800 | 8 / 800 | 8.9975e-06 |

Fonte: Elaborado pelo Autor

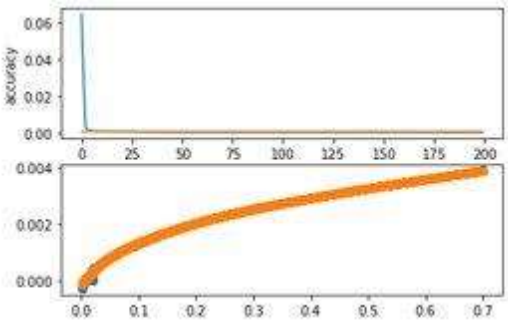
Demonstramos a seguir a figura dos gráficos que representa cada configuração de aprendizado e treinamento usando diferentes número de épocas.

Figura 24 - Gráficos do aprendizado em cada configuração Sigmoid.

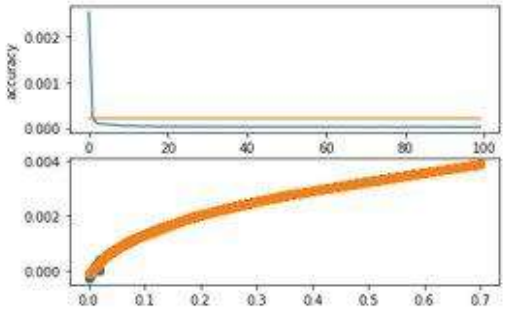
1ª configuração



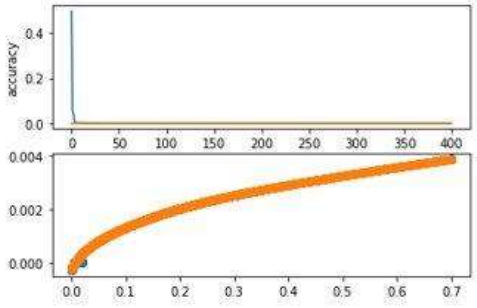

2ª configuração



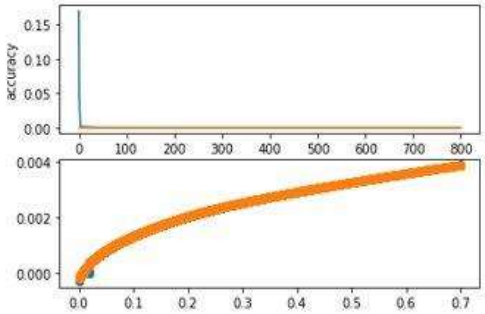

3ºconfiguração




4ª configuração




5ª configuração




Fonte: gráficos gerados pela inteligência artificial.

### 4.3.3 - *Softplus Activation Function*

É uma alternativa de funções tradicionais porque é diferenciável e sua derivada é fácil de demonstrar. Além disso, as saídas produzidas pelas funções *sigmoid* e *tanh* têm limites superior e inferior, enquanto a função *softplus* produz saídas em escala de (0, +∞). Essa é a diferença essencial.

Tabela 3 - Processo de aprendizagem para treinamento teste via algoritmo: *Softplus.*

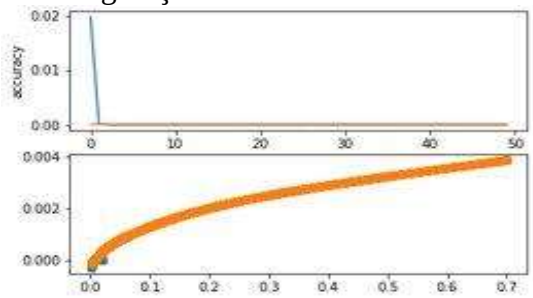
| TEST | Nº layer | Função de Ativação | Epoch | Neurônios | Loss |
|---|---|---|---|---|---|
| 1ª configuração | 02 + saída | Softplus | 50 | 16 | 1.8504e-05 |
| 2ª configuração | 03 + saída | Softplus | 100 | 16 | 1.9802e-05 |
| 3ª configuração | 03 + saída | Softplus | 200 | 16 | 1.4415e-05 |
| 4ª configuração | 03 + saída | Softplus | 400 | 16 | 1.1385e-05 |
| 5ª configuração | 03 + saída | Softplus | 800 | 16 | 8.4681e-06 |
| 6ª configuração | 03 + saída | Softplus | 1000 | 16 | 7.9332e-06 |
| 7ª configuração | 03 + saída | Softplus | 2000 | 16 | 7.5165e-06 |

Fonte: Elaborado pelo Autor

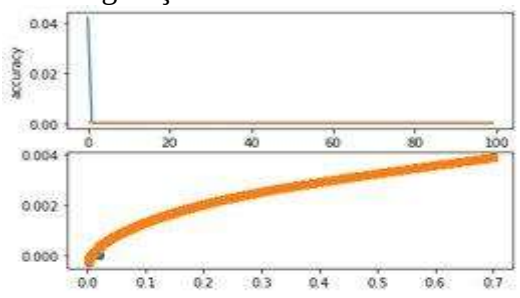
A seguir os gráficos representam cada configuração usando diferentes número de épocas. Com um comportamento de função bem similar com a ReLU, a função *softplus* tende a ter um funcionamento parecido dentro da rede neural (Glorot et al, 2011). Entretanto, em comparação ao processamento dos nossos resultados, *Softplus* se mostrou superior à *ReLU*. Todavia, uma característica a qual essa função de ativação se destaca em ser diferenciável em todo o domínio, com derivada correspondente.

Figura 25 - Gráficos do aprendizado em cada configuração *Softplus.*

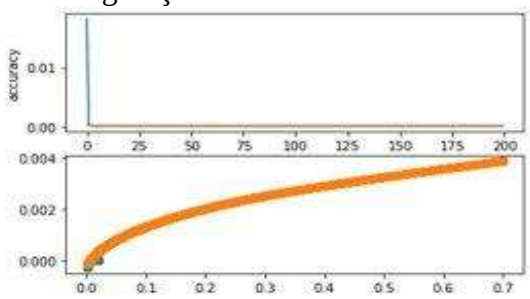
1ª configuração



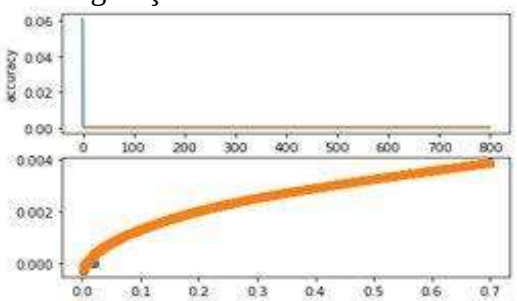

3ª configuração



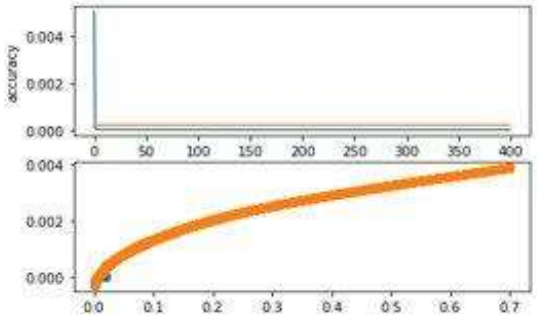

2º configuração



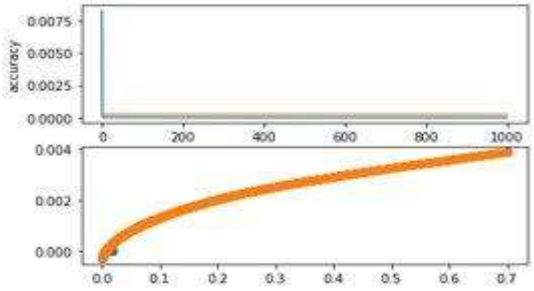

4ª configuração



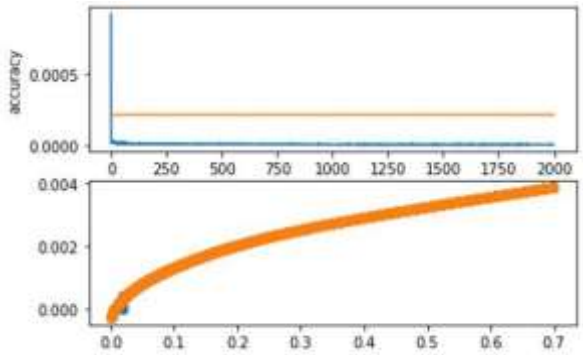
5ª configuração




6ª configuração




7ª configuração




Fonte: gráficos gerados pela inteligência artificial.

Muitos cientistas e pesquisadores tendem a usar a função de ativação que possuem derivadas significativas. Por isso, as funções *ReLu* e *Sigmoid* são as funções de ativação mais comuns na literatura. No entanto, *Softplus* é uma função mais recente que *ReLu*, *sigmoid* e *tanh*, tendo como um dos nossos melhores resultado nos testes iniciais deste trabalho. Por isso, o objetivo deste cenário é mostrar a robustez do nosso trabalho, considerando um cenário usando funções alternativas.

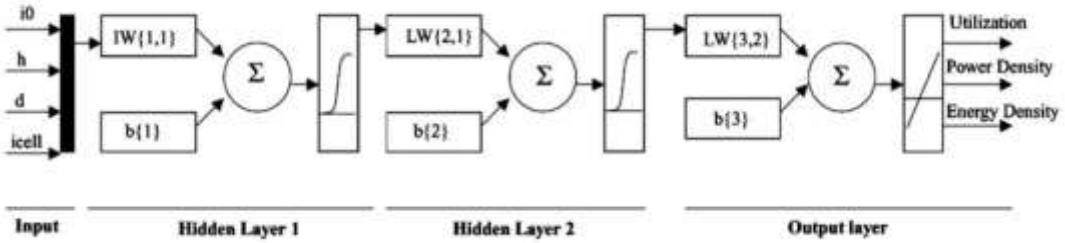
## 4.4 - COMPARAÇÃO COM A LITERATURA

O trabalho apresenta em si comparações sobre o desenvolvimento de um sistema para reconhecimento de padrões, onde os processos de identificação e de regressão são obtidos através de treinamentos de redes neurais. Na literatura, estes treinamentos no processo de rede neural tornam-se difíceis e desafiadores quando há várias camadas ocultas. Alguns exemplos como pesos em ziguezagues, problemas de gradiente e de fugas, são muito complicados, por isso, problemas de fórmula ou saturação na rede neural da função de ativação são alguns desses desafios que pesquisadores têm relatado em seus trabalhos.

Esta seção visa discutir uma comparação com diferentes visões de autores tanto prática quanto teoricamente sobre as RNA’s e funções de ativação que são usadas pela comunidade científica.

O nosso modelo de RNA é baseado no conceito e modelo do *perceptron* multicamadas. Sendo que o treinamento de nossa rede consistiu em ajustar os pesos da rede neural usando um algoritmo de aprendizagem, por isso, treinamos o modelo pelo algoritmo criado em *Python*, uma linguagem fácil, dinâmica e fácil entendimento e a mais usada na atualidade.

Figura 26 - Rede neural para os mapas de desempenho de supercapacitores.




Fonte: H. Farsi, F. Gobal / Ciência de Materiais Computacionais 39 (2007)

O nosso modelo consistiu em camadas de entradas multiplicadas aos seus respectivos pesos, a função de ativação e uma camada de saída. Além disso, a função de ativação escolhida para melhorar o desempenho das nossas RNAs, foi a *softplus*, usada de forma alternativa também para o desenvolvimento de sistema para reconhecimento de padrões em alguns trabalhos pelo mundo.

Para os dois cenários abaixo mencionados, executamos tais comparações diferentes:

1. Comparação das funções de ativação usada para otimizações nas RNA's;
2. Comparação da técnica RNA de simulação deste trabalho com o estado da arte e literatura;

Todas as comparações foram baseadas nas configurações de trabalhos anteriores baseados em neurônios ou definido por experimentos preliminares.

Existem alguns modelos disponíveis para estudar o desempenho do supercapacitor [12, 13], quanto aos artigos estudados sobre os modelos dos melhores desempenho para supercapacitor, a literatura revela que são limitadas as iniciativas tomadas para modelagem de supercapacitor por rede neural artificial (RNA). As RNA's têm sido empregadas para vários problemas por causa de seus fascinantes recursos de aprendizado, computação rápida e facilidade de implementação; problemas como reconhecimento de padrão, modelagem de sistema não linear, indústria elétrica e eletrônica, produção e captação de energia, indústria química, aplicações médicas, etc.[15 - 17].

Para H. Farsi e F. Gobal, tais autores treinaram redes neurais artificiais para previsão de alto desempenho de supercapacitor. A rede neural artificial foi utilizada para calcular o desempenho de um modelo de supercapacitor, expresso pela densidade de energia e utilização

das características sintéticas, intrínsecas e operacionais. Uma rede neural de quatro camadas com duas camadas ocultas com 6 e 15 nós mostrou-se bem capaz de simular o desempenho do capacitor com a convergência alcançada frequentemente em um número relativamente pequeno de épocas [2]. Quanto aos parâmetros de entrada, tamanho do cristal, comprimento da superfície da rede, densidade de corrente de troca do material ativo do capacitor e a corrente da célula empregada enquanto utilização, densidade de energia e densidade de potência foram as saídas.

Marie-Françoise et al. relatou sua ferramenta de modelagem para avaliação do comportamento térmico e elétrico de supercapacitores usando RNA [18]. Wu CH et al. desenvolveu o método online para previsão de conversão de energia e gerenciamento de supercapacitor usando RNA [19]. Para estes autores, a RNA também é chamada de computação neural e é uma das áreas de inteligência artificial que mais cresce no mundo.

Segundo Wu CH et al., as RNAs são treinadas pela experiência, quando aplicada uma entrada desconhecida à rede pode generalizar a partir de experiências anteriores e produziu um novo resultado que é um resultado necessário [16,17]. RNA geralmente consiste em uma camada de entrada, uma camada oculta e uma camada de saída. Assim, as RNA's também são conhecidas como modelos orientados a conexão ou processamento paralelo distribuído.

Assim, consideramos este cenário do trabalho como representativo de um cenário mais complexo, que é a parte mais difícil do que aquele cenário tipicamente enfrentado na literatura anterior sobre os simples testes das funções de ativação.

## 4.5 - ANÁLISES DE CURVAS E CÁLCULOS DE CAPACITÂNCIAS

De acordo com os dados dos voltamogramas recebidos [13], para todas as velocidades de varredura desde a menor velocidade de varredura (0,005 V/s) à maior velocidade de varredura (0,3 V/s), foram calculados os valores de capacitância específicas através da Equação (15). Os parâmetros físicos de fabricação com uma variação de massa de 20 g até 140 g, submetido a uma variação de pressão 15 kgf/cm² até 250 kgf/cm²,

$$C = 2 \int \frac{I\, dV}{\Delta V . v} \qquad \textit{Equação 18}$$

A partir das curvas foram calculados os valores de capacitância específica para todas densidades de corrente aplicadas. Os valores obtidos para cada configuração obtivemos um valor de capacitância específica, e com isso adicionamos o resultado a um arquivo de dados.

Por exemplo, estão apresentados na Tabela 4, os cálculos de capacitância com os dados somente das amostras de 20 g sujeito a uma variação de 15 kgf/cm² até 250 kgf/cm².

Tabela 4 - Cálculo de capacitância com amostras de 20 g.

| Massa (g) | Scanrate (V/s) | Capacitância (Faraday) |
|---|---|---|
| 20 | 0,005 | 0.04539764169273388 |
| 20 | 0,01 | 0.0671472049392749 |
| 20 | 0,02 | 0.11722067842092738 |
| 20 | 0,03 | 0.15912860312734847 |
| 20 | 0,05 | 0.2269574006537094 |
| 20 | 0,08 | 0.2982328752397377 |
| 20 | 0,09 | 0.31397912849530013 |
| 20 | 0,1 | 0.332223824529579 |
| 20 | 0,2 | 0.4700107616155108 |
| 20 | 0,3 | 0.5625188373124105 |

Fonte: Elaborado pelo autor.

Sendo assim, como dito acima, somente são mencionados os valores de capacitâncias de 20 g quando submetidas a uma variação de 15 kgf/cm² até 250 kgf/cm². No entanto, foram obtidos os valores para todas as amostras de 30 g até 140 g, variando os *scanrate* de (0,005 V/s) à (0,3 V/s), e, em cada configuração obtivemos um valor de capacitância específica executando a integral. E, com isso foi-se adicionando os resultados de capacitâncias sequencialmente a um arquivo de texto (arquivo de dados) *workbook*, em formato (*.dat*) separados por vírgulas. Assim montando o nosso arquivo completo com todos os dados necessários para entregar à inteligência artificial para aprender usando a técnica de RNA (Redes Neurais Artificiais).

Analisando as curvas em diferentes taxas de varredura pôde ser observado nas curvas CV que existem influências na variação da pressão de compactação e da massa. Portanto, a evolução da capacitância específica de acordo com os cálculos, apresentaram maiores valores para as menores pressões e com altos valores de massa. A performance energética dos supercapacitores se mantém praticamente constante a determinados valores de pressão acima de 80 kgf/cm², diferenciando-se em relação onde os valores de capacitância específica são muito maiores com pressões baixíssimas com aproximadamente 15 kgf/cm².

## 4.6 - RESULTADO DO MÉTODO DE RNA'S

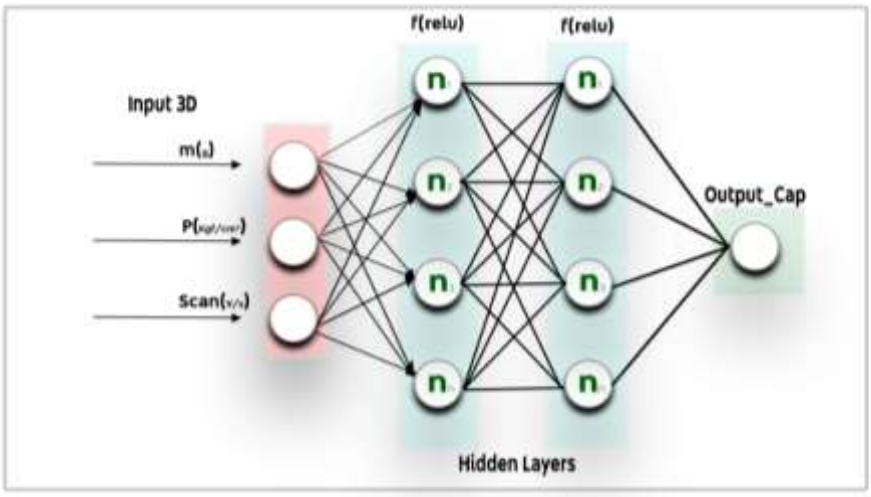
O nosso modelo *perceptron* simplificados multicamadas para supercapacitor é mostrado na figura a seguir, uma arquitetura de RNA inteligente onde os parâmetros de

treinamento são descritos algebricamente, com o intuito de encontrar uma regra de aprendizado que nos permite encontrar melhoramentos sucessivos.

Uma rede neural artificial é um sistema de processamento paralelo de informações constituído pela interconexão de unidades básicas de processamento, denominadas neurônios artificiais, que têm a propensão natural para armazenar conhecimento experimental e torná-lo disponível para o uso. Todo conhecimento adquirido pela rede se dá através de um algoritmo de aprendizagem, cuja função é modificar os pesos de conexões entre os neurônios da rede, conhecidos como pesos sinápticos, de forma ordenada a fim de alcançar o mapeamento desejado.

A arquitetura da rede neural artificial está relacionada com a maneira pela qual os neurônios dela estão organizados. As camadas de entrada e saída são intuitivas e representam o número de entradas e saídas do problema em questão. Normalmente uma rede neural possui uma camada de entrada, uma camada de saída e camadas escondidas, na qual é definido empiricamente e varia de acordo com o problema. Com isso forma-se uma rede de múltiplas camadas.

Figura 27 - Arquitetura da rede neural artificial




Fonte: Elaborado do autor.

Para os autores Kheraa e Khanb, que publicaram um artigo contando uma abordagem de RNA para determinar a condição de circuito em tempo real de capacitores alvo em sistemas eletrônicos de potência, considerando o efeito da frequência de operação e da temperatura simultaneamente. Percebe-se pelos resultados que a RNA proposta por estes autores é treinada off-line usando conjuntos de dados de treinamento e testes obtidos experimentalmente do banco de testes desenvolvido. Com objetivos finais diferentes, este trabalho usa uma estratégia metodológica muito similar a este artigo.

## 4.7 - RESULTADOS DOS TESTES DAS FUNÇÕES DE ATIVAÇÃO (1ªETAPA)

Para os resultados de todos os experimentos das funções de ativação, apresentamos neste trabalho em forma de tabelas e gráficos para melhor visualização de todas as redes neurais.

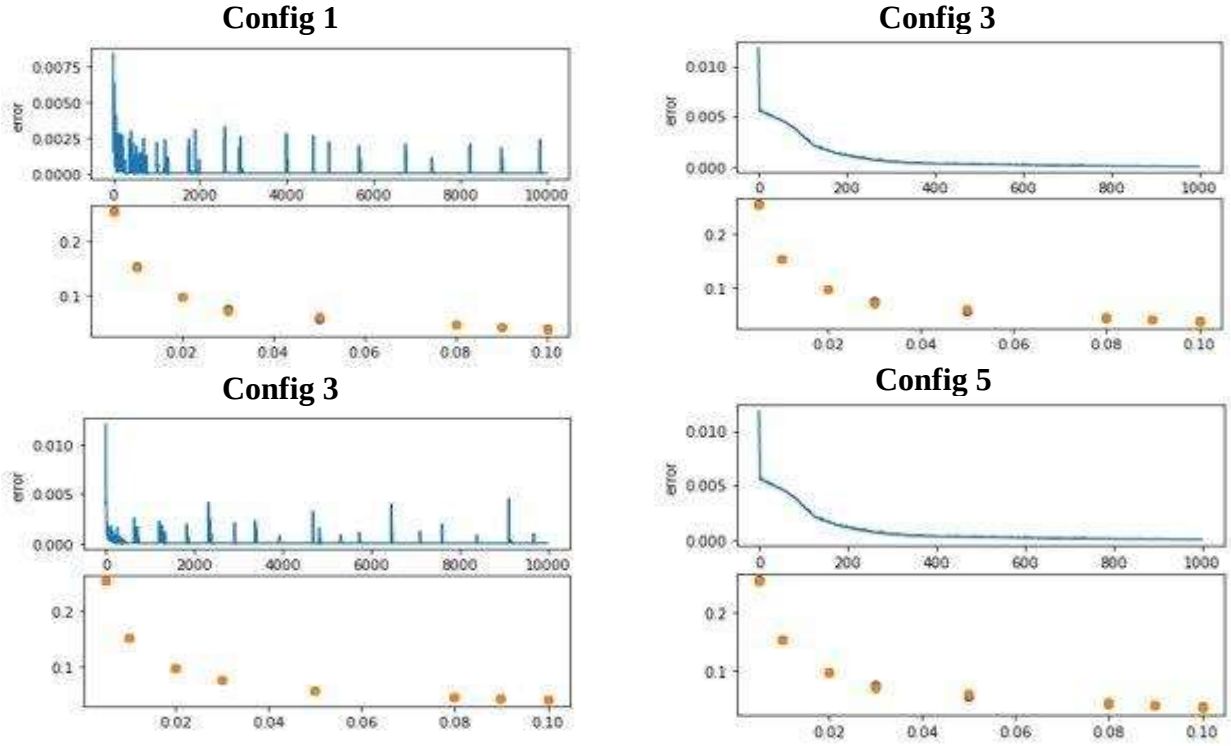
### 4.7.1 - Configuração A - *Relu Function Activation*

Tabela 5 - Processo de aprendizagem para treinamento teste via algoritmo: *ReLu.*

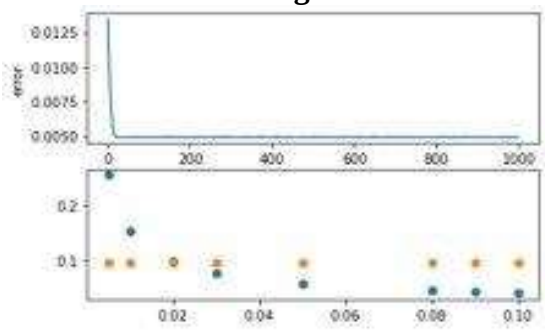
| TEST | Nº layers | Números de Neurônios | Épocas | Erro | MeanSquaredError | Capacitância prevista |
|---|---|---|---|---|---|---|
| Config 1 | Input + 3 layers + output | 1000 | 10000 | 7.8493e-06 | 7.8493e-06 | [[0.05124097]] |
| Config 2 | Input + 3 layers + output | 100 | 10000 | 1.3706e-07 | 1.3706e-07 | [[0.04920918]] |
| Config 3 | Input + 3 layers + output | 10 | 1000 | 9.4071e-06 | 9.4071e-06 | [[0.05013456]] |
| Config 4 | Input + 3 layers + output | 8 | 1000 | 0.0020 | 0.0020 | [[0.05547142]] |
| Config 5 | Input + 3 layers + output | 6 | 100 | 1.3092e-05 | 1.3092e-05 | [[0.05160636]] |
| Config 6 | Input + 3 layers + output | 4 | 100 | 0.0049 | 0.0049 | [[0.09480281]] |
| Config 7 | Input + 3 layers + output | 2 | 50 | 0.0049 | 0.0049 | [[0.09557226]] |

Fonte: Elaborado pelo autor.

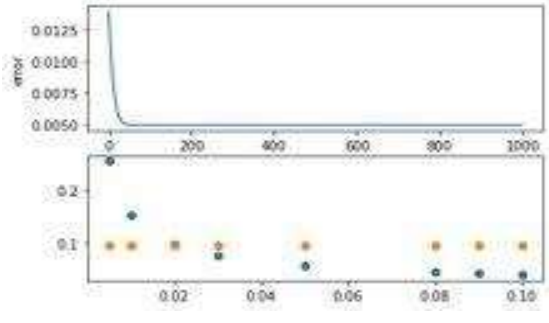
Figura 28- Gráficos do aprendizado em cada configuração ReLu.



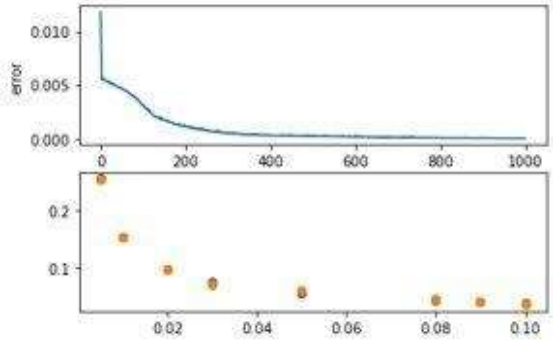
**Config 4**




**Config 6**




**Config 7**




Fonte: gráficos gerados pela inteligência artificial.

### 4.7.2 - Configuração B - Sigmoid Function Activation

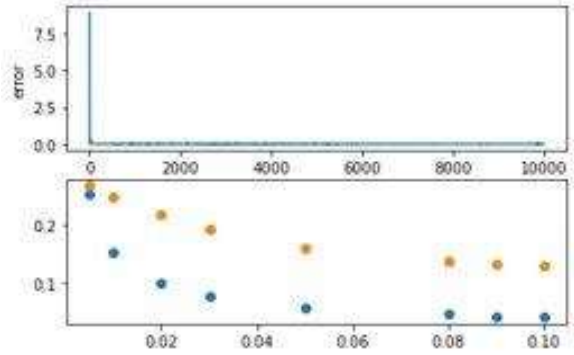
Tabela 6 - Processo de aprendizagem para treinamento teste via algoritmo: *Sigmoid.*

| TEST | Nº layers | Número de Neurônios | Épocas | Erro | MeanSquaredError | Capacitância prevista |
|---|---|---|---|---|---|---|
| Config 1 | Input + 3 layers + output | 1000 | 10000 | 0.0070 | 0.0070 | [[0.14244601]] |
| Config 2 | Input + 3 layers + output | 100 | 10000 | 0.0054 | 0.0054 | [[0.10372876]] |
| Config 3 | Input + 3 layers + output | 10 | 1000 | 0.0049 | 0.0049 | [[0.09285121]] |
| Config 4 | Input + 3 layers + output | 8 | 1000 | 0.0048 | 0.0048 | [[0.09274849]] |
| Config 5 | Input + 3 layers + output | 6 | 100 | 0,049 | 0,049 | [[0.09585831]] |
| Config 6 | Input + 3 layers + output | 4 | 100 | 0,049 | 0,049 | [[0.09525388]] |
| Config 7 | Input + 3 layers + output | 2 | 50 | 0,049 | 0,049 | [[0.09587823]] |

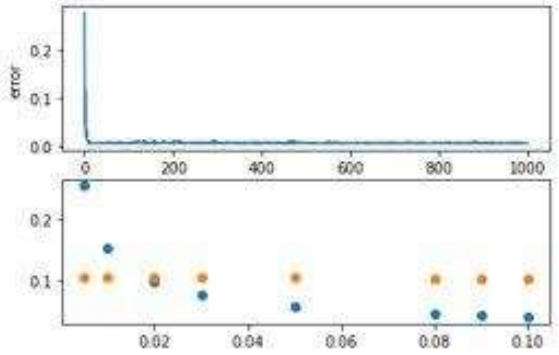
Fonte: Elaborado pelo autor.

Figura 29 - Gráficos do aprendizado em cada configuração *Sigmoid*.

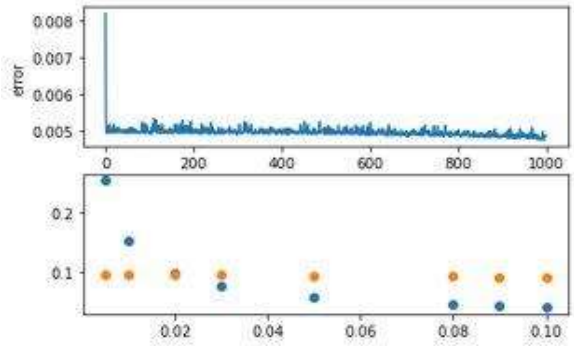
**Config 1**



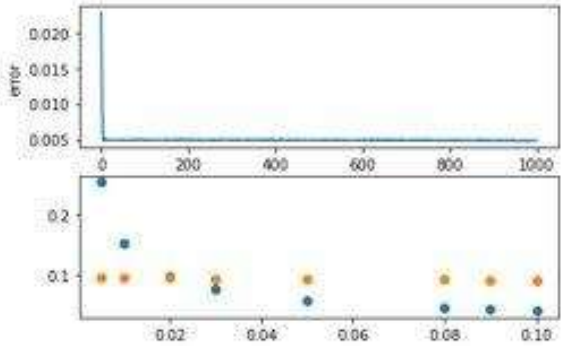

**Config 2**



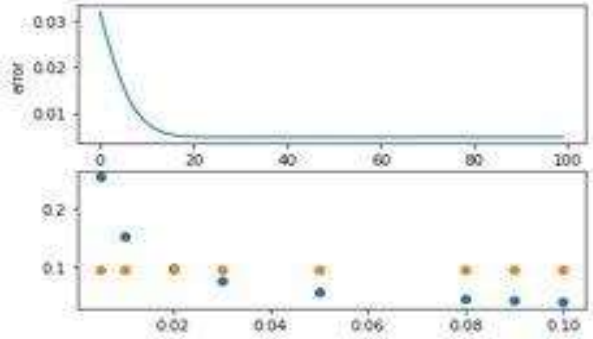

**Config 3**



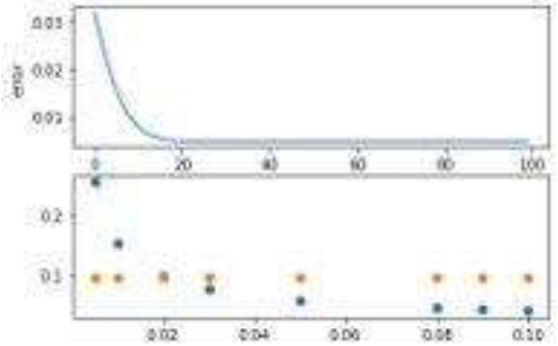

**Config 4**



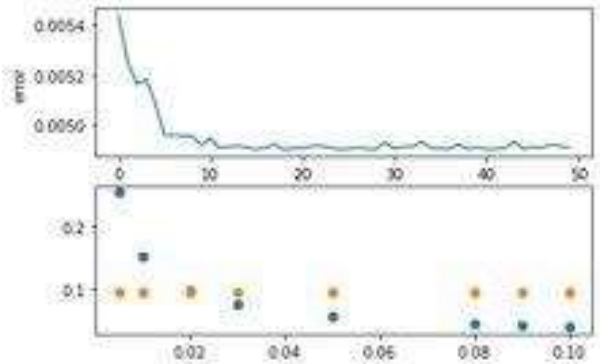

**Config 5**




**Config 6**




**Config 7**




Fonte: gráficos gerados pela inteligência artificial.

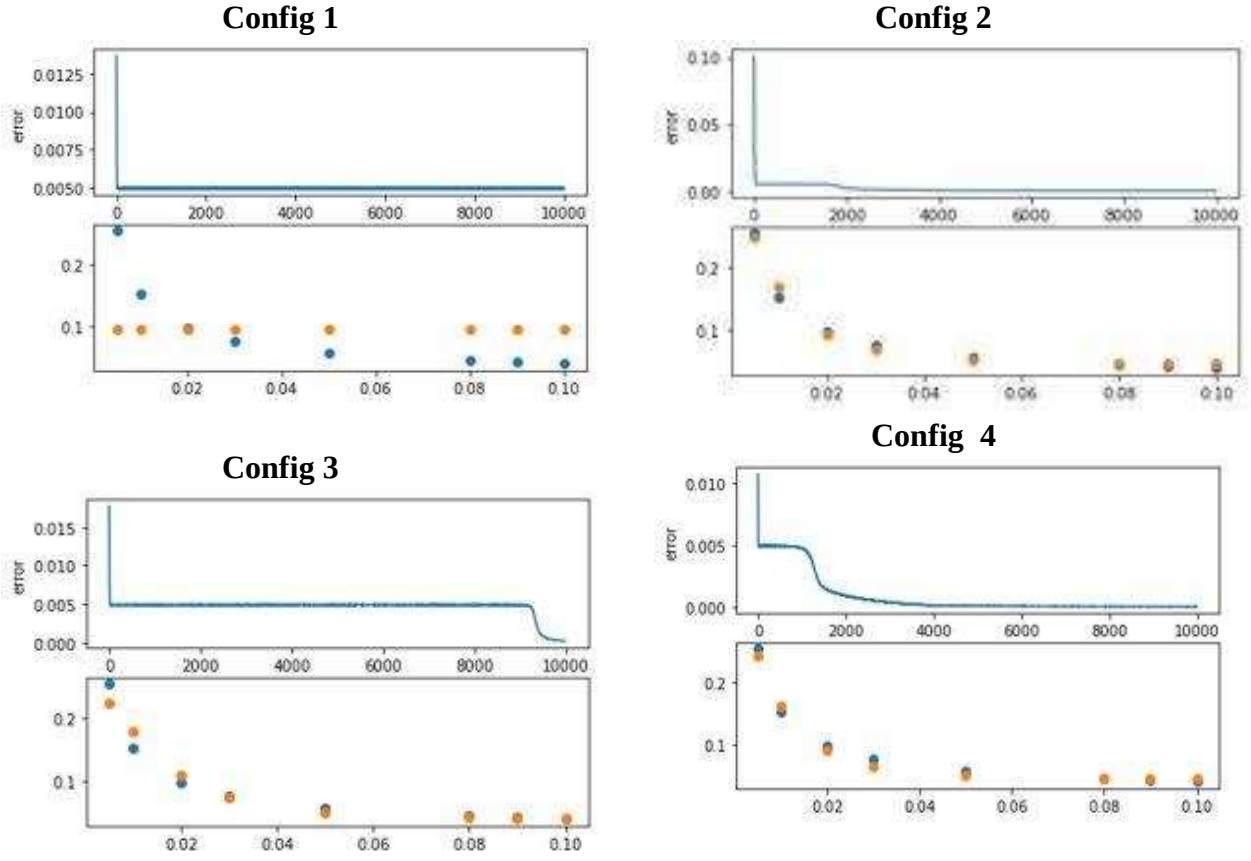
### 4.7.3 -Configuração C - *Function Activation Softmax*

Tabela 7 - Processo de aprendizagem para treinamento teste via algoritmo: *Softmax*.

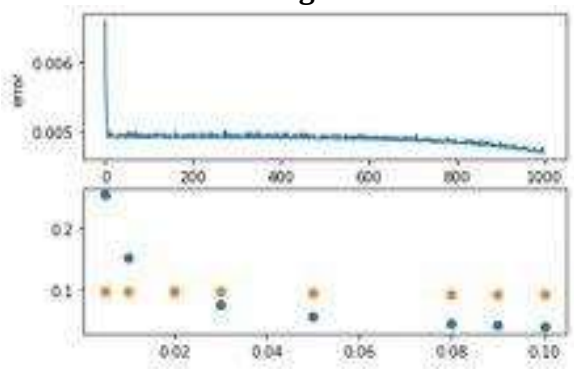
| TEST | Nº layers | Número de Neurônios | Épocas | Erro | MeanSquaredError | Capacitância prevista |
|---|---|---|---|---|---|---|
| **Config 1** | Input + 3 layers + output | 1000 | 10000 | 0.0049 | 0.0049 | [[0.09689048]] |
| **Config 2** | Input + 3 layers + output | 100 | 10000 | 2.2996e-04 | 2.2996e-04 | [[0.04356765]] |
| **Config 3** | Input + 3 layers + output | 10 | 10000 | 7.0782e-05 | 7.0782e-05 | [[0.04635696]] |
| **Config 4** | Input + 3 layers + output | 8 | 10000 | 5.1330e-05 | 5.1330e-05 | [[0.0483404]] |
| **Config 5** | Input + 3 layers + output | 6 | 10000 | 6.3133e-05 | 6.3133e-05 | [[0.04912537]] |
| **Config 6** | Input + 3 layers + output | 4 | 10000 | 5.0940e-05 | 5.0940e-05 | [[0.04794693]] |
| **Config 7** | Input + 3 layers + output | 2 | 10000 | 6.3126e-05 | 6.3126e-05 | [[0.04670158]] |

Fonte: Elaborado pelo autor.

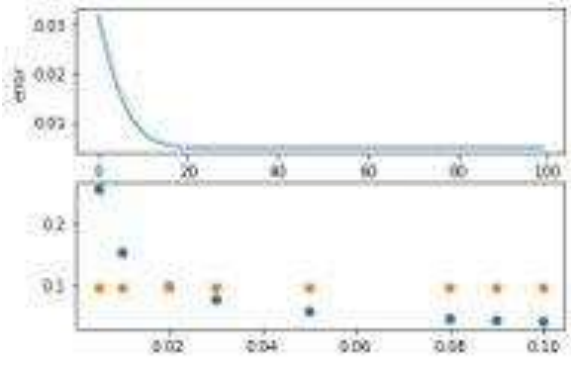
Figura 30 - Gráficos do aprendizado em cada configuração Softmax.



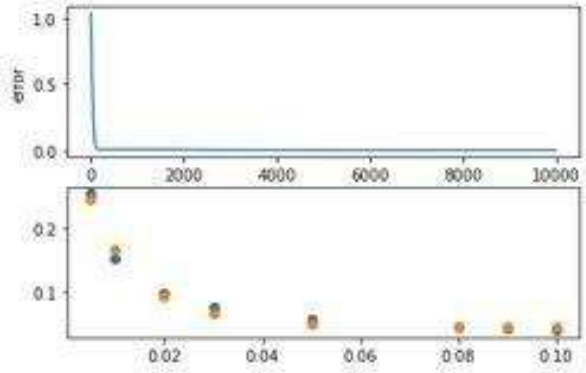
**Config 5**




**Config 6**




**Config 7**




Fonte: gráficos gerados pela inteligência artificial..

### 4.7.4 -Configuração D - *Function Activation Softplus*

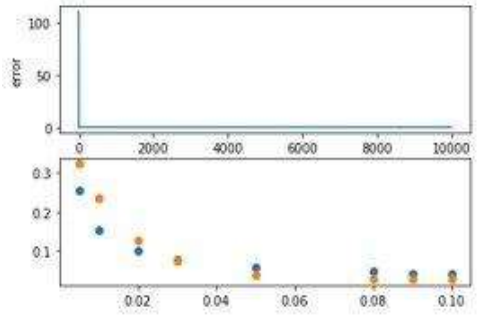
Tabela 8 - Processo de aprendizagem para treinamento teste via algoritmo: *Sofplus*

| TEST | Nº layers | Número de Neurônios | Épocas | Erro | MeanSquaredError | Capacitância prevista |
|---|---|---|---|---|---|---|
| Config 1 | Input + 3 layers + output | 1000 | 10000 | 2.0305e-04 | 2.0305e-04 | [[0.02853657]] |
| Config 2 | Input + 3 layers + output | 100 | 10000 | 5.1666e-04 | 5.1666e-04 | [[0.0343825]] |
| Config 3 | Input + 3 layers + output | 10 | 10000 | 5.2458e-04 | 5.2458e-04 | [[0.03953518]] |
| Config 4 | Input + 3 layers + output | 8 | 10000 | 4.7105e-04 | 4.7105e-04 | [[0.02502351]] |
| Config 5 | Input + 3 layers + output | 6 | 10000 | 0.0015 | 0.0015 | [[0.05347293]] |
| Config 6 | Input + 3 layers + output | 4 | 10000 | 0.0019 | 0.0019 | [[0.06637691]] |
| Config 7 | Input + 3 layers + output | 2 | 10000 | 0.0015 | 0.0015 | [[0.05381072]] |

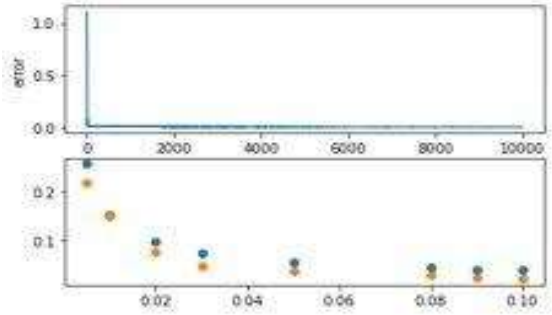
Fonte: Elaborado pelo autor.

Figura 31 - Gráficos do aprendizado em cada configuração Softplus.

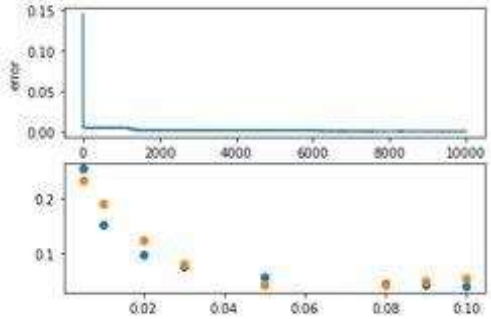
**Config 1**



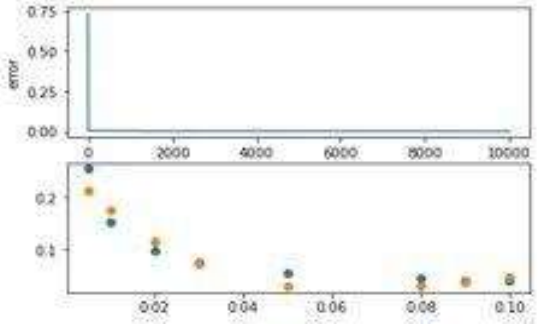

**Config 2**



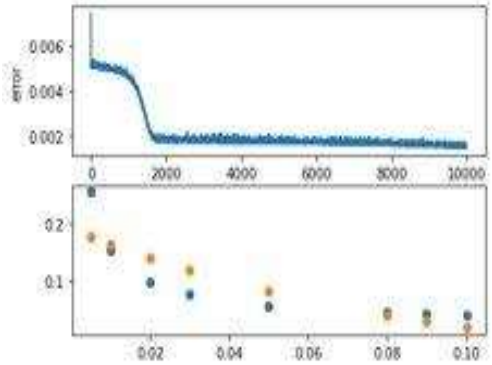

**Config 3**



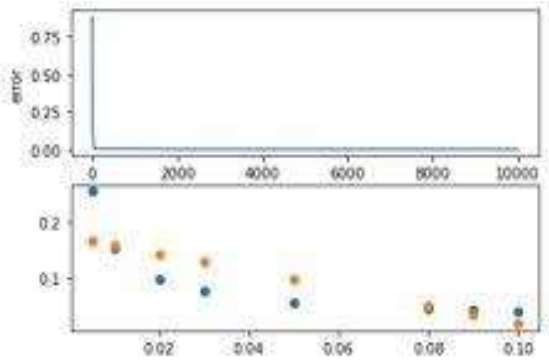

**Config 4**



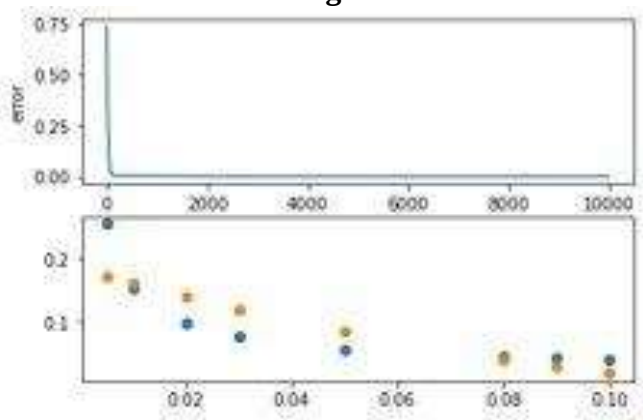

**Config 5**




**Config 6**




**Config 7**




Fonte: gráficos gerados pela inteligência artificial.

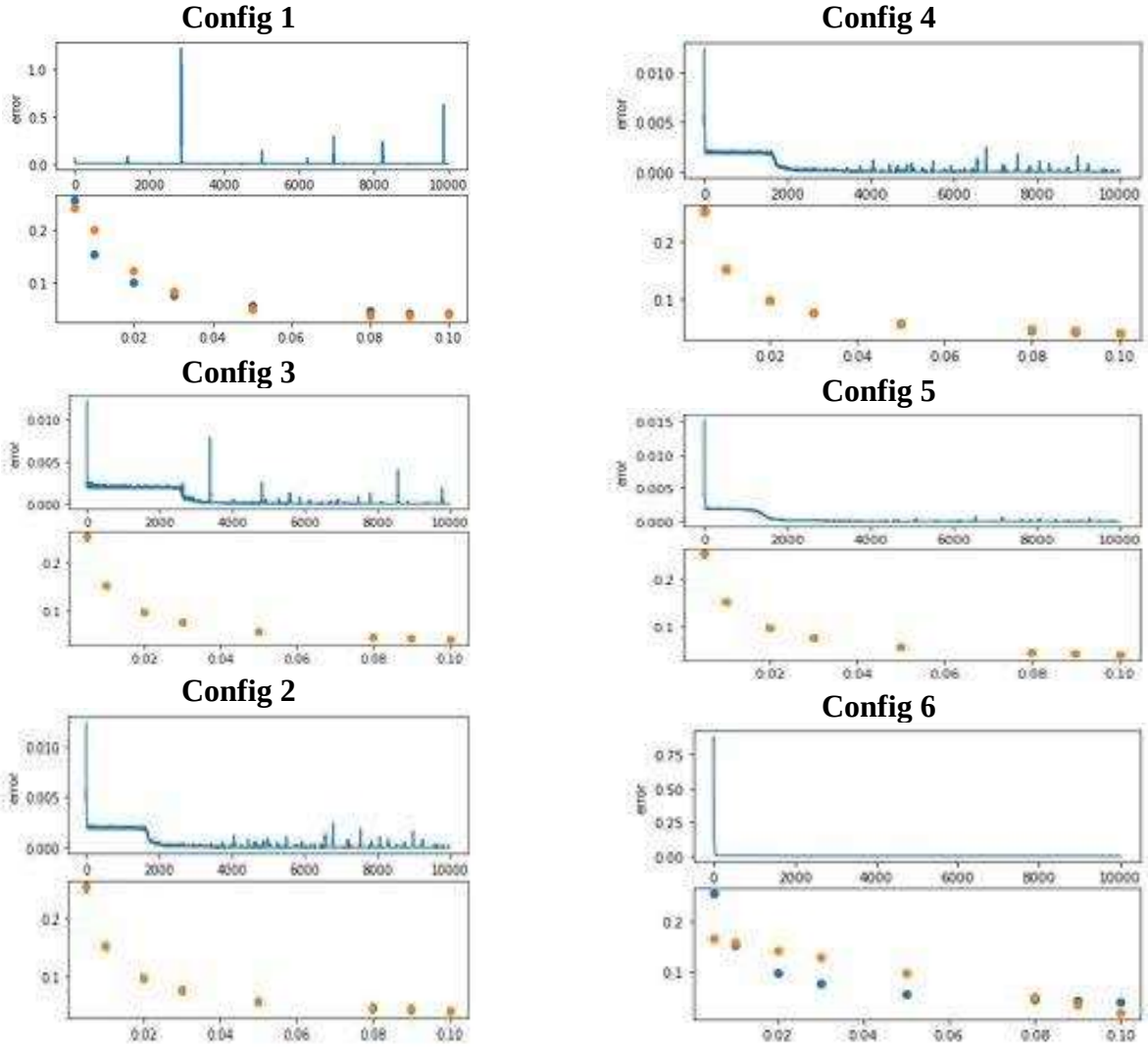
### 4.7.5 - Configuração E - *Function Activation tanh*

Tabela 9 - Processo de aprendizagem para treinamento teste via algoritmo: *Tanh.*

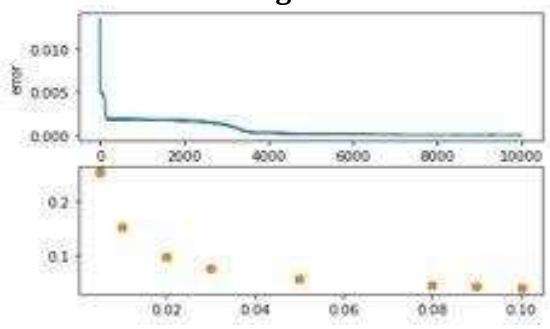
| TEST | Nº layers | Número de Neurônios | Épocas | Erro | MeanSquaredError | Capacitância prevista |
|---|---|---|---|---|---|---|
| **Config 1** | Input + 3 layers + output | 1000 | 10000 | 4.1100e-04 | 4.1100e-04 | [[0.0363836]] |
| **Config 2** | Input + 3 layers + output | 100 | 10000 | 1.2422e-07 | 1.2422e-07 | [[0.04832466]] |
| **Config 3** | Input + 3 layers + output | 10 | 10000 | 9.4071e-06 | 9.4071e-06 | [[0.05013456]] |
| **Config 4** | Input + 3 layers + output | 8 | 10000 | 3.9451e-06 | 3.9451e-06 | [[0.04947824]] |
| **Config 5** | Input + 3 layers + output | 6 | 10000 | 8.0633e-06 | 8.0633e-06 | [[0.04725267]] |
| **Config 6** | Input + 3 layers + output | 4 | 10000 | 4.4994e-06 | 4.4994e-06 | [[0.04906556]] |
| **Config 7** | Input + 3 layers + output | 2 | 10000 | 3.0411e-06 | 3.0411e-06 | [[0.04704253]] |

Fonte: Elaborado pelo autor.

Figura 32 - Gráficos do aprendizado em cada configuração Tanh.



**Config 7**




Fonte: gráficos gerados pela inteligência artificial.

## 4.8 - TESTES FINAIS (REDES NEURAIS ARTIFICIAIS) *Machine Learning*

### 4.8.1 - Configuração F (3D) - *Function Activation "Relu"*

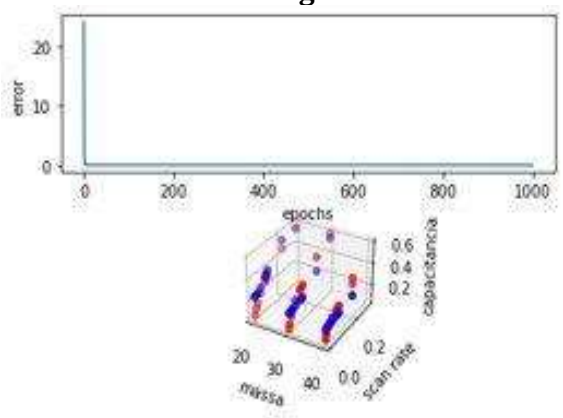
Tabela 10 - Processo de aprendizagem para treinamento teste via algoritmo: *ReLu.*

| TEST | Nº layers | Número de neurônios | Épocas | Erro | MeanSquaredError |
|---|---|---|---|---|---|
| Config 1 | Input + 3 layers + output | 10 | 1.000 | 0.0197 | 0.0197 |
| Config 2 | Input + 3 layers + output | 100 | 1.000 | 0.0076 | 0.0076 |
| Config 3 | Input + 3 layers + output | 500 | 1.000 | 0.0194 | 0.0194 |
| Config 4 | Input + 3 layers + output | 20 | 10.000 | 9.2775e-04 | 8.5456e-04 |
| Config 5 | Input + 3 layers + output | 10 | 10.000 | 0.0021 | 0.0021 |
| Config 6 | Input + 3 layers + output | 7 | 10.000 | 0.0023 | 0.0023 |
| Config 7 | Input + 3 layers + output | 5 | 10.000 | 6.0454e-04 | 6.0454e-04 |

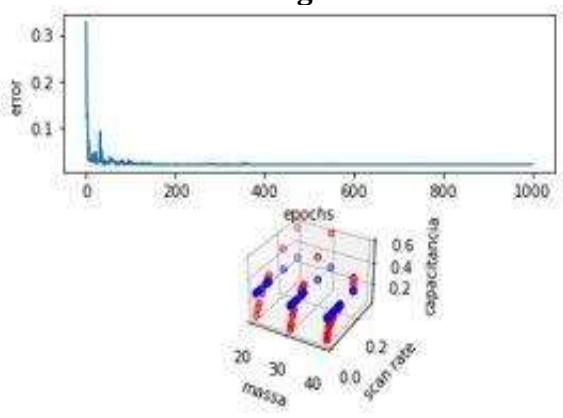
Fonte: Elaborado pelo autor.

Figura 33 - Gráficos do aprendizado em cada configuração ReLu (3D).

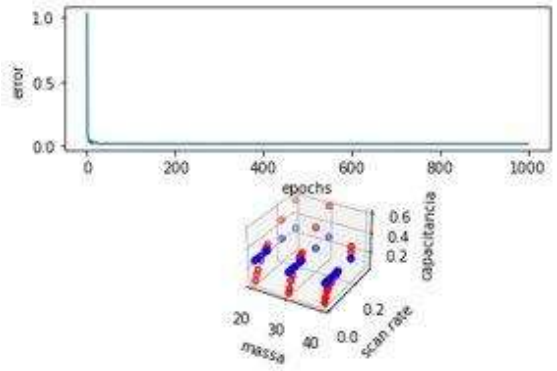
**Config 1**



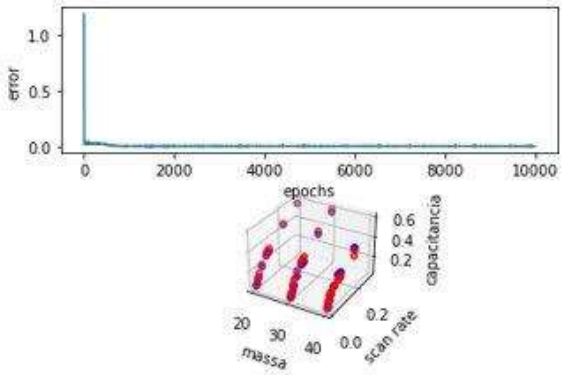

**Config 2**



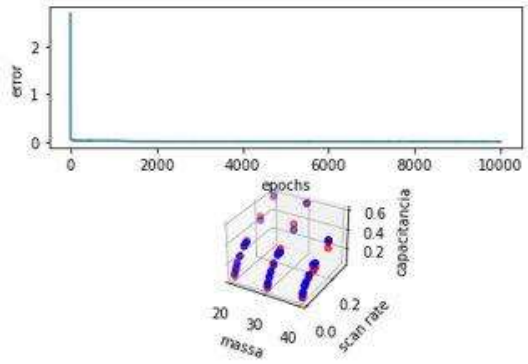
**Config 3**



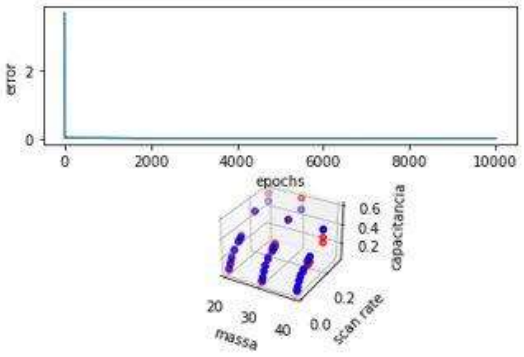

**Config 4**



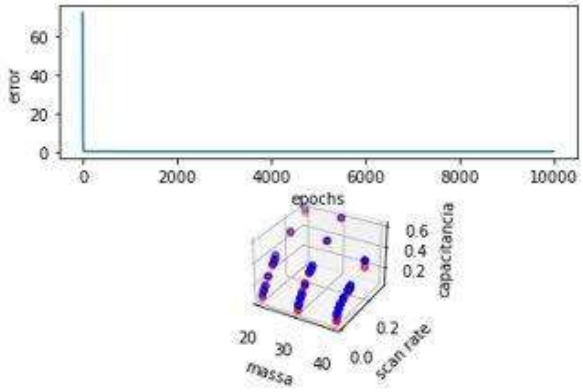

**Config 5**




**Config 6**




**Config 7**




Fonte: gráficos gerados pela inteligência artificial.

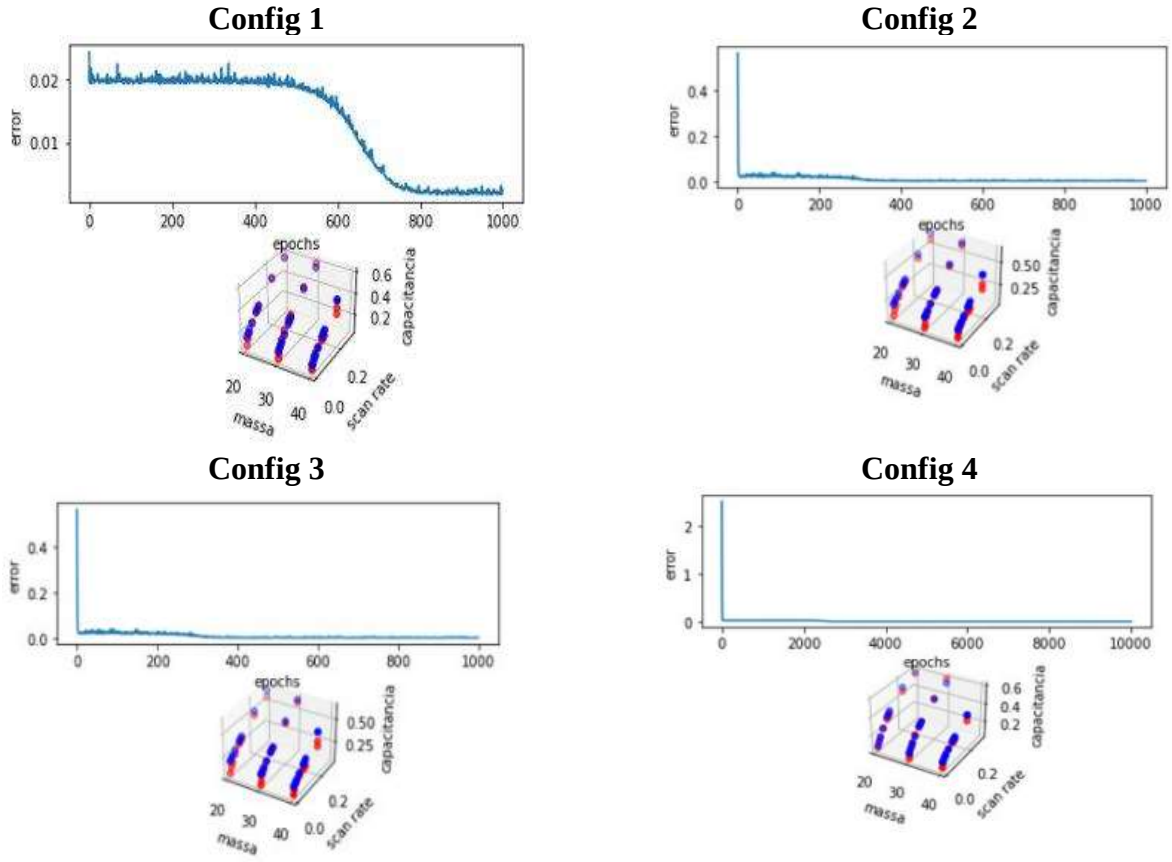
### 4.8.2 - Configuração G (3D) - Function Activation “Sigmoid”

Tabela 11 - Processo de aprendizagem para treinamento teste via algoritmo: *Sigmoid.*

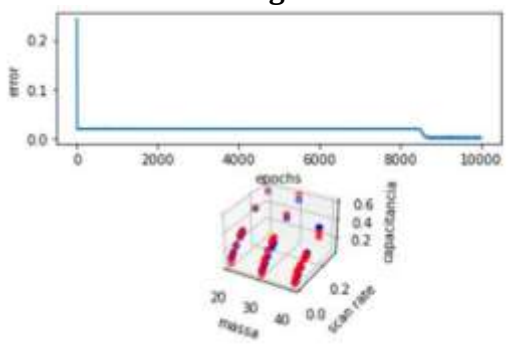
| TEST | Nº layers | Número de Neurônios | Épocas | Erro | MeanSquaredError |
|---|---|---|---|---|---|
| Config 1 | Input + 3 layers + output | 10 | 1.000 | 0.0025 | 0.0025 |
| Config 2 | Input + 3 layers + output | 100 | 1.000 | 0.0031 | 0.0031 |
| Config 3 | Input + 3 layers + output | 500 | 1.000 | 0.0031 | 0.0031 |
| Config 4 | Input + 3 layers + output | 20 | 10.000 | 8.1768e-04 | 8.1768e-04 |
| Config 5 | Input + 3 layers + output | 10 | 10.000 | 0.0019 | 0.0019 |
| Config 6 | Input + 3 layers + output | 7 | 10.000 | 0.0195 | 0.0195 |
| Config 7 | Input + 3 layers + output | 5 | 10.000 | 0.0195 | 0.0195 |

Fonte: Elaborado pelo autor.

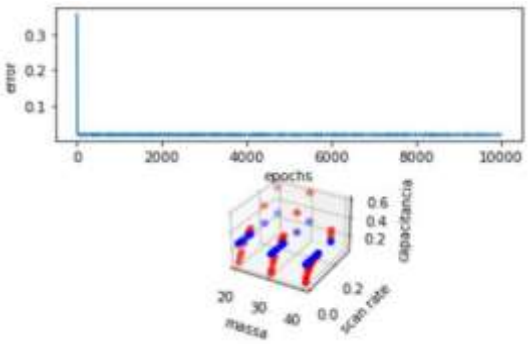
Figura 34 - Gráficos do aprendizado em cada configuração Sigmoid (3D).



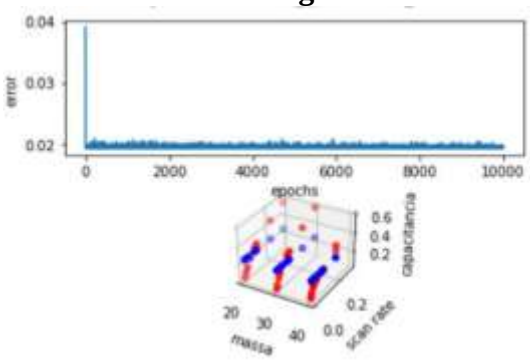
**Config 5**




**Config 6**




**Config 7**




Fonte: gráficos gerados pela inteligência artificial.

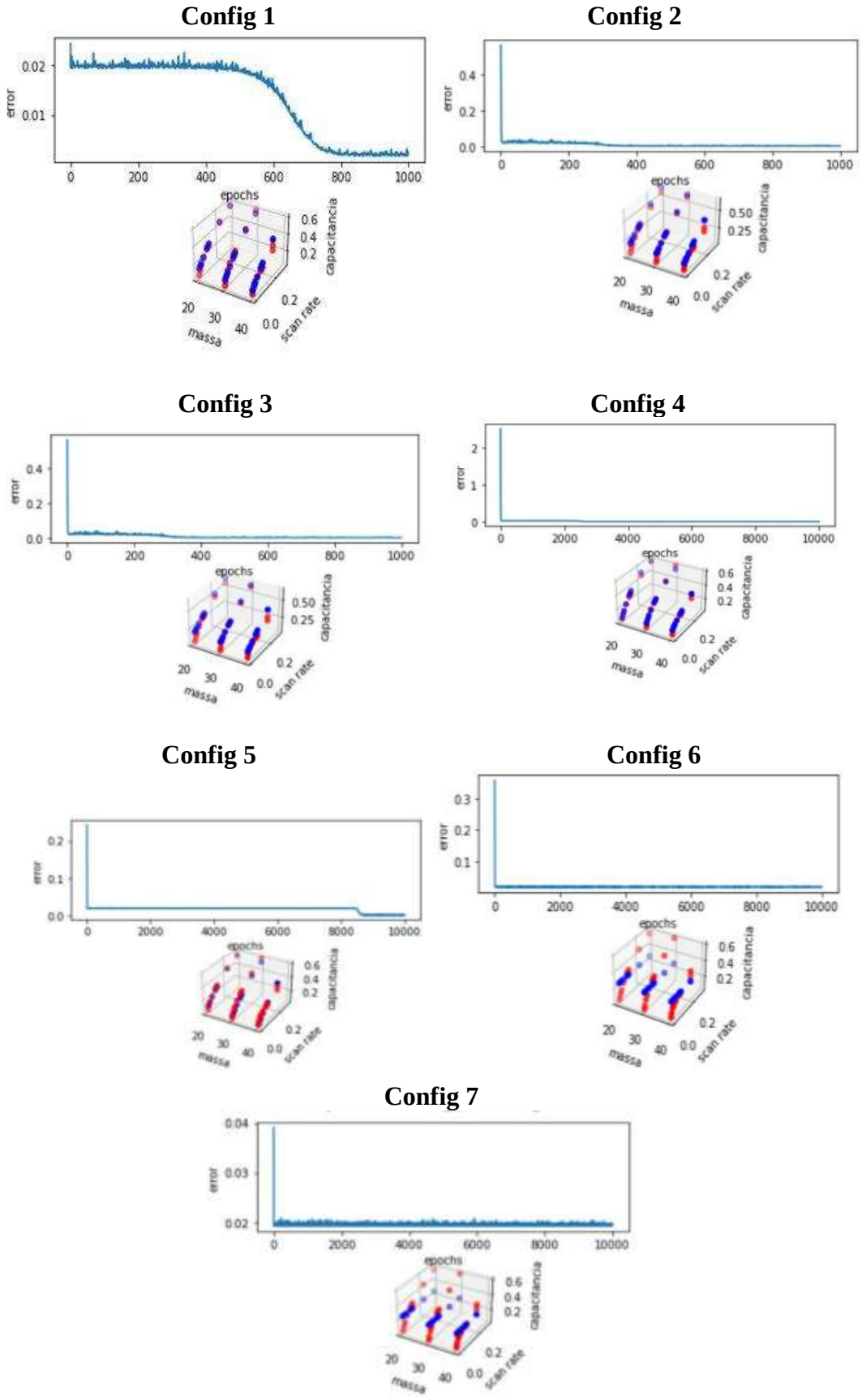
### 4.8.3 - Configuração H (3D) - *Function Activation "Softmax"*

Tabela 12 - Processo de aprendizagem para treinamento teste via algoritmo: *Softmax.*

| TEST | Nº layers | Número de Neurônios | Épocas | Erro | MeanSquaredError |
|---|---|---|---|---|---|
| Config 1 | Input + 3 layers + output | 10 | 1.000 | 0.0025 | 0.0025 |
| Config 2 | Input + 3 layers + output | 100 | 1.000 | 0.0031 | 0.0031 |
| Config 3 | Input + 3 layers + output | 500 | 1.000 | 0.0031 | 0.0031 |
| Config 4 | Input + 3 layers + output | 20 | 10.000 | 8.1768e-04 | 8.1768e-04 |
| Config 5 | Input + 3 layers + output | 10 | 10.000 | 0.0019 | 0.0019 |
| Config 6 | Input + 3 layers + output | 7 | 10.000 | 0.0195 | 0.0195 |
| Config 7 | Input + 3 layers + output | 5 | 10.000 | 0.0195 | 0.0195 |

Fonte: Elaborado pelo autor.

Figura 35 - Gráficos do aprendizado em cada configuração Softmax (3D).




Fonte: gráficos gerados pela inteligência artificial.

### 4.8.4 - Configuração I (3D) - *Function Activation "Softplus"*

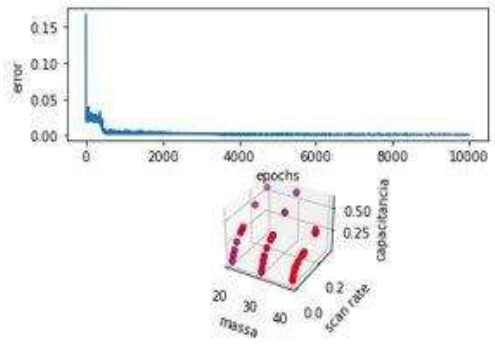
Tabela 13 – Processo de aprendizagem para treinamento teste via algoritmo: *Sofplus.*

| TEST | Nº layers | Número de neurônios | Épocas | Erro | MeanSquaredError |
|---|---|---|---|---|---|
| Config 1 | Input + 3 layers + output | 50 | 10000 | 2.9047e-04 | 2.9047e-04 |
| Config 2 | Input + 3 layers + output | 40 | 10000 | 1.5944e-04 | 1.5944e-04 |
| Config 3 | Input + 3 layers + output | 30 | 100000 | 9.1377e-05 | 9.1377e-05 |
| Config 4 | Input + 3 layers + output | 20 | 100000 | 5.7891e-05 | 5.7891e-05 |
| Config 5 | Input + 3 layers + output | 10 | 100000 | 7.7654e-04 | 7.7654e-04 |
| Config 6 | Input + 3 layers + output | 7 | 100000 | 6.2086e-05 | 6.2086e-05 |
| Config 7 | Input + 3 layers + output | 5 | 1000000 | 5.0909e-06 | 5.0909e-06 |

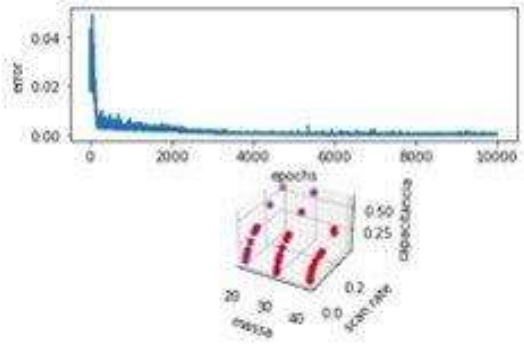
Fonte: Elaborado pelo autor.

Figura 36 - Gráficos do aprendizado em cada configuração Softplus (3D)

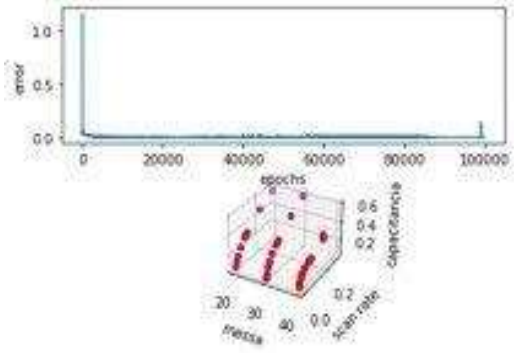
**Config 1**



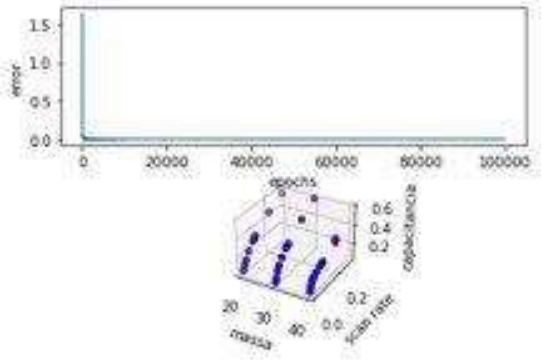

**Config 2**



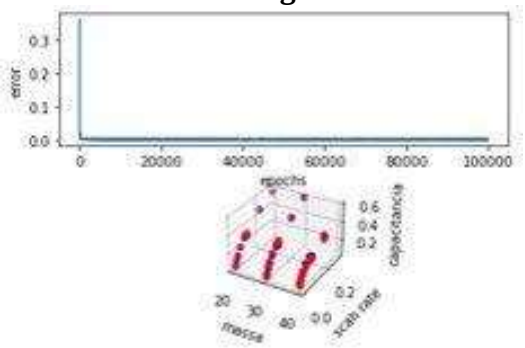

**Config 3**



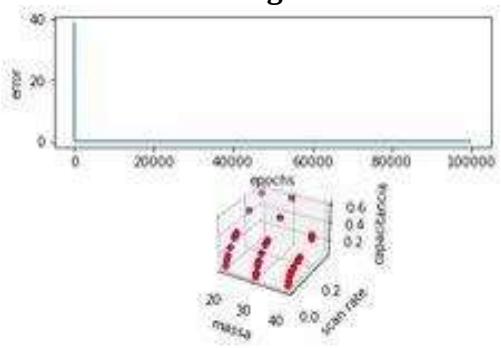

**Config 4**



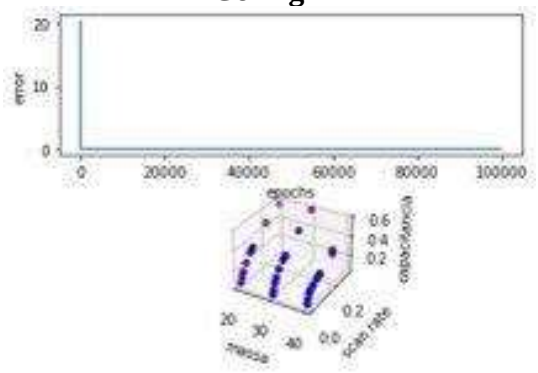
**Config 5**




**Config 6**




**Config 7**




Fonte: gráficos gerados pela inteligência artificial

.

## 4.9 - TESTES COM EXECUÇÃO EM LONGO PERÍODO (PRIMEIRA ETAPA)

Nossos testes de RNA's também se estenderam por um longo período de aprendizado usando a máquina da universidade usando apenas a função de ativação softplus. Essa máquina foi projetada especialmente pelo professor orientador, onde ela possuía uma memória RAM melhor e capaz de executar o algoritmo por um longo período sem nenhum problema. A estratégia apresentou bons resultados em situações de dependência a longo prazo. Entretanto, este modelo tem sua desvantagem quando se trata em tempo de processamento devido à sua complexidade, e, este processamento pode ser observado na tabela a seguir.

Tabela 14 - Processo de aprendizagem para treinamento longos com algoritmo: *Softplus.*

| TEST 1 | Função de Ativação | Nº layers | Número de neurônios | Épocas | Erro | MeanSquaredError |
|---|---|---|---|---|---|---|
| Config 1 | 'softplus' | Input + 3 layers + output | 7 | 100000 | 7.7654e-04 | 7.7654e-04 |
| Config 2 | 'softplus' | Input + 3 layers + output | 7 | 1.000.000 | 6.2086e-05 | 6.2086e-05 |
| Config 3 | 'softplus' | Input + 3 layers + output | 7 | 10.000.000 | 2.23549-06 | 2.23549-06 |

Fonte: Elaborado pelo autor.

Não é possível demonstrar o desempenho dos modelos em gráficos nesta etapa, pois o conjunto de dados possuía quatro dimensões, além do espaço em 3 dimensões. Porém, nesta seção, obtivemos os melhores resultados de minimização do erro e isso pode ser visualizado na tabela. Para analisarmos os testes desta performance foi fixado o número de neurônios em 7, por razões de tempo de execução, e utilizando um número bem alto de épocas consecutivas. Deste modo, pode ser acompanhado como o modelo se adaptou, e se poderia continuar o treinamento sem dar nenhum erro de execução.

## 4.10 – SEGUNDA ETAPA (TREINAMENTOS E TESTES USANDO O COLAB)

Para desenvolvermos a segunda etapa desta dissertação de Mestrado, foi utilizado o *Google Colab*, um ambiente online de livre acesso para qualquer navegador, basta ter internet. Com ele foi possível ter uma infinidade de bibliotecas disponíveis e gratuitas na montagem e otimização do código, como *Keras, Numpy, Pandas*, Scikit-learn, *Matplotlib e o TensorFlow*.

Usamos os mesmos conjuntos de dados que foram usados no ambiente anterior *Spyder*, porém com alterações e otimizações no código. Com isso, todos os nossos conjuntos de dados foram normalizados e reavaliados minuciosamente entre possíveis erros no arquivo final da IA. A melhoria nesta etapa foi significativa, uma vez que além de analisarmos graficamente as curvas específicas de massas, pressão, *scanrate* e capacitância, também foi otimizado a forma de treinamento e teste da nossa rede neural artificial. Nosso código em si, todos os nossos dados experimentais foram divididos em 70% dos dados para treinamento e 30% para os testes.

A técnica de normalização em que usamos neste trabalho foi a *MinMaxScaler*, cuja biblioteca é a *Scikit-learn*, no qual inserimos em nosso código em Python. Com essa técnica normalizamos os nossos dados para valores entre zero e um, para reduzir a complexidade do problema e tempo de treinamento do modelo. E ainda, por possuirmos os limiares extremos (superior e inferior) de todos os dados experimentais, utilizamos um algoritmo genético (AG) [125] que também consiste numa otimização para resolvermos problemas computacionais usando algoritmo genético. Assim, com todas essas informações pôde-se prever um conjunto de dados resultantes de capacitâncias e seus possíveis parâmetros físicos ideais de fabricação, tais como massa, pressão e *scanrate*.

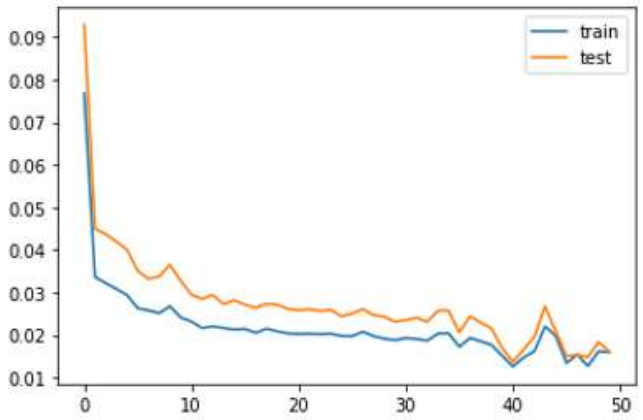
Nesta etapa usando *Colab*, todos os programas foram acoplados num único código e executados sequencialmente usando a automação do Pycaret, que é uma biblioteca que foi capaz de automatizar o *machine learning* nessa segunda etapa. Essa ferramenta nos auxiliou e

também acelerou exponencialmente o nosso ciclo de trabalho, tornando-o muito mais produtivo.

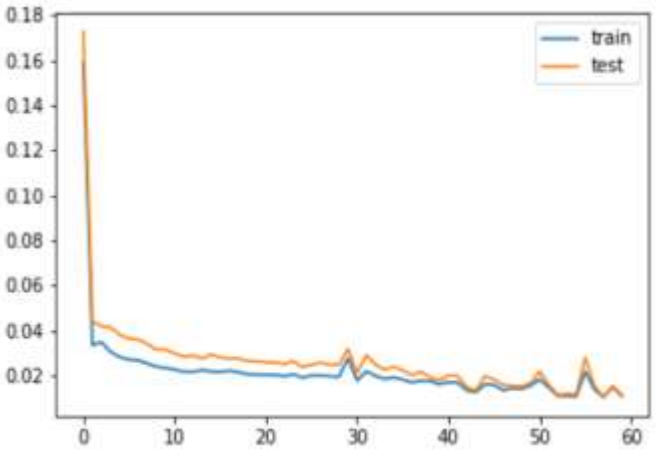
Dentro da nossa proposta principal de investigação e trabalho, que foi criar um software no qual entregássemos os conjuntos de dados experimentais, baseado nos parâmetros físicos de fabricação de supercapacitores. A princípio nosso modelo funcionou bem e obtivemos os seguintes gráficos a seguir para a curva de treinamento (linha azul) e teste (linha laranja) do algoritmo usando a função de ativação *ReLu.*

Figura 37 – Rede neural e gráficos de aprendizado em diferentes números de épocas

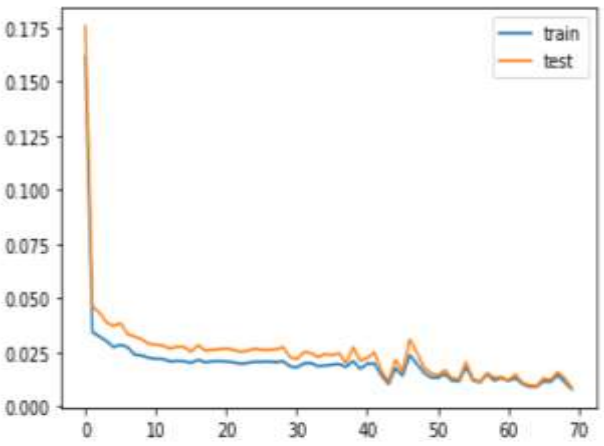
```
model.add(Dense(200, input_dim=3, kernel_initializer='normal', activation='relu'))
model.add(Dense(100, kernel_initializer='normal', activation='relu'))
model.add(Dense(50, kernel_initializer='normal', activation='relu'))
model.add(Dense(1, kernel_initializer='normal'))
```



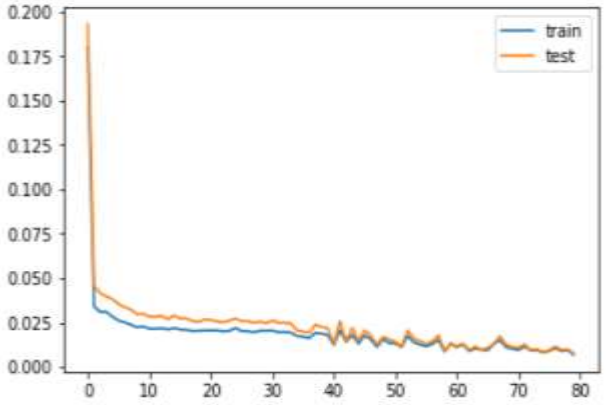

**Épocas: 50**




**Épocas: 60**




**Épocas: 70**




**Épocas: 80**

Fonte: gráficos gerados pela inteligência artificial.

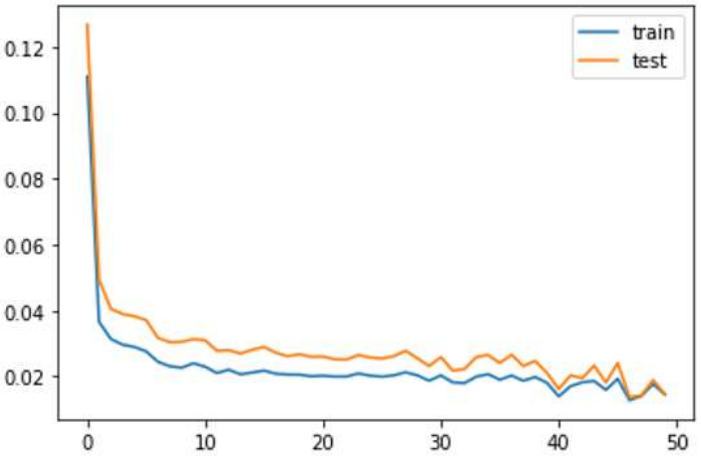
Essencialmente estes gráficos nos representam o aprendizado da rede neural em diferentes condições e configurações da rede, variando número de camadas, número de neurônios e principalmente a função de ativação para nossa aplicação de interesse.

Podemos observar que as linhas azuis e laranjas se interceptaram em torno de um intervalo entre 40 e 50 épocas, significando que essas curvas de treinamento e teste representam um aprendizado quando se cruzaram e permaneceram juntas constantemente. É notável ainda que o erro caiu drasticamente bem logo no início, bem antes da décima (10) épocas em todos os gráficos demonstrados, e, depois disso tais curvas foram se estabilizando até que se encontrassem e continuassem sempre juntas, como pode ser observado em todos os modelos com 50, 60, 70 e 80 épocas.

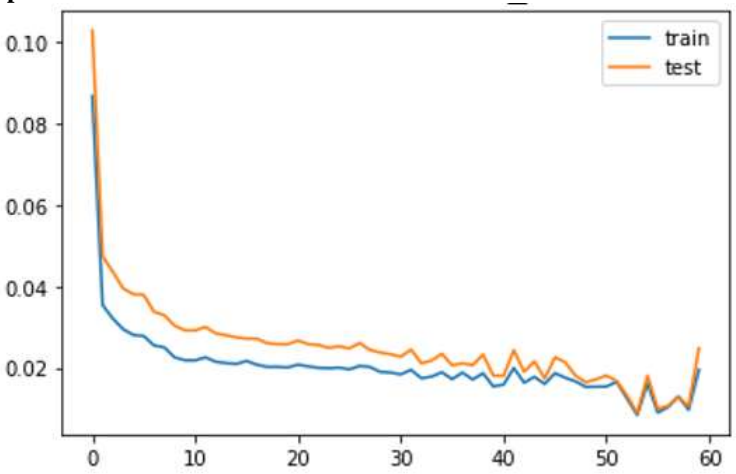
Os seguintes gráficos a seguir representam a estrutura de treinamento com mais camadas (linha azul) e teste (linha laranja) do algoritmo usando a função de ativação: *ReLu*

Figura 38 – Rede neural e gráficos de aprendizado em diferentes números de épocas

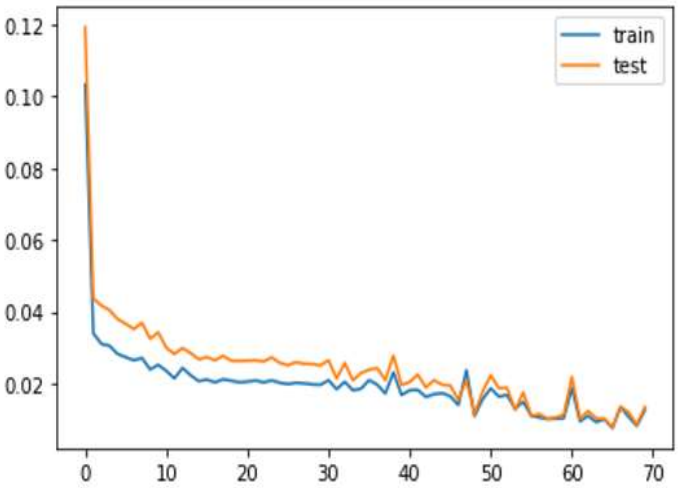
```
model.add(Dense(300, input_dim=3, kernel_initializer='normal', activation='relu'))
model.add(Dense(150, kernel_initializer='normal', activation='relu'))
model.add(Dense(75, kernel_initializer='normal', activation='relu'))
model.add(Dense(25, kernel_initializer='normal', activation='relu'))
model.add(Dense(1, kernel_initializer='normal'))
```



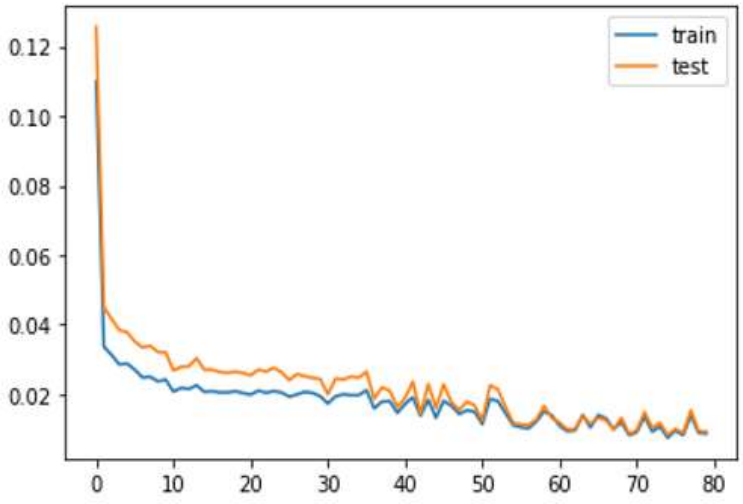

**Épocas: 50**




**Épocas: 60**




**Épocas: 70**



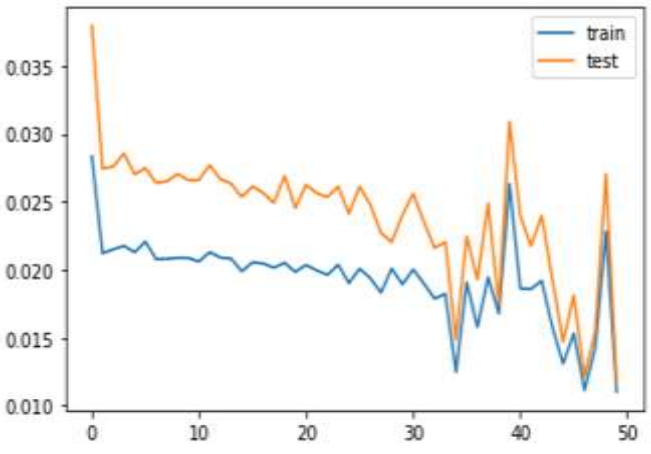

**Épocas: 80**
Fonte: gráficos gerados pela inteligência artificial.

Podemos observar nos gráficos que ao aumentarmos o número de camadas e mantermos a função de ativação *ReLu*, não tivemos tantas mudanças significativas. Somente no gráfico de 80 épocas, no qual tivemos que as curvas foram se estabilizando até que se encontrassem e continuaram sempre juntas, em torno de 35 a 40 épocas. Um resultado considerado interessante e bom, se comparado com a configuração anterior usando um modelo com menos camadas e menos neurônios.

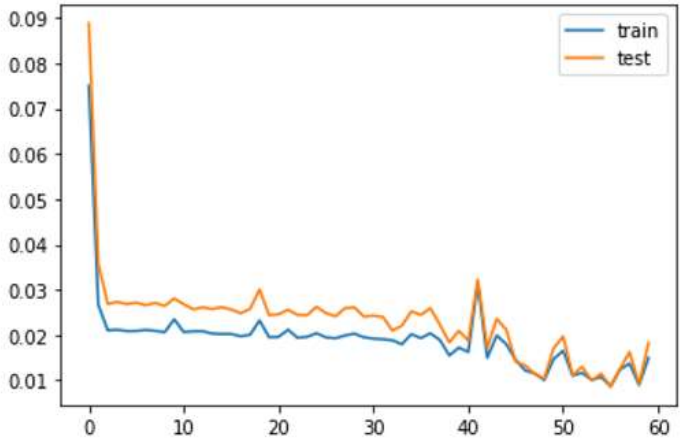
Os seguintes gráficos a seguir representam a estrutura de treinamento mesclada (linha azul) e teste (linha laranja) do algoritmo usando a função de ativação: *ReLu e Softplus.*

Figura 39 – Rede neural e gráficos de aprendizado em diferentes números de épocas

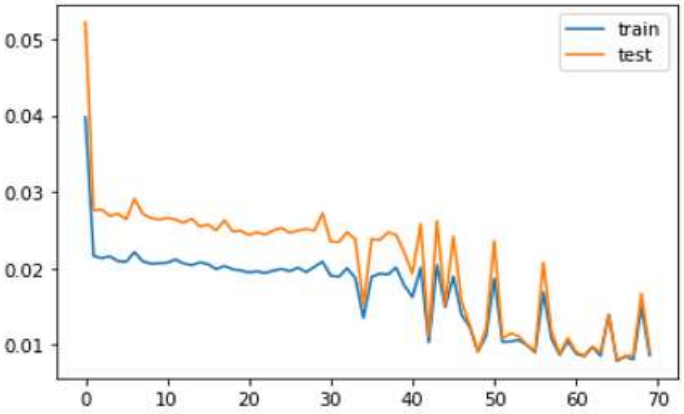
```
model.add(Dense(300, input_dim=3, kernel_initializer='normal', activation='relu'))
model.add(Dense(150, kernel_initializer='normal', activation='softplus'))
model.add(Dense(75, kernel_initializer='normal', activation='relu'))
model.add(Dense(25, kernel_initializer='normal', activation='softplus'))
model.add(Dense(1, kernel_initializer='normal'))
```



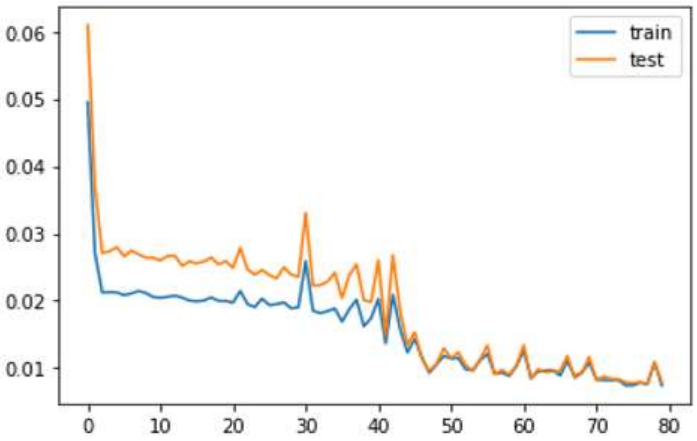

**Épocas: 50**




**Épocas: 60**




**Épocas: 70**




**Épocas: 80**

Fonte: gráficos gerados pela inteligência artificial.

Neste caso, os resultados da conversão das curvas de treinamento e testes nos demonstraram também um aprendizado muito significativo. No entanto a rede neural teve uma maior dificuldade de aprender. Isso pode ser observado através dos pequenos e grandes picos de ajustes em que a rede neural criou para se ajustar. Ou seja, estes picos também

podem representar as diferentes conexões entre os neurônios nas camadas ocultas até se ajustarem a uma diferença mínima possível, entre as curvas de treinamentos e as curvas de testes.

Podemos observar que ao mesclarmos as funções de ativação *ReLu e Softplus*, obtivemos diferenças substanciais e significativas, com relação aos outros resultados. Até que as curvas se estabilizaram e se encontrassem. Porém, a desvantagem é o tempo para que o erro e que o aprendizado aconteça, ele é maior. Neste modelo de configuração as curvas de treinamento e testes se ajustaram em torno de 50 ~ 60 épocas.

Em todas as curvas de aprendizado apresentadas, ocorreu um pré-processamento dos dados da rede neural, em que resumidamente é um processo em que os dados são inseridos na rede neural através da camada de entrada, e, estas por sua vez se comunicam com as camadas ocultas. Então este processamento acontece nessas camadas através de um sistema de conexões ponderadas. Os nós nas camadas ocultas combinam os dados da camada de entrada com um conjunto de coeficientes e atribui diferentes pesos para as entradas. Os resultados dessas entradas avaliadas são, então, somados. A soma passa pela função de ativação de um nó, que determina a extensão em que um sinal deve progredir na rede para afetar o resultado. Finalmente, as camadas ocultas ligam-se a camada de saída de onde os resultados são obtidos.

Os erros da (IA) nos treinamentos e testes da rede neural do nosso modelo, bem como os valores (coeficientes) obtidos de métricas de avaliação de *machine learning* são apresentados a seguir:

Figura 40 – Coeficientes de erros e métricas de (ML).

| **MÉTRICA** | **Coeficiente de Erro** |
|---|---|
| MAE | 0.0011 |
| MSE | 0.0079 |
| RMSE | $6.304.\ 10^{-5}$ |
| RMSLE | $3.378\ \ 10^{-5}$ |
| $R^2$ | 0.9977 |

Fonte: imagem gerada pela (IA).

Em todos os gráficos, as curvas de treinamentos e testes praticamente em todas as configurações alcançaram precisões ótimas. Sobretudo usando a função *ReLu*, somente, o erro é minimizado drasticamente, chegando a $6{,}304\ 10^{-5}$ na métrica RMSE, avaliado como o melhor resultado. Assim como em todos os coeficientes de erros, MAE, MSE, RMSLE e $R^2$ a acurácia das métricas e os pequenos valores dos coeficientes, nos demonstraram que ocorreu um aprendizado muito significativo neste modelo, como pode ser observado na Figura 41.

Sendo assim, estes resultados alcançaram precisão significativamente melhores do que para todas as métricas avaliadas em ambos os conjuntos de dados em toda as etapas de testes anteriores, comparando e executando a rede neural usando as funções *ReLu e Softplus.*

## 4.11 - PREDIÇÃO E MAXIMIZAÇÃO

Neste trabalho, realizamos simulações e previsões acerca da fabricação de supercapacitores usando diferentes intervalos de massas e pressões. E para os *scanrates* usamos o intervalo completo, ou seja, todos os valores de velocidade de varredura. O programa considerou todos os valores de massas, pressões e *scanrate* possíveis em cada intervalo solicitado à máquina. Isso se aplicou principalmente às variáveis do material (parâmetros físicos) de fabricação, tais como pressão e massa da matéria-prima. E por último, a máquina também previu a melhor capacitância em tal intervalo.

Logo abaixo, as simulações estão estruturadas na seguinte sequência de categorias:

- ($\boldsymbol{\Delta m_1}$) iniciados em 20 g até 40 g, categoria: **menor massa**;
- ($\boldsymbol{\Delta m_2}$) iniciados em 50 g até 100 g, categoria: **massa intermediária**;
- ($\boldsymbol{\Delta m_3}$) iniciados em 110 g até 140 g, categoria: **maior massa**.

Deste modo, simulamos as configurações dos intervalos de (**Δm**) intervalo de massas, intervalos de (**Δp**) pressões de compactação, e **(Δscan)** intervalo dos *scanrates* que é a velocidade de varredura e leitura da área superficial do material. Observe na tabela 15 que foram executadas em cada linha da tabela uma configuração de simulação diferente.

Tabela 15 - Configuração das previsões de fabricação de supercapacitores: **menor massa**

| ΔP Intervalo de pressões | Δm1 Intervalo de massas | Δscan velocidade de varredura | Pressão (kgf/cm²) | Massa (g) | Scanrate V/s | CAPACITÂNCIA / massa F/g |
|---|---|---|---|---|---|---|
| 15 ~ 30 kgf/cm² | 20 ~ 40 g | 0,005 ~ 0,3 | 15 | 20 | 0,3 | **0.00272257** |
| 30 ~ 40 kgf/cm² | 20 ~ 40 g | 0,005 ~ 0,3 | 30 | 20 | 0,3 | **0.00294355** |
| 40 ~ 50 kgf/cm² | 20 ~ 40 g | 0,005 ~ 0,3 | 40 | 20 | 0,3 | **0.00266493** |
| 50 ~ 60 kgf/cm² | 20 ~ 40 g | 0,005 ~ 0,3 | 50 | 20 | 0,3 | **0.00264478** |
| 60 ~ 70 kgf/cm² | 20 ~ 40 g | 0,005 ~ 0,3 | 60 | 20 | 0,3 | **0.00266704** |
| 70 ~ 80 kgf/cm² | 20 ~ 40 g | 0,005 ~ 0,3 | 72 | 20 | 0,3 | **0.00259619** |
| 80 ~ 90 kgf/cm² | 20 ~ 40 g | 0,005 ~ 0,3 | 80 | 21 | 0,3 | **0.00258798** |
| 90~ 100 kgf/cm² | 20 ~ 40 g | 0,005 ~ 0,3 | 90 | 20 | 0,3 | **0.00200804** |
| 100 ~ 110 kgf/cm² | 20 ~ 40 g | 0,005 ~ 0,3 | 100 | 20 | 0,3 | **0.00221077** |
| 110 ~ 120 kgf/cm² | 20 ~ 40 g | 0,005 ~ 0,3 | 110 | 20 | 0,3 | **0.00232313** |
| 120 ~ 140 kgf/cm² | 20 ~ 40 g | 0,005 ~ 0,3 | 120 | 20 | 0,3 | **0.00221226** |

Fonte: Elaborado pelo autor.

Como pode ser visto na tabela 15, todas as configurações retornam como resposta aos melhores valores e parâmetros de fabricação naquele determinado intervalo que solicitado à máquina. Tais valores como: a melhor pressão, a melhor massa e a maior capacitância (densidade de energia), respectivamente naquele determinado intervalo o qual foi solicitado.

Deste modo, cada amostra foi montada/simulada usando um intervalo de massa ($\Delta$m) aplicando em diferentes condições de pressão. Para as configurações da tabela 15 "menor massa" o programa previu que os valores de 20 g e 30 kgf/cm² sendo os melhores parâmetro de massa e pressão de compactação. Tais resultados representam relativamente o melhor nessa categoria ("menor massa"), prevendo uma capacitância de $2.943\ x\ 10^{-3}$ F/g.

Conforme pode ser visto abaixo na tabela 16 "massa intermediária", extraímos dessa categoria o nosso melhor valor de configurações de simulação deste trabalho. Obtendo assim o nosso objetivo, a maximização da eficiência energética de supercapacitores de grafeno. Para as simulações das configurações da tabela 16, o programa nos retornou valores de 80 g como o melhor parâmetro de massa e para a pressão de compactação, o melhor valor foi de 15 kgf/cm².

Tabela 16 - Configuração das previsões de fabricação de supercapacitores: **massa intermediária**

| ΔP Intervalo de pressões | Δm1 Intervalo de massas | Δscan velocidade de varredura | Pressão (kgf/cm²) | Massa (g) | Scanrate V/s | CAPACITÂNCIA / massa F/g |
|---|---|---|---|---|---|---|
| 15 ~ 30 kgf/cm² | 50 ~ 100 g | 0,005 ~ 0,3 | 15 | 80 | 0,3 | **0.0045536516** |
| 30 ~ 40 kgf/cm² | 50 ~ 100 g | 0,005 ~ 0,3 | 30 | 80 | 0,3 | **0.0044124676** |
| 40 ~ 50 kgf/cm² | 50 ~ 100 g | 0,005 ~ 0,3 | 43 | 85 | 0,3 | **0.0041220777** |
| 50 ~ 60 kgf/cm² | 50 ~ 100 g | 0,005 ~ 0,3 | 50 | 80 | 0.3 | **0.0041111719** |
| 60 ~ 70 kgf/cm² | 50 ~ 100 g | 0,005 ~ 0,3 | 60 | 80 | 0,3 | **0.0041102562** |
| 70 ~ 80 kgf/cm² | 50 ~ 100 g | 0,005 ~ 0,3 | 73 | 80 | 0,3 | **0.0041186947** |
| 80 ~ 90 kgf/cm² | 50 ~ 100 g | 0,005 ~ 0,3 | 80 | 80 | 0,3 | **0.0041128714** |
| 90~ 100 kgf/cm² | 50 ~ 100 g | 0,005 ~ 0,3 | 91 | 80 | 0,3 | **0.0041099513** |
| 100 ~ 110 kgf/cm² | 50 ~ 100 g | 0,005 ~ 0,3 | 100 | 80 | 0,3 | **0.0040869544** |
| 110 ~ 120 kgf/cm² | 50 ~ 100 g | 0,005 ~ 0,3 | 110 | 80 | 0,3 | **0.0041869497** |
| 120 ~ 140 kgf/cm² | 50 ~ 100 g | 0,005 ~ 0,3 | 120 | 80 | 0,3 | **0.0041798981** |

Fonte: Elaborado pelo autor.

Resultados os quais representaram para nós o mais alto valor de capacitância em todo o nosso trabalho, de $4.55\ x\ 10^{-3}$ F/g, um valor que se assemelha ao experimental do artigo principal [13] que é de $4.51\ x\ 10^{-3}$ F/g. O comportamento destes supercapacitores foram analisados em diferentes condições de simulação e montagem. Além disso, nessa categoria

houve melhorias significativas na capacitância. Pois, ao simularmos a configuração de todas essas amostras em um intervalo de massa e um intervalo de pressão de compactação, o programa foi capaz de fornecer uma capacitância otimizada, denominada a partir de agora como configuração de (SOE), Supercapacitor Otimizado Energeticamente.

Como pode ser observado na tabela, os efeitos dos parâmetros de massa na capacitância são bastante evidentes, com valores constantes preditos sempre em torno de 80 g, ao contrário da pressão de compactação que sempre tendeu aos menores valores no intervalo e por vezes valores aleatórios. Sobre as pressões da tabela 16 “massa intermediária", os resultados nos mostraram que a pressão de compactação não teve influência significativa nas magnitudes das densidades de energia (capacitância), uma vez que as mudanças quase não ocorreram na intensidade da capacitância. Nota-se então que a pressão não interfere tanto nas capacitâncias, pois a diferença dentre os valores extremos da tabela é de apenas 0,451181 F/g.

De acordo com a tabela 17 abaixo “maior massa”, em cada linha da tabela há um intervalo de massa de 110 g ~ 140 g em diferentes pressões. A capacitância diminui com o aumento da pressão usada para compactar a massa. Também pode ser observado que as massas mediante diferentes pressões de compactação se mantêm constante tendendo sempre a valores menos no intervalo, ou seja, de 110 g.

Tabela 17 - Configuração das previsões de fabricação de supercapacitores: **maior massa**

| ΔP Intervalo de pressões | Δm1 Intervalo de massas | Δscan velocidade de varredura | Pressão (kgf/cm²) | Massa (g) | Scanrate V/s | CAPACITÂNCIA / massa F/g |
|---|---|---|---|---|---|---|
| 15 ~ 30 kgf/cm² | 110 ~140 g | 0,005 ~ 0,3 | 20 | 110 | 0,3 | **0.00036497** |
| 30 ~ 40 kgf/cm² | 110 ~140 g | 0,005 ~ 0,3 | 30 | 110 | 0,3 | **0.00036341** |
| 40 ~ 50 kgf/cm² | 110 ~140 g | 0,005 ~ 0,3 | 44 | 110 | 0,3 | **0.00039754** |
| 50 ~ 60 kgf/cm² | 110 ~140 g | 0,005 ~ 0,3 | 51 | 110 | 0,3 | **0.00039626** |
| 60 ~ 70 kgf/cm² | 110 ~140 g | 0,005 ~ 0,3 | 64 | 110 | 0,3 | **0.00039087** |
| 70 ~ 80 kgf/cm² | 110 ~140 g | 0,005 ~ 0,3 | 70 | 110 | 0,3 | **0.00037993** |
| 80 ~ 90 kgf/cm² | 110 ~140 g | 0,005 ~ 0,3 | 81 | 110 | 0,3 | **0.00036859** |
| 90~ 100 kgf/cm² | 110 ~140 g | 0,005 ~ 0,3 | 98 | 110 | 0,3 | **0.00034759** |
| 100 ~ 110 kgf/cm² | 110 ~140 g | 0,005 ~ 0,3 | 100 | 110 | 0,3 | **0.00034513** |
| 110 ~ 120 kgf/cm² | 110 ~140 g | 0,005 ~ 0,3 | 110 | 110 | 0,3 | **0.00033336** |
| 120 ~ 140 kgf/cm² | 110 ~140 g | 0,005 ~ 0,3 | 120 | 110 | 0,3 | **0.00032302** |

Fonte: Elaborado pelo autor.

Resumindo, essa categoria comprimimos um intervalo de “maior massa” ($\Delta \boldsymbol{m}_3$) de 110g a 140g e variamos pressões sistematicamente como pode ser visto. Deste modo, foi observado que os valores de capacitância tiveram uma variação e diminuíram. Outro ponto importante, foram os valores encontrados de massa sempre de 110 g (menores valor). Em contrapartida, é notável que pressões acima de 70 kgf/cm² influenciaram nos valores de capacitâncias.

## 4.12 – DESVIO PADRÃO E VARIÂNCIA DA MELHOR EFICIÊNCIA ENERGÉTICA

Por fim, neste trabalho usamos o método do desvio padrão e este foi calculado para um número de cinco (05) amostras, formando um conjunto de valores de cinco execuções do programa de predição com a configuração máxima especificada nessa seção na tabela.

$$\text{Desvio da Média } |C - \bar{C}_M| \qquad Equação\ 19$$

$$\text{Variância } \sigma = \sum \left(\frac{C - \bar{C}_M}{n - 1}\right)^2 \qquad Equação\ 20$$

$$\text{Desvio Padrão} = \sqrt{\sigma} \qquad Equação\ 21$$

Tabela 18 - (SOE) Supercapacitor Otimizado Energeticamente

| CONFIGURAÇÃO MÁXIMA | | |
|---|---|---|
| **Pressão (kgf/cm²)** | **Massa (g)** | **Scanrate V/s** |
| 15 | 80 | 0,3 |

Fonte: Elaborado pelo autor.

Obtivemos primeiro o valor da capacitância média, que é simplesmente a soma aritmética das medidas dividido pelo número de amostras. Em seguida, foi calculado os desvios de cada uma das amostras do valor da média. Posteriormente foi calculado a variância, no qual se trata da soma de todos os valores de desvios elevando ao quadrado cada um dos valores de desvios e dividido pelo número de amostras menos 1. Por último, foi calculado a raiz quadrada da variância, no qual o valor obtido representa o nosso desvio padrão, que foi **1,26233e-4.**

Tabela 19 - Desvio Padrão na fabricação de supercapacitores: **SOE**

| C - Capacitância | Desvio $\lvert C - \overline{C}_M \rvert$ | Variância $\sigma = \sum \left(\frac{C - \overline{C}_M)^2}{n - 1}\right.$ | Desvio Padrão = $\sqrt{\sigma}$ |
|---|---|---|---|
| $n_1$ = 0,045536516 | 0,0011489164 | 1,32096e-8 | - |
| $n_2$ = 0,004224575 | 0,0021430246 | 4,59216e-8 | - |
| $n_3$ = 0,043653373 | 0,0007342266 | 5,39056e-9 | - |
| $n_4$ = 0,045356658 | 0,0009690584 | 9,39056e-9 | - |
| $n_5$ = 0,045546876 | 0,0007592764 | 5,76596e-9 | - |
| $\overline{C}_M$=0,0443875996 Capacitância média | | $\sigma$ = 1,59984e-8 Variância CALCULADA | Desvio Padrão 1,26233e-4 |

Fonte: Elaborado pelo autor.

## 4.13 - DISCUSSÃO DOS RESULTADOS

Os resultados apresentados neste capítulo foram divididos e executados em duas etapas, como citado anteriormente em outras seções. Os principais resultados, obtivemos no *Colab*, o software no computador do *Google*, onde tivemos acesso a um processador com dois núcleos, 12 GB de memória RAM e que pode ter acesso a uma TPU ou uma GPU.

Usando o *Colab*, os resultados foram muito melhores do que usando o outro computador que foram executados os primeiros testes das funções de ativação, que foi um computador pessoal comum com as especificações: i5-7400Hz CPU, 8GB de RAM e sistema operacional Windows 11. Neste computador comum usamos o ambiente *Spyder*. Também utilizamos uma máquina *Linux* com especificações desconhecidas localizada na universidade (UFTM) com acesso remoto.

Foram desenvolvidos e otimizados os códigos em *Python*, tanto da primeira etapa, quanto na segunda etapa deste trabalho. Simulamos o modelo de RNA (Rede Neural) para serem identificados os melhores resultados de funções de ativação e parâmetros físicos de fabricação de supercapacitores. Na qual essa discussão foi pautada na literatura e nos diversos dados experimentais alcançados com êxitos, além de uma série de resultados inesperados.

Sobre o modelo de RNA proposto neste trabalho tivemos que uma vez treinada adequadamente, pode ser utilizada para o nosso objetivo desse trabalho, que é maximizar a eficiência energética de supercapacitores de grafeno usando RNA (Redes Neurais Artificiais),

em diferentes sistemas e em diferentes condições de operação devido à sua capacidade de generalização.

No capítulo 3, mostramos que o método de RNA e regressão proposto, supera desafios e possíveis erros existentes na própria rede, devido à sua alta taxa de aprendizagem. Porém, em alguns momentos deste trabalho, nossa avaliação ficou restrita a erros de dados problemáticos do arquivo gerado pela inteligência artificial, sendo corrigida manualmente arquivo por arquivo. Tais problemas têm sido consistentemente abordados por trabalhos anteriores na literatura (GAO et al., 2017; ZHOU et al., 2018; PETRY et al., 2019).

### 4.13.1 – DISCUSSÃO DOS RESULTADOS DA PRIMEIRA ETAPA (*Spyder*)

Quanto aos nossos testes da (primeira etapa) as funções de ativação mais usadas pela comunidade científica, destaca-se que para menores quantidades de épocas, os modelos se adaptam melhor com duas funções de ativação em especial, sendo elas: *Softplus e Sigmoid*. No entanto, classificamos como melhor função a *sofltplus*, adotada de forma alternativa para o desenvolvimento do nosso sistema para reconhecimento de padrões neste trabalho.

Comparando o valor predito com o valor real para o modelo com função de ativação, em todas as figuras dos gráficos de aprendizado sendo eles em 1D, 2D e 3D, a linha azul, ou o ponto corresponde sempre ao valor real e a linha laranja, ou ponto sempre ao valor predito. Nossa abordagem pôde lidar com muitas dimensões, pois fomos adicionando o nosso conjunto de dados completo (arquivo.dat), novas colunas de informações adicionais que contribuíram para o aprendizado da máquina. A cada etapa deste trabalho fomos enriquecendo o nosso conjunto de dados inicial com informações como capacitâncias, *scanrate*, variação de pressões e variações de massas.

Definitivamente dos testes iniciais, limitamos o número de 50 até 800 épocas e não tivemos nenhum desempenho ruim no quesito minimização do erro, isso considerando as cinco variações de funções: *ReLu, Sigmoid e Softplus, Softmax e Tanh.* Observamos que uma função rendeu melhores resultados para a maioria das configurações do que outras funções, como pode ser observado em todos os gráficos, a curva predita laranja tem praticamente todas as configurações alcançadas com precisões ótimas. Sobretudo usando a função *ReLu*, somente na 4º configuração da (figura) o erro é minimizado drasticamente, chegando a $9{,}9157.e^{-6}$ com uma acurácia de $2.1815\,.e^{-4}$ , demonstrando-nos um aprendizado muito significativo.

Consideramos a função *Sigmoid*, a segunda melhor função em (Tabela  e Figura), todos os resultados apresentaram valores melhores que a função *ReLu*, e, somente na 5º configuração da (figura) o erro é minimizado a $8{,}9975.e^{-6}$ com uma acurácia de $2.1815\,.e^{-4}$

quando executadas um total de 800 épocas e 8 neurônios; Este resultado alcançou precisão, significativamente, melhor do que para todas as métricas avaliadas em ambos os conjuntos de dados, comparando as funções ReLu e Sigmoid.

A função *SoftPlus* possui um comportamento de função bem similar com a ReLU, e, segundo autores da literatura, tende a ter um funcionamento parecido da rede neural (Glorot et al, 2011). Entretanto, em comparação ao processamento dos nossos resultados experimentais, *Softplus* também se mostrou superior à *ReLu, Sigmoid, Softmax, e Tanh* para nossa aplicação específica (supercapacitores). Além disso, outras características essenciais na qual essa função se destacou e foi preferencialmente a escolhida, é porque, muitos cientistas e pesquisadores tendem a usar muito mais apenas funções de ativação como *ReLu* e *Sigmoid*, entre outras mais comuns na literatura. Por isso, consideramos neste trabalho ser um cenário ideal e mais atual usarmos funções alternativas em que o alcance de precisão é igual ou muito melhor em comparação às demais opções disponíveis para fins de testes de supercapacitores.

Sobre os cálculos de capacitâncias, foi observado que fisicamente para menores pressões de compactação e maiores massas utilizadas, foram encontrados os melhores resultados em Faraday (F) de densidade de energia (capacitâncias).

Similarmente, em relação a pressão, nos cálculos de capacitâncias deste trabalho é observada uma variação da densidade de energia entre os valores de menor e maior pressão, para uma densidade de energia praticamente invariável, se mostrando bastante similares com resultados obtidos experimentalmente do trabalho que forneceu os dados para tal trabalho [13].

Para a variação de massa os resultados se mostram ainda mais promissores, já que além da ampliação dos valores de capacitâncias (densidade de energia) com o aumento da massa utilizada houve um aumento da (capacitância) densidade de energia, mesmo a menor densidade de corrente aplicada.

Dos resultados dos cálculos de capacitância é possível visualizar a influência dos parâmetros físicos de pressão e massa na performance dos supercapacitores, que fica mais evidente nos gráficos de Ragone para as variações de pressão e massa, respectivamente.

### 4.13.2 – DISCUSSÃO DOS RESULTADOS DA SEGUNDA ETAPA (*Colab*)

Quanto aos nossos segundos testes (segunda etapa) usando o *Google Colab*, destaca-se que nesta etapa usamos os conjuntos de dados normalizados, menores quantidades de épocas, pouquíssimos neurônios. Realizamos a separação de treinamentos e teste, para evitar *overfitting* e com apenas duas funções de ativação em especial, sendo elas: *ReLu e Softplus*.

No entanto, classificamos como melhor função de ativação a *ReLu*, adotada para o desenvolvimento do aprendizado final da nossa rede neural artificial.

A partir disso, usamos o arquivo gerado pela (IA) no nosso sistema de previsões de capacitâncias e parâmetros físicos otimizados de fabricação de supercapacitores neste trabalho. Conjecturamos que esses resultados das funções estão relacionados com [1, 2, 3, 4, 5] principais trabalhos anteriores. Além disso, embora *ReLu* tenha sido a função de melhor desempenho do nosso método em todos os conjuntos de dados para aplicações como supercapacitores, outros novos testes precisam ser feitos para otimizações nas configurações no número de neurônios, épocas e etc.

Por fim, com base nas análises dos gráficos e resultados obtidos pela inteligência artificial, foi capaz de obtermos resultados podendo prever configurações otimizadas com o nosso método RNA. Sendo assim, o método foi consideravelmente eficiente, rápido e pode nos beneficiar no treinamento e inferência do modelo de *machine learning* para fabricar supercapacitores.

Deste modo, esperamos que este método tenha um desempenho ainda melhor em problemas com maiores conjuntos de dados experimentais fornecidos para a inteligência artificial aprender e prever. Isso porque quanto maior a quantidade dos atributos armazenados no arquivo (.dat), maior a precisão dos atributos numéricos de saída que podem ser afetados. De maneira simples, diferentes valores de entrada podem influenciar diretamente nos resultados de saída.

# 5 – CONCLUSÕES

Conclui-se que os resultados obtidos nesta dissertação a partir do modelo inteligente de RNA e os resultados experimentais apresentados na primeira e segunda etapa deste trabalho, nos mostraram um desempenho satisfatório do algoritmo proposto. As propriedades da rede em fazer uma previsão com base no aprendizado mostraram uma boa concordância entre os resultados experimentais dos supercapacitores [13] e os resultados gerados pelas RNA's deste trabalho científica.

Sobre os modelos de *machine learning* da (segunda etapa), obtivemos na maioria dos treinamentos e estes uma melhora significativa dos resultados e com intervalo de tempo muito pequenos. Fato este, que na (primeira etapa do trabalho) o tempo de execução chegou a ser até aproximadamente de 48h para o maior tempo de execução. Isso tudo dependente do número de camadas, épocas e de neurônios, onde tivemos na maior parte do tempo uma superioridade da função de ativação ReLu e Softplus. Sendo a melhor função na precisão do modelo e até mesmo para o tempo de execução. Com os modelos otimizados e treinados, conseguimos prover uma predição mais precisa. Deste modo, conclui-se que a função *ReLu* foi a função de melhor desempenho do nosso método em todos os conjuntos de dados para aplicações como supercapacitores. Contudo, outros testes precisam ser feitos, bem como novas otimizações nas configurações na rede, tais como: número de neurônios, épocas, pesos e sinapses que podem alterar os resultados.

Quanto à nossa previsão supervisionada em nosso problema de regressão, o código em *Python* conseguiu prever capacitâncias e parâmetros físicos com bastante facilidade, e, dependendo da configuração em um curto intervalo de tempo. Quando executando o diretório contendo o arquivo de dados (*.dat*) gerado pela inteligência artificial, logo após o término das execuções das épocas de aprendizado da rede neural. Dependendo dos parâmetros de desempenho do supercapacitor, eles são considerados para uma variedade de aplicações, como veículos elétricos híbridos, fontes de alimentação ininterrupta, proteção de memória de eletrônicos de computador e dispositivos celulares [2].

Dos modelos avaliados, comparamos o desempenho obtido variando entre as cinco funções de ativação selecionadas (Sigmoid, ReLU e Softplus, Softmax e Tanh) para cada uma das diferentes configurações. Tais configurações possuíram a idealização de alcançar um modelo, podendo aplicar modelos preditivos a fim de auxiliar numa melhor tomada de decisão. Dos resultados obtidos mostraram que praticamente todos os modelos treinados se adaptaram satisfatoriamente bem ao perfil, podendo concluir que a utilização da técnica RNA

teve efeito positivo neste trabalho, como foi eficaz em outros da literatura (Yu Bin Tan and Jong-Min Lee).

Sobre os cálculos de capacitâncias, foi observado que fisicamente para menores pressões de compactação e maiores massas utilizadas, foram encontrados os melhores resultados em Faraday (F) de densidade de energia (capacitâncias).

Sobre o material, destaca que o maior sucesso alcançado deste trabalho foi usando os conjuntos de dados do material grafeno. Devido às suas vantagens de uma alta capacitância específica, alta densidade de energia e ampla faixa de potencial de carga / descarga com baixa toxicidade, alta abundância natural, baixo custo, respeito ao meio ambiente e alta capacitância específica teórica.

Finalizando as considerações finais a respeito do desenvolvimento deste trabalho, pôde-se concluir que através de uma metodologia simples, foi possível desenvolver algoritmos capazes de obter uma alta taxa de aprendizagem e prever valores de capacitância e os parâmetros físicos na maximização de supercapacitores de grafeno.

Os aprendizados pessoais acerca da temática foram grandiosos, e categoricamente é muito interessante para a ciência dos materiais conseguir prever os melhores parâmetros e as configurações para fabricação de supercapacitores de grafeno, com base em informações e parâmetros previamente determinados.

De um ponto de vista do trabalho científico, concluo este trabalho com base nos resultados que o modelo de redes neurais pode ser uma ferramenta útil para outras tarefas importantes que afetam nossas vidas sociais diárias. Quanto aos aspectos sociais, permite uma melhor compreensão dos padrões digitais e tecnologias, bem como identificar padrões de configurações cada vez mais personalizadas e sustentável.

# REFERÊNCIAS

[1] Sérgio M. Rezende, Física de Materiais & Dispositivos Eletrônicos - 4ª Ed. 2015.

[2] Redes neurais artificiais. Antonio de Padua Braga, André Ponce de Leon E de Carvalho, Teresa Bernarda Ludermir, LTC - LIVROS TÉCNICOS E CIENTÍFICOS EDITORA S.A, 2000.

[3] Journal of Materials Chemistry A Graphene for supercapacitor applications Yu Bin Tan and Jong-Min Lee, School of Chemical and Biomedical Engineering, Nanyang Technological University, Singapore 637459, Singapore. This journal is The Royal Society of Chemistry 2013.

[4] Journal of Materials Chemistry, Supercapacitor Devices Based on Graphene Materials Y. Wang, Zhiqiang Shi, Yi Huang, Yanfeng Ma, Chengyang Wang, Mingming Chen, and Yongsheng Chen, School of Chemical Engineering and Technology, Tianjin UniVersity, Tianjin 300072, China, 2014.

[5] K. Zhang, L. Mao, L. L. Zhang, H. S. O. Chan, X. S. Zhao and J. Wu, Surfactant-intercalated, chemically reduced graphene oxide for high performance supercapacitor electrodes, J. Mater. Chem., 2011, 21, 7302.

[6] - K. Zhang, L. L. Zhang, X. S. Zhao and J. S. Wu, Graphene/ Polyaniline Nanobers Composites as Supercapacitor Electrodes, Chem. Mater., 2010, 22, 1392.

[7] A. Yu, I. Roes, A. Davies and Z. Chen, Ultrathin, transparent, and exible graphene lms for supercapacitor application, Appl. Phys. Lett., 2010, 96, 253105.

[8] J. Lin, J. Zhong, D. bao, J. Reiber-kyle, W. Wang, V. Vullev, M. Ozkan and C. S. Ozkan, Electrochemical supercapacitor based on exible pillar graphene.

[9] Development of nano fiber MnO2 thin film electrode and cyclic voltammetry behavior modeling using artificial neural network for supercapacitor application Autores: T.D. Dongale, P.R. Jadhav, G.J. Navathe, J.H. Kim c,nn, M.M. Karanjkar P.S. Patil.

[10] Brazilian Journal of Development, Lucas Fernandes Rocha Lago, Power demand forecasting on hybrid energy storage system in electric vehicles using Narx networks, DOI:10.34117/bjdv5n10-048, publicação: 04/10/2019.

[11] JOÃO PAULO CAMPOS TRIGUEIRO, Desenvolvimento de supercapacitores de alto desempenho baseados em eletrodos nanoestruturados e eletrólitos de líquidos iônicos, Tese.

[12] Baeza-Yates, Berthier. *Modern Information Retrieval* , 2008, 2 ed.

[13] AUGUSTO, Gabriel de Souza, Estudo e desenvolvimento de supercapacitores eletroquímicos usando eletrodo baseados em multicamadas de grafeno. Dissertação - Universidade Federal do Triangulo Mineiro, Uberaba, 2018.

[14] Economic Analysis and Policy, M. Horn, J. MacLeod, M. Liu, J. Webb, N. Motta, Supercapacitors: A new source of power for electric cars?,, V. 1, 2019, p. 93-103, https://doi.org/10.1016/j.eap.2018.08.003.

[15] Chemical Physics Letters, F. Zhang, J. Tang, N. Shinya, Lu-Chang Qin, Hybrid graphene electrodes for supercapacitors of high energy density, 2013, p. 124-129, https://doi.org/10.1016/j.cplett.2013.08.021.

[16] Yu, L., Zhang, W., Wang, J., & Yu, Y. (2017). SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient. *Proceedings of the AAAI Conference on Artificial Intelligence*, *31*(1). Retrieved from https://ojs.aaai.org/index.php/AAAI/article/view/10804

[17] E. Danila, G. Livint and D. D. Lucache, "Dynamic modelling of supercapacitor using artificial neural network technique," *2014 International Conference and Exposition on Electrical and Power Engineering (EPE)*, 2014, pp. 642-645, doi: 10.1109/ICEPE.2014.6969988.

[18] ACS Applied Energy Materials, Kian Keat Lee, Tamara L. Church, and Niklas Hedin, RNA as a Precursor to N-Doped Activated Carbon, 2018 *1* (8), 3815-3825, DOI: 10.1021/acsaem.8b00589

[19] Computers in Biology and Medicine, Yuwen Sun, Allen C. Cheng, Machine learning on-a-chip: A high-performance low-power reusable neuron architecture for artificial neural networks in ECG classifications, 2012, p. 751-757, ISSN 0010-4825.

[20] McQuisten KA, Peek AS (2009) Comparing Artificial Neural Networks, General Linear Models and Support Vector Machines in Building Predictive Models for Small Interfering RNAs. PLoS ONE 4(10): e7522. https://doi.org/10.1371/journal.pone.0007522

[21] C. A. Villacorta Cardoso and G. Lima Cruz, "Forecasting Natural Gas Consumption using ARIMA Models and Artificial Neural Networks," in *IEEE Latin America Transactions*, vol. 14, no. 5, pp. 2233-2238, May 2016, doi: 10.1109/TLA.2016.7530418.

[22] A. S. Brandao and D. C. Jorge, "Artificial Neural Networks Applied to Image Steganography," in IEEE Latin America Transactions, vol. 14, no. 3, pp. 1361-1366, March 2016, doi: 10.1109/TLA.2016.7459621.

[23] D. Valle-Cruz, E. A. Gomez, R. S. Almazan, J. Ignacio Proceedings of the 20th Annual International Conference, A Review of Artificial Intelligence in Government and its Potential from a Public Policy Perspective June 2019 Pages 91–99, doi.org/10.1145/3325112.3325242

[24] Jahanzaib Shabbir, Tarique Anwer Artificial Intelligence (cs.AI); Computer Vision and Pattern Recognition (cs.CV)Cite as:arXiv:1804.01396 [cs.AI] (or arXiv:1804.01396v1 [cs.AI] for this version) https://doi.org/10.48550/arXiv.1804.01396

[25] Mohammad Hossein Jarrahi, Artificial intelligence and the future of work: Human-AI symbiosis in organizational decision making, Business Horizons,Volume 61, Issue 4, 2018,Pages 577-586,ISSN 0007-6813, https://doi.org/10.1016/j.bushor.2018.03.007.

[26] Jean-Charles Pomerol, Artificial intelligence and human decision making, European Journal of Operational Research, Volume 99, Issue 1, 1997, Pages 3-25, ISSN 0377-2217, https://doi.org/10.1016/S0377-2217(96)00378-5.

[27] Y. Ma, Z. Wang, H. Yang and L. Yang, "Artificial intelligence applications in the development of autonomous vehicles: a survey," in *IEEE/CAA Journal of Automatica Sinica*, vol. 7, no. 2, pp. 315-329, March 2020, doi: 10.1109/JAS.2020.1003021.

[28] Hokey Min (2010) Artificial intelligence in supply chain management: theory and applications, International Journal of Logistics Research and Applications, 13:1, 13-39, DOI: 10.1080/13675560902736537

[29] Rohit Sharma, Anjali Shishodia, Angappa Gunasekaran, Hokey Min, Ziaul Haque Munim. (2022) The role of artificial intelligence in supply chain management: mapping the territory. *International Journal of Production Research* 0:0, pages 1-24.

[30] Frank Bodendorf, Philipp Merkl, Jörg Franke. (2021) Artificial neural networks for intelligent cost estimation – a contribution to strategic cost management in the manufacturing supply chain. *International Journal of Production Research* 0:0, pages 1-22.

[31] Marc A. Anderson, Ana L. Cudero, Jesus Palma, Capacitive deionization as an electrochemical means of saving energy and delivering clean water. Comparison to present desalination practices: Will it compete?, Electrochimica Acta, Volume 55, Issue 12, 2010, Pages 3845-3856, ISSN 0013-4686, https://doi.org/10.1016/j.electacta.2010.02.012.

[32] Majid Monajjemi, Cell membrane causes the lipid bilayers to behave as variable capacitors: A resonance with self-induction of helical proteins, Biophysical Chemistry, Volume 207, 2015, Pages 114-127, ISSN 0301-4622, https://doi.org/10.1016/j.bpc.2015.10.003.

[33] B.E. Conway, V. Birss, J. Wojtowicz, The role and utilization of pseudocapacitance for energy storage by supercapacitors, Journal of Power Sources, Volume 66, Issues 1–2, 1997, Pages 1-14, ISSN 0378-7753, https://doi.org/10.1016/S0378-7753(96)02474-3.

[34] Electrochemical supercapacitors for energy storage and delivery : fundamentals and applications / Aiping Yu … [et al.]. p. cm. -- (Green chemistry and chemical engineering) Includes bibliographical references and index. ISBN 978-1-4398-6989-5.

[35] Jitendra Tahalyani, M. Jaleel Akhtar, Jayesh Cherusseri Characteristics of Capacitor: Fundamental Aspects Handbook of Nanocomposite Supercapacitor Materials I, 2020, Volume 300 ISBN : 978-3-030-43008-5

[36] Electric Potential and Electric Field, Capacitors and Dielectrics, Cap. 19, Disponível em: https://pressbooks.bccampus.ca/physics0312chooge/chapter/19-5-capacitors-and-dielectrics/ Acessado em 16/04/2022 às 15h29.

[37] Brazilian Journal. Phys. 29, José A. Giacometti, Sergei FedosovMauro M. Costa, Corona charging of polymers: recent advances on constant current charging, June 1999 https://doi.org/10.1590/S0103-97331999000200009

[38] A. R. Frederickson and J. R. Dennison, "Measurement of conductivity and charge storage in insulators related to spacecraft charging," in *IEEE Transactions on Nuclear Science*, vol. 50, no. 6, pp. 2284-2291, Dec. 2003, doi: 10.1109/TNS.2003.821397.

[39] Tzu-Ho Wu, Chun-Tsung Hsu, Chi-Chang Hu, Laurence J. Hardwick, Important parameters affecting the cell voltage of aqueous electrical double-layer capacitors, Journal of Power Sources, Volume 242, 2013, Pages 289-298, ISSN 0378-7753, https://doi.org/10.1016/j.jpowsour.2013.05.080.

[40] Atsushi Nishino, Capacitors: operating principles, current market and technical trends, Journal of Power Sources, Volume 60, Issue 2, 1996, Pages 137-147, ISSN 0378-7753, https://doi.org/10.1016/S0378-7753(96)80003-6.

[41] Black, J., & Andreas, H. A. (2009). Effects of charge redistribution on self-discharge of electrochemical capacitors. *Electrochimica Acta, 54*(13), 3568-3574.

[42] LINZEN, Dirk et al. Analysis and evaluation of charge-balancing circuits on performance, reliability, and lifetime of supercapacitor systems. IEEE transactions on industry applications, v. 41, n. 5, p. 1135-1141, 2005.

[43] GUALOUS, Hamid et al. Supercapacitor thermal modeling and characterization in transient state for industrial applications. IEEE Transactions on industry applications, v. 45, n. 3, p. 1035-1044, 2009.

[44] HADARTZ, Martin; JULANDER, Martin. Battery-supercapacitor energy storage. Gothenburg, Sweden: Chalmers University of Technology, 2008.

[45] FRIVALDSKY, Michal; CUNTALA, Jozef; SPANIK, Pavol. Simple and accurate thermal simulation model of supercapacitor suitable for development of module solutions. International journal of thermal sciences, v. 84, p. 34-47, 2014.

[46] XIE, Binghe et al. Shape-tailorable graphene-based ultra-high-rate supercapacitor for wearable electronics. ACS nano, v. 9, n. 6, p. 5636-5645, 2015.

[47] Tan Yu Bin; LEE, Jong-Min. Graphene for supercapacitor applications. Journal of Materials Chemistry A, v. 1, n. 47, p. 14814-14843, 2013.

[48] IRO, Zaharaddeen S. et al. A brief review on electrode materials for supercapacitor. Int. J. Electrochem. Sci, v. 11, n. 12, p. 10628-10643, 2016.

[49] DAVIES, Aaron; YU, Aiping. Material advancements in supercapacitors: from activated carbon to carbon nanotube and graphene. The Canadian Journal of Chemical Engineering, v. 89, n. 6, p. 1342-1357, 2011.

[50] YANG, Wen et al. Graphene in supercapacitor applications. Current Opinion in Colloid & Interface Science, v. 20, n. 5-6, p. 416-428, 2015.

[51] CANAL-RODRÍGUEZ, María et al. Graphene-doped carbon xerogel combining high electrical conductivity and surface area for optimized aqueous supercapacitors. Carbon, v. 118, p. 291-298, 2017.

[52] Zhai, Rui, et al. "Construction of NiCo2S4 heterostructure based on electrochemically exfoliated graphene for high-performance hybrid supercapacitor electrode." *Journal of Alloys and Compounds* 845 (2020): 156164.

[53] TANG, Xiao-Ning et al. Graphene aerogel derived by purification-free graphite oxide for high performance supercapacitor electrodes. Carbon, v. 146, p. 147-154, 2019.

[54] XU, Yanfei et al. Screen-printable thin film supercapacitor device utilizing graphene/polyaniline inks. Advanced Energy Materials, v. 3, n. 8, p. 1035-1040, 2013.

[55] WU, Zhong-Shuai; FENG, Xinliang; CHENG, Hui-Ming. Recent advances in graphene-based planar micro-supercapacitors for on-chip energy storage. National Science Review, v. 1, n. 2, p. 277-292, 2014.

[56] CARVALHO, Wilhiam C.; BATAGLIOLI, Rodrigo P.; COURY, Denis V. Comparison of supercapacitor storage system control methods for wind power smoothing. In: 2018 Simposio Brasileiro de Sistemas Eletricos (SBSE). IEEE, 2018. p. 1-6.

[57] PARWAIZ, Shaikh et al. Machine-learning-based cyclic voltammetry behavior model for supercapacitance of Co-doped ceria/rGO nanocomposite. Journal of Chemical Information and Modeling, v. 58, n. 12, p. 2517-2527, 2018.

[58] ZHOU, Musen et al. Insights from machine learning of carbon electrodes for electric double layer capacitors. Carbon, v. 157, p. 147-152, 2020.

[59] AKKOUCHI, Kamel; RAHMANI, Lazhar; LEBIED, Ryma. New application of artificial neural network-based direct power control for permanent magnet synchronous generator. Electrical Engineering & Electromechanics,(6), p. 18-24, 2021.

[60] KAMATH, R. S. et al. Modeling mice Down syndrome through protein expression: An artificial neural network based approach. Journal of Pharmacy Research, v. 11, n. 11, p. 1300, 2017.

[61] MATHEW, Seema et al. An optimization of hybrid capacitor with respect to mass of electrode material. In: 2016 IEEE Students' Conference on Electrical, Electronics and Computer Science (SCEECS). IEEE, 2016. p. 1-4.

[62] PANDEY, Poonam et al. KELM-CPPpred: kernel extreme learning machine based prediction model for cell-penetrating peptides. Journal of proteome research, v. 17, n. 9, p. 3214-3222, 2018.

[63] RAMIREZ-VAZQUEZ, Isaias; RAMIREZ-GONZALEZ, Miguel; SALGADO-TALAVERA, J. Eduardo. Neural Network Model to Estimate Resistivity of Ground Enhancers Reinforced with Graphene Nano Particles for Transmission Lines. In: Journal of Nano Research. Trans Tech Publications Ltd, 2019. p. 139-150.

[64] MARIE-FRANÇOISE, Jean-Noël. Contribution à la commande neuronale et à la gestion d'énergie d'un système hybride batterie-supercondensateurs: application aux transports terrestres. 2004. Tese de Doutorado. Besançon.

[65] ASHER, Zachary D. et al. Enabling Prediction for Optimal Fuel Economy Vehicle Control. 2018.

[66] TAO, Jili; ZHANG, Ridong. Intelligent Feature Selection Using GA and Neural Network Optimization for Real-Time Driving Pattern Recognition. IEEE Transactions on Intelligent Transportation Systems, 2021.

[67] KUBÁŇ, Vojtěch et al. Quantitative conformational analysis of functionally important electrostatic interactions in the intrinsically disordered region of delta subunit of bacterial RNA polymerase. Journal of the American Chemical Society, v. 141, n. 42, p. 16817-16828, 2019.

[68] CHEN, Zhu et al. Advances in fully automated nucleic acid extraction system based on magnetic nanobeads. Nanoscience and Nanotechnology Letters, v. 11, n. 12, p. 1633-1643, 2019.

[69] DE VOLDER, Michael FL et al. Carbon nanotubes: present and future commercial applications. science, v. 339, n. 6119, p. 535-539, 2013.

[70] DERAKHSHANFAR, Soroosh. Application of high performance hydrogels in tissue engineering and supercapacitors. 2018. Dissertação de Mestrado.

[71] LIU, Jixin et al. Hydrodynamic numerical simulation and prediction of bionic fish based on computational fluid dynamics and multilayer perceptron. Engineering Applications of Computational Fluid Mechanics, v. 16, n. 1, p. 858-878, 2022.

[72] JI, Jian et al. Optimum scheme selection for multilayer perceptron-based Monte Carlo simulation of slope system reliability. International Journal of Geomechanics, v. 21, n. 10, p. 06021025, 2021.

[73] André, Augusto and Ferreira, and Pomilio, José and Silva, Ennio and Vaz, Diego and Cambra, Pontes, Metodologia para dimensionar múltiplas fontes de suprimento de energia de veículos elétricos, 2022.

[74] MUNI KUMAR, N.; MANJULA, R. Design of multi-layer perceptron for the diagnosis of diabetes mellitus using keras in deep learning. In: Smart intelligent computing and applications. Springer, Singapore, 2019. p. 703-711.

[75] FAROOQ, Furqan et al. Predictive modeling for sustainable high-performance concrete from industrial wastes: A comparison and optimization of models using ensemble learners. Journal of Cleaner Production, v. 292, p. 126032, 2021.

[76] CHARISOPOULOS, Vasileios; MARAGOS, Petros. Morphological perceptrons: geometry and training algorithms. In: International Symposium on Mathematical Morphology and Its Applications to Signal and Image Processing. Springer, Cham, 2017. p. 3-15.

[77] CAR, Zlatan et al. Modeling the spread of COVID-19 infection using a multilayer perceptron. Computational and mathematical methods in medicine, v. 2020, 2020.

[78] ERIKSSON, Maria et al. Spotify teardown: Inside the black box of streaming music. Mit Press, 2019.

[79] APELLA, Ignacio; ROFMAN, Rafael; ROVNER, Helena. Skills and the Labor Market in a New Era: Managing the Impacts of Population Aging and Technological Change in Uruguay. World Bank Publications, 2020.

[80] AOUINI, Zied. Traffic monitoring in home networks: from theory to practice. 2017. Tese de Doutorado. Université de La Rochelle.

[81] PALACIOS NEFFKE, Jens C. Contribution to the development of technology-enhanced education in manufacturing and energy generation. 2017.

[82] HAO, Jiangang; HO, Tin Kam. Machine learning made easy: a review of scikit-learn package in python programming language. Journal of Educational and Behavioral Statistics, v. 44, n. 3, p. 348-361, 2019.

[83] DANKWA, S.; YANG, L. Securing IoT Devices: A Robust and Efficient Deep Learning with a Mixed Batch Adversarial Generation Process for CAPTCHA Security Verification. Electronics 2021, 10, 1798. 2021.

[84] CIELEN, Davy; MEYSMAN, Arno. Introducing data science: big data, machine learning, and more, using Python tools. Simon and Schuster, 2016.

[85] THOBHANI, Alaa et al. CAPTCHA Recognition Using Deep Learning with Attached Binary Images. Electronics, v. 9, n. 9, p. 1522, 2020.

[86] RUPPERT, David. The elements of statistical learning: data mining, inference, and prediction. 2004.

[87] TANG, Chengliang; YUAN, Gan; ZHENG, Tian. Weakly Supervised Learning Creates a Fusion of Modeling Cultures. Observational Studies, v. 7, n. 1, p. 203-211, 2021.

[88] PANG, Bo; NIJKAMP, Erik; WU, Ying Nian. Deep learning with tensorflow: A review. Journal of Educational and Behavioral Statistics, v. 45, n. 2, p. 227-248, 2020.

[89] SHAZEER, Noam et al. Mesh-tensorflow: Deep learning for supercomputers. Advances in neural information processing systems, v. 31, 2018.

[90] RASCHKA, Sebastian; MIRJALILI, Vahid. Python machine learning: Machine learning and deep learning with Python, scikit-learn, and TensorFlow 2. Packt Publishing Ltd, 2019.

[91] WEISTRAND, Ola; SVENSSON, Stina. The ANACONDA algorithm for deformable image registration in radiotherapy. Medical physics, v. 42, n. 1, p. 40-53, 2015.

[92] NIKITINA, Oleg; LUKYANOVAA, Olga. Accelerating Deep Learning for Shared Facility Centers Using Tensorflow Framework Analysis Based on IBM POWER Platform. 2020.

[93] KADIYALA, Akhil; KUMAR, Ashok. Applications of Python to evaluate environmental data science problems. Environmental Progress & Sustainable Energy, v. 36, n. 6, p. 1580-1586, 2017.

[94] LAKATOS, Eva Maria. Marina de Andrade Marconi. Fundamentos de metodologia científica, v. 5, 2008.

[95 ]ZOU, Jinming; HAN, Yi; SO, Sung-Sau. Overview of artificial neural networks. Artificial Neural Networks, p. 14-22, 2008.

[96] CAMPOS L.; GARCIA, J. C. Redes neurais apoiando a tomada de decisão na análise de crédito bancário e detecção do câncer de mama, v19i1.1451 doi.org/10.20397/2177-6652/2019.

[97] YOO, Hyun Deog et al. On the challenge of developing advanced technologies for electrochemical energy storage and conversion. Materials Today, v. 17, n. 3, p. 110-121, 2014.

[98] IOSSEL, Yu Y.; KOCHANOV, E. S.; STRUNSKII, M. G. The calculation of electrical capacitance. AIR FORCE SYSTEMS COMMAND WRIGHT-PATTERSON AFB OH FOREIGN TECHNOLOGY DIVISION, 1971.

[99] BIELA, Juergen; KOLAR, Johann W. Using transformer parasitics for resonant converters-a review of the calculation of the stray capacitance of transformers. In: Fourtieth IAS Annual Meeting. Conference Record of the 2005 Industry Applications Conference, 2005. IEEE, 2005. p. 1868-1875.

[100] ZULLO, J. R.; ARRUDA, F. B. Programa computacional para ajuste de equação em dados experimentais. IAC, Campinas. 23 p. Boletim Técnico, v. 113, 1986.

[101] HANSELMAN, Duane; LITTLEFIELD, Bruce. Matlab. 1997.

[102] SIGMON, Kermit. MATLAB Primer. 1989.

[103] BARAHA, Satyakam; BISWAL, Pradyut Kumar. Implementation of activation functions for ELM based classifiers. In: 2017 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET). IEEE, 2017. p. 1038-1042.

[104] LACHER, R. Christopher et al. A neural network for classifying the financial health of a firm. European Journal of Operational Research, v. 85, n. 1, p. 53-65, 1995.

[105] XU, Xing et al. Toward effective intrusion detection using log-cosh conditional variational autoencoder. IEEE Internet of Things Journal, v. 8, n. 8, p. 6187-6196, 2020.

[106] HORNIK, Kurt. Approximation capabilities of multilayer feedforward networks. Neural networks, v. 4, n. 2, p. 251-257, 1991.

[107] HORNIK, Kurt; STINCHCOMBE, Maxwell; WHITE, Halbert. Multilayer feedforward networks are universal approximators. Neural networks, v. 2, n. 5, p. 359-366, 1989.

[108] SKOUNDRIANOS, Elias N.; TZAFESTAS, Spyros G. Finding fault-fault diagnosis on the wheels of a mobile robot using local model neural networks. IEEE Robotics & Automation Magazine, v. 11, n. 3, p. 83-90, 2004.

[109] MA, Liying; KHORASANI, Khashayar. Constructive feedforward neural networks using Hermite polynomial activation functions. IEEE Transactions on Neural Networks, v. 16, n. 4, p. 821-833, 2005.

[110] DA S GOMES, Gecynalda S.; LUDERMIR, Teresa B.; LIMA, Leyla MMR. Comparison of new activation functions in neural network for forecasting financial time series. Neural Computing and Applications, v. 20, n. 3, p. 417-439, 2011.

[111] ODA, Masaya et al. Schottky barrier diodes of corundum-structured gallium oxide showing on-resistance of 0.1 mΩ· cm2 grown by MIST EPITAXY®. Applied Physics Express, v. 9, n. 2, p. 021101, 2016.

[112] KHRABROV, Alexy; CYBENKO, George. Discovering influence in communication networks using dynamic graph analysis. In: 2010 IEEE Second International Conference on Social Computing. IEEE, 2010. p. 288-294.

[113] Activation Functions in Neural NetworksSigmoid, tanh, Softmax, ReLU, Leaky ReLU EXPLAINED, Disponível em: https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6]Acessado em: 16/04/2022 às 15h00.

[114] Funções de ativação: definição, características, e quando usar, Disponível em: https://iaexpert.academy/2020/05/25/funcoes-de-ativacao-definicao-caracteristicas-e-quando-usar-cada-uma/ Acessado em: 16/04/2022 às 15h10.

[115] DUGAS, M. J. et al. Le Questionnaire sur l'Inquiétude et l'Anxiété. Validation dans des échantillons non cliniques et cliniques. Journal de thérapie comportementale et cognitive, 2001.

[116] GLOROT, Xavier; BORDES, Antoine; BENGIO, Yoshua. Domain adaptation for large-scale sentiment classification: A deep learning approach. In: ICML. 2011.

[117] GOMES, Albino et al. Redes Neurais Artificiais Aplicadas na Predição de Algoritmos de Escalonamento em Sistema LTE.

[118] DUARTE, Felipe Machado. Acurácia de previsões para vazão em redes: um comparativo entre ARIMA, GARCH e RNA. 2014. Dissertação de Mestrado. Universidade Federal de Pernambuco.

[119] MONICO, João Francisco Galera et al. Acurácia e precisão: revendo os conceitos de forma acurada. Boletim de Ciências Geodésicas, v. 15, n. 3, p. 469-483, 2009.

[120] Accuracy and Precision, measures of *observational error*. disponível em: http://en.wikipedia.org/wiki/Accuracy_and_precision. Acesso em: 16/04/2022 às 15h33.

[121] DOS SANTOS, José Ricardo Vieira Silva; MOURÃO, Luciana. Impacto do treinamento como variável preditora da satisfação com o trabalho. Revista de Administração, v. 46, n. 3, p. 305-318, 2011.

[122] MONARD, Maria Carolina; BARANAUSKAS, José Augusto. Conceitos sobre Machine Learning . Sistemas inteligentes-Fundamentos e aplicações, v. 1, n. 1, p. 32, 2003.

[123] DRUCKER, Peter F. Knowledge-worker productivity: The biggest challenge. California management review, v. 41, n. 2, p. 79-94, 1999.

[124] CHEN, Chunhua et al. Early inhibition of HIF-1α with small interfering RNA reduces ischemic–reperfused brain injury in rats. Neurobiology of disease, v. 33, n. 3, p. 509-517, 2009.

[125] KOZA, J.R. (1992). Genetic Programming. On the Programming of Computers by Means of Natural Selection. [S.l.]: MIT

[126] GOLDBERG, David E. (1989). Genetic Algorithms in Search, Optimization, and Machine Learning. EUA: Addison-Wesley. 0-201-15767-5

[127] TURING, Alan M.; HAUGELAND, J. Computing machinery and intelligence. The Turing Test: Verbal Behavior as the Hallmark of Intelligence, p. 29-56, 1950.

[128] DEMARIA, Marco et al. An essential role for senescent cells in optimal wound healing through secretion of PDGF-AA. Developmental cell, v. 31, n. 6, p. 722-733, 2014.

[127] GOOGLE, INC. Google Colab. Disponível em: https://colab.research.google.com/ Acesso em: setembro de 2022.

[128] KERAS, INC. Api Keras. Disponível em : https://keras.io/api/ Acesso em setembro de 2022.

[129] Harris, CR, Millman, KJ, van der Walt, SJ et al. Programação de matrizes com NumPy . Natureza 585, 357–362 (2020). DOI: 10.1038/s41586-020-2649-2 . ( link da editora ).

[130] The pandas development team, pandas-dev/pandas: Pandas, feb, 2020, doi: 10.5281/zenodo.3509134, https://doi.org/10.5281/zenodo.3509134.

[131] Scikit-learn: aprendizado de máquina em Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.

[132] J. D. Hunter, "Matplotlib: A 2D Graphics Environment", Computing in Science & Engineering, vol. 9, no. 3, pp. 90-95, 2007.

[133] Pycaret.org. PyCaret, April 2020. URL https://pycaret.org/about PyCaret version 1.0.

[134] PYTHON SOFTWARE FOUNDATION. Python Language Site: Documentation, 2022. Página de documentação. Disponível em: https://www.python.org/doc/ Acesso em: 06 de Jan. de 2020.