

**UNIVERSIDADE FEDERAL DE SÃO CARLOS**  
**CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA**  
**DEPARTAMENTO DE ENGENHARIA DE MATERIAIS**

**IMPLEMENTAÇÃO DE REDE NEURAL NA  
CARACTERIZAÇÃO DE IPMC's COM LÍQUIDOS IÔNICOS  
DE ALQUILIMIDAZÓLIO INCORPORADOS: ESTUDO DE  
VIABILIDADE DE FERRAMENTA AUXILIAR**

**LIA KIM RODRIGUES**

**SÃO CARLOS -SP**  
**2021**

**IMPLEMENTAÇÃO DE REDE NEURAL NA  
CARACTERIZAÇÃO DE IPMC's COM LÍQUIDOS IÔNICOS  
DE ALQUILIMIDAZÓLIO INCORPORADOS: estudo de  
viabilidade de ferramenta auxiliar**

Trabalho de conclusão de curso apresentado ao Departamento de Engenharia de Materiais da Universidade Federal de São Carlos, como requisito para obtenção do título de bacharel em Engenharia de Materiais.

Orientador: Prof. Dr. Carlos H.  
Scuracchio

São Carlos-SP  
2021



## ATA DE DEFESA DE TRABALHO DE CONCLUSÃO DE CURSO (TCC)

**NOME:** Lia Kim Rodrigues

**RA:** 727395

**TÍTULO:** Aplicação de rede neural na caracterização de IPMC's com líquidos iônicos de alquilimidazólio incorporados: estudo de viabilidade de ferramenta

**ORIENTADOR(A):** Prof. Dr. Carlos Henrique Scuracchio

**CO-ORIENTADOR(A):**

**DATA/HORÁRIO:** 18/11/2021, 14h

### BANCA – NOTAS:

	Monografia	Defesa
Prof. Dr. Carlos Henrique Scuracchio	9,0	9,5
Prof. Dr. Ariel Gustavo Zuquello	9,0	9,5
<b>Média</b>	9,0	9,5

Certifico que a defesa de monografia de TCC realizou-se com a participação a distância dos membros Prof. Dr. Carlos Henrique Scuracchio e Prof. Dr. Ariel Gustavo Zuquello e depois das arguições e deliberações realizadas, os participantes à distância estão de acordo com as informações redigidas nesta ata de defesa.

Prof. Dr. Carlos Henrique Scuracchio

## **DEDICATÓRIA**

Dedico este trabalho à minha família que me forneceu sempre todo suporte e amor necessários para percorrer essa trajetória.

## **AGRADECIMENTO**

A conclusão deste trabalho e, assim, desta etapa que é a graduação foi possível devido ao apoio de várias pessoas durante esses últimos anos. Dessa forma, gostaria de agradecer:

Aos meus pais, irmão e avós por sempre acreditarem e me darem toda a base necessária para trilhar meu caminho.

Ao meu professor orientador que me orientou mostrando sempre o melhor caminho e as ferramentas necessárias para a elaboração deste e outros projetos.

Aos meus professores do curso que me transmitiram o conhecimento e as experiências para formarem uma profissional capacitada.

Aos meus amigos que fizeram o dia a dia dentro e fora da sala de aula ser mais leve e divertido.

## RESUMO

O presente trabalho de conclusão de curso teve por objetivo a implementação e o estudo da viabilidade de um algoritmo de rede neural em linguagem de programação Python para abordar um problema de classificação de amostras de IPMC. Para este trabalho foram considerados IPMC's (*Ionic Polymer-Metal Composites*) de Nafion®-117 – polímero eletroativo do tipo iônico – com platina depositada – camada de material condutor. Uma das aplicações do IPMC é sua utilização como atuadores ou sensores, pois apresentam movimentos suaves similares aos músculos biológicos devido à migração de cátions solvatados dentro dos canais ionoméricos do Nafion quando uma tensão é aplicada [12, 13, 14].

A partir de estudos e caracterizações eletromecânicas, observou-se que a resposta do dispositivo varia com condições externas – como a temperatura e umidade relativa [14] – de forma que a perda de água para o ambiente durante os ciclos de atuação implica na redução da efetividade do atuador, sendo este um dos desafios para o uso dos IPMCs. Assim, uma possível solução encontrada para que os dispositivos atuem com maiores capacidades por tempos mais longos é a incorporação dos chamados líquidos iônicos, neste caso foi utilizado o 1-butil-3-metilimidazólio (BMIM.Cl).

Além das caracterizações eletromecânicas em que o uso de BMIM.Cl se mostrou bastante promissor, também foram realizadas caracterizações viscoelásticas do dispositivo por meio de ensaios dinâmico-mecânicos por torção em geometrias retangulares. Dessa maneira, os dados utilizados na camada de entrada da rede neural para treinamento e teste foram os valores do módulo de armazenamento ( $G'$ ) em função da frequência, previamente obtidos em ensaios de torção realizados em um reômetro ARES da TA Instruments.

A metodologia empregada foi a adaptação do código de rede neural disponível publicamente no tutorial de Machine Learning para programação em Python [28]. A rede programada é formada por três camadas, duas ocultas e uma de saída, densamente conectadas. Após treinamento e teste da rede neural o programa fornece, então, a acurácia da rede em porcentagem.

Inicialmente, fazendo-se uso das funções de ativação “relu” e “sigmoid”, a acurácia medida foi de 16-19%. Após o estudo da influência das funções de ativação no desempenho da rede e a alteração das mesmas nas camadas intermediárias e de

saída foi obtida uma acurácia com valores superiores a 95%. Além disso, a partir do grupo de testes, o qual foi gerado de forma aleatória utilizando a função *random* disponível das bibliotecas importadas, foram adicionados deslocamentos nos eixos x e y criando dados com ruídos para simular análises de uma mesma amostra com variabilidade e, assim, avaliar a robustez da rede em classificar corretamente as amostras conforme o aumento da intensidade dos ruídos. Como era esperado, observou-se a queda no nível de acurácia ao passo que a intensidade de ruído associada cresce.

Assim, embora o problema abordado seja simples e tenha uma pouca quantidade de dados, como resultado preliminar, a implementação do algoritmo de rede neural se mostrou uma ferramenta viável com alto nível de acurácia para auxiliar na caracterização de amostras de IPMCs e de Náfion® desde que a faixa de ruído em relação aos dados experimentais fique dentro de um determinado limite. É de grande relevância ressaltar a importância de uma quantidade de dados significativa para resultados mais conclusivos.

**Palavras-chave:** Rede Neural. IPMC. Líquido Iônico.

## ABSTRACT

The present study aims to implement and evaluate the feasibility of an artificial neural network algorithm in Python to classify IPMC samples. For this work it was considered Nafion-based IPMC samples with platinum electrodes. One of the IPMC's applications is its use as soft actuators and sensors due to its smooth movements similar to biological muscles. This bending mechanism occurs because of the cationic diffusion inside Náfion®'s ionomeric channels dragging solvent molecules causing a difference in the solvent concentration along the sample thickness [12, 13, 14].

Based on studies and electromechanical characterizations, it was observed that the hydration level and external conditions influence in the actuator performance [14]. The loss of water to the environment during the actuation cycles implies a drop in the IPMC effectiveness which is a challenge for commercial applications. The incorporation of the so-called ionic liquids, in this case was used the 1-butyl-3-methylimidazole chloride (BMIM.Cl), may be one possible solution so the IPMC works with its higher capacity for longer periods.

Besides the electromechanical characterizations in which the incorporation of BMIM.Cl proved to be very promising, dynamic mechanical oscillatory shear measurements with torsional rectangular geometry were also carried out in order to study the IPMC's viscoelastic response. Therefore, the storage modulus ( $G'$ ) as a function of frequency ( $\omega$ ), previously obtained in torsion tests performed in ARES rheometer from TA Instruments, were used as input data in the neural network.

The methodology used was the adaptation of the of the publicly available neural network code in the Machine Learning tutorial for Python programming [28]. The neural network has three layers, two hidden and one output layer, densely connected. Thus, after the training and validation of the artificial neural network, the algorithm gives us the accuracy.

Firstly, using the *relu* and *sigmoid* activation functions, the accuracy measured was between 16% and 19%. After the study of the influence of the activation functions and its change in the hidden layers and in the output layer it was obtained a accuracy higher than 95%. Furthermore, random noisy data were generated in order to simulate different analysis of the same sample containing a variability. This simulated was used to evaluate the accuracy with the increase of the noise intensity. The algorithm showed that the accuracy in the capacity of rightly classify the IPMC samples drops with higher intensities noises.

So, although the problem herein approached was very simple and the quantity of data available was relatively small, from preliminary results, the implementation of the neural network algorithm showed to be a viable tool with high accuracy level to assist in the characterization of IPMC samples as long as the noise range in relation to the experimental data is within a certain limit. It is of great relevance to emphasize the importance of a significant amount of data to get more conclusive results.

**Keyword:** Neural Network. IPMC. Ionic Liquid.

## LISTA DE ILUSTRAÇÕES

- Figura 1 – Gráfico adaptado indicando número de publicações por ano abordando aprendizado de máquina na Engenharia e Ciência dos Materiais. [2] **3**
- Figura 2 – Fluxograma simplificado de um modelo supervisionado de ML adaptado de [5]. **3**
- Figura 3 – Modelo simplificado de um perceptron em que o conjunto  $X_1 - X_n$  é o vetor de entrada que será multiplicado por pesos ( $w_i$ ) – vetor de pesos – gerando um valor por meio de uma soma ponderada que será a variável da função de ativação  $\phi$  que finalmente irá retornar uma saída  $y_k$  a qual será a entrada para a próxima camada de perceptrons. **5**
- Figura 4 – Exemplo simplificado da arquitetura de uma RNA com camadas ocultas [5]. **5**
- Figura 5 – Imagem ilustrativa de polímeros eletroativos do tipo iônico e suas áreas de aplicação [12]. **7**
- Figura 6 – (a) Estrutura do Náfion® – N117 com parte hidrofóbica (laranja) e hidrofílica (azul) destacadas e (b) morfologia mostrando os canais ionoméricos contendo o cátion móvel [15]. **8**
- Figura 7 – Ilustração da deformação do IPMC ao longo do tempo de atuação [22]. **9**
- Figura 8 – Estrutura química do anel imidazólio [23] **10**
- Figura 9 – Estrutura química do líquido iônico 1-butil-3-metilimidazólio (BMIM.Cl). **10**
- Figura 10 – Ilustração de uma amostra com geometria de prisma sob torção [27] **12**
- Figura 11 – Curvas do módulo  $G'$  pela frequência obtidas de um ensaio de torção em geometria retangular para uma amostra de SBR e variação de  $G'$  com o comprimento da amostra [27] **12**
- Figura 12 – Gráfico do módulo de armazenamento,  $G'$  [Pa], por frequência, [rad/s], para as amostras de IPMC. **14**
- Figura 13 – Gráfico do módulo de armazenamento,  $G'$  [Pa], por frequência,  $\omega$  [rad/s], para as amostras de Náfion®. **15**
- Figura 14 – Diagrama ilustrativo da arquitetura da rede neural considerada para este trabalho. **16**
- Figura 15 – Gráfico ilustrativo indicando os valores de entrada utilizados para treino

em azul e, para teste, em vermelho.	<b>16</b>
Figura 16 – (a) Curva da função ReLu. (b) Curva da função sigmóide. Adaptado de [32].	<b>18</b>
Figura 17 – Princípio ilustrativo de funcionamento da função de ativação Softmax [37]	<b>19</b>
Figura 18 – Conjunto de pontos normalizados utilizados para treino, teste e teste com ruídos indicados em azul, vermelho e preto respectivamente para as amostras do tipo IPMC.	<b>20</b>
Figura 19 – Conjunto de pontos normalizados utilizados para treino, teste e teste com ruídos indicados em azul, vermelho e preto respectivamente para as amostras de Nafion.	<b>20</b>
Figura 20 – (a) Acurácia medida por percentagem de ruído associado aos dados simulados para teste para o conjunto de valores referente às amostras de IPMC (b) Valores de perda calculados por meio da função de perda em função do ruído.	<b>21</b>
Figura 21 – (a) Acurácia medida e (b) valores de perda calculados por meio da função de perda por percentagem de ruído associado aos dados simulados para teste para o conjunto de valores referente às amostras de Náfion®.	<b>22</b>

## SUMÁRIO

<b>1</b>	<b>INTRODUÇÃO</b>	<b>1</b>
<b>2</b>	<b>FUNDAMENTAÇÃO TEÓRICA</b>	<b>2</b>
2.1	INTELIGÊNCIA ARTIFICIAL E A ENGENHARIA E CIÊNCIA DOS MATERIAIS	2
2.2	REDES NEURAIS	4
2.3	POLÍMEROS ELETROATIVOS E COMPÓSITOS DE POLÍMERO IONOMÉRICO METAL (IPMC)	6
2.4	LÍQUIDOS IÔNICOS	9
2.5	ANÁLISE DINÂMICO MECÂNICA EM TORÇÃO	10
<b>3</b>	<b>MATERIAIS E MÉTODOS</b>	<b>13</b>
3.1	ALGORITMO DA REDE NEURAL EM PYTHON	13
<b>4</b>	<b>RESULTADOS E DISCUSSÕES</b>	<b>16</b>
4.1	ANÁLISE DA ACURÁCIA DE REDES NEURAIS NA CLASSIFICAÇÃO DE AMOSTRAS DE IPMC E NAFION	16
4.2	ANÁLISE DA ACURÁCIA EM FUNÇÃO DA INTENSIDADE DE RUÍDO	17
<b>5</b>	<b>CONCLUSÕES E CONSIDERAÇÕES FINAIS</b>	<b>20</b>
<b>6</b>	<b>REFERÊNCIAS</b>	<b>20</b>
<b>7</b>	<b>APÊNDICE A - CÓDIGO DA REDE NEURAL EM PYTHON</b>	<b>24</b>

## 1 INTRODUÇÃO

É possível observar a implementação de Inteligência Artificial (IA) nas mais diversas áreas como marketing, ciência do clima, mídias sociais [3, 4] e, também, na Ciência e Engenharia dos Materiais. O uso de IA dessa maneira se deve, principalmente, à enorme disponibilidade de dados [3] e ao avanço do poder computacional nas últimas décadas fazendo com que a ciência movida por dados ganhasse grande visibilidade. A utilização de redes neurais – modelos computacionais pertencentes à área de aprendizagem de máquina dentro da IA –, mesmo com arquiteturas mais simples, se tornasse uma forma de abordar problemas mais sofisticados.

As redes neurais artificiais (RNA) são denominadas dessa forma pelo fato de sua lógica de funcionamento ser inspirada no funcionamento do cérebro humano e na transmissão de informações dos neurônios biológicos [8]. Simplificadamente, um neurônio matemático recebe dados de entrada que serão computados numa soma ponderada e aplicados a uma função de ativação a qual fará o tratamento não linear dos dados e gerará um dado de saída que será, por sua vez, a entrada para os neurônios da próxima camada.

Desse modo, este trabalho tem por objetivo principal implementar e discutir a viabilidade da IA, mais especificamente a Aprendizagem de Máquina por meio de algoritmo de rede neural em linguagem de programação Python, como uma ferramenta no auxílio da caracterização de IPMCs – materiais da forma sanduíche formados por uma fina camada intermediária de um polímero eletroativo e duas camadas externas de um material condutor, neste caso, um metal nobre [11, 15]. Esses materiais se destacam por apresentarem movimentos suaves quando uma tensão é aplicada e, dessa forma, serem aplicados em diversas áreas como a biomedicina e a indústria aeroespacial [12].

Para obter melhores resultados, busca-se entender a relevância dos parâmetros da arquitetura da rede neural e como sua alteração pode implicar numa maior efetividade da rede. Ademais, pretende-se simular dados pertencentes ao domínio dos valores utilizados para treino e teste da rede, porém com ruídos adicionados aos valores nos eixos x e y, ou seja, valores que simulem análises de uma mesma amostra. Com estes dados gerados, será verificada qual a taxa de

acertos da rede neural conforme a intensidade do ruído dos dados em análise.

## **2 FUNDAMENTAÇÃO TEÓRICA**

### **2.1 Inteligência Artificial e a Engenharia e Ciência dos Materiais**

A Inteligência Artificial é um termo que já não faz parte somente do vocabulário de ficções científicas, é uma ciência que está cada vez mais presente no dia a dia e em todas as áreas do conhecimento, como, por exemplo, em algoritmos de redes sociais, jogos de estratégia – como Xadrez e Go –, reconhecimento de imagens e tradução simultânea [1, 2]. A aplicação de IA dessa maneira só é possível devido ao grande volume de dados disponível e à rápida evolução pela qual as unidades de processamento gráfico, poder computacional, passaram nas últimas décadas [8]. Desse modo, atrelada a um grande potencial econômico, a implementação da IA na Engenharia e Ciência dos Materiais também vem se mostrando um campo repleto de oportunidades.

Durante muito tempo o entendimento dos quatro vértices do famoso tetraedro da Ciência dos Materiais – processamento, estrutura, propriedade e aplicação – se baseou em métodos de tentativa e erro por meio de análises experimentais e simulações [3]. Porém, com o advento da Ciência de Dados e da Inteligência Artificial, o uso de métodos orientados por dados, conhecidos também por *data-driven*, para a descoberta e o *design* de novos materiais vem se mostrando uma alternativa para reduzir custos, riscos e tempo [1, 3].

Assim, dentro da grande área de IA, existe um ramo que conhecemos por *Machine Learning* (ML), ou aprendizado de máquina, no qual algoritmos permitem a aceleração do processo de detecção de regras a partir de uma extensa base de dados. Previsão de diagramas de fases, de estruturas cristalinas e de propriedades; auxílio na caracterização de materiais – como na interpretação de imagens de microscopias – são alguns exemplos já consolidados da aplicação do aprendizado de máquina na ciência dos materiais [1, 4], ramo este da ciência computacional que vem crescendo visivelmente nos últimos anos, como é possível observar pela Figura 1 abaixo.

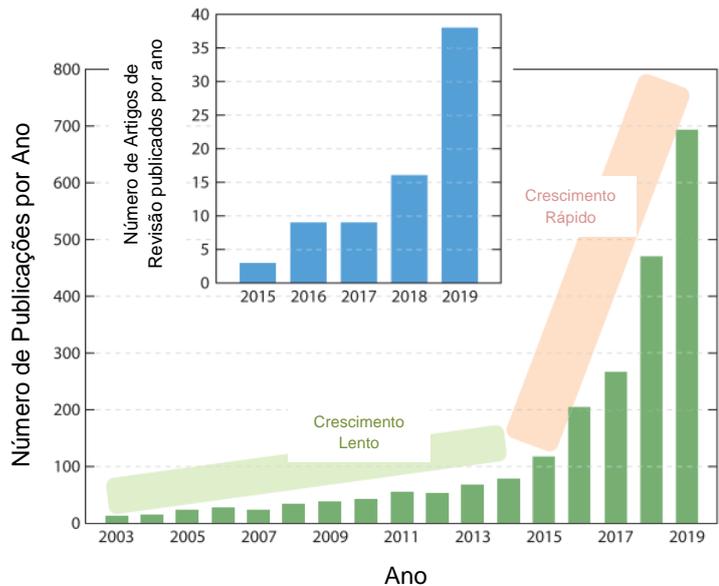


Figura 1: Gráfico adaptado indicando número de publicações por ano abordando aprendizado de máquina na Engenharia e Ciência dos Materiais. [2]

O aprendizado de máquina pode ser dividido em duas categorias: supervisionado e não supervisionado. No primeiro caso, o algoritmo é capaz de identificar uma função a partir de dados de entrada – que podem ser propriedades físicas e estruturais ou parâmetros de processamento, por exemplo – e de saída correspondentes. Na Figura 2 é possível observar o fluxo de um modelo supervisionado em que, ao final da otimização, são obtidas previsões de determinada propriedade ao inserir informações novas [5].

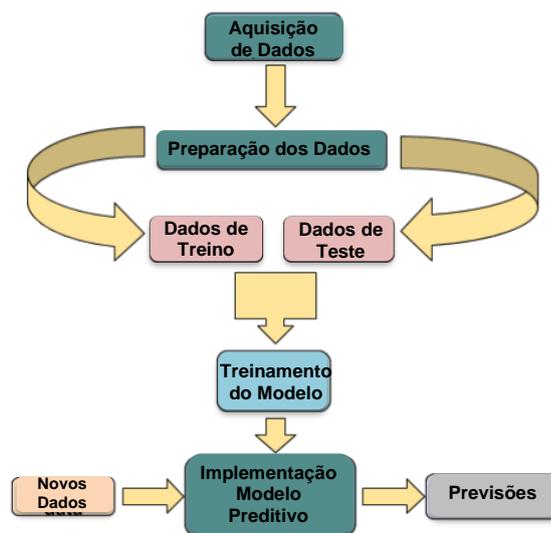


Figura 2: Fluxograma simplificado de um modelo supervisionado de ML adaptado de [5].

Já no segundo caso, modelos não supervisionados, não há a determinação de dados de saída e a máquina deve encontrar um padrão com base apenas nos dados fornecidos [1, 2]. Apesar da crescente geração de dados, sejam experimentais ou simulados, um desafio ainda encontrado para aplicar modelos computacionais de análise é a falta de uma base de dados grande e consistente disponível.

## 2.2 Redes Neurais

Redes Neurais Artificiais são um modelo matemático de Aprendizado de Máquina inspirado no funcionamento biológico das estruturas neurais de organismos inteligentes [8]. As pesquisas com RNAs começaram com o modelo computacional baseado na lógica de limiar (*threshold logic*) de Warren McCulloch e Walter Pitts em 1943 [8]. A partir desse modelo, abriram-se possibilidades para os estudos dos anos subsequentes, como o Perceptron (1958), que resolvia problemas com dados apenas linearmente separáveis e ficaria conhecido como o algoritmo de aprendizagem de arquitetura mais simples, desenvolvido por Rosenblatt [8, 28], os modelos ADALINE e MADALINE (1960) e, finalmente após o período conhecido como primeiro Inverno da IA (final da década de 70) em que houve uma baixa nos investimentos nessa área de pesquisa devido à falta de resultados mais promissores, foi proposto o modelo de redes multicamadas em meados dos anos 80 o qual permitiu, então, a construção de arquiteturas mais elaboradas e a resolução de problemas mais complexos [8].

RNAs multicamadas são formadas por um conjunto de neurônios, ou perceptrons, densamente interconectados e associados cada um a pesos, bias e uma função de ativação, sendo esta última um elemento matemático extremamente relevante no funcionamento da rede responsável pela transformação não linear dos dados recebidos, ou seja, permitindo que problemas de classificação de sistemas não lineares sejam abordados [32]. A Figura 3 [8, 28] representa o modelo simplificado de um perceptron em que o conjunto  $X_1 - X_n$  é o vetor de entrada que será multiplicado por pesos ( $w_i$ ) – vetor de pesos – gerando um valor por meio de uma soma ponderada que será a variável da função de ativação  $\phi$  que finalmente irá retornar uma saída  $y_k$  a qual será a entrada para a próxima camada de perceptrons.

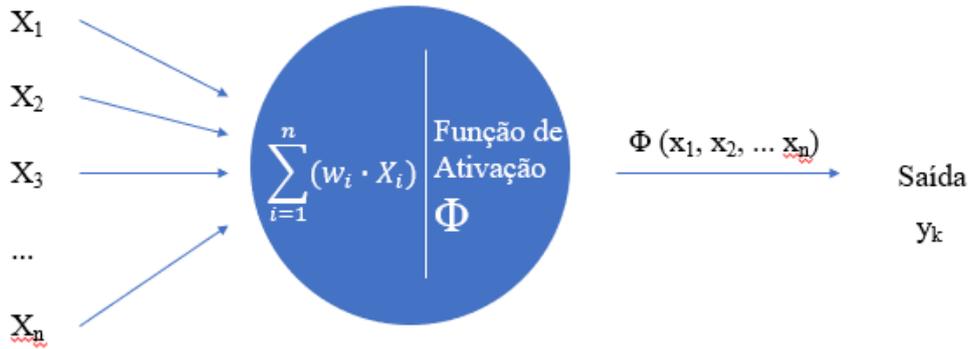


Figura 3: Modelo simplificado de um perceptron

A arquitetura da rede neural varia conforme o problema abordado e são vários os parâmetros que definem a arquitetura de uma rede neural, como o número de camadas e a função de ativação associada a cada camada. No modelo simplificado na figura abaixo, o fluxo de informações segue uma única direção de forma que os dados de saída de uma camada entram como vetor de entrada para a próxima camada [6, 7, 8, 28]. É comum que redes neurais possuam mais de três camadas: a camada de entrada, a de saída e as camadas ocultas como indicado no esquema representado pela Figura 4 [4].

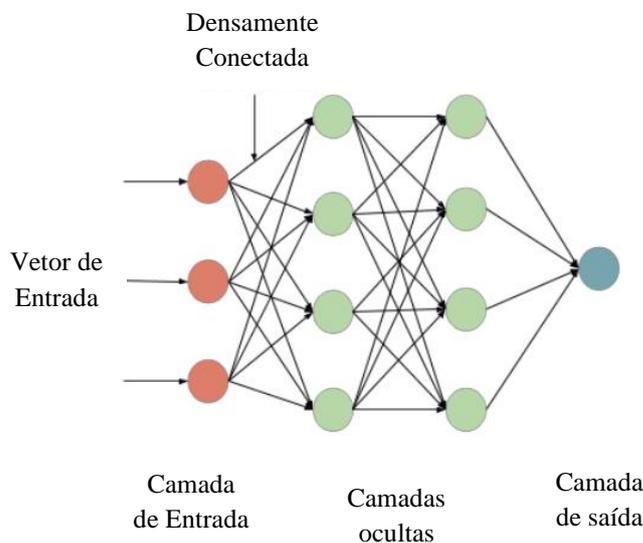


Figura 4: Exemplo simplificado da arquitetura de uma RNA com camadas ocultas [5].

A partir de dados, experimentais ou simulados, de entrada e de saída, as redes neurais são treinadas para que posteriormente novos dados do mesmo domínio possam ser inseridos e soluções ou previsões do problema abordado sejam devolvidas como dados de saída. Um dos principais algoritmos padrões de treinamento utilizado é o que conhecemos por algoritmo de retropropagação (ou

*backpropagation algorithm* no inglês) [8, 9, 28] que, apesar de ter evoluído significativamente em 1970, só foi aplicado às RNAs em 1985 [8].

No modelo de retropropagação os pesos associados às unidades de processamento são ajustados para reduzir o erro da rede – diferença entre o valor obtido e o valor de saída esperado – e, assim, otimizar o modelo de forma que a rede mapeie corretamente os dados de entrada [4, 8, 9, 28]. Essa otimização é feita calculando-se o gradiente da função de perda por meio da regra da cadeia em várias iterações até se obter o resultado desejado.

Dessa forma, as redes neurais têm sido muito utilizadas para obter as relações existentes entre variáveis de difícil percepção ao cérebro humano [4] e resolver problemas complexos de engenharia que contam, geralmente, com uma extensa base de dados que não possuem uma relação matemática simples.

Nesse âmbito, a área de compósitos, que vem ganhando cada vez mais espaço nas indústrias e aplicações do cotidiano, também mostrou um enorme potencial de crescimento com o auxílio dos algoritmos de RNA [10], sendo alguns exemplos: a previsão e otimização de propriedades mecânicas de compósitos reforçados com fibras naturais como demonstrado por Mulenga *et al.* (2021); previsão de propriedades dinâmico-mecânicas de compósitos [10] e a previsão da atuação de compósitos de polímero iônico e metal (IPMC) [11]. Assim, o tema abordado neste trabalho está relacionado com a última aplicação exemplificada.

## **2.2 Polímeros Eletroativos e Compósitos de Polímero Iônico e Metal (IPMC)**

Materiais inteligentes ou responsivos são materiais que possuem uma ou mais propriedades alteradas após um estímulo, físico ou químico, do ambiente em que estão inseridos. Materiais piezoelétricos, ligas de memória de forma e polímeros eletroativos são exemplos de materiais inteligentes [12]. No caso do último grupo, estes sofrem uma deformação física reversível quando submetidos a um estímulo elétrico [13, 12]. Esses materiais são divididos em dois grupos: os do tipo eletrônico (eEAP) e do tipo iônico (iEAP). Exemplos de aplicações de iEAP's são mostrados na Figura 5.

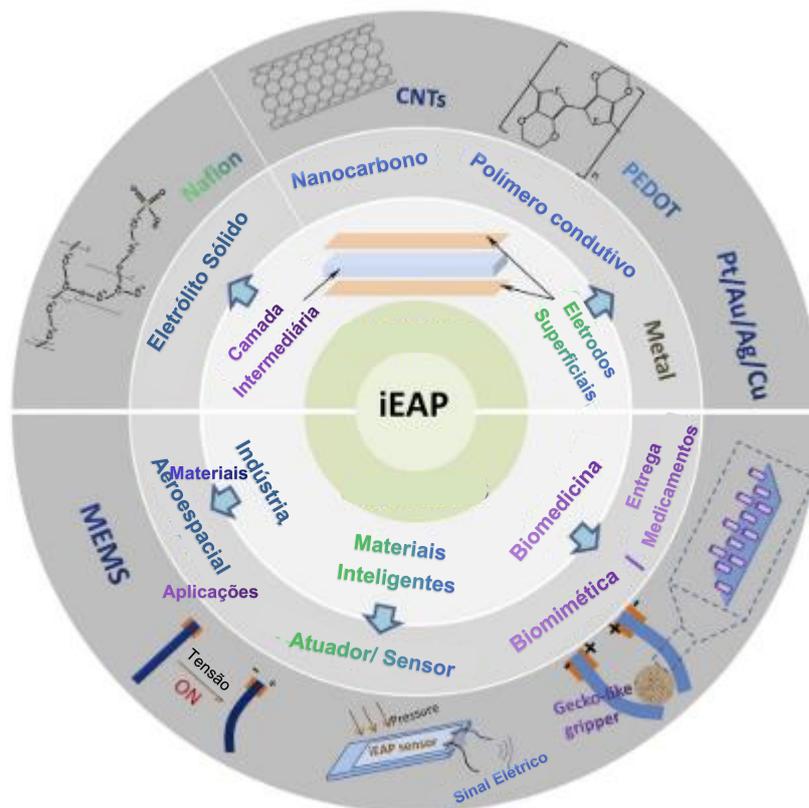


Figura 5: Imagem ilustrativa de polímeros eletroativos do tipo iônico e suas áreas de aplicação [12].

Dentro do grupo de iEAP's, existem os Compósitos de Polímero Ionomérico-Metal ou IPMC's, como é mais conhecido devido à adaptação do inglês – *Ionomeric Polymer- Metal Composites* –, que ganharam visibilidade por se apresentarem como uma possibilidade em futuras aplicações na robótica e biomedicina atuando, em baixas tensões, como músculos artificiais devido ao seu deslocamento suave similar a movimentos orgânicos [12, 13, 14]. IPMC's são compostos por uma fina membrana polimérica eletroativa recoberta por eletrodos metálicos em ambas as faces como indicado na Figura 5. Neste trabalho foi utilizado o Nafion® - N117 produzido pela DuPont, um dos principais polímeros estudados nesta aplicação [13, 15, 16, 17], possuindo elevada condutividade iônica e estabilidade química e mecânica [18].

O Náfion® – N117 é constituído por uma cadeia principal semicristalina apolar e hidrofóbica, semelhante ao Teflon (politetrafluoretileno), responsável por conferir as propriedades mecânicas do polímero. Já as cadeias laterais, parte amorfa, são hidrofílicas sulfonadas responsáveis pela condução iônica, como é possível observar na Figura 6 (a) [15]. O polímero apresenta uma morfologia ainda não totalmente

conhecida, sendo que uma das teorias mais aceitas é a de Gierke et al. [19] em que é proposta a existência de aglomerados iônicos interconectados por estreitos canais. A morfologia em questão poderia explicar a alta condutividade iônica do Náfion® [13, 20]. Porém, mesmo com a comprovação da existência desses canais ionoméricos [13], ainda não se tem um único modelo morfológico definido.

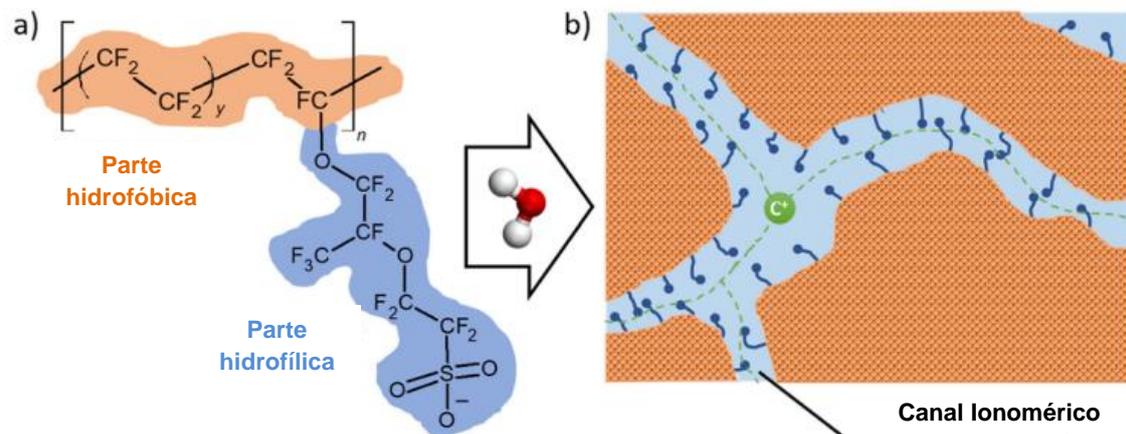


Figura 6: (a) Estrutura do Náfion® – N117 com parte hidrofóbica (laranja) e hidrofílica (azul) destacadas e (b) morfologia mostrando os canais ionoméricos contendo o cátion móvel [15].

Para este trabalho, o IPMC foi produzido conforme método de Oguro [17], em que é depositada, por meio da redução de íons metálicos, uma camada de metal sobre a membrana de Náfion® [17]. Esse metal deve ser química e eletroquimicamente estável, tenaz e ter resistência à fadiga [15], assim, os metais nobres são os principais candidatos, sendo que dentro deles o metal utilizado foi a platina.

O IPMC é, então, imerso em solução iônica [17] para incorporação do contraíon nos canais ionoméricos. O tamanho do contraíon incorporado se mostrou uma variável de influência no desempenho do atuador como mostrou Onishi, Sewa, Asaka, Fujiwara e Oguro (2001) [21]. Quando o IPMC com o contraíon incorporado é, então, submetido a uma diferença de potencial, é possível observar seu deslocamento o qual se dá devido à migração dos contraíons arrastando consigo moléculas de determinado solvente, como a água, por exemplo. Surge, dessa forma, uma diferença na concentração de água ao longo da espessura do atuador resultando na deformação do dispositivo como indicado pela Figura 7 [13, 18, 22].

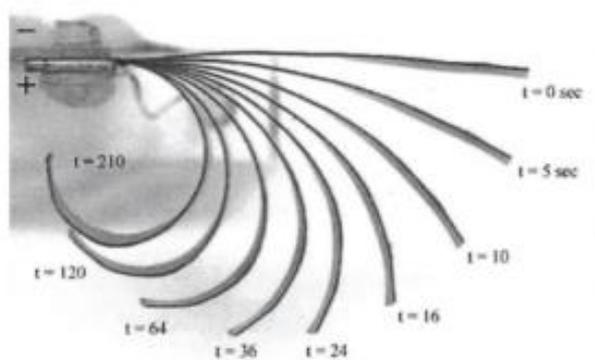


Figura 7: Ilustração da deformação do IPMC ao longo do tempo de atuação [22].

O desempenho do dispositivo pode variar com uma série de fatores, sendo alguns deles: o metal depositado na membrana, o solvente utilizado e a espessura [23]. Assim, embora o uso de IPMC's à base de Nafion® seja bastante promissor, ainda existem algumas limitações para o uso comercial, como o elevado custo associado à deposição de metais nobres sobre as superfícies da membrana e a perda de água para o ambiente e conseqüente queda de performance [24]. Quanto ao último ponto, a questão da desidratação durante os ciclos de atuação do dispositivo, uma possível solução que vem sendo estudada é a incorporação de líquidos iônicos nos canais ionoméricos [16, 24, 25].

### 2.3 Líquidos Iônicos

Sais iônicos líquidos, também chamados de líquidos iônicos (LI), são substâncias geralmente compostas por um cátion orgânico e um ânion de origem inorgânica [23]. Sua descoberta, principalmente dos sistemas com baixas temperaturas de fusão viabilizando seu estado líquido à temperatura ambiente, possibilitou o acesso a íons orgânicos livres de solventes abrindo novos caminhos em aplicações nas áreas de baterias, da eletroquímica e de reações orgânicas [25].

LI's cujos cátions têm como base o anel imidazólio – estrutura química representada pela Figura 8 – possibilita a formação de um composto quaternário a partir do átomo de nitrogênio na posição 3 [23], ganharam grande visibilidade por apresentarem propriedades como estabilidade térmica, alta condutividade iônica e caráter anfótero [23]. Esta classe de LI's ainda apresenta grande variação de suas

propriedades físico-químicas por conta do ânion e dos substituintes alquila combinados ao anel imidazólio [25, 23]. Por exemplo, os líquidos iônicos com cátions 1-alkil-3-metilimidazólio com ânion cloreto mostraram diferentes temperaturas de fusão com a variação do tamanho da cadeia alquila segundo Consorti, Souza e Dupont (2001) [25].

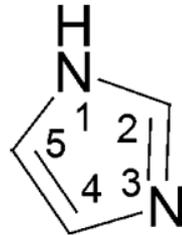


Figura 8: Estrutura química do anel imidazólio [23]

Com o avanço dos estudos relacionados à síntese e caracterização de líquidos iônicos, devido às propriedades citadas anteriormente, o 1-butil-3-metilimidazólio (BMIM.Cl), Figura 9, vem sendo estudado para ser incorporado em IPMC's à base de Nafion®. Pesquisas mostram que sua incorporação conferiu um aumento na condutividade do Náfion® além de aumentar a estabilidade do atuador exposto ao ar. O dispositivo com o LI incorporado operou sem queda significativa de performance por quase 9 vezes o número de ciclos que um dispositivo contendo água e cátions metálicos operaria [20]. Outras propriedades bastante interessantes que motivam pesquisas para a aplicação em questão é a baixa viscosidade, baixa volatilidade, alta mobilidade iônica e estabilidade eletromecânica dos LI's – esta última permitindo que o IPMC atue com tensões maiores [20, 23].

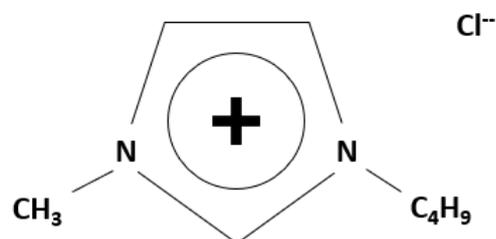


Figura 9: Estrutura química do líquido iônico 1-butil-3-metilimidazólio (BMIM.Cl).

## 2.4 Análise dinâmico-mecânica em torção

Todo material quando submetido a uma tensão ou forças externas gera uma resposta, seja na forma de deformação ou tensão, e, por meio de estudos reológicos, essas respostas dos materiais são estudadas. Um material pode ser considerado um sólido elástico cujo comportamento ideal pode ser descrito pela Lei de Hooke, Equação 1, [29, 30] em que a deformação ( $\gamma$ ) é instantânea sendo função apenas da tensão aplicada ( $\tau$ ). O material pode, também, ser classificado como um fluido viscoso de forma que irá apresentar deformações irreversíveis dissipando todo o trabalho externo. Um fluido viscoso ideal pode ter seu comportamento descrito pela Lei de Newton conforme Equação 2 em que a viscosidade é indicada por  $\eta$  e a taxa de deformação por  $\dot{\gamma}$  [30].

$$\tau = G \cdot \gamma$$

(Equação 1)

$$\tau_{xy} = \eta \dot{\gamma}$$

(Equação 2)

Além disso, o material pode ter um comportamento intermediário, ou seja, apresentar propriedades elásticas e viscosas, de modo que parte da energia será armazenada e, outra parte, dissipada. Esses materiais são chamados de viscoelásticos e é o caso dos polímeros fundidos e sólidos, nanocompósitos poliméricos e blendas poliméricas [30].

Assim, uma das formas de se estudar a resposta viscoelástica de polímeros no estado sólido é realizando ensaios dinâmico-mecânicos de torção com variação de frequência [26] por meio dos quais é possível obter os módulos de armazenamento e de perda,  $G'$  e  $G''$ , respectivamente [26, 27, 31]. O módulo  $G'$  está associado à energia armazenada em cada ciclo e,  $G''$ , à energia dissipada e ambos os módulos ajudam a definir as propriedades viscoelásticas do polímero [31]. Testes dinâmico-mecânicos de caracterização são fundamentais para o maior entendimento do material polimérico, já que em diversas aplicações os materiais podem estar sujeitos a deformações torcionais.

Podem ser utilizadas tanto amostras com geometria cilíndrica quanto com geometria retangular. Neste trabalho serão consideradas amostra retangulares, como mostra a Figura 10. Devido à existência de tensões extensionais e compressivas, principalmente ao redor das presilhas em que a amostra é afixada [27], a análise

torna-se mais complexa do que a primeira geometria mencionada.

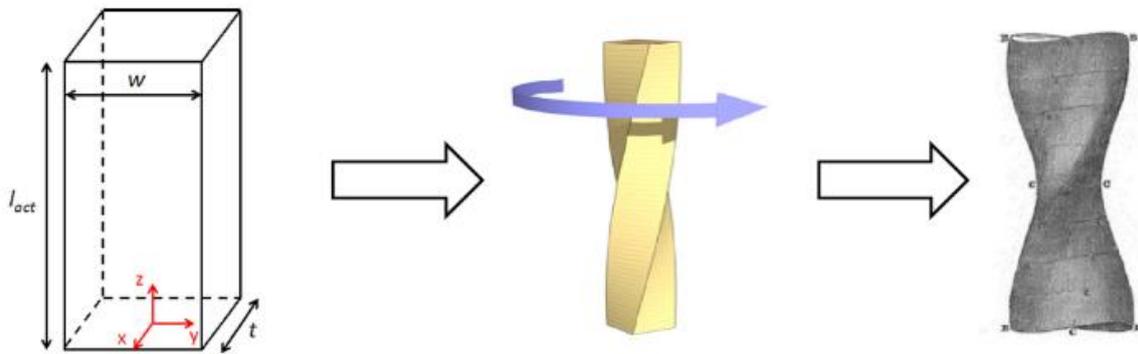


Figura 10: Ilustração de uma amostra com geometria de prisma sob torção [27]

As dimensões da amostra e a forma como a amostra é fixada no aparelho são alguns dos principais parâmetros que influenciam nas curvas obtidas de modo que é crucial que sejam realizadas triplicatas, ou mais, para a minimizar os erros. A partir de testes com amostras retangulares de SBR, como demonstrou Dessi *et al.* (2016), na Figura 11 é possível observar as curvas esperadas do módulo  $G'$  em função da frequência para diferentes razões de aspecto comprimento/espessura. Assim, sabendo-se dessa dependência da razão de aspecto na leitura de  $G'$  com a frequência, as dimensões das amostras requeridas para realização dos ensaios são indicadas nos manuais dos reômetros comerciais de acordo com a espessura do material polimérico [26] para que os dados obtidos sejam coerentes.

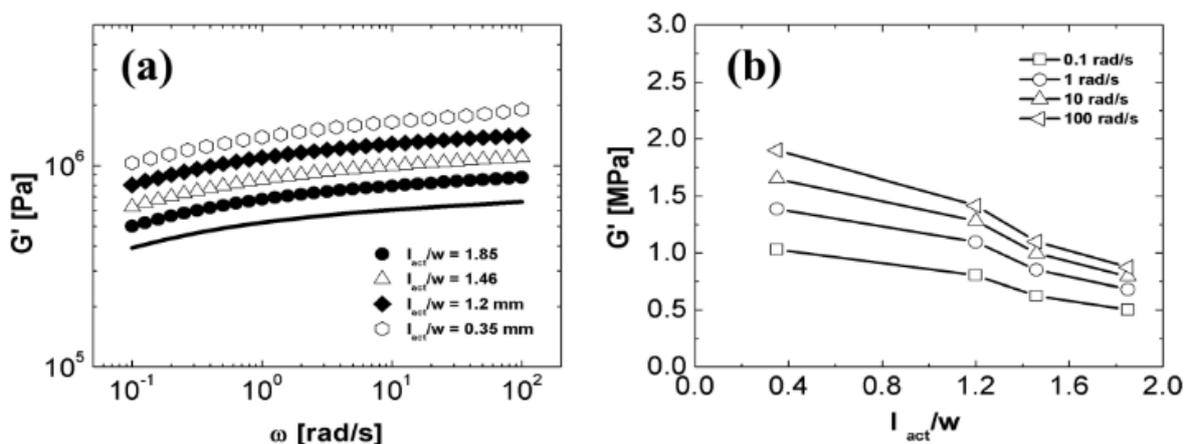


Figura 11: (a) Curvas do módulo  $G'$  pela frequência obtidas de um ensaio de torção em geometria retangular para uma amostra de SBR e (b) variação de  $G'$  com o comprimento da amostra [27]

Diani e Gilormini (2017) compararam os dados experimentais de ensaios

dinâmico-mecânicos de torção com simulações numéricas por elementos finitos. A ferramenta computacional se mostrou viável, atentando-se ao fato de que de acordo com a geometria considerada, é necessário um coeficiente de correção a fim de determinar o módulo de forma mais precisa [26].

Neste trabalho, para estudar a viabilidade de algoritmos de aprendizado de máquina, foram considerados dados de  $G' \times$  Frequência angular ( $\omega$ ) a partir de ensaios de torção em Nafion® e IPMC em temperatura ambiente.

### **3 MATERIAIS E MÉTODOS**

#### **3.1 Algoritmo de Rede Neural em Python**

Para este trabalho, o código foi escrito em linguagem de programação Python baseado num algoritmo de rede neural para reconhecimento de imagens de números manuscritos disponível publicamente no tutorial de *Machine Learning* para programação em Python [28]. Foram feitas adaptações para o problema de classificação de amostras de IPMC aqui explorado e o código adaptado consta no Apêndice A. Além disso, foram empregadas as bibliotecas *NumPy*, *Random*, *Matplotlib* e *Keras*, também disponíveis para uso aberto, com a finalidade de auxiliar na geração de gráficos, valores randomizados, e disponibilizar funções matemáticas para o desenvolvimento de modelos de redes neurais de aprendizagem profunda.

Os dados utilizados foram obtidos por meio de ensaios dinâmico-mecânicos de reometria de torção com geometria retangular previamente feitos em reômetro ARES da TA Instruments do laboratório de Polímeros do Departamento de Engenharia de Materiais da UFSCar. As amostras consideradas foram as de Nafion® e as de IPMC a base de Nafion® e platina com os contraíons  $H^+$  e líquido iônico BMIM.Cl 0,2M e 0,5M incorporados. Além disso, a condição inicial de umidade da amostra também variou entre seca e úmida. Dessa forma, para cada tipo de amostra, Nafion® ou IPMC, é possível contar com 6 classes diferentes como indicado na Tabela 1 abaixo:

Tabela 1: Amostras com diferentes contraíons incorporados e condições iniciais de umidade utilizadas para realização dos ensaios de torção.

CLASSE	AMOSTRA	CONTRAÍON	CONDIÇÃO
1	Náfion®	H+	Seco
2	Náfion®	BMIM.Cl 0,2M	Seco
3	Náfion®	BMIM.Cl 0,5M	Seco
4	Náfion®	H+	Úmido
5	Náfion®	BMIM.Cl 0,2M	Úmido
6	Náfion®	BMIM.Cl 0,5M	Úmido
1	IPMC	H+	Seco
2	IPMC	BMIM.Cl 0,2M	Seco
3	IPMC	BMIM.Cl 0,5M	Seco
4	IPMC	H+	Úmido
5	IPMC	BMIM.Cl 0,2M	Úmido
6	IPMC	BMIM.Cl 0,5M	Úmido

Para cada classe em um dado tipo de amostra, Nafion® ou IPMC, foram coletados 20 pontos - módulo de armazenamento  $G' (T)$  [Pa] medido em função da frequência aplicada ( $\omega$ ) [rad/s], conforme ilustrado nas Figuras 12 e 13. É possível observar que as amostras cujas condições iniciais de hidratação são secas um módulo mais alto, já que a umidade apresenta um efeito tenacificador no polímero.

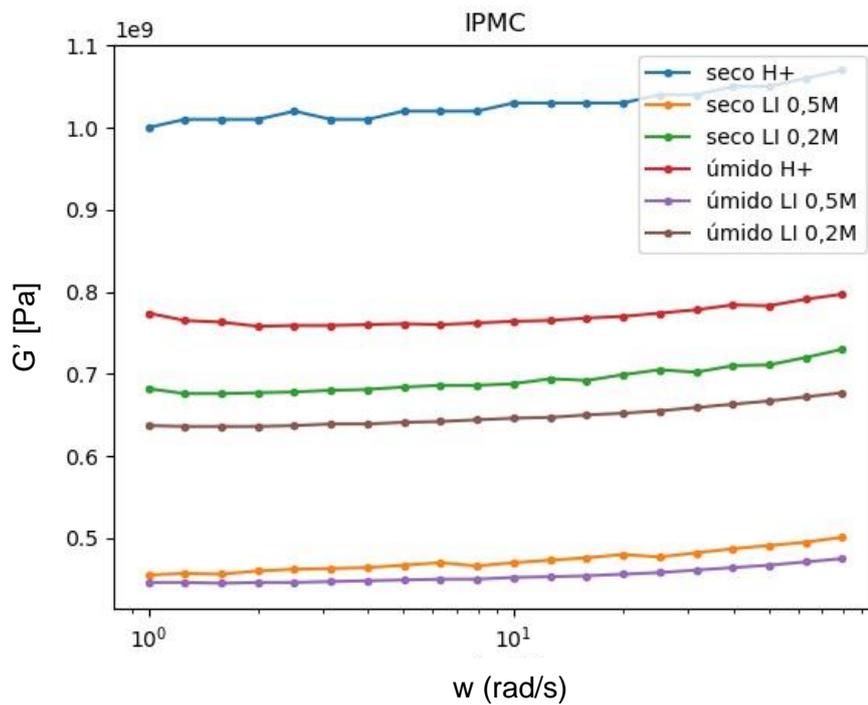


Figura 12: Gráfico do módulo de armazenamento,  $G'$  [Pa], por frequência, [rad/s], para as amostras de IPMC.

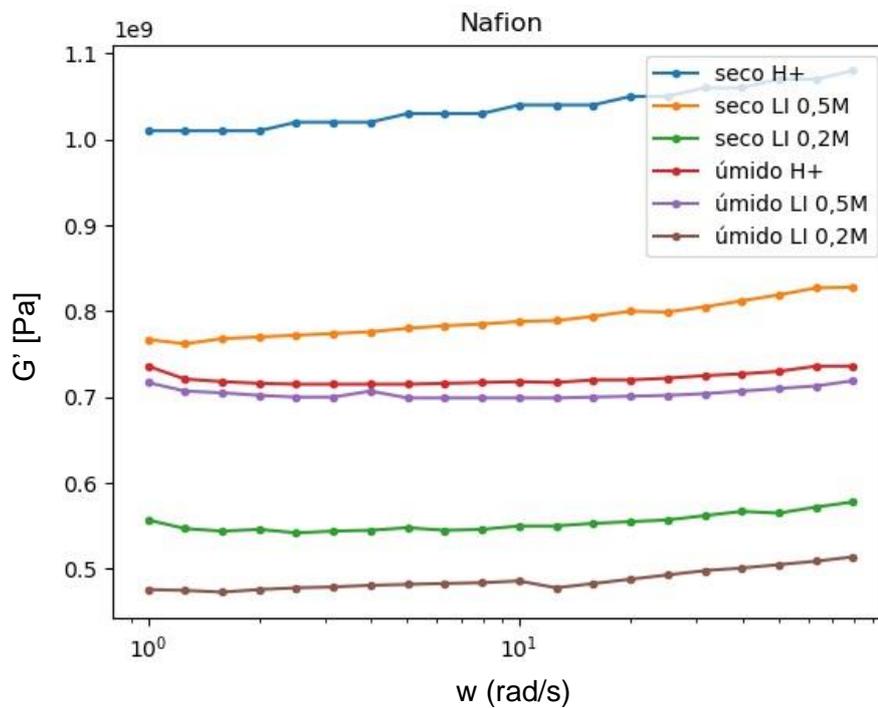


Figura 13: Gráfico do módulo de armazenamento,  $G'$  [Pa], por frequência,  $\omega$  [rad/s], para as amostras de Nafion®.

Desse modo, os dados  $W$  e  $T$  foram fornecidos na camada de entrada da rede e, assim, ao final da etapa de treinamento, a rede deve ser capaz de classificar a amostra, ou seja, identificar qual amostra se trata. Logo, trata-se de um problema de classificação multiclases com 6 classes.

No código original disponível no tutorial, a quantidade de dados analisada era da ordem de dezenas de milhares e a rede era muito mais extensa contando com 100 neurônios ocultos. Como para o problema deste trabalho a base de dados e sua complexidade são menores, foi escolhido de forma arbitrária um número de neurônios bem menor. Assim, a arquitetura da rede utilizada neste trabalho consta de três camadas densamente conectadas, sendo duas ocultas com 16 neurônios cada e uma de saída com 6 neurônios, conforme ilustração do esquema abaixo.

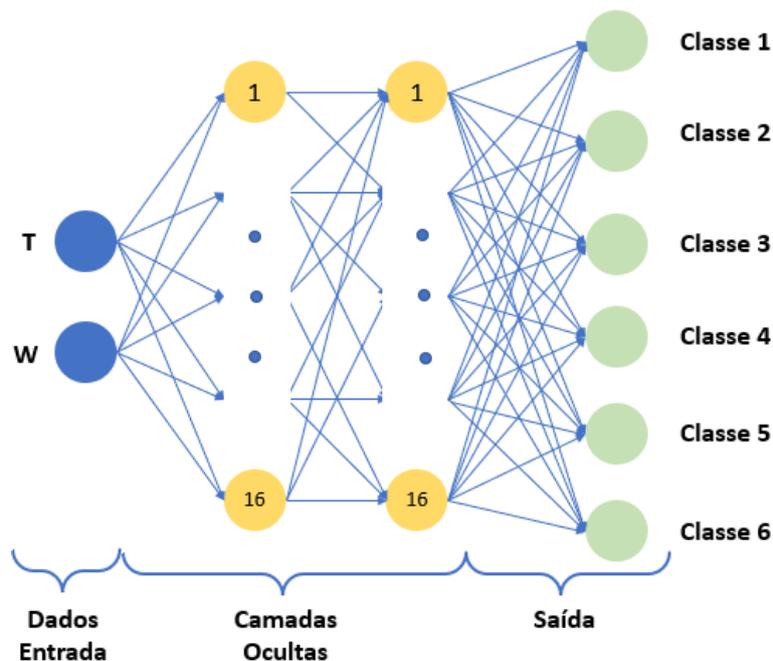


Figura 14: Diagrama ilustrativo da arquitetura da rede neural considerada para este trabalho.

De todos os 120 dados disponíveis para cada tipo de amostra, 5/6 foram utilizados para treino da rede neural e 1/6 para validação. Os dados foram selecionados de forma aleatória com o auxílio da função *random* disponível nas bibliotecas do Python, como mostra o gráfico abaixo, em que os pontos em azul indicam os valores utilizados para treino e os em vermelho, para teste.

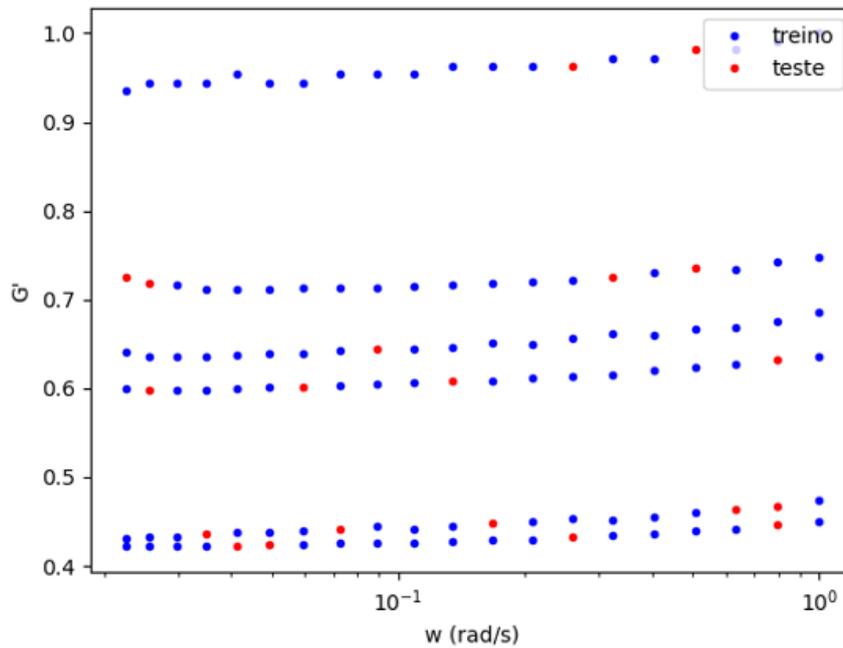


Figura 15: Gráfico ilustrativo indicando os valores de entrada normalizados utilizados para treino em azul e, para teste, em vermelho.

## 4 RESULTADOS E DISCUSSÕES

### 4.1 ANÁLISE DA ACURÁCIA DE REDES NEURAIS NA CLASSIFICAÇÃO DE AMOSTRAS DE IPMC E NÁFION®

Seguindo o Tutorial de Aprendizado de Máquina em Python [28] inicialmente foram utilizadas as funções de ativação *relu*, a qual é comumente aplicada em perceptrons multicamadas e está representada pela Figura 16 (a), nas duas camadas iniciais e a função *sigmoid*, ideal para problemas de classificação binários cuja curva em formato de S está representada pela Figura 16 (b) [32, 33], na camada de saída. Após os ciclos de treino e teste da rede obteve-se uma acurácia de 16 a 19%, que foi considerada baixa já que não se trata de um problema complexo de classificação.

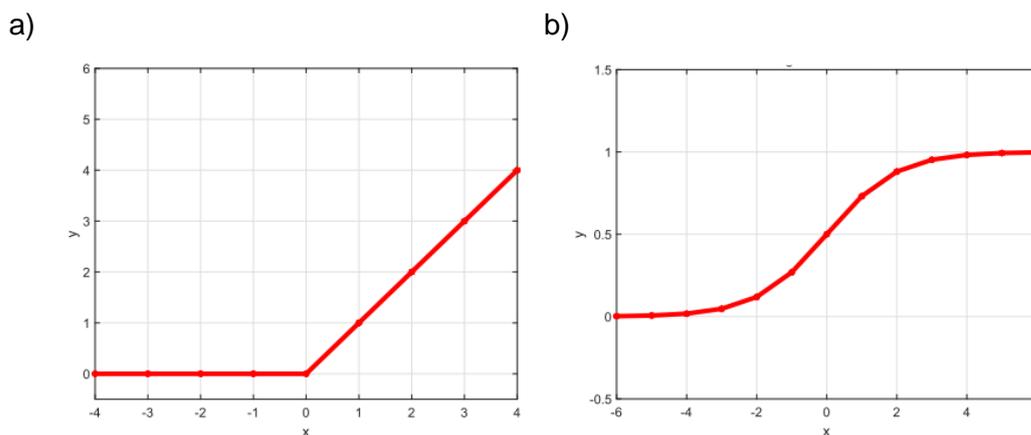


Figura 16: Gráficos das curvas (a) da função ReLu (b) e da sigmóide. Adaptado de [32].

Assim, com o objetivo de aumentar a acurácia de classificação da rede, foram revisados os parâmetros da rede que poderiam ser alterados, como o número de camadas, número de ciclos de treinamento – um número excessivo de ciclos pode levar ao *overfitting* da rede de modo que a mesma perca o poder de generalização – e as funções de ativação – elemento matemático responsável pela transformação não-linear dos sinais [8, 32, 33, 34]. Para o modelo de rede abordado neste trabalho, a escolha da função de ativação tem grande influência para o bom desempenho da rede neural, assim como foi demonstrado por Wang, Li, Song e Rong (2020) [32] ao estudarem a influência da função de ativação na acurácia da taxa de reconhecimento de expressões faciais por classificação de imagens via redes neurais convolucionais.

Por fim, após alterações nas funções de ativação do modelo, foi, então, utilizado um modelo sequencial com as funções *tanh* nas duas primeiras camadas e *softmax* na camada de saída. Esta última é similar à função sigmoide, porém opera com vetores, enquanto a outra, com escalares. A função softmax normaliza o vetor dos dados de saída de forma que os novos valores estejam contidos num intervalo de 0 a 1 podendo ser interpretados como probabilidades [33, 34], como ilustrado na Figura 17 [37], o que faz com que seja largamente utilizada em modelos de redes neurais de classificação multiclases em que as classes são mutualmente exclusivas [33, 34]. Dessa forma, com uma função de ativação na camada de saída mais adequada ao problema aqui abordado contendo 6 classes, a acurácia obtida foi superior a 95% e não mais em torno de 16 – 19%.

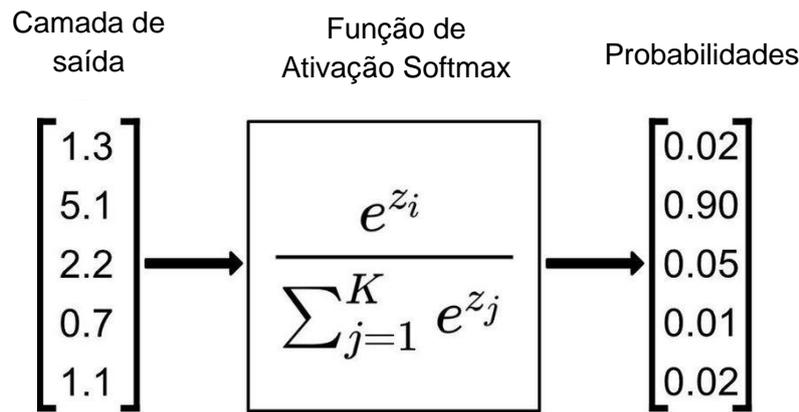


Figura 17: Princípio ilustrativo de funcionamento da função de ativação Softmax [37]

#### 4.2 ANÁLISE DA ACURÁCIA EM FUNÇÃO DA INTENSIDADE DE RUÍDO

Quando várias análises de uma mesma amostra são feitas, existe uma distribuição de medidas, pois os valores medidos em cada ensaio nunca serão exatamente os mesmos e irão apresentar uma diferença em relação ao valor “verdadeiro”, ou seja, possuem o que chamamos de erro experimental associado, é um erro inerente. Ademais, esse erro deve estar dentro de uma faixa considerada aceitável.

A proposta de uma rede neural de classificação multiclases é que sua performance ocorra de acordo com as características presentes nos dados de entrada e não de acordo com dados que apresentem valores exatamente iguais, ou seja, deve apresentar uma boa acurácia levando em consideração o desvio padrão aceitável. Caso houvessem mais pontos experimentais disponíveis para uma mesma frequência, os mesmos seriam utilizados para treino da rede e o algoritmo aprenderia a faixa de variação de valores para cada classe.

Dessa forma, para estudar a robustez da rede neural, o programa descrito no Apêndice A também gerou dados com ruídos a partir do grupo selecionado para teste de forma aleatória, ou seja, houve deslocamento nos eixos x e y em relação aos pontos de referência medidos experimentalmente. Essa simulação de dados foi feita para analisar a acurácia da rede neural em função da intensidade dos ruídos nos dados de entrada.

Os valores normalizados com ruídos simulados estão indicados em preto pelos gráficos das Figuras 18 e 19 para as amostras de IPMC e Nafion® respectivamente.

Conjunto de teste com ruído (normalizado) 1,2% - IPMC

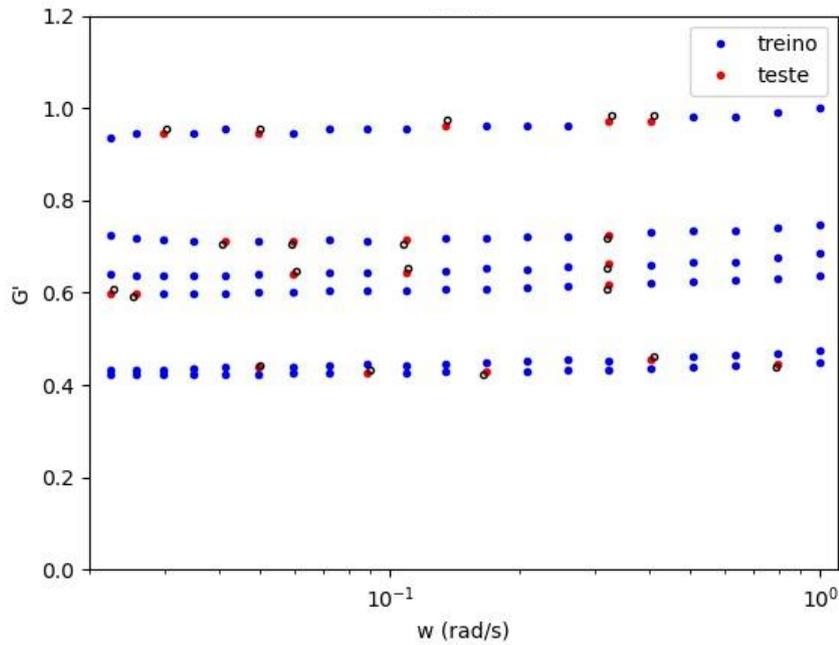


Figura 18: Conjunto de pontos normalizados utilizados para treino, teste e teste com ruídos indicados em azul, vermelho e preto respectivamente para as amostras do tipo IPMC.

Conjunto de teste com ruído (normalizado) 1,2% - Nafion

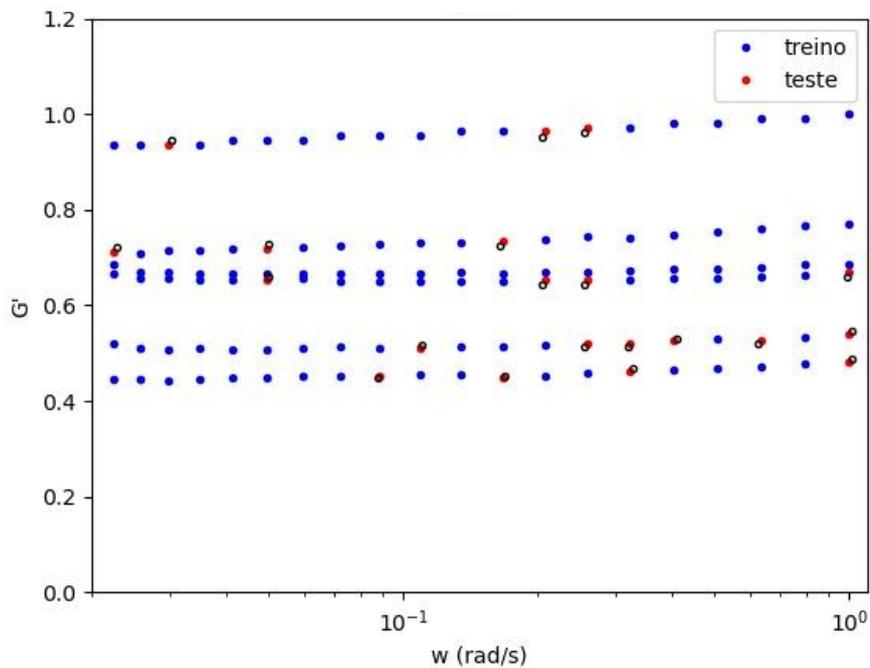


Figura 19: Conjunto de pontos normalizados utilizados para treino, teste e teste com ruídos indicados em azul, vermelho e preto respectivamente para as amostras de Nafion®.

Nos gráficos das Figuras 20 (a) e 21 (a) é possível observar a acurácia da rede em função dos testes realizados com os dados simulados variando-se a intensidade de ruído associada. Nota-se que para uma porcentagem de ruídos em até 1% a acurácia se mantém acima de 80%, porém, de acordo com o esperado, conforme a divergência dos valores gerados em relação aos experimentais aumenta, a rede se torna menos eficaz para classificar corretamente as amostras. A rede é considerada adequada para classificação das amostras de IPMC se a mesma apresentar uma boa acurácia dentro dos limites de variabilidade das medidas de torção, o que poderia ser observado se mais dados estivessem disponíveis.

Além disso, também foram analisados os valores de perda calculados em função do ruído. Os valores de perda são obtidos por meio da função de perda, a qual é um elemento importante no processo de otimização da rede neural pois a perda é utilizada no cálculo do gradiente para ajuste dos pesos [35, 36]. Assim, a função de perda é uma forma de avaliar a solução fornecida pela rede com o valor alvo desejado, quanto maior este valor mais distante a previsão está do valor correto [35, 36]. Neste trabalho a função utilizada foi *Categorical Crossentropy* disponível pela biblioteca Keras.

Portanto, nas Figuras 20 (b) e 21 (b) é possível observar que conforme a porcentagem de ruído aumenta, os valores de perda são maiores, ou seja, a previsão da rede se distancia do valor correto, o que está coerente com o gráfico das Figuras 20 (a) e 21 (a) em que se nota a queda da acurácia.

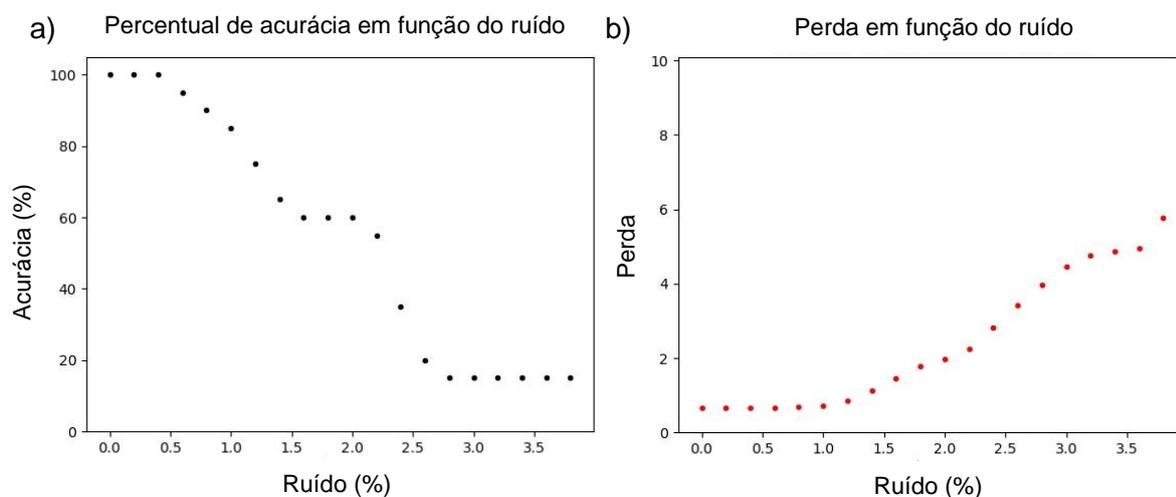


Figura 20: (a) Acurácia medida e (b) valores de perda calculados pela função de perda por porcentagem de ruído associado aos dados simulados para teste para o conjunto de valores referente às amostras de IPMC.

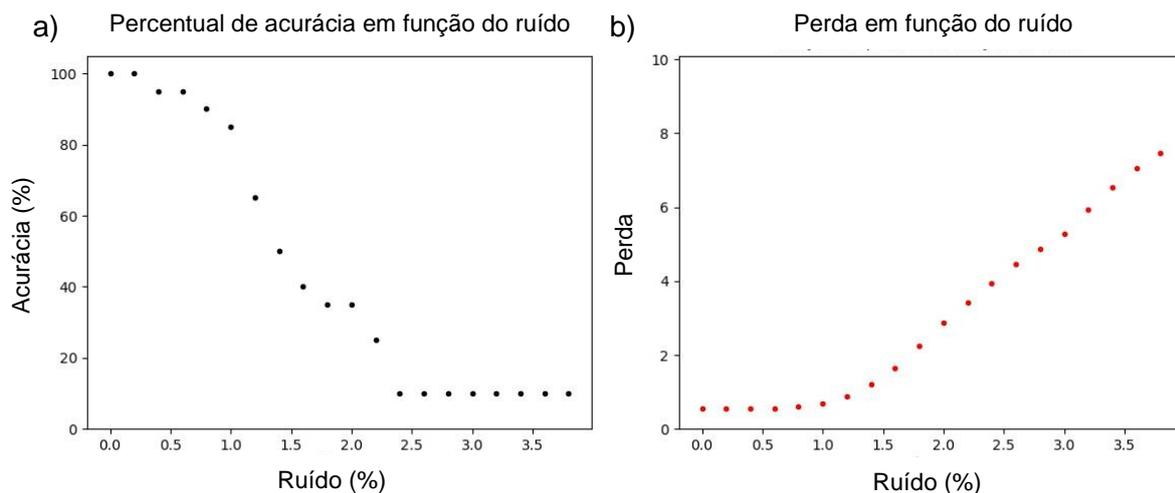


Figura 21: (a) Acurácia medida e (b) valores de perda calculados por meio da função de perda por porcentagem de ruído associado aos dados simulados para teste para o conjunto de valores referente às amostras de Nafion®.

## 6 CONCLUSÕES E CONSIDERAÇÕES FINAIS

Este trabalho permitiu estudar a aplicação de redes neurais multicamadas como ferramenta para classificar amostras de IPMC e Nafion® submetidas a diferentes tratamentos (contração incorporado e condição inicial de umidade) com base nas repostas viscoelásticas obtidas por ensaios dinâmico-mecânicos de torção. A programação da rede neural foi feita em linguagem de programação Python com a utilização das bibliotecas *NumPy*, *Matplotlib*, *Random* e *Keras*.

Após o treinamento e teste da rede neural, observou-se que com a escolha adequada da função de ativação para a rede, obteve-se uma medida de acurácia superior a 95% indicando, como resultado preliminar, que o perceptron multicamadas adaptado descrito no Apêndice A apresenta ser viável como uma ferramenta auxiliar para classificar amostras de IPMC e Nafion®. Foi possível perceber a relevância dos parâmetros da arquitetura da rede neural e como estes influenciam da efetividade do algoritmo.

Além disso, ao utilizar dados de entrada simulados com ruídos, notou-se que há uma queda significativa na acurácia medida com uma porcentagem de ruídos maior que 1%, tanto para as amostras de IPMC quanto para as de Nafion®. O gráfico dos valores de perda em função da porcentagem de ruído também apresentou um

comportamento coerente com os gráficos de acurácia. Um ponto para melhoria seria verificar qual é a variabilidade aceitável encontrada nas medidas do módulo de armazenamento,  $G'$ , para, assim, avaliar a robustez da rede em classificar corretamente as amostras com os diferentes tratamentos. Portanto, uma melhor performance da rede poderia ser alcançada caso houvesse um maior volume de dados disponíveis para realizar o treinamento da rede neural. Além disso, é interessante ter certeza a respeito da sensibilidade do equipamento e da validade dos valores medidos, talvez pudesse ser, inclusive, realizada uma contraprova utilizando outro equipamento.

Apesar do problema de classificação aqui abordado não ser muito complexo e não demandar a necessidade da implementação de uma rede neural, o trabalho teve como intuito estudar de forma preliminar o funcionamento e a aplicação do algoritmo de aprendizado de máquina. Dessa forma, por meio deste trabalho, observou-se que, assim como em outras áreas da engenharia de materiais, as redes neurais podem ser uma ferramenta complementar bastante útil na previsão e caracterização de materiais utilizando seu comportamento viscoelástico, neste caso, de amostras de Nafion® e IPMC, uma classe de materiais inteligentes com grande potencial de crescimento e aplicação.

## REFERÊNCIAS

[1] MUELLER, Tim; KUSNE, Aaron Gilad; RAMPRASAD, Rampi. MACHINE LEARNING IN MATERIALS SCIENCE: RECENT PROGRESS AND EMERGING APPLICATIONS. In: PARRILL, Abby L.; LIPKOWITZ, Kenny B. (ed.). **Reviews in Computational Chemistry**. 1ª Ed. New Jersey: Wiley, 2016. Cap. 4. p. 186-273. (29).

[2] MORGAN, Dane; JACOBS, Ryan. Opportunities and Challenges for Machine Learning in Materials Science. **Annual Review Of Materials Research**, [S.L.], v. 50, n. 1, p. 71-103, 1 jul. 2020. Annual Reviews. <http://dx.doi.org/10.1146/annurev-matsci-070218-010015>.

[3] AGRAWAL, Ankit; CHOUDHARY, Alok. Deep materials informatics: applications of deep learning in materials science. **Mrs Communications**, [S.L.], v. 9, n. 3, p. 779-792, set. 2019. Springer Science and Business Media LLC. <http://dx.doi.org/10.1557/mrc.2019.73>.

[4] SHA, Wuxin; GUO, Yaqing; YUAN, Qing; TANG, Shun; ZHANG, Xinfang; LU, Songfeng; GUO, Xin; CAO, Yuan-Cheng; CHENG, Shijie. Artificial Intelligence to

Power the Future of Materials Science and Engineering. **Advanced Intelligent Systems**, [S.L.], v. 2, n. 4, p. 1900143, 24 mar. 2020. Wiley. <http://dx.doi.org/10.1002/aisy.201900143>.

[5] MULENGA, Timothy K.; UDE, Albert U.; VIVEKANANDHAN, Chinnasamy. Techniques for Modelling and Optimizing the Mechanical Properties of Natural Fiber Composites: a review. **Fibers**, [S.L.], v. 9, n. 1, p. 6, 14 jan. 2021. MDPI AG. <http://dx.doi.org/10.3390/fib9010006>.

[6] CHERNODED, Andrey; DUDKO, Lev; MYAGKOV, Igor; VOLKOV, Petr. Deep Learning Neural Networks and Bayesian Neural Networks in Data Analysis. **Epj Web Of Conferences**, [S.L.], v. 158, p. 06008, 2017. EDP Sciences. <http://dx.doi.org/10.1051/epjconf/201715806008>.

[7] DE CARVALHO, Andre. **Redes Neurais Artificiais**. Disponível em: <<https://sites.icmc.usp.br/andre/research/neural/>>. Acesso em: 15 ago. 2021.

[8] Data Science Academy. **Deep Learning Book, 2021**. Disponível em: <<https://www.deeplearningbook.com.br/>>. Acesso em: 26 Set. 2021

[9] JOHSON, Daniel. Back Propagation Neural Network: What is Backpropagation Algorithm in Machine Learning?. Disponível em: <https://www.guru99.com/backpropogation-neural-network.html> . Acesso em: 26 Set. 2021.

[10] ZHANG, Z; FRIEDRICH, K. Artificial neural networks applied to polymer composites: a review. **Composites Science And Technology**, [S.L.], v. 63, n. 14, p. 2029-2044, nov. 2003. Elsevier BV. [http://dx.doi.org/10.1016/s0266-3538\(03\)00106-4](http://dx.doi.org/10.1016/s0266-3538(03)00106-4).

[11] YANG, Liang; ZHANG, Dongsheng; ZHANG, Xining; TIAN, Aifen. Prediction of the Actuation Property of Cu Ionic Polymer–Metal Composites Based on Backpropagation Neural Networks. **Acs Omega**, [S.L.], v. 5, n. 8, p. 4067-4074, 19 fev. 2020. American Chemical Society (ACS). <http://dx.doi.org/10.1021/acsomega.9b03725>.

[12] DONG, Yuqing; YEUNG, Ka-Wai; TANG, Chak-Yin; LAW, Wing-Cheung; TSUI, Gary Chi-Pong; XIE, Xiaolin. Development of ionic liquid-based electroactive polymer composites using nanotechnology. **Nanotechnology Reviews**, [S.L.], v. 10, n. 1, p. 99-116, 1 jan. 2021. Walter de Gruyter GmbH. <http://dx.doi.org/10.1515/ntrev-2021-0009>.

[13] HIRANO, Laos A.; ESCOTE, Márcia T.; MARTINS-FILHO, Luiz S.; MANTOVANI, Gerson L.; SCURACCHIO, Carlos H.. Development of Artificial Muscles Based on Electroactive Ionomeric Polymer-Metal Composites. **Artificial Organs**, [S.L.], v. 35, n. 5, p. 478-483, maio 2011. Wiley. <http://dx.doi.org/10.1111/j.1525->

[14] HIRANO, Laos Alexandre; ACERBI, Lenin Wagner; KIKUCHI, Kunitomo; TSUCHITANI, Shigeki; SCURACCHIO, Carlos Henrique. Study of the Influence of the Hydration Level on the Electromechanical Behavior of Nafion Based Ionomeric **Polymer-metal Composites Actuators**. *Materials Research*, [S.L.], v. 18, n. 2, p. 154-158, 24 nov. 2015. FapUNIFESP (SciELO). <http://dx.doi.org/10.1590/1516-1439.353214>

[15] GONÇALVES, R.; TOZZI, K. A.; SACCARDO, M. C.; ZUQUELLO, A. G.; SCURACCHIO, C. H.. Nafion-based ionomeric polymer/metal composites operating in the air: theoretical and electrochemical analysis. **Journal Of Solid State Electrochemistry**, [S.L.], v. 24, n. 8, p. 1845-1856, 16 fev. 2020. Springer Science and Business Media LLC. <http://dx.doi.org/10.1007/s10008-020-04520-6>.

[16] KIKUCHI, Kunitomo; TSUCHITANI, Shigeki. Nafion®-based polymer actuators with ionic liquids as solvent incorporated at room temperature. **Journal Of Applied Physics**, [S.L.], v. 106, n. 5, p. 053519, set. 2009. AIP Publishing. <http://dx.doi.org/10.1063/1.3204961>.

[17] SACCARDO, Matheus Colovati; ZUQUELLO, Ariel Gustavo; TOZZI, Kaique Afonso; GONÇALVES, Roger; HIRANO, Laos Alexandre; SCURACCHIO, Carlos Henrique. Counter-ion and humidity effects on electromechanical properties of Nafion®/Pt composites. **Materials Chemistry And Physics**, [S.L.], v. 244, p. 122674, abr. 2020. Elsevier BV. <http://dx.doi.org/10.1016/j.matchemphys.2020.122674>.

[18] LEE, Jang-Woo; HONG, Soon Man; KOO, Chong Min. High-performance polymer ionomer–ionic liquid membrane IPMC actuator. **Research On Chemical Intermediates**, [S.L.], v. 40, n. 1, p. 41-48, 26 out. 2013. Springer Science and Business Media LLC. <http://dx.doi.org/10.1007/s11164-013-1453-0>.

[19] Gierke T, Munn G, Wilson F. Perfluorinated ionomer membranes. *American Chemical Society*; 1982. p. 195e216 [chapter 10]

[20] BENNETT, Matthew D.; LEO, Donald J.; WILKES, Garth L.; BEYER, Frederick L.; PECHAR, Todd W.. A model of charge transport and electromechanical transduction in ionic liquid-swollen Nafion membranes. **Polymer**, [S.L.], v. 47, n. 19, p. 6782-6796, set. 2006. Elsevier BV. <http://dx.doi.org/10.1016/j.polymer.2006.07.061>.

[ 21] ONISHI, Kazuo; SEWA, Shingo; ASAKA, Kinji; FUJIWARA, Naoko; OGURO, Keisuke. The effects of counter ions on characterization and performance of a solid polymer electrolyte actuator. **Electrochimica Acta**, [S.L.], v. 46, n. 8, p. 1233-1241, jan. 2001. Elsevier BV. [http://dx.doi.org/10.1016/s0013-4686\(00\)00695-2](http://dx.doi.org/10.1016/s0013-4686(00)00695-2).

[22] MIRVAKILI, Seyed M.; HUNTER, Ian W.. *Artificial Muscles: mechanisms,*

applications, and challenges. **Advanced Materials**, [S.L.], v. 30, n. 6, p. 1704407, 18 dez. 2017. Wiley. <http://dx.doi.org/10.1002/adma.201704407>.

[23] GREEN, Matthew D.; LONG, Timothy E.. Designing Imidazole-Based Ionic Liquids and Ionic Liquid Monomers for Emerging Technologies. **Polymer Reviews**, [S.L.], v. 49, n. 4, p. 291-314, 30 out. 2009. Informa UK Limited. <http://dx.doi.org/10.1080/15583720903288914>.

[24] AKLE, Barbar J.; BENNETT, Matthew D.; LEO, Donald J.. High-strain ionomeric-ionic liquid electroactive actuators. **Sensors And Actuators A: Physical**, [S.L.], v. 126, n. 1, p. 173-181, jan. 2006. Elsevier BV. <http://dx.doi.org/10.1016/j.sna.2005.09.006>.

[25] ONSORTI, Crestina S.; SOUZA, Roberto F. de; DUPONT, Jairton. LÍQUIDOS IÔNICOS CONTENDO O CÁTION DIALQUILIMIDAZÓLIO: ESTRUTURA, PROPRIEDADES FÍSICO-QUÍMICAS E COMPORTAMENTO EM SOLUÇÃO. **Quim. Nova**, Porto Alegre - Rs, v. 24, n. 6, p. 830-837, 23 mar. 2001.

[26] DIANI, Julie; GILORMINI, Pierre. On necessary precautions when measuring solid polymer linear viscoelasticity with dynamic analysis in torsion. **Polymer Testing**, [S.L.], v. 63, p. 275-280, out. 2017. Elsevier BV. <http://dx.doi.org/10.1016/j.polymertesting.2017.08.025>.

[27] DESSI, Claudia; TSIBIDIS, George D.; VLASSOPOULOS, Dimitris; CORATO, Marco de; TROFA, Marco; D'AVINO, Gaetano; MAFFETTONE, Pier Luca; COPPOLA, Salvatore. Analysis of dynamic mechanical response in torsion. **Journal Of Rheology**, [S.L.], v. 60, n. 2, p. 275-287, mar. 2016. Society of Rheology. <http://dx.doi.org/10.1122/1.4941603>.

[28] KLEIN, Bernd. **Python Machine Learning Tutorial: neural network**. Neural Network. 2020. Disponível em: [https://www.python-course.eu/neural\\_network\\_mnist.php](https://www.python-course.eu/neural_network_mnist.php). Acesso em: 06 set. 2021.

[29] MORRISON, Faith A.. **Understanding Rheology**. Nova York: Oxford University Press, 2001. 538 p.

[30] CALLISTER, William D.; RETHWISCH, David G.. **Materials Science and Engineering: an introduction**. 9. ed. New York: Wiley, 2014. 937 p.

[31] CANEVAROLO JUNIOR, Sebastião V.. **TÉCNICAS DE CARACTERIZAÇÃO DE POLÍMEROS**. São Paulo: Artliber Editora Ltda., 2004. 448 p.

[32] WANG, Yingying; LI, Yibin; SONG, Yong; RONG, Xuewen. The Influence of the Activation Function in a Convolution Neural Network Model of Facial Expression Recognition. **Applied Sciences**, [S.L.], v. 10, n. 5, p. 1897, 10 mar. 2020. MDPI AG. <http://dx.doi.org/10.3390/app10051897>.

[33] BROWNLEE, Jason. **Softmax Activation Function with Python**. 2020. Disponível em: <https://machinelearningmastery.com/softmax-activation-function-with-python/>. Acesso em: 11 out. 2021.

[34] WOOD, Thomas. **What is the Softmax Function?** Disponível em: <https://deeptai.org/machine-learning-glossary-and-terms/softmax-layer>. Acesso em: 19 out. 2021.

[35] BROWNLEE, Jason. **Loss and Loss Functions for Training Deep Learning Neural Networks**. 2019. Disponível em: <https://machinelearningmastery.com/loss-and-loss-functions-for-training-deep-learning-neural-networks/>. Acesso em: 27 out. 2021.

[36] ALGORITHMIA. **Introduction to loss functions**. 2018. Disponível em: <https://algorithmia.com/blog/introduction-to-loss-functions>. Acesso em: 26 out. 2021.

[37] Nabiyev, Nabi & Malekzadeh, Saber. (2021). Anomalous Sound Localization Estimation. 10.13140/RG.2.2.25949.95201.

## APÊNDICE A – CÓDIGO EM PYTHON DA REDE NEURAL IMPLEMENTADA NESTE TRABALHO

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Sun Aug  8 18:11:21 2021
@author: lia

Your First Deep Learning Project in Python with Keras Step-By-Step
https://machinelearningmastery.com/tutorial-first-neural-network-python-keras/

Neural Network
#https://www.python-course.eu/neural\_network\_mnist.php

"""
import matplotlib.pyplot as plt
import random as rnd

from numpy import loadtxt
import numpy as np

# Multilayer Perceptron (MLP) for multi-class softmax classification
# https://faroit.com/keras-docs/1.0.1/getting-started/sequential-model-guide/
from keras.models import Sequential
from keras.layers import Dense, Dropout, Activation
from keras.optimizers import SGD

#=====
def carrega_dados(qual):
```

```

# load the dataset
if qual=='Nafion':
    dd = loadtxt('Nafion.csv', delimiter=',')
    plt.figure(1)

elif qual=='IPMC':
    dd = loadtxt('IPMC.csv', delimiter=',')
    plt.figure(2)

#-----
# split into input (X) and output (y) variables
X = dd[:,0]
y = dd[:,1:7]

#-----
# gera graficos dos dados
nome = ['seco H+', 'seco LI 0,5M', 'seco LI 0,2M', 'úmido H+', 'úmido LI
0,5M', 'úmido LI 0,2M']
for i in range(6):
    #plt.plot(X,y[:,i],'.-')
    plt.title(qual)
    plt.semilogx(X,y[:,i],'.-',label=nome[i])
    #plt.loglog(X,y[:,i],'.-',label=nome[i])
    plt.xlabel('w (rad/s)')
    plt.ylabel("G'")
    plt.legend()
#plt.axis([0.01,1.1,0,1.2])
if qual=='Nafion':
    plt.savefig('Nafion_medidas'+'.jpg')
elif qual=='IPMC':
    plt.savefig('IPMC_medidas'+'.jpg')

return dd

#=====
def prepara_dados(dd):
    #rnd.seed(15) # to reproduce the sequences given by a pseudo-random
number generator
    #-----
    #coloca os dados tabelados medidos na forma de 3 colunas: classe, freq.
e G'
    dados_em_3_colunas = np.zeros((numero_exemplos,3))

    linha = 0
    for j in range(num_classes):
        for i in range(num_frequencias):
            dados_em_3_colunas[linha,0] = j # tratamento (classe)
            dados_em_3_colunas[linha,1] = dd[i,0] # frequencia
            dados_em_3_colunas[linha,2] = dd[i,j+1] # G'
            linha = linha + 1

    #-----
    #separa os dados entre treinamento e teste
    lista_sorteio = list(range(numero_exemplos)) # indices dos exemplos
para sorteio
    ordem = rnd.sample(lista_sorteio,numero_exemplos) # embaralha a ordem
de treinamento

    #print(ordem)
    #print(ordem[0:num_treino])

```

```

train_data = dados_em_3_colunas[ordem[0:num_treino],:]
test_data = dados_em_3_colunas[ordem[num_treino:],:]

return train_data, test_data

#=====
def prepara_imgs(train_data, test_data, dd):
    train_imgs = np.zeros((num_treino,2))
    test_imgs = np.zeros((num_teste,2))

    fac = 0.99 / np.max(dd[:,0]) # pega o maximo da frequencia
    train_imgs[:,0] = train_data[:,1] * fac + 0.01 # normaliza a frequencia
    test_imgs[:,0] = test_data[:,1] * fac + 0.01 # normaliza a frequencia

    fac = 0.99 / np.max([np.max(train_data[:,2]),np.max(test_data[:,2])]) #
pega o maximo de G'
    train_imgs[:,1]=train_data[:,2] * fac + 0.01 # normaliza
    test_imgs[:,1]=test_data[:,2] * fac + 0.01 # normaliza

    plt.figure(3)
    plt.semilogx(train_imgs[:,0],train_imgs[:,1],'.b', label='treino')
    plt.semilogx(test_imgs[:,0],test_imgs[:,1],'.r', label='teste')
    #plt.plot(train_imgs[:,0],train_imgs[:,1],'.b', label='treino')
    #plt.plot(test_imgs[:,0],test_imgs[:,1],'.r', label='teste')
    plt.xlabel('w (rad/s)')
    plt.ylabel("G'")
    plt.axis([0.02,1.1,0,1.2])
    plt.legend()
    plt.title('conjunto de teste (normalizado)')
    if qual=='Nafion':
        plt.savefig('Nafion_teste_normalizado'+'.jpg')
    elif qual=='IPMC':
        plt.savefig('IPMC_teste_normalizado'+'.jpg')

    return train_imgs, test_imgs

#=====
def prepara_labels(train_data, test_data):
    train_labels = np.asarray(train_data[:, :1])
    test_labels = np.asarray(test_data[:, :1])

    lr = np.arange(num_classes)

    # transforma labels em representação one hot
    train_labels_one_hot = (lr==train_labels).astype(float)
    test_labels_one_hot = (lr==test_labels).astype(float)

    print('.train_labels')
    print(train_labels[1:5])
    print('.train_labels_one_hot')
    print(train_labels_one_hot[1:5])

    # como não desejamos zeros nem uns nos labels:
    train_labels_one_hot[train_labels_one_hot==0] = 0.01
    train_labels_one_hot[train_labels_one_hot==1] = 0.99
    test_labels_one_hot[test_labels_one_hot==0] = 0.01
    test_labels_one_hot[test_labels_one_hot==1] = 0.99

    print('.train_labels_one_hot centrado')
    print(train_labels_one_hot[1:5], '\n')

```

```

return train_labels_one_hot, test_labels_one_hot

#=====
# grafico do dado de teste com ruído
def grafico_ruído(train_imgs, test_imgs, sinal):
    plt.figure(4)
    plt.semilogx(train_imgs[:,0],train_imgs[:,1],'.b', label='treino')
    plt.semilogx(test_imgs[:,0],test_imgs[:,1],'.r', label='teste')

    amplitude=0.012 # não é porcentagem aqui
    for i in range(len(sinal)):
        tmpx = (1 + amplitude*sinal[i]) * test_imgs[i,0]
        tmpy = (1 + amplitude*sinal[i]) * test_imgs[i,1]
        plt.semilogx(tmpx,tmpy, '.k', mfc='none')

    plt.xlabel('w (rad/s)')
    plt.ylabel("G'")
    plt.axis([0.02,1.1,0,1.2])
    plt.legend()
    titulo = 'conjunto de teste com ruído
(normalizado)\n'+str(amplitude*100)+'%'
    plt.title(titulo)
    if qual=='Nafion':
        plt.savefig('Nafion_teste_normalizado_ruído'+'.jpg')
    elif qual=='IPMC':
        plt.savefig('IPMC_teste_normalizado_ruído'+'.jpg')

#=====
# grafico da acurácia em função do ruído
def grafico_acuracia_ruído(x,y):
    plt.figure(5)
    #plt.plot(abs(ruído),score[0],'.r',abs(ruído),score[1],'.k')
    plt.plot(x,y, '.k')
    plt.title('acurácia em função do ruído')
    plt.xlabel('ruído (%)')
    plt.ylabel('acurácia')
    plt.ylim(0,1.1)
    if qual=='Nafion':
        plt.savefig('Nafion_acuracia_ruído'+'.jpg')
    elif qual=='IPMC':
        plt.savefig('IPMC_acuracia_ruído'+'.jpg')

#=====
# programa principal
#=====
# O conjunto de dados experimentais tem 20 pontos medidos (frequencias)
# de cada uma das 6 classes (tratamento da amostra)

# Quando fazemos o treinamento sem ruído, são 20 * 6 = 120 exemplos no
total

# Cada linha do conjunto de dados para treinamento/teste consiste de 3
números (colunas):
#   -primeiro: a classe, entre 0 e 6
#   -segundo: a frequencia, entre 1 e 80
#   -terceiro: o módulo de perda G', entre 400 e 1100 (verificar
necessidade de reescalar)

# O primeiro número de cada linha é a classe (ou label) -- equivalente aos
dígitos de 0 a 9
# os outros 2 números são equivalentes à imagem dos dígitos

```

```

# no problema dos CEPs, havia:
#   -60000 linhas de treino
#   -10000 linhas de teste

# w (rad/s)
# 0 - seco H+
# 1 - seco LI 0,5M
# 2 - seco LI 0,2M
# 3 - úmido H+
# 4 - úmido LI 0,5M
# 5 - úmido LI 0,2M

num_classes = 6
num_frequencias = 20

num_treino = 100
num_teste = 120-num_treino

numero_exemplos=num_classes*num_frequencias # 6 classes * 20 medidas = 120
exemplos

#-----
#qual='Nafion'
qual='IPMC'
d = carrega_dados(qual)

#-----
train_data, test_data = prepara_dados(d)

#-----
train_imgs, test_imgs = prepara_imgs(train_data, test_data, d)

#-----
train_labels_one_hot, test_labels_one_hot = prepara_labels(train_data,
test_data)

#-----

X_train = train_imgs
y_train = train_labels_one_hot

X_test = test_imgs
y_test = test_labels_one_hot

#-----
# define a rede neural
model = Sequential()
# Dense(16) is a fully-connected layer with 16 hidden units.
# in the first layer, you must specify the expected input data shape:
# here, 20-dimensional vectors.
model.add(Dense(16, input_dim=20))
model.add(Activation('tanh'))
model.add(Dropout(0.1))

model.add(Dense(16))
model.add(Activation('tanh'))
model.add(Dropout(0.1))

model.add(Dense(6))
model.add(Activation('softmax'))

```

```

sgd = SGD(lr=0.1, decay=1e-6, momentum=0.9, nesterov=True)
model.compile(loss='categorical_crossentropy',
              optimizer=sgd,
              metrics=['accuracy'])

#-----
# treina a rede neural
model.fit(X_train, y_train, epochs=250, batch_size=160)

#-----
# testa a rede neural
print('----teste')
score = model.evaluate(X_test, y_test, batch_size=160)

#-----
# testa a rede neural com ruído
print('----teste com ruído')
# grafico da acurácia em função do ruído
graf_ruído_x=[]
graf_ruído_y=[]

sinal = [] # sinal aleatório do ruído
for j in range(len(X_test)):
    sinal.append(2.0*rnd.randint(0, 1) - 1.0)

# faz o gráfico dado de teste com ruído
grafico_ruído(train_imgs, test_imgs, sinal)

for i in range(20): # loop para aumentar intensidade do ruído
    X_ruído = X_test
    amplitude = i/500
    for j in range(len(X_test)):
        ruído = 1.0 + amplitude * sinal[j]
        X_ruído[j,0] = ruído * X_test[j,0]
        X_ruído[j,1] = ruído * X_test[j,1]

    print('ruído: ', amplitude, ' ==> ', (1.0-abs(ruído))*100)
    score = model.evaluate(X_ruído, y_test, batch_size=160)

    graf_ruído_x.append(amplitude*100)
    graf_ruído_y.append(score[1])

# plota o gráfico do ruído
grafico_acuracia_ruído(graf_ruído_x, graf_ruído_y)

```

