



UNIVERSIDADE FEDERAL DE SANTA CATARINA  
CENTRO TECNOLÓGICO  
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Jone Follmann

**Aplicação de redes neurais pulsadas no processamento de sinais de  
mecanorreceptores táteis eletrônicos**

Florianópolis  
2023

Jone Follmann

**Aplicação de redes neurais pulsadas no processamento de sinais de  
mecanorreceptores táteis eletrônicos**

Dissertação submetida ao Programa de Pós-Graduação  
em Engenharia Elétrica da Universidade Federal de  
Santa Catarina para a obtenção do título de mestre  
em Engenharia Elétrica.

Orientador: Prof. Cesar Ramos Rodrigues, Dr.

Florianópolis  
2023

Ficha de identificação da obra elaborada pelo autor,  
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Follmann, Jone

Aplicação de redes neurais pulsadas no processamento de  
sinais de mecanorreceptores táteis eletrônicos / Jone  
Follmann ; orientador, Cesar Ramos Rodrigues, 2023.

80 p.

Dissertação (mestrado) - Universidade Federal de Santa  
Catarina, Centro Tecnológico, Programa de Pós-Graduação em  
Engenharia Elétrica, Florianópolis, 2023.

Inclui referências.

1. Engenharia Elétrica. 2. Detecção de escorregamento.  
3. Detecção de toque. 4. Redes Neurais Pulsadas. 5.  
Próteses. I. Ramos Rodrigues, Cesar. II. Universidade  
Federal de Santa Catarina. Programa de Pós-Graduação em  
Engenharia Elétrica. III. Título.

Jone Follmann

**Aplicação de redes neurais pulsadas no processamento de sinais de mecanorreceptores táteis eletrônicos**

O presente trabalho em nível de mestrado foi avaliado e aprovado por banca examinadora composta pelos seguintes membros:

Prof. Cesar Ramos Rodrigues, Dr.  
Universidade Federal de Santa Catarina - UFSC

Prof. (a) Daniela Ota Hisayasu Suzuki, Dr.  
Universidade Federal de Santa Catarina - UFSC

Prof. Érico Marcelo Hoff do Amaral, Dr.  
Universidade Federal do Pampa - UNIPAMPA

Certificamos que esta é a **versão original e final** do trabalho de conclusão que foi julgado adequado para obtenção do título de mestre em Engenharia Elétrica.

---

Prof. Telles Brunelli Lazzarin, Dr.  
Coordenação do Programa de  
Pós-Graduação

---

Prof. Cesar Ramos Rodrigues, Dr.  
Orientador

Florianópolis, 2023.

Este trabalho é dedicado a Deus.

## **AGRADECIMENTOS**

Agradeço a Deus pela oportunidade que me deste de aprender, pela força e a Fé que me permitiram realizar este trabalho.

Agradeço ao meus pais, Marlene Oberger Follmann e Albino José Follmann, por todo o apoio durante estes anos de estudos. Sem o suor de suas camisas, chegar até aqui não seria possível.

Agradeço ao meu orientador, Prof. Cesar Ramos Rodrigues pelas infinitas orientações, pela dedicação com este trabalho, pela inspiração, motivação, carinho e conhecimento. Sempre digo que fui agraciado com orientadores ativos e eficientes em minha jornada acadêmica até aqui.

Agradeço a todos os professores e colegas que de alguma forma contribuíram para minha formação.

Agradeço aos meus familiares, aos amigos de aquário (GAEA), minha namorada, por todo o apoio e amizade. E agradeço a todos aqueles que de alguma forma me auxiliaram a realizar estes estudos.

Que todos os seres sejam Felizes, que todos seres sejam Ditosos, que todos os serem estejam em Paz.

AOM

*“Quien quiera conocer  
todas las maravillas de la naturaleza,  
debe estudiarlas dentro de sí mismo”.*  
*(V.M Samael Aun Weor)*

## RESUMO

As próteses mioelétricas poliarticuladas de membro superior aumentaram muito a destreza e as possibilidades de integração às estruturas esqueléticas, musculares e até mesmo neurais dos pacientes. Toda tarefa de manipulação tem como condições elementares a identificação do toque e do deslizamento. Neste cenário, quando não há qualquer interação entre o dedo da prótese e os objetos agarrados, a manipulação de objetos se torna um desafio, podendo levar a instabilidade ou escorregamento dos objetos manipulados. Como alternativa às abordagens algorítmicas atuais para detecção de toque e deslizamento, investigamos a implementação de redes neurais pulsadas visando futura integração de *feedback* tátil em próteses de mão, o que permitiria ao usuário a percepção de sensações táteis. Para isso, treinamos uma rede neural de pulsada para identificar eventos de toque e deslizamento durante a apreensão realizada pela mão robótica IH2 Azzurra. Os dados do sensor acoplado à mão robótica são codificados por uma população heterogênea de neurônios, modelados para corresponder à atividade pulsada das fibras mecanorreceptoras. Com base na atividade pulsátil contendo informações de tato, a camada de saída da rede classifica os eventos de toque ou deslizamento. Desta forma, o modelo é capaz de detectar eventos usando apenas o componente de força normal. É feita uma comparação com os resultados obtidos pelos autores em um trabalho anterior, bem como com trabalhos correlatos. Além das vantagens inerentes à abordagem da rede neural pulsada, como plausibilidade biológica, o método obteve uma alta precisão na detecção de toque e 100% de precisão na classificação de deslizamento e não deslizamento. Além disso, o potencial implementação de *hardware* de ultrabaixa potência é favorável em comparação com processadores, podendo diminuir em até 162 vezes menos consumo de energia na implantação das redes em *hardwares* neuromórficos. Desta forma, os resultados promissores sugerem a validação da utilização de redes neurais pulsadas para a classificação de toque e escorregamento em mãos protéticas.

**Palavras-chave:** Detecção de escorregamento. Detecção de toque. Redes Neurais Pulsadas. Próteses. Biologicamente plausível.



## ABSTRACT

Poliarticulated myoelectric upper limb prostheses have greatly enhanced dexterity and integration possibilities to patients skeletal, muscular and even neural structures. Every manipulation task has, as elemental conditions, touch and slippage identification. Therefore, when there is no interaction between the prosthetic fingers and the grasped objects, the manipulation becomes a challenge, which might cause instability or slipping of the manipulated objects. As an alternative to the algorithmic approach to touch and slippage detection, we investigate the implementation of spiking neural networks aiming the future integration of tactile feedback into hand prosthesis, which would allow the user to perceive tactile sensations. To this end, we trained a spiking neural network for identifying touch and slippage events during grasping performed by the IH2 Azzurra robotic hand. The sensor data is encoded by a heterogeneous population of neurons, modeled to match the spiking activity of mechanoreceptor cells. Given the spiking activity that contains tactile information, the output layer classifies the touch or slip events. Therefore, the model is capable of detecting events using only the normal force component. A comparison with the results obtained from the authors in a previous work is made, also with related works. In addition to the inherent advantages of the spiking neural network approach, such as biological plausibility, with the method we achieved a great accuracy of touch detection and 100% accuracy in the classification of slip and non-slip events. Furthermore, the potential implementation of ultra-low-power hardware is favorable compared to processors, and can decrease up to 162 times less power consumption when deploying the networks on neuromorphic hardware. Thus, the promising results suggest the validation of using spiking neural networks for touch and slip classification in prosthetic hands.

**Keywords:** Slippage detection. Touch detection. Spiking Neural Networks. Prostheses. Biologically-inspired.

## LISTA DE FIGURAS

Figura 1 – Informações mecanossensoriais transmitidas da ponta do dedo para o cérebro (Adaptado de (PURVES D., 2004)). . . . .	20
Figura 2 – Mecanorreceptores humanos. (a) Seção transversal da pele glabra (KANDEL <u>et al.</u> , 2000). (b) O campo receptivo de cada tipo de mecanorreceptores (KANDEL <u>et al.</u> , 2000). (c) Os trens de pulsos de cada tipo de mecanorreceptores em resposta a um estímulo específico (KANDEL <u>et al.</u> , 2000). (d) A densidade de cada tipo de mecanorreceptores (VALLBO, A. B.; JOHANSSON, R. S., 1984). . . . .	21
Figura 3 – Sistema biomimético para reproduzir o sentido do tato humano (Adaptado de (OSBORN <u>et al.</u> , 2018; YI; ZHANG; PETERS, 2018)) . . . .	25
Figura 4 – Modelo de mecanorreceptor proposto por Kim (Adaptado de (KIM, S. S.; SRIPATI; BENSMAIA, Sliman J, 2010)). . . . .	26
Figura 5 – Respostas de simulação para fibras aferentes SA e FA-I (Adaptado de (KIM, S. S.; SRIPATI; BENSMAIA, Sliman J, 2010)). . . . .	27
Figura 6 – Resumo dos princípios de representação e transformação do NEF (Adaptado de (BEKOLAY <u>et al.</u> , 2014)) . . . . .	31
Figura 7 – Ilustração do funcionamento da validação cruzada k-fold (Adaptado de (RASCHKA, 2018)) . . . . .	33
Figura 8 – Resumo do trabalho de Veiga et al. (2015) (Adaptado de (VEIGA <u>et al.</u> , 2015)). . . . .	35
Figura 9 – Resumo do trabalho de Begalinova et al. (2020) (Adaptado de (BEGALINOVA <u>et al.</u> , 2020)). . . . .	36
Figura 10 – Configuração para a primeira sessão experimental (GENTILE <u>et al.</u> , 2020). . . . .	39
Figura 11 – Demonstração de deslizamento da sonda sobre o sensor FSR (GENTILE <u>et al.</u> , 2020). . . . .	40
Figura 12 – Exemplo de resultado da primeira sessão experimental para a combinação da força C e tempo C (Adaptado de (GENTILE <u>et al.</u> , 2020)).	41
Figura 13 – (a) A mão robótica IH2 segurando o cubo equipado com o manipulador passivo e efetor tipo garra; (b) Configuração para a segunda sessão experimental (GENTILE <u>et al.</u> , 2020). . . . .	42
Figura 14 – 30 escorregamentos induzidos pelo robô KUKA (Adaptado de (GENTILE <u>et al.</u> , 2020)). . . . .	43
Figura 15 – Arquitetura da SNN para classificação de toque . . . . .	44
Figura 16 – Exemplo de referência criada para detecção do toque . . . . .	46
Figura 17 – Ilustração da etapas de busca dos melhores hiperparâmetros para a arquitetura de detecção de toque. . . . .	48

Figura 18 – Arquitetura SNN para classificar o deslizamento . . . . .	49
Figura 19 – Referência criada para detecção de escorregamento . . . . .	50
Figura 20 – Etapas de seleção dos melhores modelos de detecção de escorregamento. . . . .	51
Figura 21 – Gráfico paralelo a média dos resultados na variação dos hiperparâmetros para a SNN de detecção de toque. . . . .	54
Figura 22 – Matriz de gráficos de dispersão dos resultados para a arquitetura da SNN de detecção de toque. . . . .	55
Figura 23 – Resultados da simulação com os dados do teste Força A e Velocidade A da primeira sessão experimental, para a SNN de detecção de toque . . . . .	56
Figura 24 – Gráfico do consumo total de energia estimado por classificação para os <i>hardwares</i> Loihi, SpiNNaker 2, CPU e ARM (SNN de classificação de toque). . . . .	57
Figura 25 – Gráfico paralelo dos resultados na variação dos hiperparâmetros para a SNN de detecção de escorregamento. . . . .	58
Figura 26 – Matriz de gráficos de dispersão dos resultados para a arquitetura da SNN de detecção de escorregamento. . . . .	59
Figura 27 – Resultados da simulação com os dados da segunda sessão experimental, para a SNN de detecção de escorregamento . . . . .	60
Figura 28 – Gráfico do consumo total de energia estimado por classificação para os <i>hardwares</i> Loihi, SpiNNaker 2, CPU e ARM (SNN de classificação de escorregamento). . . . .	61

## LISTA DE TABELAS

Tabela 1 – Características dos mecanorreceptores humanos (YI; ZHANG; PETERS, 2018). . . . .	22
Tabela 2 – Parâmetros de força e tempo usados na primeira sessão experimental para deslizamento da sonda sobre o sensor FSR. . . . .	40
Tabela 3 – Parâmetros do neurônio LIF. . . . .	45
Tabela 4 – Resultados dos treinamentos para a SNN de toque que obtiveram melhor performance para cada quantidade de neurônios na primeira camada. . . . .	56
Tabela 5 – Resultados dos treinamentos para a SNN de escorregamento que obtiveram melhor performance para cada quantidade de neurônios na primeira camada. . . . .	60
Tabela 6 – Desempenho da SNN e abordagens da literatura. . . . .	68

## LISTA DE ABREVIATURAS E SIGLAS

AUC	<i>Area Under the ROC Curve</i>
CPU	<i>Central Processing Unit</i>
CV	<i>Cross-Validation</i>
DNN	<i>Deep Neural Network</i>
DoFs	<i>Degrees-of-Freedom</i>
FA	<i>Fast Adaption</i>
FA-I	Corpúsculos de Meissner
FA-II	Corpúsculo de Pacini
FN	falsos positivos
FP	falsos positivos
FSR	<i>Force Sensitive Resistor</i>
HH	Hodgkin-Huxley
HMM	Modelos Ocultos de Markov
LIF	<i>Leakly Integrate-and-Fire</i>
LTSM	<i>Long Short-Term Memory</i>
MAC	<i>Multiply-Accumulate</i>
MEMS	<i>Micro-Electro-Mechanical System</i>
NEF	<i>Neural Engineering Framework</i>
NI	<i>National Instruments</i>
RNA	Redes Neurais Artificiais
SA	<i>Slow Adaption</i>
SA-I	Discos de Merkel
SA-II	Cilindros de Ruffini
SNC	Sistema Nervoso Central
SNN	<i>Spiking Neural Network</i>
SVM	<i>Support Vector Machine</i>
VN	verdadeiros negativos
VP	verdadeiros positivos

## SUMÁRIO

<b>1</b>	<b>INTRODUÇÃO</b>	<b>15</b>
1.1	OBJETIVOS	17
<b>1.1.1</b>	<b>Objetivo Geral</b>	<b>17</b>
<b>1.1.2</b>	<b>Objetivos Específicos</b>	<b>17</b>
1.2	ORGANIZAÇÃO DA DISSERTAÇÃO	18
<b>2</b>	<b>REFERENCIAL TEÓRICO</b>	<b>19</b>
2.1	SENTIDO HUMANO DO TATO	19
<b>2.1.1</b>	<b>Mecanorreceptores</b>	<b>20</b>
<b>2.1.2</b>	<b>Percepção de estímulos táteis</b>	<b>22</b>
2.2	SENTIDO BIOMIMÉTICO DO TATO	24
<b>2.2.1</b>	<b>Codificação de estímulos táteis</b>	<b>25</b>
2.2.1.1	Modelos de mecanorreceptores	26
2.2.1.2	Modelos de Neurônios	27
2.3	REDES NEURAIS PULSADAS	28
<b>2.3.1</b>	<b>Nengo</b>	<b>30</b>
<b>2.3.2</b>	<b>Técnicas de treinamento e avaliação de redes neurais</b>	<b>32</b>
2.4	DETECÇÃO DE ESCORREGAMENTO	34
<b>3</b>	<b>CLASSIFICAÇÃO DE TOQUE E ESCORREGAMENTO EM MÃOS PROTÉTICAS UTILIZANDO REDES NEURAIS PULSADAS</b>	<b>38</b>
3.1	MATERIAIS E MÉTODOS	38
<b>3.1.1</b>	<b>Coleta e preparação de dados</b>	<b>38</b>
3.1.1.1	Primeira sessão experimental	39
3.1.1.2	Segunda sessão experimental	41
<b>3.1.2</b>	<b>Arquitetura da SNN para detecção de toque</b>	<b>43</b>
3.1.2.1	Rotulagem dos dados	46
3.1.2.2	Aprendizado e seleção do melhor modelo	46
<b>3.1.3</b>	<b>Arquitetura da SNN para detecção de escorregamento</b>	<b>49</b>
3.1.3.1	Rotulagem de dados	50
3.1.3.2	Simulação e seleção do melhor modelo	50
<b>4</b>	<b>RESULTADOS</b>	<b>53</b>
4.1	CLASSIFICAÇÃO DE TOQUE	53
4.2	CLASSIFICAÇÃO DE ESCORREGAMENTO	58
<b>5</b>	<b>DISCUSSÕES</b>	<b>62</b>
5.1	CLASSIFICAÇÃO DE TOQUE	62
5.2	CLASSIFICAÇÃO DE ESCORREGAMENTO	65
5.3	COMPARAÇÕES COM OUTRAS ABORDAGENS	67
<b>6</b>	<b>CONCLUSÕES E CONSIDERAÇÕES FINAIS</b>	<b>69</b>

	<b>Referências . . . . .</b>	<b>71</b>
	<b>APÊNDICE A – CÁLCULOS DE CONSUMO DE ENERGIA . . . .</b>	<b>78</b>
	<b>APÊNDICE B – PUBLICAÇÕES . . . . .</b>	<b>80</b>
B.1	<b>ARTIGOS EM CONGRESSOS . . . . .</b>	<b>80</b>

## 1 INTRODUÇÃO

A amputação de membros superiores, especialmente a perda de mãos e dedos, resulta em deficiências severas para suas vítimas. Segundo as estimativas usando os resultados da carga global de doenças em 2017, 57,7 milhões de pessoas viviam com amputação de membros devido a causas traumáticas em todo o mundo (MCDONALD *et al.*, 2020). A remoção de um membro do corpo, parcial ou total, pode ser realizada devido a diferentes motivos como, por exemplo, acidentes graves e diabetes (DORLAN, 1999), e afeta diretamente a capacidade física dos indivíduos, incluso a manipulação de objetos em tarefas do dia a dia e a perda do sentido de toque.

A perda de um membro ou parte dele é um fator que esteve sempre presente na humanidade, e o homem procurou desenvolver diferentes maneiras de substituição, uma delas as próteses. Podemos defini-la como sendo um dispositivo artificial, cujo objetivo é substituir membros ausentes ou má formação congênita (BOCCOLINI, 2000).

Ao longo dos anos, houve vários avanços tecnológicos, e desta forma, melhores e mais sofisticadas próteses são desenvolvidas usando robótica, para recuperar não só o controle do usuário (posição e movimento), mas a função sensorial responsável pelo *feedback* tátil (LUIS; MONCAYO, 2019).

Nesse sentido, uma das maiores limitações para um amputado que usa uma prótese sem *feedback* sensorial, é a dificuldade de manejar eventos inesperados de forma autônoma. Assim, um interesse crescente tem sido direcionado às ações de preensão e manipulação realizadas por mãos robóticas e protéticas. Em tarefas de preensão e manipulação, a possibilidade de deslizamento do objeto é alta. Por este motivo, um dos principais aspectos a serem considerados na avaliação é a estabilidade do objeto agarrado, que poderia ser garantida pela prevenção do deslizamento do objeto (GENTILE *et al.*, 2020)

Entre as várias sensações humanas, o sentido de tato nos fornece a sensibilidade que permite discriminar várias texturas superficiais, agarrar e manipular objetos com precisão. Ao contrário dos outros sentidos baseados em órgãos sensoriais, o sentido de toque surge de receptores distribuídos por todo o corpo (CHAUDHURI, 2011). A modalidade de toque na categoria cutâneo (tátil), depende de receptores embutidos na pele, e para um ser humano apto, os estímulos de força e vibração podem ser detectados por mecanorreceptores (BOFF; KAUFMAN; THOMAS, 1986).

Existem quatro tipos principais de mecanorreceptores, e cada um é responsável pela recepção de estímulos específicos. Estes quatro incluem os corpúsculos de Pacini, corpúsculos de Meissner, discos de Merkel e cilindros de Ruffini, todos os quais podem ser divididos em duas categorias, mecanorreceptores de adaptação lenta (*Slow Adaption* (SA)) e mecanorreceptores de adaptação rápida (*Fast Adaption* (FA)) (JAMALI; SAMMUT, 2010). Os mecanorreceptores de adaptação lenta, respondem a estímulos



de baixa frequência e descrevem as propriedades estáticas de um estímulo. Em contraste, os estímulos de alta frequência disparam ao excitar os mecanorreceptores de adaptação rápida. Em resposta, os mecanorreceptores geram trem de picos, que por sua vez, são transportados para o Sistema Nervoso Central (SNC) para percepção no córtex somatossensorial (KIM, S. S. *et al.*, 2009).

Além disso, o mecanorreceptor biológico discutido pode ser substituído por um artificial, desde que seu sinal de saída possa ser interpretado pelo cérebro. Um mecanorreceptor tátil eletrônico, pode ser projetado usando sensores de pressão e mecanismos que possibilitem traduzir a força aplicada em uma quantidade elétrica (LEE *et al.*, 2019). Posteriormente, através do processamento dos sinais é necessário traduzir o sinal de entrada em um formato compreensível pelo cérebro (KIM, E. K. *et al.*, 2012), codificado como uma frequência de descarga e, em seguida, inseridos em eletrodos para realizar a estimulação cerebral (BORETIUS *et al.*, 2010). Desta forma, eletronicamente é possível desenvolver tecnologias em próteses, que permitam aos usuários recuperar a sensação tátil.

Na literatura, diferentes abordagens têm sido propostas para obter informações sobre o deslizamento de objetos durante as ações de apreensão e manipulação. As técnicas para detecção de escorregamento geralmente utilizam filtros de Kalman, transformada rápida de Fourier, aprendizado de máquina e modelos de limiar (GENTILE *et al.*, 2020). As principais desvantagens dos métodos acima são seu alto custo computacional, o que limita sua incorporação em eletrônicos de baixa potência, e a dificuldade de aplicação das metodologias e ferramentas tradicionais em sistemas embarcados. Também, como está bem estabelecido, muitas aplicações de robótica requerem baixa latência e economia de energia (HASLER; MARR, 2013; DEWOLF; JAWORSKI; ELI-ASMITH, 2020).

Quando tratamos de aprendizado de máquina, as Redes Neurais Profundas (*Deep Neural Network* (DNN)) possuem um desempenho impressionante em uma variedade de aplicações de classificação de imagem e sinal. Essa classe de algoritmo de aprendizado de máquina depende intrinsecamente de cálculos intensivos de multiplicação e acumulação (YANG *et al.*, 2020). Além disso, a diferença fundamental entre DNN, e outros algoritmos de Redes Neurais Artificiais (RNA) onde às contrapartes biológicas inspiraram seus princípios é a forma como a informação é representada e transmitida entre os neurônios (NGUYEN; TRAN; IACOPI, 2021).

Na terceira geração de redes neurais (PAUGAM-MOISY; BOHTE, 2012), também conhecidas como Redes Neurais Pulsadas (*Spiking Neural Network* (SNN)), a informação é codificada no tempo através de sequências de pulsos nos potenciais de ação, o que poderia facilitar sua integração aos tecidos nervosos biológicos. Além disso, as SNN consomem consideravelmente menos energia do que outras implementações de RNA e são consideradas adequadas para implementar aprendizado em tempo real

(YANG et al., 2020).

De acordo com essas características, investigamos uma abordagem neuromórfica para detectar o toque e deslizamento de objetos durante as ações de preensão e manipulação. Para isso, foram realizadas simulações com o Nengo, uma biblioteca Python desenvolvida para mapear funções para uma SNN biologicamente plausível (ELIASMITH; ANDERSON, 2003). As redes Nengo podem ser transferidas para processadores neuromórficos como o SpiNNaker e o Intel Loihi, ou para originar implementações customizadas em silício, que é o objetivo de longo prazo da pesquisa a qual este trabalho faz parte. Porém, inicialmente se faz necessário simular em *software* o correto funcionamento destas redes.

Portanto, dentro do contexto apresentado, esta dissertação tem como objetivo investigar uma solução biologicamente plausível para detectar toque e deslizamento em mãos protéticas. O método baseia-se na conversão de uma representação biológica da percepção do toque em uma rede artificial de neurônios pulsados, que é então treinada para realizar a função desejada. Para resumir, o estímulo físico é convertido em atividades pulsáteis em mecanorreceptores, uma camada de neurônios subsequente mapeia a atividade para extrair elementos salientes e, em seguida, decodifica os pulsos resultantes para classificar se ocorreu ou não um toque ou escorregamento.

## 1.1 OBJETIVOS

Nas seções abaixo estão descritos o objetivo geral e os objetivos específicos.

### 1.1.1 Objetivo Geral

Desenvolver soluções bioinspiradas utilizando redes neurais pulsadas para detecção de toque e deslizamento em mãos protéticas, através de sinais de mecanorreceptores táteis eletrônicos.

### 1.1.2 Objetivos Específicos

- Investigar topologias e parâmetros de SNN para mimetizar o comportamento de mecanorreceptores frente a sinais táteis;
- Avaliar e implementar uma arquitetura para SNN de classificação de toque;
- Avaliar e implementar uma arquitetura para SNN de classificação de escorregamento;
- Treinar SNN para obtenção dos hiperparâmetros que proporcionem o melhor desempenho na classificação de toque e escorregamento;
- Avaliar o desempenho da rede para simulações em tempo real;

- Realizar uma análise comparativa para o consumo de energia da solução em diferentes *hardwares*;
- Discutir e analisar os resultados obtidos;

## 1.2 ORGANIZAÇÃO DA DISSERTAÇÃO

Esta dissertação está estruturada da seguinte forma:

- Capítulo 2: Referencial teórico

Este capítulo apresenta uma revisão sobre os vários tópicos essenciais à fundamentação teórica desta dissertação. Aborda o sentido humano do tato, mecanismos biomiméticos que buscam representar esse sentido artificialmente, conceitos relacionados às redes neurais pulsadas e aos trabalhos relacionados ao tema desta dissertação.

- Capítulo 3: Classificação de toque e escorregamento em mãos protéticas utilizando redes neurais pulsadas

Este capítulo apresenta todo o processo de desenvolvimento das redes neurais pulsadas. Inicialmente são descritos os materiais e métodos utilizados, incluindo a coleta e preparação do dados. Em seguida são apresentadas as arquiteturas propostas para classificação de toque e escorregamento.

- Capítulo 4: Resultados

Este capítulo traz os resultados adquiridos com o treinamento das redes neurais propostas, as simulações realizadas, bem como as estimativas dos consumos de energia por classificação para diferentes implementações em *hardware*, incluindo processadores neuromórficos.

- Capítulo 5: Discussão

Este capítulo apresenta uma ampla discussão dos resultados adquiridos neste trabalho. A discussão aborda as conclusões obtidas acerca dos treinamentos realizados nas redes neurais pulsadas e a relação entre cada um dos parâmetros treinados. Além disso, são discutidos os benefícios da implementação em *hardwares* neuromórficos em termos de consumo de energia.

- Capítulo 6: Conclusão e considerações finais

Este capítulo traz as considerações finais deste trabalho, as perspectivas para o campo de sensores táteis, bem como, são resumidas as contribuições desta dissertação.

## 2 REFERENCIAL TEÓRICO

Neste capítulo é apresentada uma revisão sobre os vários tópicos essenciais à fundamentação desta dissertação. São abordados o sentido humano do tato e os mecanismos biomiméticos que buscam representar esse sentido artificialmente, conceitos relacionados às redes neurais pulsadas e os trabalhos correlatos.

### 2.1 SENTIDO HUMANO DO TATO

Entre várias sensações humanas como visão, audição, paladar e olfato, o tato é uma sensação necessária para interagir com os ambientes que circundam o ser humano (TIWANA; REDMOND; LOVELL, 2012). A sensibilidade proporcionada pelo sentido do tato nos permite discriminar diferentes texturas, agarrar e manipular objetos com precisão. Ao contrário dos outros sentidos, baseados em órgãos sensoriais discretos, o sentido do tato surge de receptores distribuídos por todo o corpo (CHAUDHURI, 2011). Nesse sentido, a modalidade de toque pode ser dividida em três categorias: cutânea (tátil), cinestésica e háptica, sendo que cada uma é caracterizada pelos seus circuitos neurais e pela dinâmica correspondente (JONES, 2018).

Os sistemas cutâneo e cinestésico diferem em termos da localização dos mecanorreceptores em resposta às entradas sensoriais. Os primeiros contam com os receptores embutidos na pele, onde capturam informações dos estímulos, tais como força, pressão e vibração, aplicados sobre a superfície da pele. Já a resposta do sistema cinestésico é originada nos receptores localizados nas articulações, músculos e tendões, fornecendo informações sobre a postura corporal (BOFF; KAUFMAN; THOMAS, 1986). Já o sistema háptico abrange diretamente a sensação tátil ativa, ou seja, se faz presente quando movemos nossa mão sobre algum tecido e compreendemos o quão liso ou rugoso ele é. Essa percepção resulta da integração, pelo sistema háptico, das informações cinestésicas geradas pelo movimento e das sensações táteis formadas pelo contato entre a pele a superfície explorada (JONES, 2018).

Existem dois caminhos principais para permitir que a informação sensorial alcance o SNC, a via coluna dorsal-lemnisco medial e a via espinotalâmica (KANDEL et al., 2000). A via coluna dorsal-lemnisco medial concentra-se principalmente na transmissão da informação tátil e proprioceptiva, enquanto a via espinotalâmica realiza principalmente a transmissão de substâncias nocivas, viscerais e informações térmicas (KANDEL et al., 2000). A Figura 1 mostra o caminho das informações mecanossensoriais transmitidas da ponta do dedo para o cérebro.

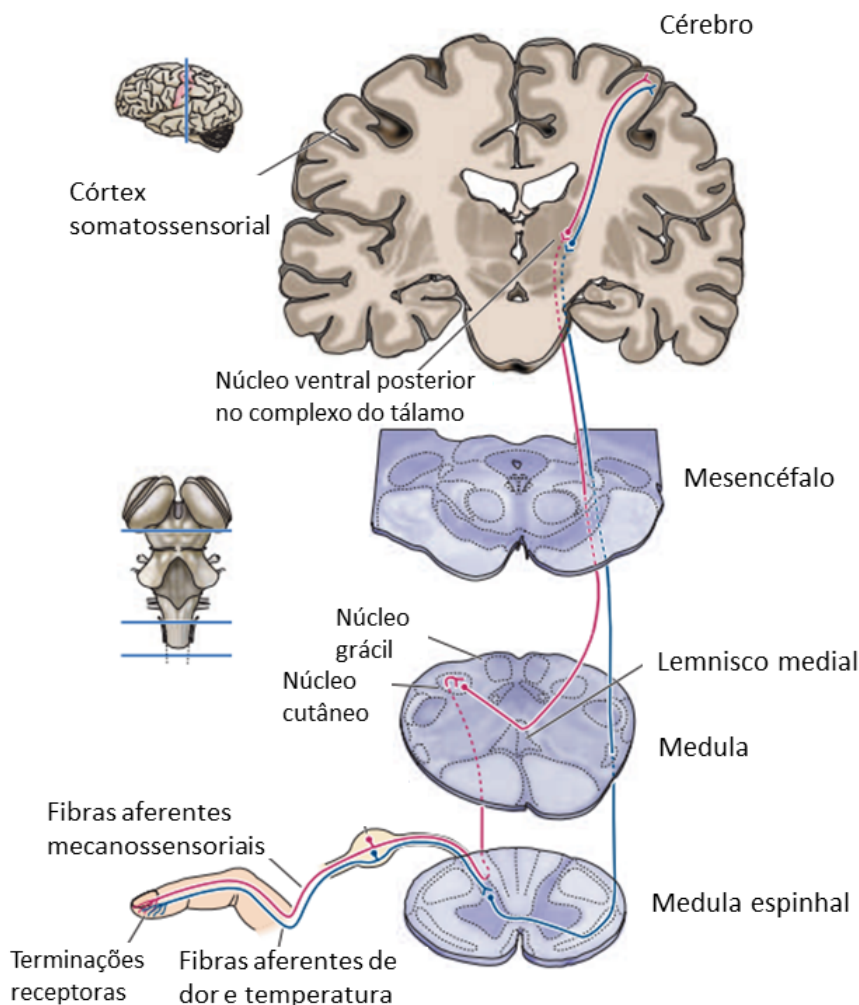


Figura 1 – Informações mecanossensoriais transmitidas da ponta do dedo para o cérebro (Adaptado de (PURVES D., 2004)).

Portanto, para um humano saudável, estímulos como força e vibração podem ser detectadas pelos mecanorreceptores, que respondem gerando trens de pulsos aferentes, sendo direcionados ao SNC para a percepção (KIM, S. S. et al., 2009). Assim sendo, a próxima seção irá abordar maiores detalhes sobre os diferentes tipos de mecanorreceptores e seu funcionamento.

### 2.1.1 Mecanorreceptores

O sistema tátil é composto por diferentes mecanorreceptores, termorreceptores e nociceptores, que detectam pressão e vibração, temperatura e estímulos nociceptivos, respectivamente (KANDEL et al., 2000). A densidade de cada tipo de receptor na pele é diferente de acordo com a sua localização no corpo, alterando a sensibilidade de tal região a diferentes estímulos táteis. A pele pode ser dividida em dois grandes grupos: a pele glabra que não possui pelos e cobre regiões como a palma das mãos e a planta dos pés, e a pele pilosa que recobre a maior parte de nosso corpo (WEINSTEIN, 1968).

Existem quatro tipos principais de mecanorreceptores incorporados na pele humana por todo o corpo (NAJARIAN; DARGAHI; MEHRIZI, 2009), e cada um é responsável pela recepção de estímulos específicos. Porém, é a ação conjunta de todos eles que cria a nossa experiência tátil, capturando as características de contato com a pele, como a força e a textura da superfície (KNIBESTÖL; VALLBO, Å. B., 1970; JOHANSSON, Roland S; VALLBO, Ake B, 1979).

Os quatro tipos de mecanorreceptores incluem corpúsculos de Pacini, corpúsculos de Meissner, discos de Merkel e cilindros de Ruffini, que por sua vez, podem ser divididos em duas categorias, a saber, mecanorreceptores de adaptação lenta (SA) e adaptação rápida (FA). Os Discos de Merkel (SA-I) e os Cilindros de Ruffini (SA-II) respondem a estímulos de baixa frequência até as grandezas estáticas (JAMALI; SAMMUT, 2010). Essas fibras são sensíveis à pressão constante e codificam a força aplicada sobre a pele (JOHANSSON, Roland S.; FLANAGAN, 2009). Em contrapartida, estímulos de alta frequência são reconhecidos por mecanorreceptores de adaptação rápida, ou seja, os Corpúsculos de Meissner (FA-I) e Corpúsculo de Pacini (FA-II) (JAMALI; SAMMUT, 2010). Essas fibras de adaptação rápida são sensíveis a estímulos transientes e, assim, codificam movimento relativo sobre a pele e vibrações (JOHANSSON, Roland S.; FLANAGAN, 2009).

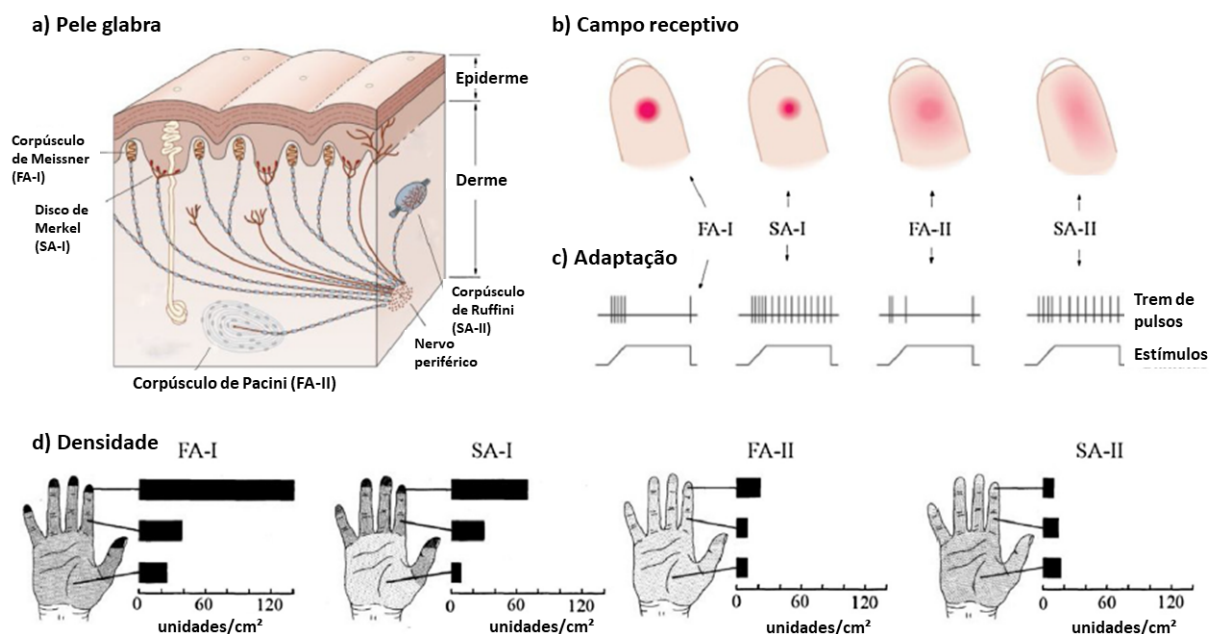


Figura 2 – Mecanorreceptores humanos. (a) Seção transversal da pele glabra (KANDEL et al., 2000). (b) O campo receptivo de cada tipo de mecanorreceptores (KANDEL et al., 2000). (c) Os trens de pulsos de cada tipo de mecanorreceptores em resposta a um estímulo específico (KANDEL et al., 2000). (d) A densidade de cada tipo de mecanorreceptores (VALLBO, A. B.; JOHANSSON, R. S., 1984).

Conforme pode ser visto na Tabela 1, o tamanho do campo receptivo diferencia os mecanorreceptores SA-I dos SA-II e os mecanorreceptores FA-I dos FA-II. Os campos receptivos dos mecanorreceptores SA-I e FA-I têm bordas mais distintas e tamanhos menores do que os mecanorreceptores SA-II e FA-II, além de serem encontrados nas camadas mais superficiais da pele glabra.

Tabela 1 – Características dos mecanorreceptores humanos (YI; ZHANG; PETERS, 2018).

Mecanorreceptores	Diâmetro da borda(mm)	Frequência (Hz)	Parâmetros de detecção
Discos de Merkel(SA-I)	3-4	DC-30	Curvatura da pele local
Cilindros de Ruffini(SA-II)	>10	DC-15	Alongamento direcional da pele
Corpúsculos de Meissner(FA-I)	3-4	10-60	Alongamento da pele
Corpúsculo de Pacini (FA-II)	>20	50-1000	Vibração não localizada

Em resposta, de acordo com estímulo recebido, os mecanorreceptores geram pulsos, que permitem ao ser humano a percepção destes estímulos táteis pelo córtex somatossensorial.

### 2.1.2 Percepção de estímulos táteis

A percepção de informações táteis é identificada através de pulsos (*spikes*) nos potenciais de ação gerados pelos mecanorreceptores espalhados na pele. Os padrões espaço-temporais codificados de diferentes formas pelos neurônios sensoriais do sistema tátil, podem ser utilizados para compreender a natureza do estímulo tátil (JOHANSSON, Roland S.; FLANAGAN, 2009). Estudos realizados sobre as atividades desses mecanorreceptores, demonstram a possibilidade de identificar a direção da força aplicada sobre a pele e características como a textura, a curvatura, dureza e as bordas dos objetos (JOHANSSON, Roland S.; FLANAGAN, 2009; SURESH; SAAL; BENSMAIA, Sliman J., 2016; SRINIVASAN; WHITEHOUSE; LAMOTTE, 1990; JOHNSON, K. O.; LAMB, 1981; PHILLIPS; JOHANSSON, R S; JOHNSON, K O, 1990).

Graças à ação dos mecanorreceptores, principalmente mecanorreceptores SA-I e FA-I, os seres humanos podem reconhecer várias formas ao tocar os objetos (JOHANSSON, Roland S.; FLANAGAN, 2009). Descobriu-se, que um único mecanorreceptor não pode oferecer informação suficiente para codificar a forma do objeto, e a percepção da forma é codificada por uma população espacial de ambos os mecanorreceptores (KHALSA et al., 1998). Segundo Lamotte et al. (1987), mecanorreceptores SA-I oferecem mais informações em termos de características espaciais de uma forma. Já, os mecanorreceptores FA-I desempenham um papel importante na discriminação de diferenças na nitidez .

A capacidade de perceber sensivelmente diferentes materiais também é uma capacidade dos seres humanos (SKEDUNG et al., 2013), porém, permanece o questionamento de como os seres humanos codificam a rugosidade da superfície durante

o interações físicas dedo-superfície. No entanto, a discriminação da rugosidade tátil depende principalmente da sensibilidade à vibração, sendo que, os mecanorreceptores de adaptação rápida tipo I e tipo II são geralmente considerados responsáveis por modular a vibração estímulos que ocorrem na interface das superfícies exploradas e os dedos (ZHENGKUN, 2017)

Também, é proposto na literatura (SRINIVASAN; WHITEHOUSE; LAMOTTE, 1990) a influência da micro-geometria de superfície na detecção de deslizamento. Como resposta, a detecção de deslizamento foi alcançada devido aos aferentes FA. Mais especificamente, os aferentes FA-I e FA-II forneceram um código espaço-temporal confiável quando um deslizamento ocorreu, segundo experimentos realizados em uma placa de ponto único e uma placa texturizada, respectivamente. Além disso, a direção do deslizamento pode ser percebida pelo aferente SA-I. Também, Johansson et al. (2004) propõe que, para o escorregamento de um dado objeto sobre a pele, foi demonstrado que tais informações são geradas pelos primeiros pulsos, visto a necessidade de serem processadas rapidamente, o que permite rápida reação em casos de escorregamento.

De acordo com Johansson et al. (2004) a classificação rápida dos estímulos táteis é realizada no núcleo cuneiforme, atuando como extratores de características e detectores de coincidências dos estímulos por meio de uma combinação espaço-temporal. Desta forma, tais neurônios de segunda ordem seriam importantes na detecção de bordas e curvaturas de objetos. Ademais, Bologna et al. (2011) propõe que deve haver algum mecanismo neural antes das informações chegarem ao córtex somatossensorial, que combina as atividades dos diferentes tipos de mecanorreceptores.

Tendo em vista a gama de informações propostas na literatura sobre os princípios da percepção dos estímulos táteis, também existem diversas propostas que visam construir sensores e algoritmos bioinspirados buscando simular o sistema tátil biológico. Para tal, o campo da engenharia neuromórfica vem sendo aplicada no desenvolvimento de sistemas táteis artificiais. Portanto, a próxima seção irá discutir algumas abordagens encontradas na literatura acerca da concepção dos sistemas neuromórficos e suas aplicações.



## 2.2 SENTIDO BIOMIMÉTICO DO TATO

O sentido do tato desempenha um papel importante em diversas aplicações em robótica, cirurgia minimamente invasiva, próteses avançadas e indústrias transformadoras (YI; ZHANG; PETERS, 2018). Em especial, quando tratamos de desenvolvimento de próteses, estudos quando a anatomia, fisiologia e biomecânica são necessários para guiar uma representação mecatrônica fiel a sua contraparte biológica, a mão humana. Nesse sentido, o desenvolvimento de articulações, atuadores e sistema de controle são pensados para garantir os graus de liberdade necessários para sua finalidade. Em contrapartida, os estudos do sistema tátil investigam meios de desenvolvimento de peles artificiais, que por sua vez, são cobertas por sensores táteis que buscam capturar informações de pressão e vibração obtidas por meio do contato com objetos (LUCAROTTI *et al.*, 2013). Para tanto, devem ser capazes de realizar tarefas complexas como agarrar objetos com formas e texturas desconhecidas sem permitir o deslizamento destes objetos (YOUSEF; BOUKALLEL; ALTHOEFER, 2011; MELCHIORRI, 2000). Assim sendo, a capacidade de detecção tátil torna-se um recurso particularmente valioso e desejável.

Nesse contexto, durante os últimos anos, enormes esforços e progressos foram feitos para indústria e a academia para o desenvolvimento de sensores e algoritmos bioinspirados, comumente chamada de engenharia neuromórfica (YI; ZHANG; PETERS, 2018). Tais modelos promovem a conversão de sinais analógicos de sensores tradicionais em sinais análogos aos que percorrem por neurônios sensoriais (RASOULI *et al.*, 2018). Desta forma, para representar essa engenharia neuromórfica presente nos sistemas biomiméticos, a Figura 3 demonstra um resumo contendo elementos que permitam reproduzir o sentido do tato humano. Como podemos observar, o processo de implementação dos sistemas táteis neuromórficos, é basicamente composto por sensores táteis, a codificação dos estímulos táteis em pulsos que irão simular o funcionamento dos mecanorreceptores biológicos, também conhecidos como mecanismos de transdução (WEI, 2016). Tal resposta, pode ser processada através de SNN que permitam a detecção de eventos, como toque ou escorregamento. Em seguida, através de uma resposta neuromórfica é possível a estimulação do nervo transcutâneo mediante uma retroalimentação sensorial (por exemplo (SAAL; BENSMAIA, Sliman J., 2015; ODDO *et al.*, 2016)), que conseqüentemente permitiria a percepção sensorial pelo paciente no córtex somatossensorial. Sendo essa, umas das principais vantagens de utilização de mecanismos com respostas neuromórficas. Ainda, além destes sistemas permitirem a retroalimentação sensorial, é possível que tais respostas sejam incorporadas aos sistemas de controles das próteses, auxiliando os pacientes a agarrar e manipular objetos, conforme é apresentado no canto inferior direito da Figura 3.

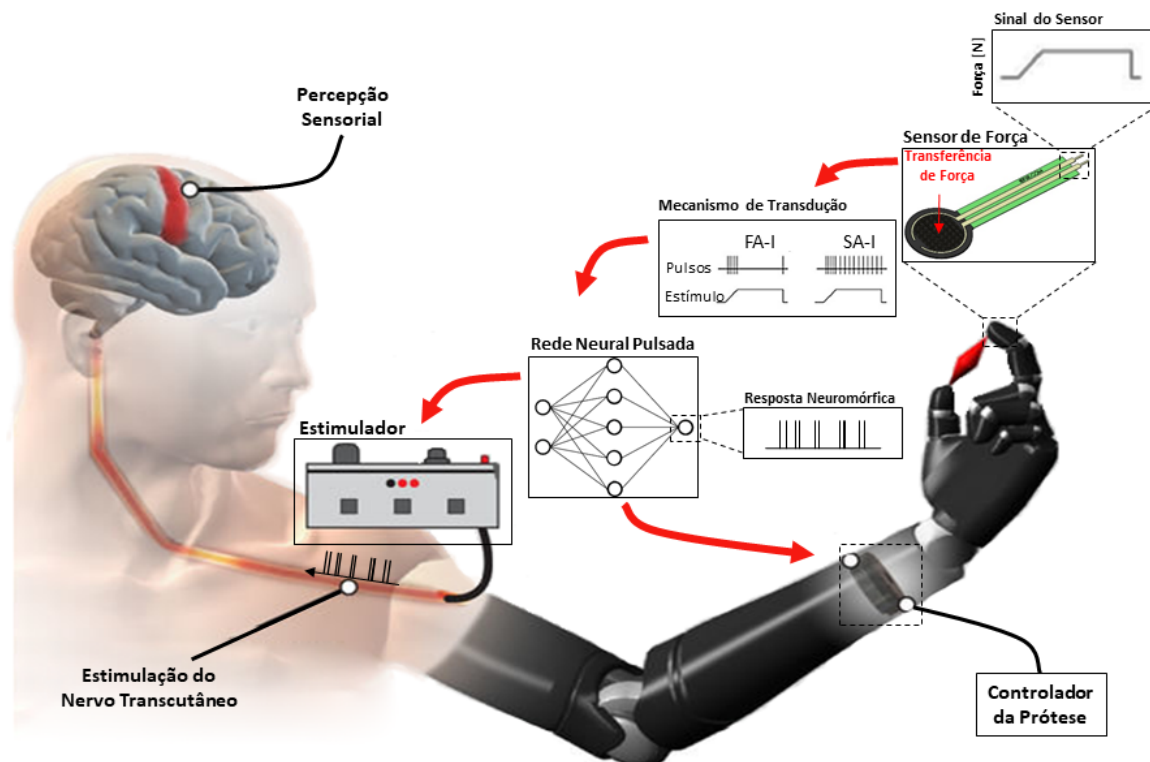


Figura 3 – Sistema biomimético para reproduzir o sentido do tato humano (Adaptado de (OSBORN *et al.*, 2018; YI; ZHANG; PETERS, 2018))

Seguindo nesse contexto, os mecanismos apresentados comportam-se de maneira similar ao sentido humano do tato. E para aprofundar, as próximas sessões abordam em detalhes cada um destes mecanismos propostos na literatura.

### 2.2.1 Codificação de estímulos táteis

Conforme apresentado na Figura 3, o mecanismo inicial que permite a reprodução do sentido humano do tato é composto por sensores táteis, e seus diversos modos de transdução e características foram relatados em detalhes (DAHIYA; VALLE, 2013; SICILIANO; KHATIB; KRÖGER, 2008). Desta forma, uma mão robótica ao tocar um objeto realiza uma transferência de força em tais sensores, e o sinal analógico gerado necessita de mecanismos de transdução destes estímulos que reproduzam o comportamento dos mecanorreceptores humanos.

A transdução dos estímulos táteis pode ocorrer basicamente com o uso de modelos computacionais de neurônios para transformar a força capturada pelos sensores táteis em pulsos ou até construir circuitos eletrônicos que geram os pulsos diretamente. Tais respostas, buscam representar a atividade dos mecanorreceptores (LIU; DELBRUCK, 2010). Para tanto, a próxima seção irá abordar os modelos de mecanorreceptores propostos na literatura.

### 2.2.1.1 Modelos de mecanorreceptores

A fim de reproduzir a capacidade tátil dos mecanorreceptores humanos, modelos de transdução de sinais de sensores táteis foram desenvolvidos em conjunto com a utilização de modelos computacionais de neurônios pulsáteis. Os modelos de neurônios pulsáteis desempenham um papel vital na descrição das atividades de disparo e são usados para converter os sinais gerados por sensores táteis artificiais em trens de pulsos (ZHENGKUN, 2017).

Desta forma, Kim et al. (2010) propuseram um modelo de mecanorreceptor que reproduz com precisão a resposta de fibras do tipo FA-I, SA-I e FA-II em uma variedade de estímulos, conforme ilustra a Figura 4. Cada neurônio é excitado por uma entrada que consiste em oito variáveis no total. A entrada principal (posição) é o sinal de força detectado pelo sensor de força. Outras variáveis consistem na derivada de primeira (velocidade), segunda (aceleração) e terceira (arranque) ordem deste sinal. Cada sinal é separado em suas componentes positiva (—) e negativa(- - -) e filtrados através de um filtro de linear. Por fim, as entradas transformadas são somadas e formam a entrada para um neurônio *Leakly Integrate-and-Fire* (LIF). Os parâmetros do filtro linear passaram por um procedimento similar a regressão para que a resposta do modelo final se aproximasse mais dos dados eletrofisiológicos registrados em macacos.

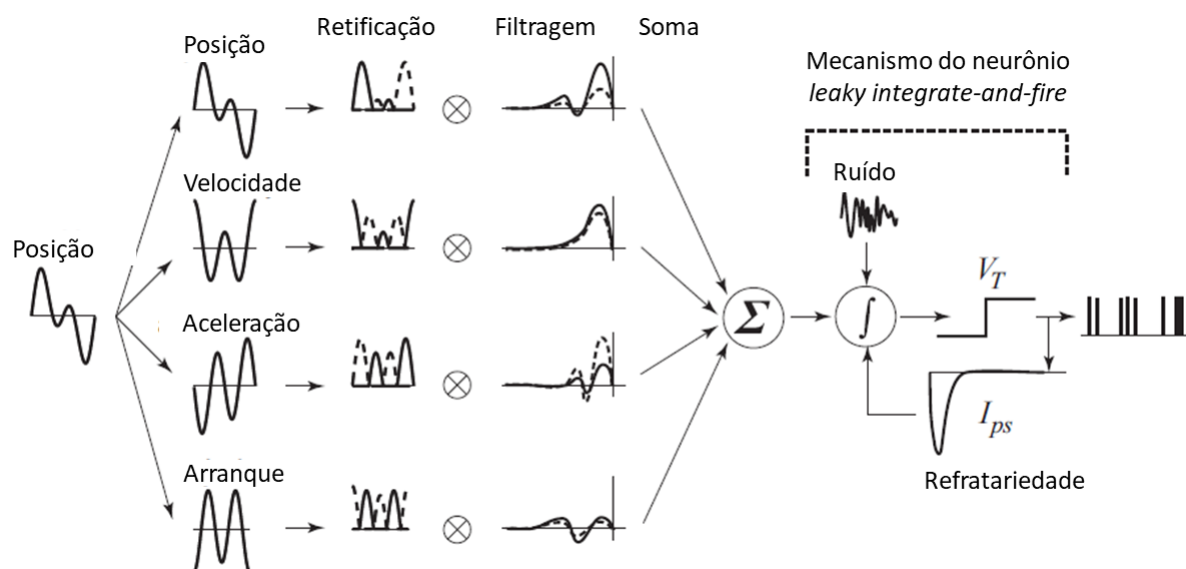


Figura 4 – Modelo de mecanorreceptor proposto por Kim (Adaptado de (KIM, S. S.; SRIPATI; BENSMAIA, Sliman J, 2010)).

Durante o processo de testes do modelo, diferentes combinações de entradas foram testadas a fim de identificar as respostas que mais se aproximam as saídas dos diferentes tipos de mecanorreceptores. Como resultado, a fibra SA-I é mais sensível ao sinal de força e sua primeira derivada, já a fibra FA-I é sensível apenas à primeira derivada. Na Figura 5, é possível observar que a taxa de disparo do aferente SA-I é

maior durante a rampa de ativação do que durante a fase de retenção do estímulo. Se esta fibra fosse sensível apenas à posição, sua taxa de disparo seria menor durante a rampa. A fibra FA dispara apenas durante as rampas de ativação e desativação do estímulo. Se esta fibra fosse sensível apenas a velocidades positivas (retificação de meia onda), ela dispararia durante a rampa de ativação, mas não durante a rampa de saída (KIM, S. S.; SRIPATI; BENSMAIA, Sliman J, 2010).

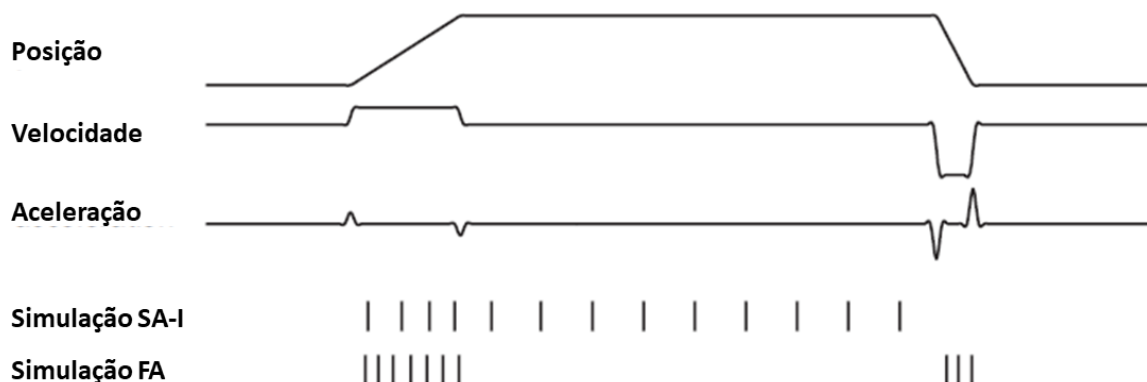


Figura 5 – Respostas de simulação para fibras aferentes SA e FA-I (Adaptado de (KIM, S. S.; SRIPATI; BENSMAIA, Sliman J, 2010)).

Visto isso, é perceptível que os modelos de neurônios biologicamente plausíveis exercem uma função crucial na descrição das atividades pulsáteis, sendo empregues para converter os sinais gerados por sensores táteis artificiais em trens de pulsos (ZHENGKUN, 2017). Assim sendo, a próxima seção irá relatar os modelos de neurônios e seu comportamento.

### 2.2.1.2 Modelos de Neurônios

Do modelo mais realista e sofisticado, o modelo Hodgkin-Huxley (HH) (HODGKIN; HUXLEY, 1952), ao mais simples e computacionalmente eficiente, o modelo integra-e-dispara com vazamento (LIF), os modelos de neurônios possuem vários níveis de abstração, ou seja, maior ou menor complexidade. O modelo HH é considerado custoso computacionalmente, pois descreve de forma detalhada a biofísica das células excitáveis, ou seja, a dinâmica dos canais iônicos. Em contrapartida, modelos reduzidos como o LIF foram criados com o objetivo de descrever formalmente apenas o processo de integração que leva ao disparo de um potencial de ação.

O LIF é um dos mais populares modelos de neurônios devido à sua implementação simples, o que o torna um modelo computacionalmente eficiente e com recursos neurocomputacionais (ARSALAN; SANTRA; ISSAKOV, 2022). Os neurônios LIF são inspirados pela tensão da membrana vazando em neurônios biológicos, com seus estados diminuindo ao longo do tempo se não houver picos de entrada presen-

tes, tornando-o mais biologicamente plausível (HAN et al., 2020). A equação a seguir descreve a dinâmica do neurônio LIF (DUTTA et al., 2017):

$$C_m \frac{dv}{dt} = g_L(V(t) - E_L) + I(t) \quad (1)$$

onde  $C_m$  significa a capacitância da membrana,  $g_L$  denota a condutância do canal de fuga,  $E_L$  define o potencial de equilíbrio do canal de fuga,  $V$  refere-se à tensão,  $t$  denota a duração e  $I$  é a quantidade total da corrente de entrada. O LIF envia pulsos síncronos em resposta a entradas assíncronas, simulando os padrões de disparo dos neurônios no cérebro humano.

Cada neurônio pulsado mantém um valor que reflete sua condição atual. Os neurônios aceitam os picos como entrada e atualizam seus estados integrando os pesos correspondentes. Quando o estado excede um valor limite, o neurônio é acionado e envia um pico para o próximo neurônio, que recebe o pico após um atraso de propagação pré-determinado e atualiza seus estados de acordo. Apesar de esses princípios operacionais serem universais, o comportamento dos neurônios varia de acordo com o modelo (HAN et al., 2020).

Também, buscando a integração de tais neurônios em um conjunto que permita a construção de modelos ou redes mais complexos e sofisticados, surgiram as SNN. Estas redes, permitem capturar as propriedades computacionais neurais dos neurônios biológicos (KASABOV et al., 2013). Além disso, as SNN podem ser treinadas para realizar uma função desejada, e em seguida será abordado os conceitos a cerca deste tema.

### 2.3 REDES NEURAIS PULSADAS

Nos últimos anos, houve um aumento no número de pesquisas no campo de redes neurais pulsadas, também conhecidas como a terceira geração das redes neurais. Em comparação com as redes neurais artificiais (RNAs), as SNN criam modelos de computador que mapeiam com mais precisão o funcionamento do sistema nervoso permitindo construir modelos de redes neurais biologicamente mais realistas, sendo que os neurônios são excitados através de pulsos. Já as RNAs, são incapazes de capturar as complexas propriedades computacionais neurais dos neurônios biológicos, bem como, os neurônios são excitados através de valores contínuos (KASABOV et al., 2013).

Também, o interesse nestas pesquisas vem da disponibilidade de plataformas de computação neuromórfica que foram otimizadas para simulações baseadas no cérebro, implementadas em *hardwares* analógicos dedicados ou digitais de picos (DIEHL et al., 2015). Existem diversos simuladores neurais dedicados à construção de modelos em larga escala (por exemplo (BEKOLAY et al., 2014; EPPLER et al., 2009; GOODMAN; BRETTE, 2009; HINES; DAVISON; MULLER, 2009)). No entanto, o Nengo

(BEKOLAY *et al.*, 2014) é único a ser construído sobre uma estrutura teórica que permite uma arquitetura cognitiva que mantém um alto nível de plausibilidade biológica. O projeto mais relacionado em termos de design de *software* com o Nengo, são o PyNN (DAVISON *et al.*, 2009) e Topographica (BEDNAR, 2009), ambos fornecem uma interface de alto nível para simuladores neurais de baixo nível.

As APIs do Nengo e do PyNN são semelhantes, mas diferem significativamente em como os grupos de neurônios são conectados. No Nengo, as conexões comumente descrevem a operação matemática que é realizada através da conexão entre dois conjunto de neurônios. Já em PyNN, as conexões geralmente descrevem características da matriz de pesos de conexão entre duas populações. Essa diferença reflete a diferença fundamental de que o Nengo é construído sobre uma estrutura teórica que permite aos modeladores pensar sobre o processamento de informações no cérebro em um nível conceitual.

Além disso, simuladores neurais como Brian (GOODMAN; BRETTE, 2009), NEST (EPPLER *et al.*, 2009) e NEURON (HINES; DAVISON; MULLER, 2009) também são amplamente utilizados. Uma diferença fundamental entre os simuladores, é o desempenho em execução das redes neurais, na qual, estudo mostra (BEKOLAY *et al.*, 2014) que o Nengo foi capaz de simular dois modelos de *benchmark* muito mais rápido do que Brian, NEST e NEURON. Isso ocorre, em parte, porque o Nengo armazena os fatores da matriz de pesos de conexão, em vez de armazenar a matriz inteira. Os simuladores neurais como Brian, NEST e NEURON são, atualmente mais adequados para simular uma ampla gama de modelos de célula única, enquanto o Nengo é projetado para grandes redes de modelos neurais simples que são conectados de acordo com os princípios do *Neural Engineering Framework* (NEF) e pode simular esses tipos de modelos de forma mais eficiente (BEKOLAY *et al.*, 2014).

Além disso, outra diferença fundamental entre o simulador Nengo e os simuladores neurais tradicionais é a possibilidade de integração com outras bibliotecas. O núcleo do ecossistema Nengo é a biblioteca Python Nengo, que inclui os objetos e um simulador baseado em NumPy. Para exemplificar, algumas das bibliotecas integradas, são: O NengoGUI, uma ferramenta interativa de construção e visualização de modelo baseado em navegador da web; o NengoDL, que simula modelos Nengo usando a biblioteca TensorFlow para interagir facilmente com redes de aprendizado profundo, bem como usar procedimentos de treinamento de aprendizado profundo para otimizar os parâmetros do modelo Nengo. Também, possibilita a simulação das redes do Nengo em *hardwares* neuromórficos, dentre eles o NengoLoihi, que executa modelos Nengo no *hardware* neuromórfico Loihi da Intel, para que os modelos possam ser prototipados antes de serem executados em *hardware* real. O NengoSpiNNaker simula modelos Nengo usando a arquitetura do *hardware* neuromórfico SpiNNaker.

Portanto, em comparação com os diferentes simuladores, o Nengo é o que

melhor se adéqua aos objetivos deste trabalho. Ele possibilita a simulação e treinamento das redes neurais pulsadas de forma eficiente, e ainda com a possibilidade de integração com diferentes ferramentas de simulação em *hardwares* neuromórficos no avanço das pesquisas. Assim sendo, a próxima seção irá abordar com mais detalhes o funcionamento do Nengo.

### 2.3.1 Nengo

Conforme apresentado na seção anterior, o Nengo é uma biblioteca projetada para permitir a construção e simulação de modelos neurais em larga escala. Esta biblioteca foi construída com base no NEF (ELIASMITH; ANDERSON, 2003) que fornece um método para mapear funções para SNN biologicamente plausíveis.

O NEF é uma ferramenta para simulação de redes neurais pulsadas com aplicações em ciência cognitiva, psicologia, inteligência artificial e neurociência (BEKOLAY et al., 2014). Ele tem sido usado para desenvolver modelos de neurônios pulsáteis, estudo em aplicações como controle motor, classificação de imagens, raciocínio indutivo e seleção de ações usando diferentes métodos de aprendizagem (RASMUSSEN, 2019). Para isso, o NEF é definido em três princípios fundamentais: 1) representação, como os neurônios representam um valor do mundo real; 2) transformação, como os neurônios computam funções (estáticas) em valores representados; e 3) dinâmica, como os neurônios podem implementar sistemas ou funções dinâmicas (VOELKER, 2019).

Para uma melhor compreensão dos princípios de representação e transformação citados, a Figura 6 apresenta um resumo na forma de exemplo. Portanto, o princípio da representação busca codificar a informação de entrada por populações de neurônios e pode ser dividida em dois outros princípios: a codificação e a decodificação (Figura 6.a e 6.b, respectivamente). Na codificação, os sinais são codificados em populações neurais com base na curva de ajuste (*tuning curve*) de cada neurônio (Figura 6.a1, onde cada neurônio da população representa uma curva). A curva de ajuste descreve o quanto um determinado neurônio irá disparar (eixo y, em frequência) em função do sinal de entrada. Neste exemplo, o sinal de entrada é representado pela Figura 6.a2, já a Figura 6.a3 representa a resposta em padrão de disparo de cada neurônio de acordo com sua curva de ajuste e o sinal de entrada. Também, pelo princípio da decodificação a atividade de pico de uma população neural é decodificada para recuperar o sinal de entrada original. Inicialmente, o padrão de disparo mostrado na Figura 6.b1 é filtrado com um filtro exponencial decrescente (Figura 6.b2). Em seguida, a resposta é então somada com um conjunto de pesos que aproxima o sinal de entrada (Figura 6.b3, verde). Cabe salientar, que a precisão da estimativa decodificada aumenta à medida que o número de neurônios também aumenta (BEKOLAY et al., 2014).

Também, o princípio de transformação determina como podemos decodificar

trens de pico para calcular transformações lineares e não lineares de sinais codificados em uma população de neurônios. Por exemplo, suponha que estamos representando uma onda senoidal codificada por uma população A (Figura 6.c1). O negativo desse sinal é projetado para a população B (Figura 6.c2) e o quadrado desse sinal é projetado para a população C (Figura 6.c3). Desta forma as populações de neurônios podem enviar sinais para outra população decodificando a função desejada da primeira população e repetir o processo de acordo com objetivo (BEKOLAY et al., 2014).

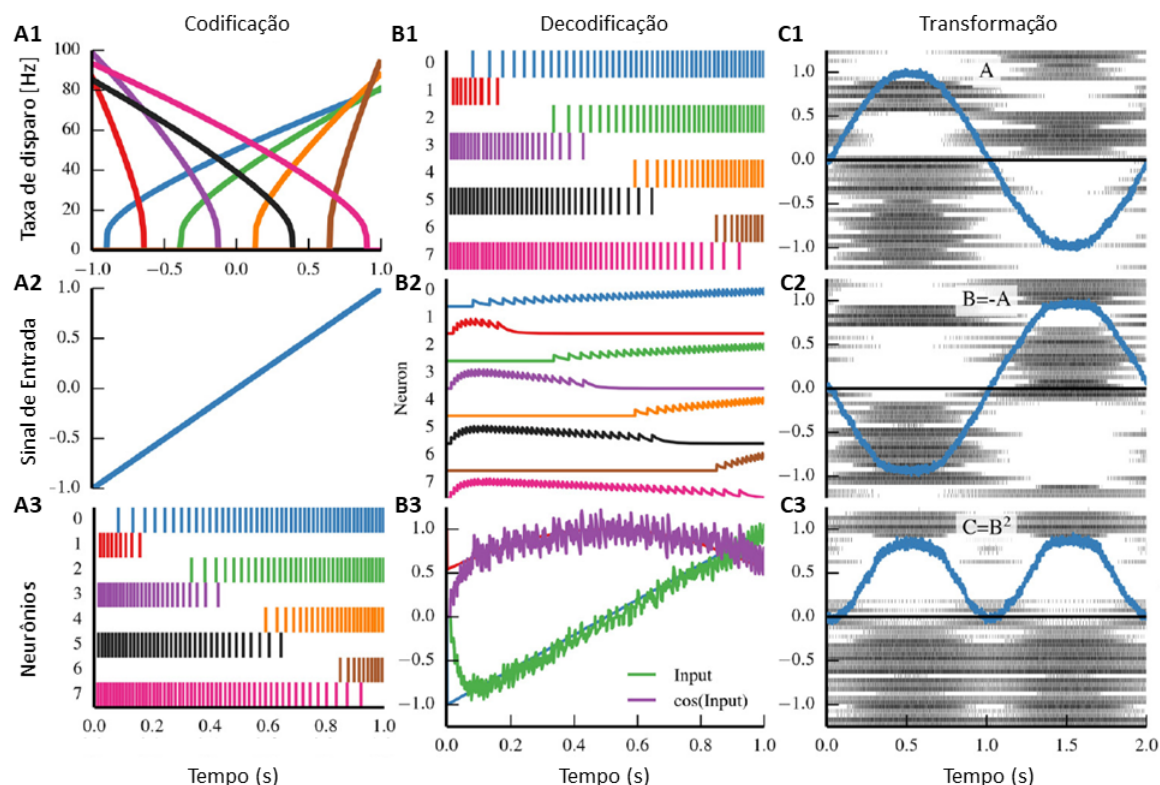


Figura 6 – Resumo dos princípios de representação e transformação do NEF (Adaptado de (BEKOLAY et al., 2014))

De acordo com estes princípios, modelos podem ser construídos como componentes conectáveis que descrevem sistemas neurais, assim como um diagrama de circuito descreve um circuito eletrônico. Para isso, o Nengo define seis objetos principais:

- População (*Ensemble*): contém um conjunto de neurônios que codifica um vetor variável no tempo de números reais.
- Nó (*Node*): representa informações não neurais, como entradas sensoriais e saídas motoras.
- Conexão (*Connection*): descreve como os nós e conjuntos são conectados.
- Prova (*Probe*): coleta dados durante uma simulação para análise posterior.
- Rede (*Network*): encapsula um grupo funcionalmente relacionado de nós e conjuntos interconectados.



- Modelo (*Model*): encapsula um modelo Nengo.

Esses seis objetos contêm informações simbólicas sobre um modelo Nengo, permitindo uma divisão estrita entre construção do modelo e simulação. Isso permite que um modelo Nengo seja executado em vários simuladores, como o NengoDL. O NengoDL é um pacote Python que combina Nengo e TensorFlow (kit de ferramentas de aprendizado de máquina do Google (ABADI *et al.*, 2016)) em uma estrutura hibridizada que automatiza a retropropagação (*backpropagation*) em redes neurais recorrentes pulsadas, não pulsadas e estilo NEF, permitindo o treinamento destas redes. (RASMUSSEN, 2019). Mais especificamente, a retropropagação visa minimizar o erro de resposta dos neurônios na camada de saída de uma rede neural durante o treinamento. A medida que as amostras são apresentadas para a rede neural juntamente com a referência (*ground truth*), é calculado o erro entre a saída e a referência, sendo esse propagado do sentido da saída para a entrada nas camadas intermediárias, realizando o ajuste de pesos (RUSSELL, 2010). Contudo, existem diversos métodos que permitem o treinamento destes pesos e a avaliação de desempenho, que por sua vez, serão apresentados na próxima seção.

### 2.3.2 Técnicas de treinamento e avaliação de redes neurais

Quando tratamos de algoritmos de aprendizados de máquina nos deparamos com diversos métodos de treinamento e avaliação. Para o treinamento, os algoritmos de aprendizado de máquina pode ser divididos em três grupos principais: aprendizado supervisionado, aprendizado não supervisionado e aprendizado por reforço. No aprendizado supervisionado, as amostras inseridas no treinamento são rotuladas (referência), o que quer dizer que o algoritmos já tem as respostas ideais para cada amostra de entrada. Para isso, o ajuste de seu modelo interno se baseia na minimização do erro entre a saída calculada e a referência. No aprendizado não supervisionado, não há referência das amostras inseridas, por isso o treinamento é baseado por semelhança entre grupos, cada grupo será formado por amostras que compartilham uma característica em comum. Já para o aprendizado por reforço, o algoritmo deve buscar atingir certo objetivo, sendo que o modelo é ajustado com base em recompensas e punições em relação ao objetivo do treinamento (RUSSELL, 2010).

Também, para controlar o comportamento dos algoritmos de aprendizado de máquina, existem alguns botões de ajuste, chamados de hiperparâmetros. Exemplos de hiperparâmetros são: número de neurônios de uma camada, a quantidade de épocas de treinamento e a taxa de aprendizado. Estes devem ser definidos antes do início do processo de aprendizado. Para encontrar os melhores hiperparâmetros de uma rede neural, podem ser utilizadas diversas técnicas, como otimização Bayesiana, pesquisa randômica ou pesquisa em forma de grade (*grid search*). Esta última pode ser considerada a mais simples de todas, visto que, unicamente são realizadas todas

as combinações possíveis entre os hiperparâmetros analisados (RASCHKA, 2018).

Além disso, podem ser empregues diferentes métodos de validação para avaliação e seleção dos melhores modelos. Uma delas comumente utilizadas é a técnicas de validação cruzada *K-fold Cross-Validation* (CV), usada para classificar modelos de várias configurações de hiperparâmetros e estimar o quão bem eles generalizam um conjunto de dados. A ideia por trás da validação cruzada é que cada amostra do conjunto de dados tenha a oportunidade de ser testada. A validação cruzada *k-fold* é um caso especial de validação cruzada em que iteramos sobre um conjunto de dados *k* vezes. Em cada rodada, dividimos o conjunto de dados em *k* partes, na qual, uma parte é usada para validação e a restante *k - 1* partes são mescladas em um subconjunto de treinamento para treinar os parâmetros do modelo (RASCHKA, 2018). A Figura 17 ilustra o processo de validação cruzada de 5 vezes, ou seja, *5-fold cross-validation*.

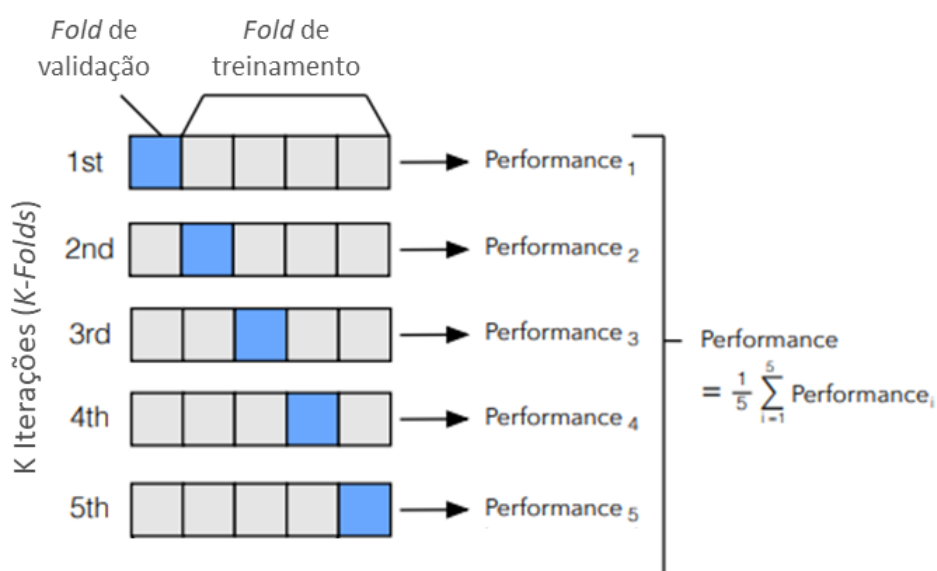


Figura 7 – Ilustração do funcionamento da validação cruzada *k-fold* (Adaptado de (RASCHKA, 2018))

Desta forma, a validação cruzada *k-fold* usa todos os dados para treinamento e teste. Também, garante que cada amostra seja usada para validação (RASCHKA, 2018).

Portanto, como forma de aplicação das técnicas de aprendizado de máquina, foram empregues em diferentes tarefas, como: rastreamento e navegação, visão computacional, proteção de ameaças, locomoção de robôs autonomamente (SINGH; KUMAR; PRATAP, 2022), bem como, estudos na detecção de escorregamento de objetos em mãos robóticas (VEIGA *et al.*, 2015; BEGALINOVA *et al.*, 2020). Nesse contexto, quando tratamos de próteses de mão, um problema relevante na área da robótica é a detecção de escorregamento, visto que à medida que os robôs saem dos laboratórios de um mundo idealizado, eles são obrigados a realizar tarefas de manipulação em ambientes não estruturados, ou seja, objetos com tamanhos e formas diferentes

(YUSEF; BOUKALLEL; ALTHOEFER, 2011). Portanto, a próxima seção irá abordar aspectos e trabalhos relacionados onde os autores utilizaram diferentes algoritmos de aprendizado de máquina para a detecção de escorregamento de objetos.

## 2.4 DETECÇÃO DE ESCORREGAMENTO

Quando tratamos da manipulação de objetos buscando manter a estabilização da preensão, é importante a detecção do deslizamento entre o manipulador e o objeto que está sendo agarrado, pois esta informação pode representar um retorno ao sistema informando se a força de preensão está correta. Desta forma, o escorregamento pode ser tratado como um problema de contato entre um objeto e uma prótese ou mão biônica. Este princípio pode ser visto na Figura 3, onde a prótese agarra em forma de pinça um objeto de massa  $m$ , através da força de preensão aplicada em cada dedo. Ainda, esta força de preensão depende do coeficiente de atrito dada pelo objeto.

Nesse sentido, o escorregamento do objeto pode ocorrer de diferentes maneiras: 1) a força de preensão do manipulador é insuficiente, o que pode ocorrer quando as propriedades do objeto são estimadas incorretamente (por exemplo, a massa); 2) quando há uma perturbação externa ao objeto, como uma colisão no objeto agarrado; ou 3) quando o coeficiente de atrito do objeto diminuiu, por exemplo, quando o objeto fica molhado e mais escorregadio, fazendo com o que a força de preensão não seja suficiente para manter a estabilidade do objeto (PRACH *et al.*, 2017). Para tanto, dada importância para a presente dissertação, dois trabalhos correlatos que utilizem técnicas de inteligência artificial serão detalhados a seguir para uma melhor compreensão dos materiais e métodos utilizados.

Veiga *et al.* (2015) apresentaram uma abordagem baseada em aprendizado de máquina para *feedback* de controle e estabilização de objetos, sendo que o controlador depende da previsão do deslizamento através da sensação tátil. Os dados táteis são extraídos do BioTac, um sensor tátil multicanal cujo *design* foi inspirado no dedo humano. O experimento é realizado colocando um objeto entre o dedo do robô e um plano vertical (Fig 8.a), em seguida o braço de um robô (Mitsubishi PA-10, robô com sete graus de liberdade) se move lentamente para longe do plano a uma velocidade constante (1 cm/s) até que ocorra um deslizamento entre o objeto e o dedo. O robô continua se movendo até que o objeto finalmente caia no chão. Para estes experimentos, foram selecionamos objetos de diferentes formas e rigidez, como: uma bolinha, bloco de madeira, copo, marcador, palito, fita adesiva e regador. A Figura 8.b apresenta um exemplo do experimento com o bloco de madeira. Para cada objeto foram realizado 10 experimentos e a rotulagem dos dados foi realizada com o auxílio de uma câmera (Asus Xtion Pro). Para classificação, os métodos utilizados neste trabalho são máquinas de vetores de suporte (*Support Vector Machine* (SVM)) e classificadores de florestas aleatórias (*Random Forest*), onde os dados de treinamento e teste foram

divididos em, respectivamente, 70% e 30%. Também, o treinamento ocorreu tanto para cada objeto separado, como para todo o conjunto de dados. Em resumo, os métodos obtiveram resultados que variaram de 64% à 97% na métrica f-score.

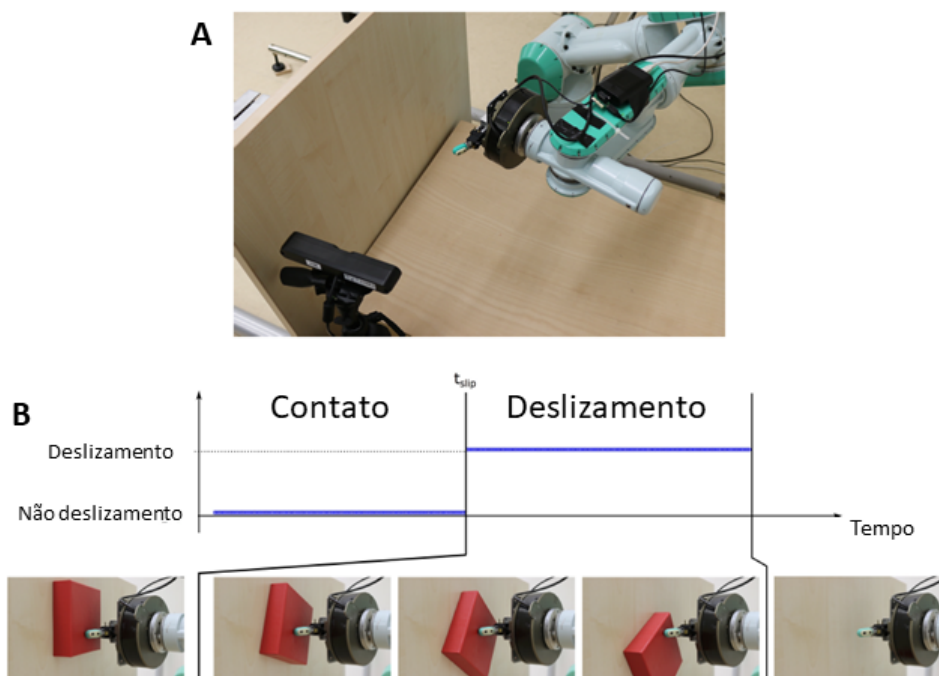


Figura 8 – Resumo do trabalho de Veiga et al. (2015) (Adaptado de (VEIGA et al., 2015)).

Analisando trabalho de Veiga et al. (2015), umas das vantagens desse método é possuir um sensor tátil bioinspirado em um dedo humano, o que é vantajoso pois melhor representa o sentido humano do tato em uma mão protética. Além disso, a utilização de experimentos com sete diferentes objetos se torna interessante, pois permite a validação do método para diferentes texturas, tamanhos e formas de objetos. Por outro lado, o trabalho apresenta certas desvantagens. Para entendermos, conforme apresentado na seção 2.2.1, um mecanismo que busque uma maneira realista representar o sistema tátil humano, é necessária uma resposta neuromórfica, ou seja, que utilize métodos de transdução com uma saída pulsada. Desta forma, estes mecanismos apresentados, não permitiriam uma retroalimentação sensorial, o que consequentemente não possibilitaria um *feedback* tátil ao paciente. Sendo esse um processo vital quando tratamos de um sistema biomimético completo. Além disso, esse tipo de solução limitam a portabilidade do sistema como um todo, pois dificultam a implementação em um *hardware* embarcado, pelo alto custo de processamento.

Também Begalinova et al. (2020), apresenta um trabalho interessante na perspectiva da utilização de técnicas de aprendizado de máquinas para detecção de escorregamento. O método é baseado na aplicação de uma técnica de classificação sequencial (uma variante de redes neurais recorrentes conhecida como memória de longo prazo (*Long Short-Term Memory* (LSTM))). Na Figura 9, que apresenta um re-

sumo do método proposto, é possível notar a sequência de passos utilizada pelo autor para treinamento da rede. Inicialmente, os dados são coletados através de sensores de barômetro Takktile *Micro-Electro-Mechanical System* (MEMS), anexados a uma luva (Figura 9.b). Esta luva é vestida, e segurado um copo de plástico com água. Em seguida o manipulador solta ligeiramente o objeto (mas ainda segura o objeto enquanto o manipula), provocando o escorregamento pela força gravitacional. Este experimento foi repetido 105 vezes. Também, foi implementado um algoritmo que automatiza o processo de rotulagem dos dados (*labeling*) e normaliza-os. Os dados então são divididos em treinamento, validação e teste (60% - 15% - 25%) e introduzidos a uma rede LSTM, treinando o modelo. Como resultado, foi obtido mais de 95% na classificação *offline* e 89% na classificação online usando o Robô Sawyer. Além disso, os autores realizaram uma comparação com outro método, o Modelos Ocultos de Markov (HMM), se sobressaindo em até 23 pontos percentuais na métrica f1-score.

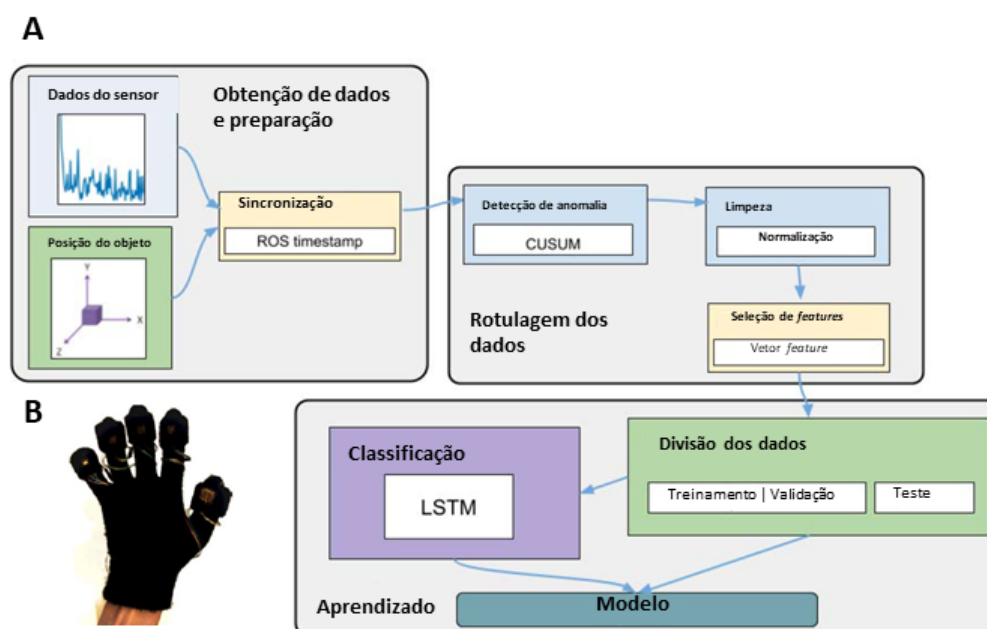


Figura 9 – Resumo do trabalho de Begalinova et al. (2020) (Adaptado de (BEGALINOVA et al., 2020)).

Dentro do contexto apresentado por Begalinova et al. (2020), o método tem a vantagem de possuir uma grande variedade de dados para o treinamento das redes, afinal foram 105 experimentos. Também, os resultados obtidos com o treinamento da rede LSTM foram promissores em relação aos métodos comparados, conforme apresentado anteriormente. Em contrapartida, o método não apresenta uma abordagem neuromórfica, o que no ponto de vista biomimético e de integração com uma retroalimentação sensorial é inviável. Também, a implementação em um *hardware* em embarcado, por exemplo, em controlador de uma prótese poderia ser custoso. Além disso, nota-se que o treinamento das redes foi realizado através do método simples de validação (divisão dos dados em treinamento, validação e teste), sendo que esta técnica não é ideal para

estimar o quão bem o conjunto de dados completo é generalizado.

Portanto, através do estudos apresentados é possível observar as diversas maneiras de solucionar o problema do escorregamento de objetos, sendo que cada um dos métodos possui suas vantagens e desvantagens. Buscando solucionar a desvantagens dos métodos propostos atualmente, o presente treina redes neurais pulsadas para detecção de toque e escorregamento, que através de um método de transdução de sinais analógicos de um sensor, obtenha uma resposta biologicamente plausível permitindo a retroalimentação sensorial a usuários de próteses de mão. Também, com a utilização de SNN é possível a implementação destes sistemas em *hardwares* embarcados neuromórficos com baixo custo de energia. Além disso, através de um método de validação que propicie que cada amostra em nosso conjunto de dados tenha a oportunidade de ser testada, permite a generalização do conjuntos de dados por completo.

### 3 CLASSIFICAÇÃO DE TOQUE E ESCORREGAMENTO EM MÃOS PROTÉTICAS UTILIZANDO REDES NEURAI PULSADAS

Neste capítulo, soluções biologicamente plausíveis para detecção de toque e deslizamento são apresentados. O conteúdo contempla a implementação de arquiteturas de SNN, que codificam o movimento de objetos capturados por um sensor de força (*Force Sensitive Resistor* (FSR)) em pulsos. Estas redes são treinadas para obtenção dos hiperpâmetros que proporcionem o melhor desempenho em dois experimentos distintos. Estes mecanismos permitem uma solução neuromórfica, que viabilizam a construção de *hardwares* neuromórficos de baixo consumo de energia, bem como fornecem uma resposta compreensível pelo cérebro. A Figura 3 apresentada anteriormente, representa de maneira ilustrativa os mecanismos utilizados nestas soluções.

#### 3.1 MATERIAIS E MÉTODOS

Nesta seção, será descrito todos os matérias e métodos utilizados na descrição das topologias de SNN para classificação de eventos de toque e escorregamento em sensores de próteses de mão. Esta pesquisa pode ser caracterizada como aplicada, visto que pretende gerar conhecimento válido e aplicável, buscando solucionar um problema específico. A forma de abordagem da pesquisa pode ser classificada como quâli-quantitativa, considerando que alguns elementos podem ser abordados em forma de números e requerem uso de recursos estatísticos, ao mesmo tempo, outros aspectos possuem uma avaliação subjetiva (SILVA, 2001).

Com relação aos objetivos, pode ser considerada exploratória, visto o fato de obter uma maior familiaridade com o tema proposto. Em relação aos procedimentos técnicos, a construção da pesquisa pode ser realizada a partir de materiais documentados (SEVERINO, 2017). Assim sendo, com o intuito de atender como resposta efetiva às demandas do problema de pesquisa levantado, em seguida são apresentados os métodos utilizados para extração dos dados utilizados como entradas das redes neurais propostas.

##### 3.1.1 Coleta e preparação de dados

Foram realizadas duas sessões experimentais para validar a abordagem proposta. Os dados adquiridos foram utilizados no treinamento de redes neurais pulsadas, tanto para detecção de toque, bem como, classificação de escorregamento em mãos protéticas.

Na primeira sessão experimental, foram adquiridos dados para a detecção do toque através da força normal medida pelo sensor FSR. Para isto, os experimentos combinaram diferentes tempos de duração do toque e a força aplicada. Já no segundo experimento, foram adquiridos dados que possibilitam a detecção de escorregamento

através de uma perturbação externa conduzida por um braço robótico, em objetos segurados por uma mão protética.

À seguir, cada sessão experimental é descrita em detalhes. Cabe salientar, que os dados destas sessões experimentais foram gerados em trabalho anterior (GENTILE *et al.*, 2020). Além disso, este estudo não inclui experimentos com humanos ou animais.

### 3.1.1.1 Primeira sessão experimental

Na primeira sessão experimental, um sensor FSR (Modelo 402 da Interlink Electronics) foi montado no topo do Sensor Multi-Eixo Força-Torque JR3. O FSR pode fornecer informações sobre forças normais de até 20 N com um limiar discriminante de 0,2 N, enquanto o JR3 mede as três componentes de força em uma faixa de  $\pm 200$  N com resolução de 0,015 N. Uma sonda desenvolvida para este fim foi posicionada no efetor final de um braço robótico antropomórfico (Kuka Light Weight Robot 4+ (ALBUSCHÄFFER *et al.*, 2007)) para aplicar forças nos sensores, como demonstra a Figura 10.

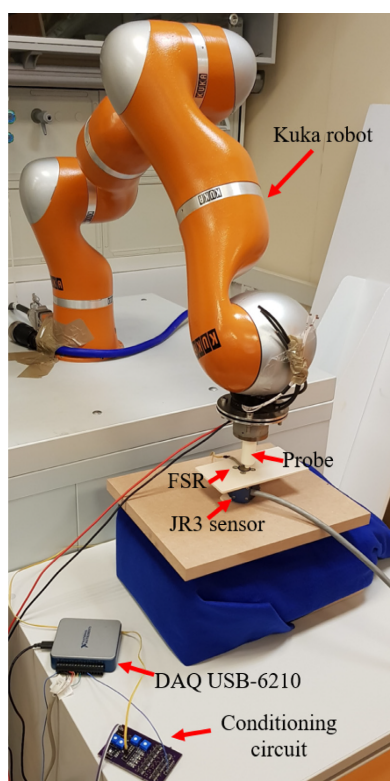


Figura 10 – Configuração para a primeira sessão experimental (GENTILE *et al.*, 2020).

Essa configuração experimental pode registrar simultaneamente a componente de força normal do sensor FSR e as componentes de força normal e tangencial do sensor JR3. O sinal condicionado FSR e as componentes de força JR3 foram adquiridas por um DAQ USB-6210 da *National Instruments* (NI) de forma síncrona. Para



estabelecer uma relação entre o sinal de saída do valor de tensão (V) do FSR e o valor de força (F), o sensor FSR foi estatisticamente caracterizado, conforme explicado em Gentile et al. (2020).

O braço robótico é caracterizado por sete graus de liberdade (*Degrees-of-Freedom* (DoFs)) ativos monitorados por sensores de posição e torque nas articulações. Um controlador paralelo de força/posição foi utilizado para deslizar a sonda exploratória ao longo da superfície do sensor FSR com força constante simulando um toque. A Figura 11.a apresenta a posição inicial e Fig 11.b demonstra a posição final do deslizamento da sonda sobre o sensor.

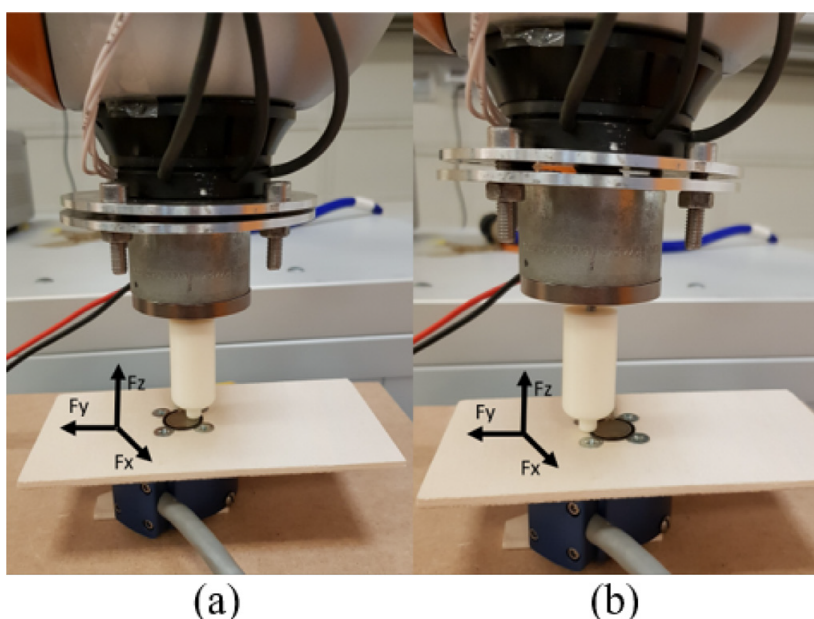


Figura 11 – Demonstração de deslizamento da sonda sobre o sensor FSR (GENTILE et al., 2020).

Para verificar o método em diferentes condições deslizamento sobre sensor, foram escolhidos três valores de força exercida pelo braço do robô e três durações do deslizamento da sonda, conforme apresenta a Tabela 2. O braço do robô foi controlado para mover a sonda sobre a superfície do sensor FSR nas condições descritas na Tabela 2.

Tabela 2 – Parâmetros de força e tempo usados na primeira sessão experimental para deslizamento da sonda sobre o sensor FSR.

	A	B	C
Força [N]	1,2	3,6	4,7
Tempo [s]	5	3	1

Desta forma, são nove combinações de experimentos, na qual, cada experimento foi repetido cinco vezes, somando um total de 45 testes realizados na primeira sessão experimental. Como resultado, a Figura 12 apresenta um exemplo dos sinais

gerados da saída dos sensores para a combinação da força C e tempo C (ver Tabela 2). É possível notar um deslocamento de força nas três componentes ( $F_x$ ,  $F_y$ ,  $F_z$ ) medidas pelo sensor JR3. No instante de tempo  $t_0$ , observa-se uma variação em todos os sinais de força (Figura 12.a força FSR, Figura 12.c  $F_x$ , Figura 12.d  $F_y$ , Figura 12.e  $F_z$ ) indicando que a sonda estava em contato com a superfície FSR. A sonda permanece estável até  $t_1$ , quando começa a deslizar na direção Y do sistema de referência JR3 (ver Figura 11), apenas parando ao ultrapassar a superfície FSR. Por este motivo, as forças ainda são aplicadas no sensor JR3 após  $t_2$ .

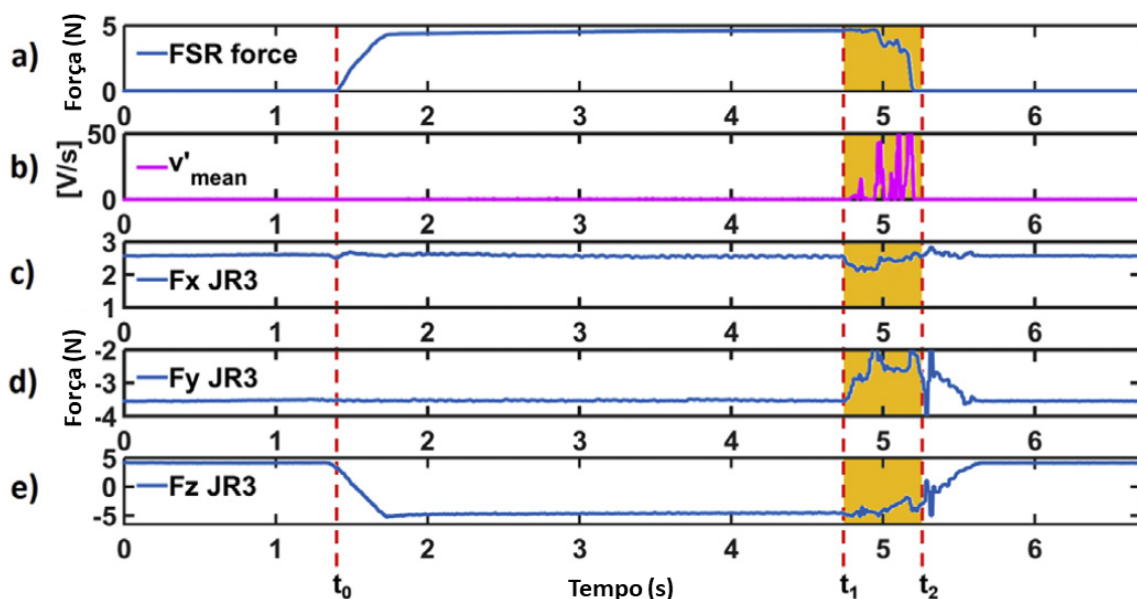


Figura 12 – Exemplo de resultado da primeira sessão experimental para a combinação da força C e tempo C (Adaptado de (GENTILE et al., 2020)).

Assim sendo, os resultados destes testes serão utilizados no treinamento de uma SNN capaz de detectar o contato da sonda com a superfície do sensor FSR em diferentes condições, inclusive durante as vibrações causadas pelo evento de deslizamento da sonda.

### 3.1.1.2 Segunda sessão experimental

Na segunda sessão experimental, dois sensores FSR foram colocados nos dedos polegar e indicador da mão robótica IH2 Azzurra (Prensilia srl) e cobertos com uma luva. Os dedos foram projetados pelos autores (GENTILE et al., 2020) com uma superfície plana para evitar problemas causados por contato incompleto entre os dedos e o objeto. A mão robótica foi ajustada para segurar um cubo (33 x 33 x 33 mm, 15,32g) equipado com um sensor magneto-inercial (M-IMU, XSENS MTw) para medir os componentes da aceleração induzida pelo deslizamento.

O BTS SMART-D Motion Capture System<sup>1</sup> foi usado para rastrear marcadores

<sup>1</sup> [www.btsbioengineering.com/it/](http://www.btsbioengineering.com/it/)

posicionados no índice da mão robótica e no cubo para detectar o deslocamento do objeto devido ao deslizamento. A distância euclidiana entre o marcador no cubo e a marcação no dedo indicador da mão robótica foi usada como referência do deslocamento. As medições foram realizadas com um sistema de análise de movimento composto por 7 câmeras com taxa de captura de 60 Hz e precisão de 0,1 mm em uma área de 2 x 2 m.

Também, um manipulador com 4 DoFs passivos e um efector tipo garra foi usado para garantir a repetitividade das condições experimentais (Figura 13.a). A pinça pode repetir a passagem do objeto na mão protética da mesma forma em cada tarefa. Além disso, para não interferir no trabalho, o manipulador pode alterar sua orientação retornando à sua posição original após liberar o objeto.

O braço robótico KUKA foi utilizado como apalpador para transmitir uma força impulsiva ao objeto agarrado, fazendo-o escorregar, conforme apresenta a Figura 13.b. Foram causados 30 eventos de deslizamento através da sonda conectada ao efector do braço robótico, após o cubo ser segurado firmemente pela mão robótica.

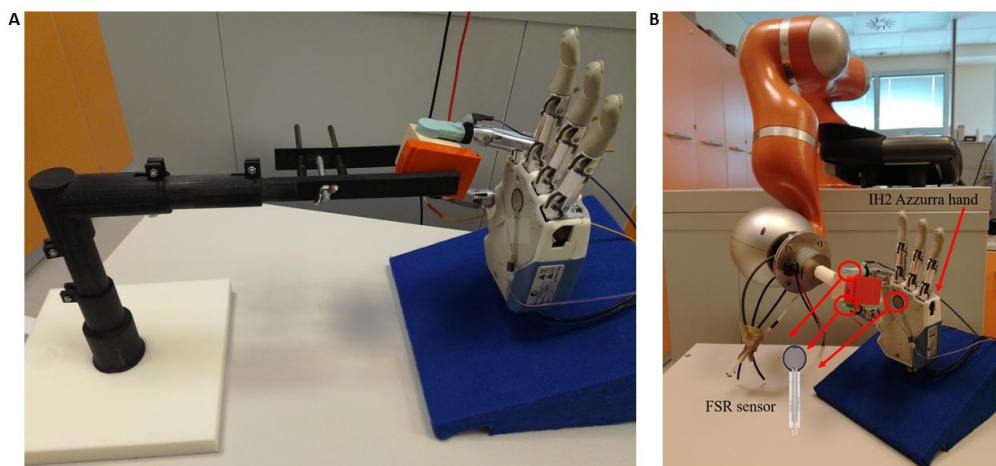


Figura 13 – (a) A mão robótica IH2 segurando o cubo equipado com o manipulador passivo e efector tipo garra; (b) Configuração para a segunda sessão experimental (GENTILE et al., 2020).

Como resultado do experimento, a Figura 14 mostra a) força, b) derivada do sinal de força, c) aceleração e d) deslocamentos relacionados à tarefa acima.

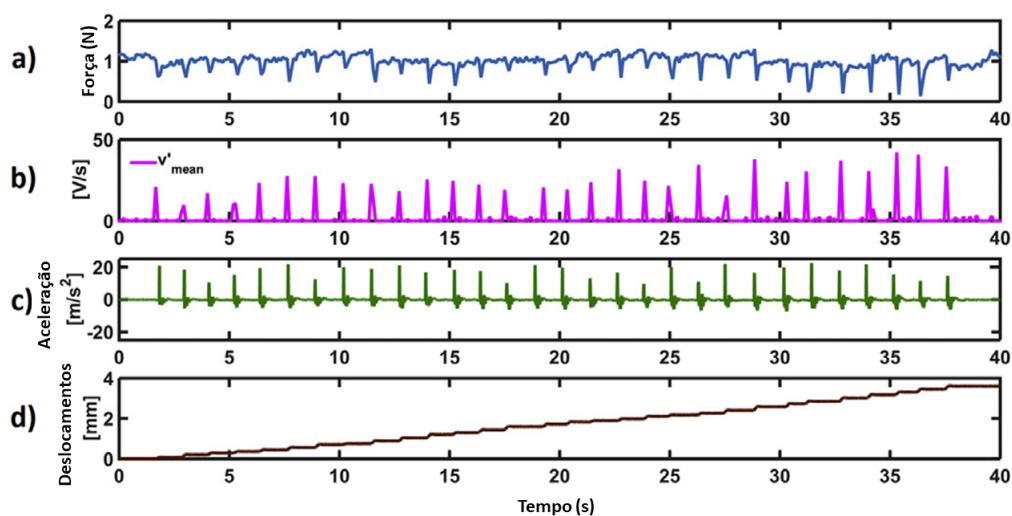


Figura 14 – 30 escorregamentos induzidos pelo robô KUKA (Adaptado de (GENTILE et al., 2020)).

Portanto, tais resultados do experimento acima descrito, serão utilizados no treinamento de uma SNN capaz de classificar o escorregamento dos 30 eventos causados por perturbação externa ao objeto agarrado.

### 3.1.2 Arquitetura da SNN para detecção de toque

A arquitetura da SNN do nosso sistema para detecção de toque, é definida por um problema de classificação binária supervisionada. Para isso, utilizamos nosso próprio métodos de rotulagem (*labeling*). Desta forma, pontos de dados individuais em nossos dados sequenciais podem ser classificados em uma das duas categorias: toque ou não toque. Assim sendo, a arquitetura do nosso sistema para classificação de toque, consiste em duas camadas: uma população de mecanorreceptores e uma camada de saída que classifica o toque (ver Figura 15). Cada uma dessas camadas será discutida a seguir.

A camada de entrada foi implementada com mecanorreceptores baseados no modelo proposto por Kim (KIM, S. S.; SRIPATI; BENSMAIA, Sliman J, 2010). De acordo com o modelo de Kim, descrito na seção 2.2.1.1, os mecanorreceptores do tipo SA-I são sensíveis ao sinal de força e sua primeira derivada. Desta forma, tendo em vista a variável estática transmitida pelos mecanorreceptores SA-I (que determina a duração do toque), esse é empregue na arquitetura.

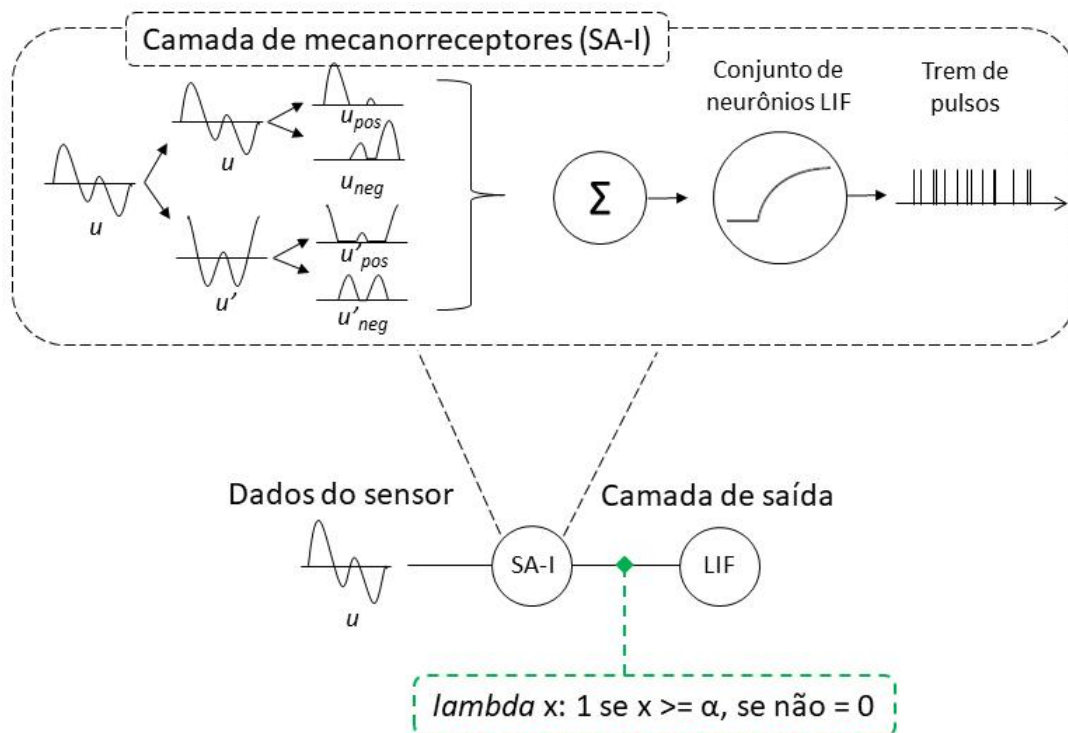


Figura 15 – Arquitetura da SNN para classificação de toque

Assim sendo, tomamos o resultado da força aplicada no sinal do sensor FSR  $u(t)$  e sua primeira derivada  $u'(t)$ . A derivada do sinal é calculada por diferenciação reversa, conforme a equação a seguir:

$$u' = \frac{d}{dt}u = \frac{u(i) - u(i-1)}{\text{cycle time}} \quad (2)$$

Posteriormente geramos sinais parcialmente retificados ( $u_{pos}$  e  $u'_{pos}$  para meia onda positiva,  $u_{neg}$  e  $u'_{neg}$  para meia onda negativa). Finalmente, os sinais retificados somados e normalizados para formar a corrente de entrada para um conjunto (*ensemble*) de neurônios do tipo LIF. A Eq. (3) demonstra os cálculos realizados.

$$sa-i = u_{pos} + u_{neg} + u'_{pos} + u'_{neg} \quad (3)$$

Além disso, os valores de  $sa-i$  são normalizados. A normalização é uma técnica frequentemente aplicada na etapa de preparação dos dados, com o objetivo de colocá-los em um intervalo de valores comuns, sem distorcer as diferenças nos intervalos de valores nem perder informações (PATRO; SAHU, 2015). Desta forma, foi possível adaptar os dados de entrada à faixa dinâmica da função de ativação da rede neural, na qual valores muito altos podem saturar a função de ativação, prejudicando a convergência da rede neural. A fórmula para normalizar os dados entre o intervalo 0 e 1 é fornecida abaixo.

$$z_i = \frac{x_i - \min(x)}{\max(x) - \min(x)} \quad (4)$$

onde,  $x_i$  é o valor da saída dos modelos de transdução;  $\min(x)$  é valor mínimo do conjunto de dados;  $\max(x)$  é o valor máximo do conjunto de dados; e  $z_i$  é a saída normalizada;

Por fim, para completar o os modelos de mecanorreceptores, a saída normalizada forma a corrente de entrada para um conjunto (*ensemble*) de neurônios do tipo LIF. O comportamento desse neurônio é descrito no Nengo com o seguintes parâmetros: *tau\_rc* é a constante de tempo RC da membrana, em segundos. Ele afeta a rapidez com que a tensão da membrana decai para zero na ausência de entrada. *tau\_ref* é o período refratário absoluto, em segundos. Este parâmetro representa o tempo em que a tensão da membrana é mantida em zero após um pico. *min\_voltage* é o valor mínimo para a tensão da membrana e *amplitude* é o fator de escala na saída do neurônio e corresponde à amplitude relativa dos picos de saída do neurônio. Neste trabalho, foram utilizados os valores padrões dos parâmetros do Nengo, fornecidos na Tabela 3.

Tabela 3 – Parâmetros do neurônio LIF.

Parâmetro	Valores	Tipo
<i>tau_rc</i>	0,02	float
<i>tau_ref</i>	0,002	float
<i>min_voltage</i>	0	float
<i>amplitude</i>	1	float

Em seguida, dada a atividade da população de mecanorreceptores, foram conectados a uma camada de saída com apenas um neurônio do tipo LIF. Nesta conexão, onde todas as saídas da primeira camada são ligadas ao neurônio de saída, é aplicada uma função de limiarização *lambda* (Eq. (5)) que corresponde à tomada de decisão quanto às duas possíveis classes. Assim sendo, quando o resultado da saída da camada de mecanorreceptores for maior ou igual a  $\alpha$ , a resposta na camada de saída deve ser 1, caso contrário, zero. A variável  $\alpha$  foi definida com o valor 0,3, o mesmo valor definido para o variável  $\alpha$  na rotulagem dos dados (seção 3.1.2.1).

$$f(x) = \begin{cases} 1, & \text{if } x \geq \alpha. \\ 0, & \text{else.} \end{cases} \quad (5)$$

Portanto, como resultado, o neurônio será acionado quando detectar um evento de toque, mais especificamente, a camada de saída classifica uma das duas categorias: toque (1) ou não toque (zero). Visto a definição de um problema de aprendizado supervisionado, o treinamento desta arquitetura conta com uma referência, na qual o método de rotulagem será apresentado a seguir.



### 3.1.2.1 Rotulagem dos dados

Para identificar os momentos exatos de toque e utilizá-los como referência (*ground truth*) da arquitetura de detecção de toque, foi criada uma função em *python* para que cada ponto no gráfico o algoritmo apresente se houve toque ou não. Para isso, a função detecta um toque quando há um valor maior ou igual a  $\alpha$  retornando o valor 1, caso contrário, não há toque e resulta no valor zero (a Equação (5) também descreve a função implementada). Como forma de validar o algoritmo, a saída foi verificada manualmente para todos os 45 testes em relação as componentes tangenciais do sensor JR3 (assim como verificado no trabalho anterior), a força e a derivada do sensor FSR. Como resultado, o valor de  $\alpha$  definido foi de 0,3. A Figura 16 apresenta o resultado para um dos dados de teste, sendo que a Figura 16.a demonstra o sinal da força FSR e o limiar  $\alpha$  aplicado, bem como, na Figura 16.b apresenta o resultado do algoritmo.

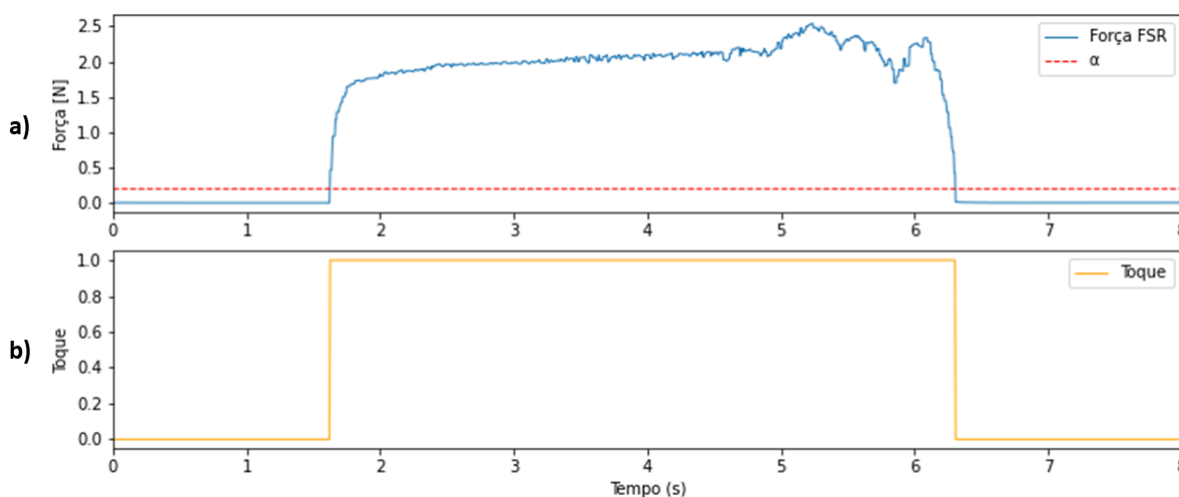


Figura 16 – Exemplo de referência criada para detecção do toque

Cabe salientar, que para cada um dos testes realizados na primeira sessão experimental foi aplicado este algoritmo, resultando em 45 sinais que serão utilizados como referência no aprendizado e seleção do melhor modelo para a arquitetura de detecção de toque.

### 3.1.2.2 Aprendizado e seleção do melhor modelo

Nos próximos parágrafos serão discutidas as etapas de aprendizado e busca dos melhores modelos para a SNN de detecção de toque. Para isso, inicialmente vamos definir dois conceitos importantes, a diferença dos parâmetros e hiperparâmetros. Em um algoritmo *machine learning*, os parâmetros são ajustados diretamente pelo processo de aprendizado e influenciam na performance do algoritmo. Um exemplos de parâmetros são os pesos de conexão entre as camadas da rede. Já os hiperparâmetros são variáveis do algoritmo definidas antes do treinamento, como por exemplo,

o número de neurônios de uma camada, a taxa de aprendizado e algoritmo de treinamento. Assim, abaixo são definidas as etapas do processo de aprendizado, bem como a Figura 17 esquematiza a sequência de etapas.

1. Inicialmente, dividimos aleatoriamente nossos dados disponíveis em dois subconjuntos: um conjunto de treinamento (80%) e um conjunto de teste (20%). Este procedimento é realizado para manter os dados de testes independentes. Desta forma o conjunto de teste deve representar dados novos e não vistos para o modelo. Ainda, o conjunto de dados de treinamento é subdividido em treinamento (80%) e validação (20%), permitindo comparar diferentes modelos e hiperparâmetros.
2. Utilização de validação cruzada k-fold para avaliar a performance do modelo através de uma pesquisa de hiperparâmetros em grade (*grid search*).
  - a) Quatro diferentes hiperparâmetros foram combinados de maneira sistemática na busca pela configuração de melhor desempenho na classificação de toque: números de neurônios, otimizador, taxa de aprendizado e quantidade de épocas. O número de neurônios foi variado de 200 à 10 em um intervalo de 10 neurônios, totalizado 20 valores diferentes; os otimizadores utilizados foram o Adam e o RMSprop, visto que são dois dos algoritmos estocásticos adaptativos mais influentes para o treinamento de redes neurais profundas (ZOU *et al.*, 2019); a taxa de aprendizado foi variada em três valores: 0,01, 0,001 e 0,0001; e o número de épocas de treinamento foi variado em 20, 40 e 60. Para combinar tais hiperparâmetros, foi realizada uma pesquisa em forma de grade, ou seja, todas as possibilidades dentre as opções. Para implementar este procedimento foi utilizado o painel *HParams*, um *plugin* do TensorBoard que permite implementação da variação de hiperparâmetros em redes neurais, bem como, fornece um painel de visualização dos resultados.
  - b) Para cada hiperparâmetro configurado, aplicamos o método de validação cruzada k-fold cross-validation no conjunto de treinamento, resultando em múltiplos modelos e estimativas de desempenho. Desta forma, é possível validar toda a extensão de dados do experimento. A validação cruzada *k-fold* é apresentada na seção 2.3.2. Além disso, a função *Kfold* da biblioteca *sklearn* foi utilizada para implementar esta funcionalidade.
  - c) Da performance resultante foi calculada a média entres todos os cinco valores adquiridos de cada *fold* da validação cruzada, para



cada métrica avaliada. Os treinamentos foram avaliados em termos de precisão, revocação e f1-score, bem como adquiridos os tempos de treinamento.

3. Em seguida, através dos melhores resultados obtidos na etapa anterior são escolhidos os melhores hiperparâmetros.
4. Por fim, usamos o conjunto de teste independente que retivemos anteriormente (Etapa 1) e utilizamos este conjunto de teste para avaliar o modelo obtido na Etapa 3.

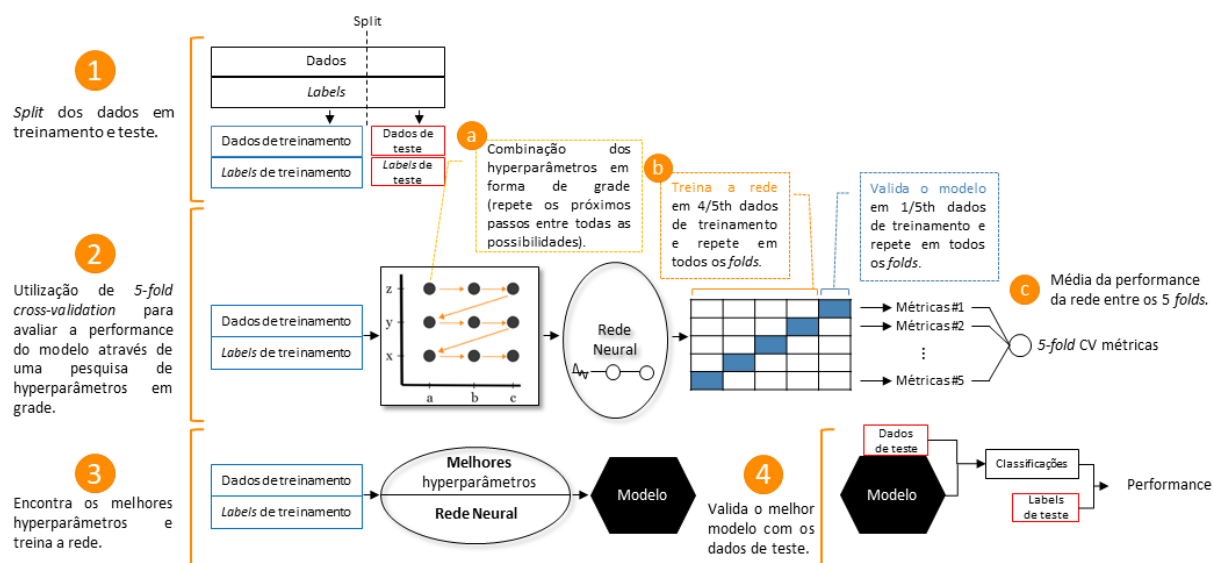


Figura 17 – Ilustração da etapas de busca dos melhores hiperparâmetros para a arquitetura de detecção de toque.

Com o fluxo de aprendizado composto destas etapas, foi executado um total de 360 treinamentos para cada *fold* da validação cruzada. Como resultado, foram selecionados os melhores conjuntos de hiperparâmetros para cada quantidade de neurônios na primeira camada, que por sua vez, serão descritos no capítulo 4.

### 3.1.3 Arquitetura da SNN para detecção de escorregamento

Para a detecção de escorregamento, tomamos como base um problema de classificação binária, que também utiliza o próprio método de rotulagem. Desta forma, pontos de dados individuais nos dados sequenciais podem ser classificados em uma das duas categorias: escorregamento ou não escorregamento. A arquitetura da rede neural para detecção de escorregamento consiste em duas camadas: uma população de mecanorreceptores e uma camada de saída que classifica entre as duas categorias (ver Figura 18). Portanto, cada uma dessas camadas será discutida em seguida.

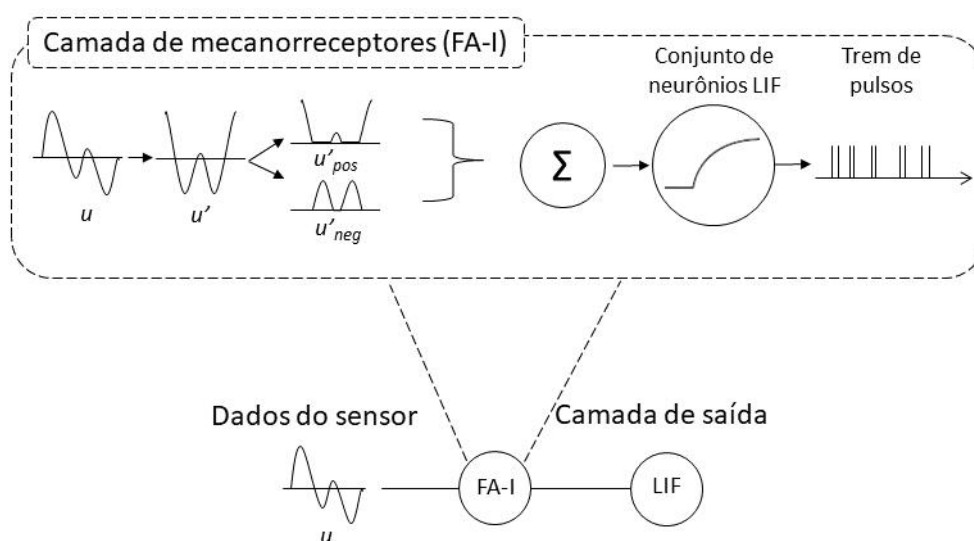


Figura 18 – Arquitetura SNN para classificar o deslizamento

Para a construção da camada de mecanorreceptores, foi seguido o modelo de mecanorreceptores de Kim (KIM, S. S. et al., 2011), da mesma forma que na arquitetura para detecção de toque. A diferença é a utilização do conceito de mecanorreceptores do tipo FA-I ao invés dos aferentes SA-I, visto que, de acordo com a literatura (capítulo 2) os aferentes FA fornecem um código espaço-temporal confiável de quando um deslizamento ocorreu.

De acordo com o modelo de Kim, os mecanorreceptores do tipo FA-I são sensíveis apenas à primeira derivada. A Eq. (6) demonstra seu funcionamento.

$$fa-i = u'_{pos} + u'_{neg} \quad (6)$$

Em seguida, para completar o modelo de mecanorreceptor FA-I, a saída normalizada (Eq. (4)) forma a corrente de entrada para um conjunto (*ensemble*) de neurônios do tipo LIF. As características dos neurônios são fornecidas na tabela 3. Por fim, dada a atividade de pico da população de mecanorreceptores, a primeira camada foi conectada a uma camada de saída com apenas um neurônio também do tipo LIF. Como resultado, o neurônio de saída ativa sua saída em resposta à detecção de um evento

de deslizamento, bem como permanece inativo quando nenhum deslizamento for identificado.

Ainda, para a simulação desta arquitetura e permitir a correta validação da mesma, foi implementado um procedimento de rotulagem do dados, ou seja, criação de uma referência de saída ideal para os escorregamentos induzidos da segunda sessão experimental.

### 3.1.3.1 Rotulagem de dados

Para capturar os momentos exatos de deslizamento, usamos a informação de mudança repentina na força do objeto disponível na Figura 14. Especificamente, os dados do sensor FSR foram processados usando SciPy (VIRTANEN *et al.*, 2020) e verificados manualmente em relação aos dados de deslocamento e as medições do BTS SMART-D Motion Capture System. O SciPy é uma extensão Python que contém uma biblioteca de métodos matemáticos e funções utilitárias. Ao comparar os valores circundantes, a função *find peaks* através de um *array* 1-D descobre todos os mínimos locais. O algoritmo retorna os índices de picos em que satisfaçam todas as condições dadas, atribuindo 1 (*slip label*) aos índices de picos. Aos pontos de dados restantes é atribuído zero. A Figura 19.b mostra o sinal de referência criado.

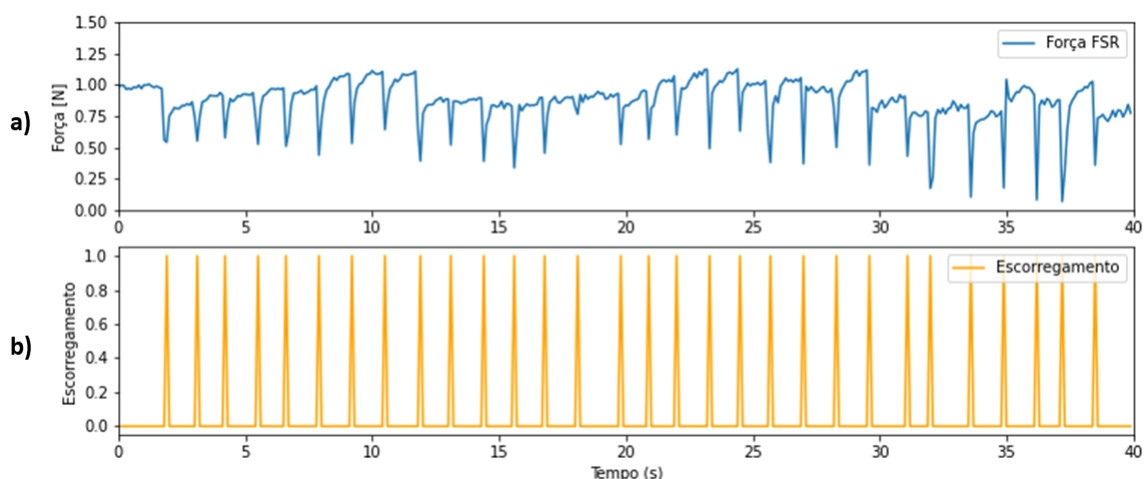


Figura 19 – Referência criada para detecção de escorregamento

Como resultado, temos, respectivamente, 380 e 30 amostras de dados rotuladas de não deslizamento e deslizamento. Estes dados serão utilizados para simulação e seleção dos melhores hiperparâmetros para a arquitetura de detecção de escorregamento.

### 3.1.3.2 Simulação e seleção do melhor modelo

Nesta seção são discutidas as etapas adotadas na seleção dos melhores modelos de SNN para a detecção de escorregamento. Diferentemente das etapas de

busca de hiperparâmetros para a detecção de toque apresentadas na seção 3.1.2.2, a SNN para a detecção de escorregamento impõe algumas diferenças devido à sua forma de construção e objetivo. Inicialmente, os dados não são divididos em dois conjuntos (treinamento e teste), tendo em vista que, neste caso, não há a necessidade de treinamento de pesos sinápticos entre as camadas, ou seja, os parâmetros. Este fato ocorre, pois com testes iniciais os resultados demonstram alta taxa de acerto com as configurações padrões de pesos sinápticos. Assim sendo, será realizada apenas a simulação com todos os dados do experimento e busca dos melhores hiperparâmetros. Portanto, a Figura 20 apresentada a estas seguidas para seleção do melhor modelo de detecção de escorregamento, bem como são descritas a seguir:

1. Foram combinados de maneira sistemática três hiperparâmetros diferentes na busca das combinações que produzem o melhor desempenho na classificação do escorregamento, são eles: números de neurônios e dois hiperparâmetros específicos do neurônio da segunda camada, os *intercepts* e *max rates*. Tais hiperparâmetros nos permitem controlar, respectivamente, o limiar e a frequência de disparo do neurônio. O número de neurônios foi variado de 200 à 10 em um intervalo de 10 neurônios, totalizado 20 valores diferentes; o *intercepts* foram variados em três valores: 0,005, 0,01 e 0,015; e o *max rates* de treinamento foi variado em 5, 10 e 15. Para combinar tais hiperparâmetros, foi realizada uma pesquisa em forma de grade, ou seja, todas as combinações dentre as opções. Para implementar este procedimento foi utilizado o painel *HParams*.
2. Através dos melhores resultados obtidos na etapa anterior são escolhidos os melhores hiperparâmetros.

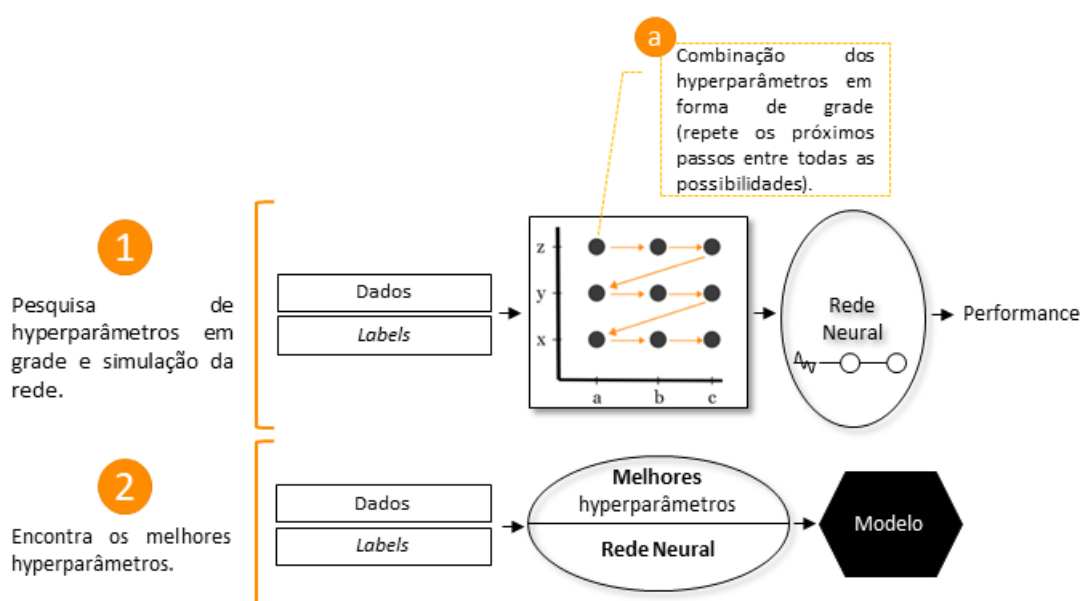


Figura 20 – Etapas de seleção dos melhores modelos de detecção de escorregamento.

Com as etapas acima seguidas, foram realizados um total de 180 simulações diferentes. Portanto, como resultado, foram encontrados os melhores conjuntos de hiperparâmetros para cada quantidade de neurônios na primeira camada, permitindo posterior análise no capítulo 4.

## 4 RESULTADOS

Para analisar o desempenho dos modelos de SNN para classificação de toque e escorregamento, serão apresentados os resultados no processo de ajuste de parâmetros e hiperparâmetros durante o treinamento das redes, o tempo de treinamento, cálculos de consumo de *hardware* e as simulações em tempo real realizadas.

O desempenho dos modelos foi avaliado usando uma combinação de três métricas: precisão, revocação e f1-score. Primeiramente, é importante lembrar algumas definições. A taxa de verdadeiros positivos (VP), é a probabilidade de que os escorregamentos sejam classificados corretamente. A taxa de verdadeiros negativos (VN), é a probabilidade de que nenhum escorregamento seja classificado corretamente. A taxa de falsos positivos (FP) é a probabilidade de que nenhum escorregamento seja classificado falsamente. A taxa de falsos negativos (FN) é a probabilidade de que os deslizamentos sejam classificados falsamente. Assim sendo, podemos definir as três métricas (precisão, revocação e f1-score). A precisão, também conhecida como valor preditivo positivo, indica qual porcentagem dos eventos classificados é esperada como correta ( $VP/(VP+FP)$ ). Para uma alta precisão o classificador deve ter uma baixa taxa de falsos positivos. A sensibilidade ou revocação indica a habilidade do classificador de reconhecer uma grande porcentagem dos eventos positivos como tal, ou seja, para uma alta sensibilidade, a taxa de falsos negativos deve ser baixa ( $VP/(VP+FN)$ ). A métrica f1-score representa a média harmônica entre os valores da revocação e precisão (HOSSIN; SULAIMAN, 2015).

Visto isso, em seguida serão apresentados separadamente cada um dos resultados dos modelos de classificação de toque e escorregamento. Também, cabe salientar que neste capítulo são apresentados os resultados, sendo que apenas serão discutidos no capítulo de discussões.

### 4.1 CLASSIFICAÇÃO DE TOQUE

Durante o treinamento da SNN para detecção de toque foram variados diversos parâmetros, a fim de descobrir as melhores configurações da rede para essa tarefa. Desta forma, de maneira sistemática diversas combinações dos parâmetros como números de neurônios, otimizador, taxa de aprendizado e número de épocas foram realizadas. Tais combinações geraram 360 treinamentos diferentes. Além disso, com a utilização de validação cruzada cada um dos treinamentos foi repetido cinco vezes, cada teste em uma rodada diferente.

A Figura 21 apresenta um gráfico de coordenadas paralelas, com a média dos resultados obtidos em cada rodada de teste. Neste tipo de visualização, cada variável recebe seu próprio eixo vertical e todos os eixos são dispostos em paralelo entre si. Cada eixo pode ter uma escala diferente, já que cada variável trabalha com uma

unidade de medida distinta. Os valores são representados como uma série de linhas conectadas em cada um dos eixos. Isso significa que cada linha é uma coleção de pontos colocados em cada eixo, que foram conectados entre si, ou seja, cada linha é conectada com os hiperparâmetros utilizados em um treinamento, juntamente com o resultado adquirido. Assim sendo, da direita para esquerda, visualizamos o número de neurônios, o otimizados, a taxa de aprendizado, as épocas a métrica f1-score. Desta forma, os gráficos de coordenadas paralelas permitem comparar muitas variáveis e ver as relações entre elas. Cabe salientar, que para uma melhor visualização o gráfico é apresentado apenas com os resultados da métrica f1-score. Também, a direita do gráfico é exibida uma paleta de cores que representa o número de neurônios, na qual cores frias são valores menores e a medida que os número de neurônios aumenta as cores mais quentes são distribuídas.

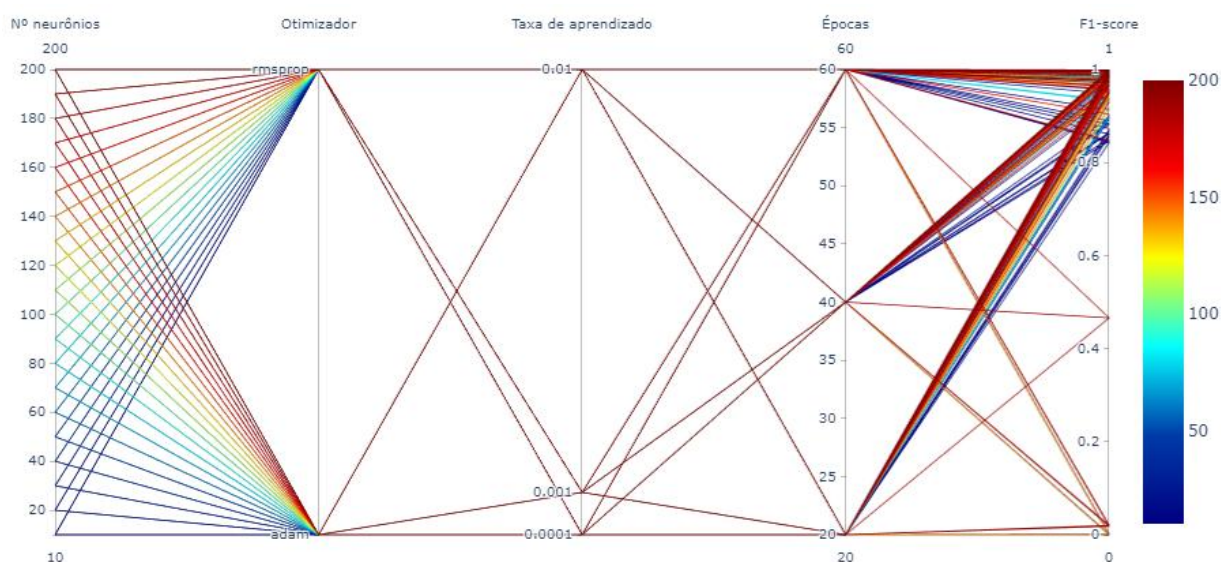


Figura 21 – Gráfico paralelo a média dos resultados na variação dos hiperparâmetros para a SNN de detecção de toque.

Também, na Fig 22 é possível visualizar uma matriz de gráficos de dispersão ou *scatter plots* que nos permite analisar minuciosamente as correlações entre os parâmetros. Os gráficos utilizam de pontos para denotar essa relação, onde cada ponto representa o valor de uma variável no eixo horizontal e o valor de outra variável no eixo vertical. Desta forma, cada gráfico de dispersão na matriz apresenta a relação entre um par de variáveis, permitindo que muitas relações sejam exploradas em um gráfico. As cores vermelho e azul representam, respectivamente, o hiperparâmetro otimizador Adam e RMSprop. Também, os gráficos na diagonal apresentam a distribuição para cada variável.

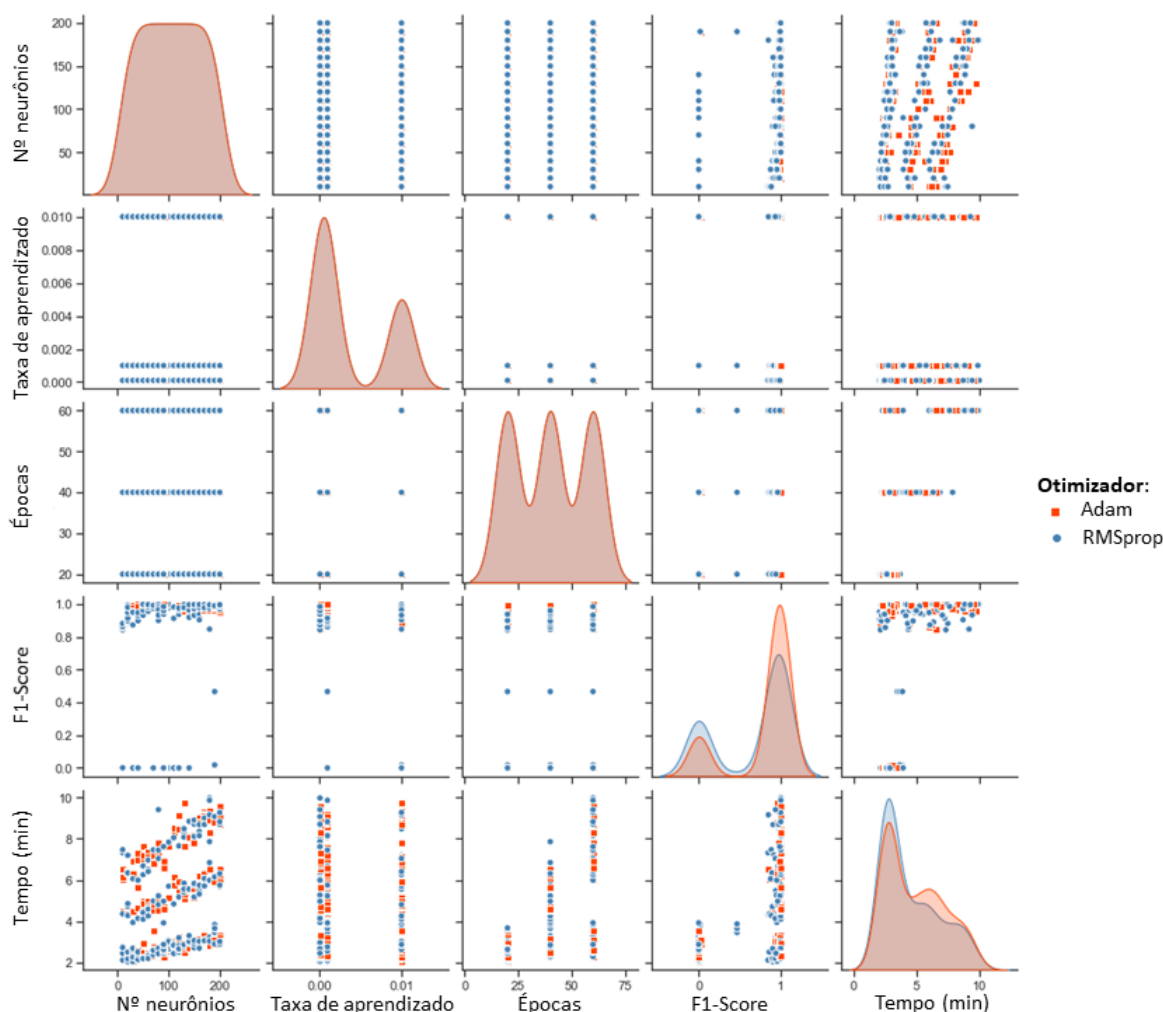


Figura 22 – Matriz de gráficos de dispersão dos resultados para a arquitetura da SNN de detecção de toque.

A Tabela 4 apresenta o resultado das melhores performances em para cada quantidade de neurônios. Em alguns casos, para certa quantidade de neurônios na primeira camada foi obtido mais de um modelo com o mesmo resultado, portanto, foi selecionado o modelo com a performance obtida em menor tempo de treinamento.

Na Figura 23 é apresentado um exemplo dos resultados da simulação com os dados de um teste da primeira sessão experimental. O teste combina a força A com a velocidade A (Tabela 2), e é apresentado quando treinado com o melhor resultado obtido para 20 neurônios (Tabela 4). A Figura 23.a representa a força aplicada sobre o sensor, já a Figura 23.b mostra a atividade de pico da primeira camada da rede e a 23.c apresenta atividade de pico da segunda camada da rede, ou seja, o resultado da simulação. Note que, durante todo o tempo de toque há a o disparo do neurônio na segunda camada, por esse motivo a saída é similar a uma tarja preta. Para melhor visualização, é possível observar a ampliação do resultado (ver Figura 23.c), onde a cada instante de tempo da simulação um pulso é gerado.



Tabela 4 – Resultados dos treinamentos para a SNN de toque que obtiveram melhor performance para cada quantidade de neurônios na primeira camada.

Nº de neurônios	Otimizador	Taxa de aprendizado	Épocas	Precisão	Sensibilidade	F1-Score	Tempo (min)
200	RMSprop	0,0001	40	0,999	0,998	0,999	5,687
190	Adam	0,0001	40	0,999	0,996	0,998	6,032
180	RMSprop	0,001	60	0,999	0,998	0,999	9,865
170	RMSprop	0,0001	20	0,999	0,998	0,999	2,997
160	RMSprop	0,001	40	0,999	0,998	0,999	5,323
150	Adam	0,0001	40	1,000	0,996	0,998	5,763
140	Adam	0,0001	20	1,000	0,996	0,998	2,913
130	RMSprop	0,001	40	0,999	0,998	0,999	5,406
120	Adam	0,0001	40	1,000	0,997	0,998	5,413
110	Adam	0,0001	20	1,000	0,996	0,998	2,904
100	Adam	0,0001	40	0,999	0,998	0,999	5,119
90	Adam	0,0001	20	1,000	0,996	0,998	2,393
80	RMSprop	0,001	20	0,999	0,994	0,997	2,683
70	Adam	0,0001	40	1,000	0,996	0,998	4,543
60	RMSprop	0,001	20	0,999	0,996	0,998	2,319
50	RMSprop	0,01	40	1,000	0,996	0,998	4,099
40	Adam	0,001	20	1,000	0,995	0,997	2,266
30	Adam	0,0001	40	1,000	0,996	0,998	4,555
20	Adam	0,0001	40	0,999	0,994	0,996	4,626
10	Adam	0,01	40	0,799	0,995	0,886	4,483

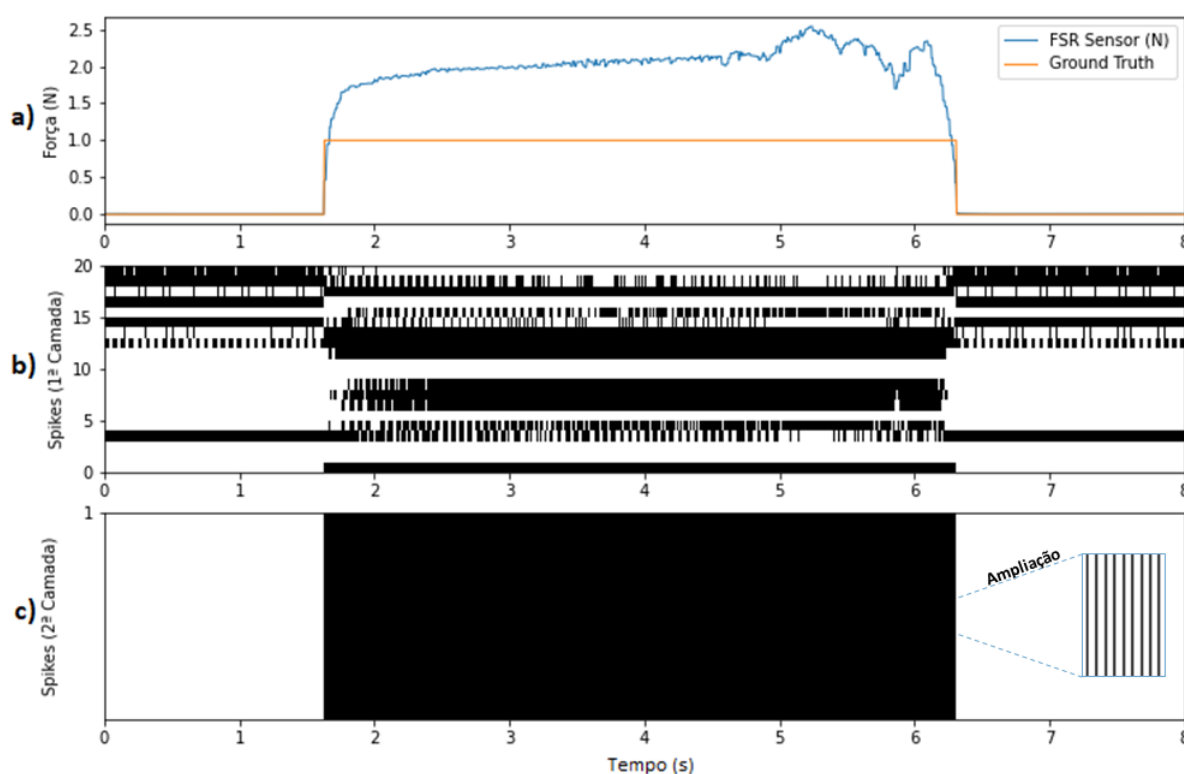


Figura 23 – Resultados da simulação com os dados do teste Força A e Velocidade A da primeira sessão experimental, para a SNN de detecção de toque

Além do cenário de aprendizado da arquitetura e sua resposta, uma das principais motivações para usar SNN é o potencial de economia significativa de energia em relação às técnicas padrão. Portanto, é útil poder estimar quanta energia seria usada por um modelo em diferentes dispositivos, para que possamos ter uma ideia de como diferentes parâmetros de modelo ou dispositivo afetam o uso de energia antes de realizar uma implantação completa. Portanto, a Figura 24 apresenta um gráfico do

consumo de energia total estimado por classificação/inferência para dois *hardwares* neuromórficos: Loihi e SpiNNaker 2, bem como, para uma unidade central de processamento (*Central Processing Unit* (CPU)) e um processador ARM. A CPU utilizada nas análises é o *Intel i7-4960X*, já o processador ARM é o Cortex-A5. Os *hardwares* analisados são representados em diferentes cores, sendo vermelho para o Loihi, verde para o SpiNNaker 2, azul para a CPU e laranja para o ARM. Para uma melhor visualização, foram escolhidos cinco valores de número de neurônios na primeira camada (200, 100, 50, 20 e 10). Desta forma, no eixo y é possível visualizar a quantidade de neurônios e no eixo x os valores totais do consumo de energia do modelo de classificação de toque apresentado. O apêndice A demonstra em detalhes os cálculos do consumo de energia realizados.

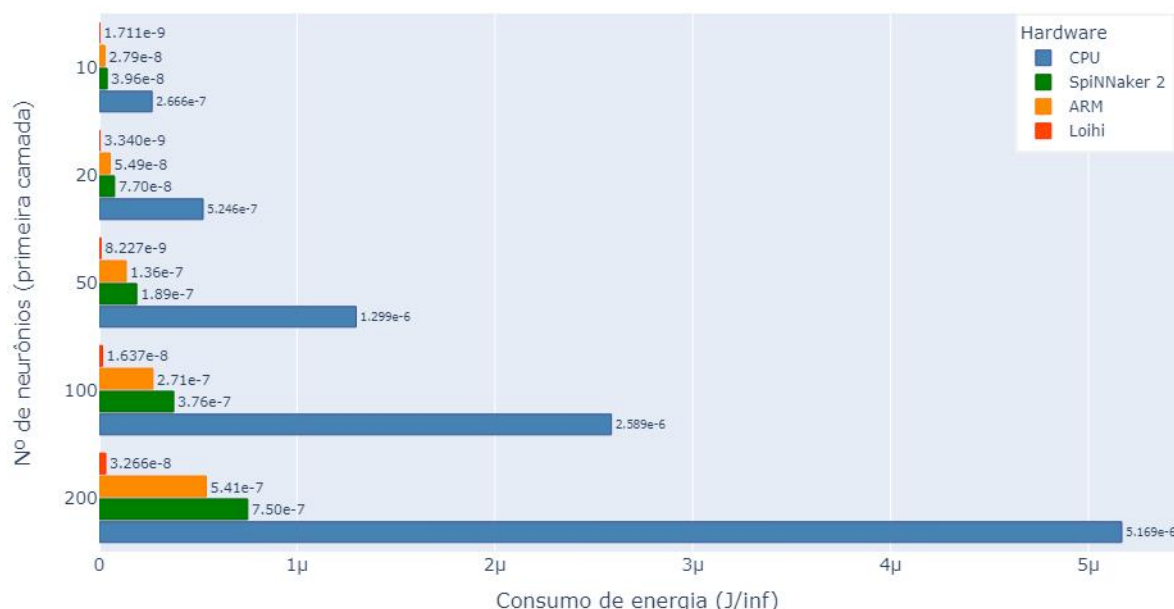


Figura 24 – Gráfico do consumo total de energia estimado por classificação para os *hardwares* Loihi, SpiNNaker 2, CPU e ARM (SNN de classificação de toque).

Cabe salientar, que quando tratamos de analisar diferentes custos de energia em processadores não pulsados, ou seja, a CPU e o ARM, assumimos que cada *synapse* e atualização do estado do neurônio é realizado com base na métrica de energia acumulada multiplicada (*Multiply-Accumulate* (MAC)) por operação nos processadores, conforme descrito no apêndice A.

Sendo assim, foram apresentados os resultados adquiridos do modelo de SNN para classificação de toque. Cabe salientar, que os gráficos e imagens apresentadas foram realizadas utilizando bibliotecas da linguagem python como: Plotly<sup>1</sup> e Seaborn<sup>2</sup>. No capítulo 5 serão discutidos os resultados apresentados.

<sup>1</sup> <https://www.plotly.com>

<sup>2</sup> <https://www.seaborn.pydata.org>

## 4.2 CLASSIFICAÇÃO DE ESCORREGAMENTO

Nesta seção serão apresentados os resultados dos treinamentos realizados para a arquitetura de classificação de escorregamento. Para isso, conforme apresentado na seção 3.1.3.2 foram variados alguns hiperparâmetros, a fim de descobrir as melhores configurações da rede para esse objetivo. Os hiperparâmetros combinados de maneira sistemática foram o número de neurônios, *intercepts* e *max rates*. Tais combinações geraram 180 simulações diferentes.

A Figura 25 demonstra um gráfico de coordenadas paralelas com os resultados obtidos das simulações para todo conjunto de dados do segundo experimento. Para isso, cada hiperparâmetro recebe seu próprio eixo e todos os eixos são dispostos em paralelo entre si. A diferença para o gráfico da Figura 21 são as variáveis de cada eixo, neste caso, o número de neurônios, *intercepts*, *max rates* e f1-score. Portanto, cada linha no gráfico representa uma combinação de hiperparâmetros para a simulações da rede de classificação de escorregamento.

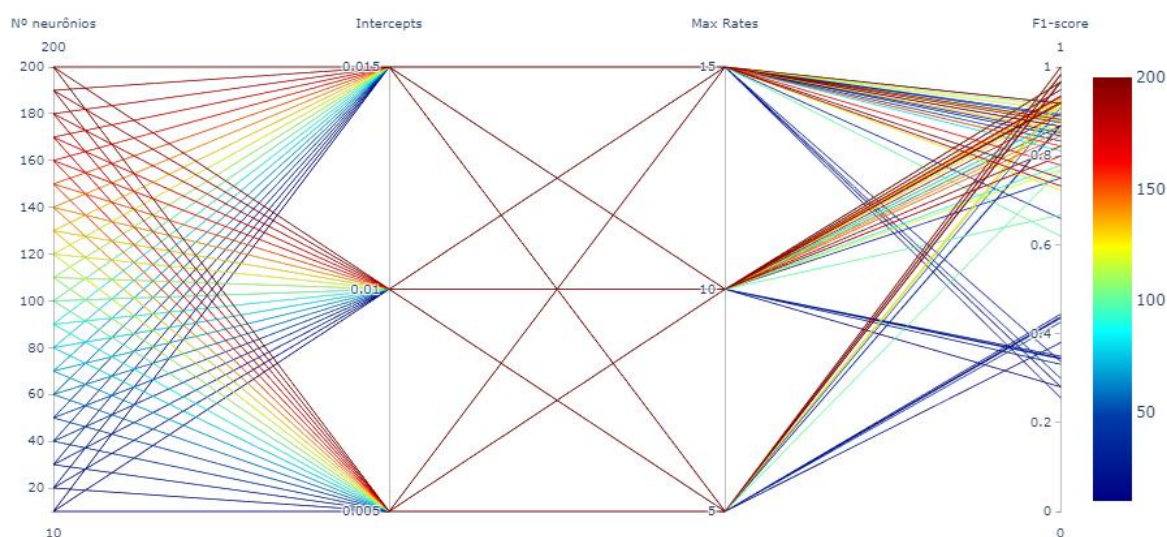


Figura 25 – Gráfico paralelo dos resultados na variação dos hiperparâmetros para a SNN de detecção de escorregamento.

Da mesma forma que foi apresentado os resultados da arquitetura de detecção de toque, a Figura 26 demonstra uma matriz de gráficos de dispersão, permitindo analisar as correlações entre os hiperparâmetros variados para rede de detecção de escorregamento. As cores do gráfico são definidas de acordo com cada valor de *max rates*, sendo 5 para vermelho, 10 para azul e 15 para verde. Permitindo assim, analisar a influência deste hiperparâmetro em todos os gráficos da matriz.

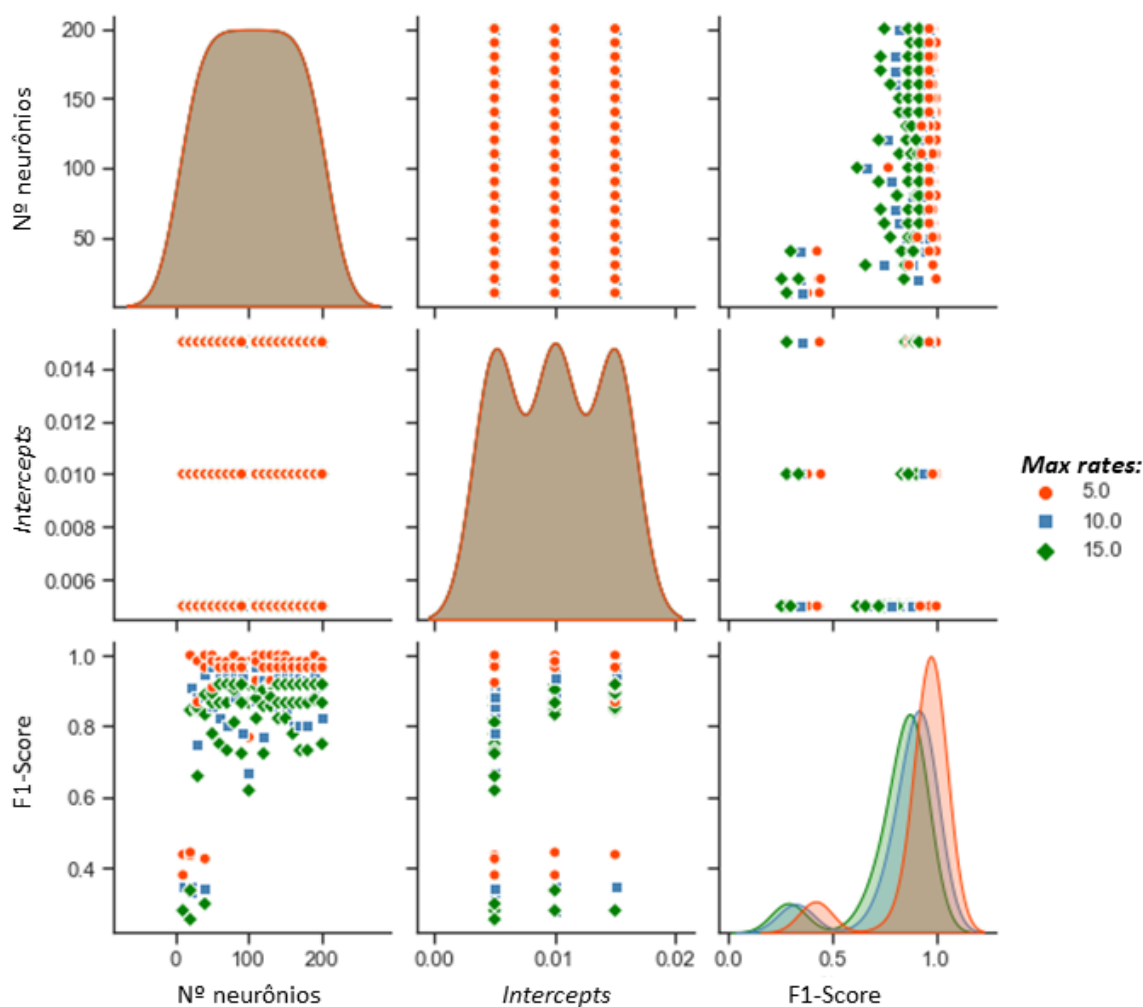


Figura 26 – Matriz de gráficos de dispersão dos resultados para a arquitetura da SNN de detecção de escorregamento.

Também, os resultados das melhores performances para cada quantidade de neurônios das simulações realizadas, são apresentados na Tabela 5. Para isso, da esquerda para a direita cada coluna representa, respectivamente, o número de neurônios, *intercepts*, *max rates* e as métricas de avaliação: precisão, sensibilidade e f1-score.

Na Figura 27 é apresentado um exemplo dos resultados da simulação com os dados da segunda sessão experimental. O teste é realizado com todo o dado experimental, ou seja, 41 segundos de simulação. O gráfico apresenta o resultado obtido para melhor performance utilizando 20 neurônios na primeira camada, conforme a Tabela 5. A Figura 27.a representa a força aplicada sobre o sensor, já a Figura 27.b mostra a atividade de pico da primeira camada da rede, ou seja, cada linha do gráfico apresenta os pulsos gerados por cada um dos 20 neurônios da camada. Já a Figura 27.c apresenta atividade de pico da segunda camada da rede, ou seja, o resultado da simulação. Neste cenário, conforme descrito na arquitetura da rede, a saída é apresentada por apenas um neurônio, portanto cada linha na vertical da Figura 27.c é um pulso deste neurônio.

Tabela 5 – Resultados dos treinamentos para a SNN de escorregamento que obtiveram melhor performance para cada quantidade de neurônios na primeira camada.

Nº de neurônios	Intercepts	Max rates	Precisão	Sensibilidade	F1-Score
200	0,005	5	1,000	0,968	0,984
190	0,005	5	1,000	1,000	1,000
180	0,01	5	0,967	1,000	0,983
170	0,01	5	0,967	1,000	0,983
160	0,01	5	0,967	1,000	0,983
150	0,005	5	1,000	1,000	1,000
140	0,005	5	1,000	1,000	1,000
130	0,005	5	1,000	1,000	1,000
120	0,01	5	1,000	1,000	1,000
110	0,005	5	1,000	1,000	1,000
100	0,01	5	0,967	1,000	0,983
90	0,01	5	0,967	1,000	0,983
80	0,005	5	1,000	1,000	1,000
70	0,01	5	0,967	1,000	0,983
60	0,005	5	1,000	0,968	0,984
50	0,005	5	1,000	1,000	1,000
40	0,01	5	1,000	1,000	1,000
30	0,01	5	1,000	0,968	0,984
20	0,015	5	1,000	1,000	1,000
10	0,015	5	1,000	0,280	0,438

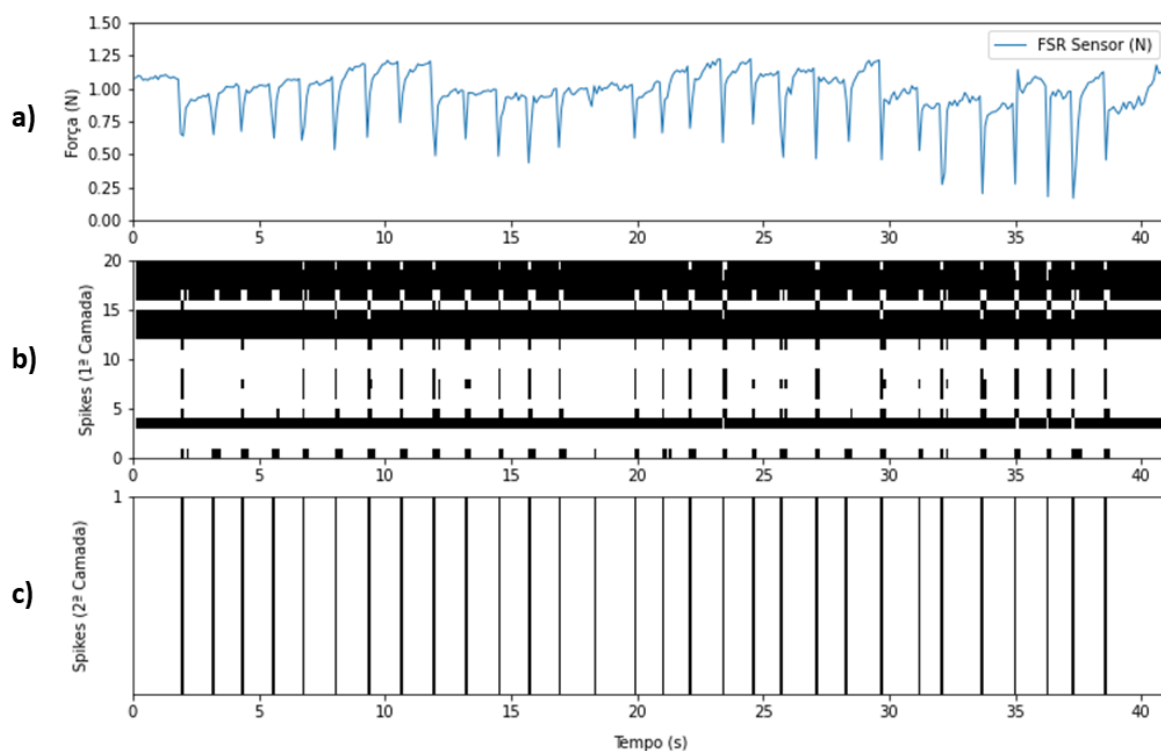


Figura 27 – Resultados da simulação com os dados da segunda sessão experimental, para a SNN de detecção de escorregamento

Por fim, conforme apresentado anteriormente, é útil poder estimar quanta energia seria usada por um modelo em diferentes dispositivos. Assim sendo, a Figura 28

apresenta um gráfico de consumo de energia total estimado por classificação/inferência para dois *hardwares* neuromórficos: Loihi (vermelho) e SpiNNaker 2 (verde). O gráfico demonstra o consumo de energia para uma CPU *Intel i7-4960X* (azul) e um processador ARM Cortex-A5 (laranja). Foram escolhidos cinco valores de número de neurônios na primeira camada (200, 100, 50, 20 e 10) para visualização do consumo de energia. Desta forma, no eixo y apresenta a quantidade de neurônios para cada *hardware*. Já no eixo x os valores totais do consumo de energia, em Joules, do modelo de classificação de escorregamento apresentado.

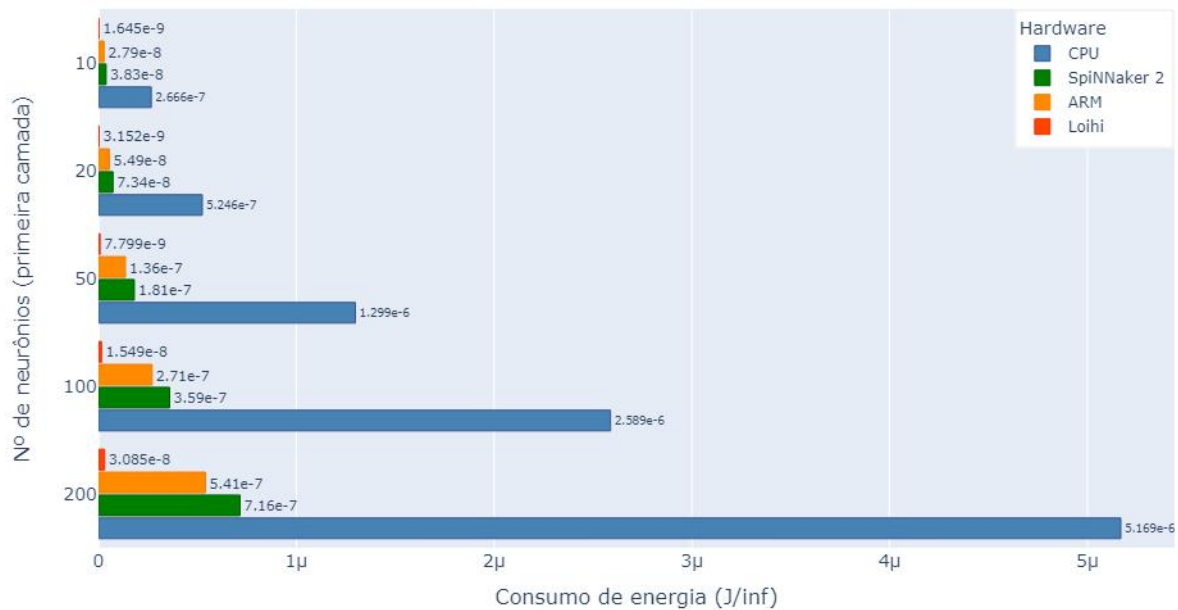


Figura 28 – Gráfico do consumo total de energia estimado por classificação para os *hardwares* Loihi, SpiNNaker 2, CPU e ARM (SNN de classificação de escorregamento).

Portanto, nesta seção foram apresentados os resultados para o modelo de classificação de escorregamento. No capítulo 5 serão discutidos os dados apresentados.

## 5 DISCUSSÕES

### 5.1 CLASSIFICAÇÃO DE TOQUE

Ao analisar os resultados do treinamento da SNN para classificação de toque apresentados na seção anterior, é possível notar a grande variedade de treinamentos realizados em busca dos melhores hiperparâmetros para a arquitetura proposta. Foram combinados de maneira sistemática 4 hiperparâmetros diferentes, são eles: números de neurônios, otimizador, taxa de aprendizado e quantidade de épocas. O número de neurônios foi variado de 200 à 10 em um intervalo de 10 neurônios, totalizado 20 valores diferentes; os otimizadores utilizados foram o Adam e o RMSprop; a taxa de aprendizado foi variada em três valores: 0,01, 0,001 e 0,0001; e o número de épocas de treinamento foi variado em 20, 40 e 60. Desta forma, combinando todos esses hiperparâmetros em uma pesquisa em forma de grade, ou seja, todas as possibilidades, foram realizados um total de 360 treinamentos para cada *fold* da validação cruzada. Também, os treinamentos foram avaliados em termos de precisão, sensibilidade, f1-score e tempos de treinamento. O resultado final é apresentado na forma de uma média entre todos os cinco valores de cada métrica, adquiridos em cada rodada da validação cruzada. Portanto, através desse método de avaliação é possível validar os resultados do modelo em toda a extensão dos dados do experimento.

Na Figura 21, podemos observar o gráfico de coordenadas paralelas representando os resultados dos treinamentos para as diferentes combinações de hiperparâmetros que foram testadas. Assim, analisando o eixo f1-score podemos observar que uma grande quantidade (247) de treinamentos atingiu desempenho superior a 90%. Este resultado sugere robustez em relação a valores dos hiperparâmetros, uma vez que, as mais variadas configurações atingiram tal performance. Por outro lado, cerca de 87 treinamentos não convergiram para um bom desempenho, resultando em uma performance inferior a 10%. Analisando as causas para este resultado, nota-se que dentre os 87 treinamentos nenhum possui a taxa de aprendizado de 0,0001. Isto nos sugere, que uma taxa de aprendizado menor permite uma convergência para um melhor resultado para essa arquitetura de rede.

Analisando a matriz de gráficos de dispersão na Figura 22, observa-se que não há um hiperparâmetro específico que realmente determine a convergência do modelo para os melhores resultados, confirmando a visualização das coordenadas paralelas na Figura 21. Além disso, os resultados mostram que os tempos de treinamento dos modelos, em média, são inferiores a cinco minutos. Note ainda, que os treinamentos com 20 épocas foram todos realizados com tempo inferior a quatro minutos, para 40 épocas inferior a oito minutos, já para 60 épocas chegou a cerca de dez minutos. Desta forma, o tempo treinamento dos modelos é principalmente afetado pelo número de épocas e pela quantidade neurônios na primeira camada. Isto ocorre, visto que



quanto maior a quantidade de neurônios, maior a quantidade de conexões e disparos por camada, portanto mais complexa computacionalmente a rede fica, bem como, quanto mais épocas de treinamento, maior o número de vezes que há o ajuste de hiperparâmetros da rede, provocando aumento no tempo de treinamento.

Também ao observar a Tabela 4, vemos o desempenho dos modelos que obtiveram melhor performance para cada número de neurônios. Primeiramente, nas colunas que apresentam os resultados de precisão, sensibilidade e f1-score, temos em sua maioria, resultados expressivos para cada quantidade de neurônios. O treinamento com dez neurônios foi único que obteve uma performance inferior a 99% da métrica f1-score. Redes contendo entre 20 e 200 neurônios na primeira camada atingiram um desempenho superior a 0,999 para a classificação de eventos de toque. Para a mesma faixa (20 e 200 neurônios) foram obtidas sensibilidades entre 0,994 e 0,998. Os elevados valores de sensibilidade e precisão sugerem que a SNN proposta possui grande probabilidade de detectar eventos de toque e baixa probabilidade de falsos positivos. Também, é possível notar que, com apenas 20 neurônios, ou seja, dez vezes menor que a maior quantidade treinada, o modelo obtém um resultado com diferença de apenas 0,001 em precisão, 0,004 em sensibilidade e 0,003 em f1-score, em comparação com o melhores resultados obtidos. Assim sendo, esta pequena diferença faz com que não haja um grande impacto na performance do modelo. Portanto, por exemplo, caso esse modelo seja implementado em *hardware*, o consumo de energia poderia ser diminuído aproximadamente na mesma proporção, conforme analisado a seguir.

Ao analisar o gráfico da Figura 24 que apresenta o consumo de energia total em Joules por classificação em diferentes *hardwares* (Loihi, SpiNNaker 2, CPU e ARM), vemos o aumento linear do consumo energia conforme aumenta o número de neurônios. Em comparação entre os dois *hardwares* neuromórficos, o consumo do SpiNNaker é 24,4 vezes maior que o Loihi. Tal variação se deve ao fato do custo maior do SpiNNaker para cada operação de *synapse* e atualização dos neurônios, ou seja, o Loihi é menos custoso em termos de consumo de energia por classificação. Além disso, tendo em vista a alta performance analisada para uma quantidade de 20 neurônios em comparação a 200 neurônios, por exemplo, reflete em um decréscimo de 9,6 vezes em média no consumo de energia por classificação para o Loihi e SpiNNaker, o que se torna vantajoso quando tratamos de implementações em *hardware*.

Quando comparamos os resultados do consumo de energia, observa-se uma diferença de cerca de 9,55 vezes maior no consumo da CPU em comparação ao ARM. Tal resultado, pode ser caracterizado devido ao baixo consumo por operação do processador ARM ( $0,9 \times 10^{-9}$  J), visto que, o Cortex-A5 foi desenvolvido pensando em alto desempenho com baixo consumo de energia, atendendo a uma variedade de aplicativos integrados e IoT (*Internet of Things* ou Internet da Coisas) com recursos



avançados. Já a CPU (Intel i7-4960X), possui um processamento mais custoso por operação ( $8,6 \times 10^{-9}$  J), visto sua utilização em aplicações maiores e mais genéricas, como computadores.

Além disso, na Figura 24 é possível notar uma grande diferença no consumo de energia da CPU em comparação com os *hardwares* neuromórficos. A CPU possui em média um consumo de, respectivamente, 162 e 7,2 vezes maior que o Loihi e o SpiNNaker 2. Essa diferença pode representar uma redução de em média até 99,4% no consumo de energia total quando utilizado um *hardware* neuromórfico (Loihi) em comparação com uma CPU. Além disso, verificando o resultado do processador ARM Cortex-A5, notamos um consumo de energia de 28,2% menor que o processador neuromórfico SpiNNaker 2. Porém, quando o comparamos ao Loihi da Intel, o processador ARM possui um custo de energia que equivale a 16,6 vezes mais consumo. Portanto, apesar do custo por inferência maior do SpiNNaker 2, ainda é vantajosa utilização do Loihi em termos de consumo de energia por inferência. Além disso, cabe salientar que os cálculos de consumo de energia aqui analisados estão descritos no apêndice A.

Também, ao observar a Figura 23 que demonstra um exemplo de simulação em tempo real utilizando o Nengo, com os dados da primeira sessão experimental, verificamos a classificação dos eventos de toque durante a preensão no sensor FSR (Figura 23.c). Este resultado é consistente com a saída esperada, com detecção correta do toque durante todo o tempo de preensão, neste exemplo, de 4,67 segundos. Além disso, na Figura 23.b é possível verificar a atividade de pico de cada um dos 20 neurônios da primeira camada, utilizados nesta simulação.

Além disso, analisando do ponto de vista prático, a detecção de toque é um aspecto importante que permite discriminar quando um objeto está sendo agarrado ou não, bem como auxiliar na de preensão e manipulação destes objetos. Para tanto, a não detecção do toque em uma mão robóticas pode acarretar no deslizamento de um objeto ou até na transmissão de uma força excessiva ao objeto agarrado. Assim, examinando na perspectiva dos resultados adquiridos (Tabela 4), a detecção do toque ocorre em 99,9% dos casos, ou seja, a cada 100 pontos de dados do sensor apenas um não seria detectado corretamente, nos sugerindo um taxa de detecção de toque expressiva na maioria dos casos.

Assim sendo, diversas análises foram realizadas buscando compreender os resultados obtidos para a SNN de classificação de toque. Tal performance obtida indicam que a arquitetura proposta obteve os resultados esperados, sendo possível validar a utilização de redes neurais pulsadas para classificação de toque em mãos protéticas. Além disso, o potencial de implementação em *hardware* de baixo consumo é promissor.

## 5.2 CLASSIFICAÇÃO DE ESCORREGAMENTO

Ao analisar a SNN de classificação de escorregamento, é possível notar a grande variedade de teste realizados em busca dos melhores hiperparâmetros para a arquitetura proposta. Para isso, foram combinados de maneira sistemática 3 hiperparâmetros diferentes, são eles: números de neurônios, *intercepts* e *max rates*. O número de neurônios foi variado de 200 à 10 em um intervalo de 10 neurônios, totalizado 20 valores diferentes; os valores para *intercepts* utilizados foram 0,005, 0,010 e 0,015; já o foi variado em: 5, 10 e 15. Desta forma, combinando todos esses hiperparâmetros em uma pesquisa em forma de grade, ou seja, todas as possibilidades, foram realizados um total de 180 treinamentos nos dados de testes. Os treinamentos foram avaliados em termos de precisão, sensibilidade e f1-score. Também, visto o não treinamento de parâmetros conforme realizado para a SNN de detecção de toque, não há a necessidade da utilização de validação cruzada, onde os dados seriam divididos em treinamento, validação e teste. Desta forma, os testes realizados com parâmetros fixos permitem que apenas com a simulação é possível validar a arquitetura proposta em todo o escopo do experimento.

Na Figura 25, podemos verificar um gráfico de coordenadas paralelas demonstrando uma série de linhas conectas representando os valores de hiperparâmetros e resultados obtidos para cada treinamento. Assim, analisando o eixo f1-score, 102 simulações obtiveram uma performance acima de 90%. Tal resultado independe de algum hiperparâmetro específico, onde diferentes configurações atingiram boa performance. Além disso, 18 treinamentos não convergiram para um bom desempenho, resultando em uma performance inferior a 50%. Já para este resultado, é possível notar que o número de neurônios é um hiperparâmetro que influencia tal performance, sendo em todos os casos a quantidade de neurônio é inferior a 40.

Analisando a matriz de gráficos de dispersão na Figura 26, observa-se que o hiperparâmetro específico *max rates* com valor 5, predomina na convergência do modelo para um resultado ótimo (métrica f1-score), sendo que no gráfico de coordenadas paralelas na Figura 25 também é possível observar esta correlação. Também, resultados intermediários que variam de 60% à 90% na métrica f1-score, tendem a utilizar um *max rates* de valor 15. Além disso, todas as 180 simulações foram realizadas em cerca de 0,89 minutos.

Também, quando observamos o desempenho dos modelos que obtiveram melhor performance para cada número de neurônios na Tabela 5, temos resultados expressivos para cada quantidade de neurônios na primeira camada, a não ser para 10 neurônios, onde o modelo obteve performance de 0,438 na métrica f1-score, porém com precisão de 1. Em termos de precisão e sensibilidade, respectivamente, 14 e 16 configurações obtiveram performance de 100%, sendo que de 20 à 200 neurônios o máximo de variações foi de 3,3%. Tais resultados, indicando uma alta taxa de clas-

sificação correta de eventos de escorregamentos, bem como capacidade do método de detectar com sucesso resultados classificados como positivos. Portanto, o melhor desempenho com a menor quantidade de neurônios na primeira camada é 20 neurônios, atingindo 100% da métrica precisão, sensibilidade e f1-score. Isto nos fornece um resultado inicial promissor, indicando que o modelo SNN é válido em toda a faixa de operação do sensor. Assim sendo, o número de neurônios não determina grandes impactos na performance do modelo, porém quando tratamos de implementação em *hardware* pode ser um aspecto vantajoso em termos de consumo de energia, o que será analisado em seguida.

Portanto, o gráfico da Figura 28 apresenta o consumo de energia total em Joules por classificação em quatro diferentes dispositivos, são eles: CPU, ARM, Loihi e SpiNNaker 2. Observa-se o aumento linear do consumo energia conforme aumenta o número de neurônios. Em comparação entre os dois *hardwares* neuromórficos, o consumo do Loihi é 24,4 vezes menor que o SpiNNaker 2, ou seja, consideravelmente menos custoso em termos de consumo de energia por classificação. Também, quando comparamos com o consumo de energia das redes neurais para detecção de toque, vemos uma pequena diferença, onde a arquitetura para classificação de escorregamento é mais custosa que para classificação de toque. Tal alteração pode ser associada aos valores de frequência de processamento ( $dt$ ) e de disparo dos neurônios (*input rate*) nas camadas. Demonstrando em números, para 20 neurônios a diferença é de, respectivamente,  $0,68 \times 10^{-9}$  J/inf e  $1,30 \times 10^{-8}$  J/inf para o Loihi e SpiNNaker 2.

Quando comparamos os resultados do consumo de energia do CPU e o ARM, observa-se o mesmo resultado que a SNN de detecção de toque. Isto se deve ao fato de que a saída de um neurônio não pulsados não é afetada pela frequência dos pulsos na entrada (*input rates*) e pelo  $dt$ , diferentemente dos *hardwares* neuromórficos (ver apêndice A). Da mesma forma, quando comparamos a diferença de consumo de energia da CPU e ARM entre os dispositivos neuromórficos, a relações realizadas são basicamente as mesmas, tornando vantajosa utilização do Loihi em termos de consumo de energia.

Além disso, explorando os resultados da Figura 27 que demonstra um exemplo de simulação em tempo real com os dados da segunda sessão experimental utilizando Nengo, verificamos a classificação 30 escorregamentos (Figura 27.c). Este resultado confirma a saída esperada e a detecção correta dos deslizamentos na mão protética. Além disso, na Figura 27.b é possível verificar a atividade pulsátil de cada um dos 20 neurônios da primeira camada.

Também, analisando na perspectiva prática da detecção de escorregamento, na seção 2.4 foi apresentado três motivos pelo qual um objeto poderia escorregar: quando força de prensão dos dedos é insuficiente, quando há uma perturbação externa ao objeto e quando o coeficiente de atrito do objeto diminuiu. Nesse sentido, a

não detecção de um escorregamento poderia acarretar em um manuseio impreciso do objeto, permitindo o objeto deslizar mais que o adequado ou até ocorrer a queda do mesmo. Assim, analisando do ponto de vista da perturbação externa de um objeto, visto estar relacionado ao experimento realizado na seção 3.1.1.2 e utilizado como dado de entrada para a rede neural de detecção de escorregamento, é possível observar através dos resultados obtidos que a detecção do escorregamento ocorreu em todos os casos. Fornecendo assim, uma boa perspectiva na detecção do escorregamento, o que permitiria um controle adequado no manuseio de um objeto quando há um perturbação externa.

Por fim, para validar a arquitetura proposta e a utilização de redes neurais pulsadas para classificação de escorregamento em mãos protéticas, a próxima seção irá apresentar uma comparação com outras abordagens encontradas na literatura.

### 5.3 COMPARAÇÕES COM OUTRAS ABORDAGENS

Fizemos uma comparação entre os resultados obtidos neste trabalho com o método de detecção de deslizamento apresentado em trabalho anterior (GENTILE et al., 2020). A proposta de Gentile et al. (2020), apresenta e valida experimentalmente um sistema de detecção de toque e deslizamento para mãos protéticas. Para detectar eventos de deslizamento, o método sugerido usa apenas o componente de força normal (ou o sinal bruto de tensão correspondente) e a derivada do mesmo. A avaliação de desempenho do método proposto para detecção de escorregamento obtém bons resultados, área sob a curva ROC (*Area Under the ROC Curve (AUC)*) igual a 0,999, com sensibilidade de 1,0 e especificidade de 0,992. Também, visto a utilização dos mesmos experimentos (seção 3.1.1) como entrada para técnica apresentada pelo autor e o trabalho desta dissertação, é possível uma comparação entre os resultados. Como pode ser observado na Tabela 6, nossa SNN atinge um resultado, apesar de mínimo, superior ao de Gentile et al. (2020) quando comparado com a métrica AUC. Isto nos sugere que não houve grandes melhorias na detecção do escorregamento, uma vez que a performance anterior já era significativa. Porém, esta dissertação acrescenta ao método a vantagem de uma resposta neuromórfica, ou seja, inteligível pelo cérebro. Este avanço permite a implementação de aplicações de *feedback* tátil aos usuários de próteses, bem como a implementação de sistemas embarcados de baixo custo energia (*hardwares* neuromórficos).

Além disso, é possível comparar os resultados obtidos neste trabalho com os resultados de outros autores (ver Tabela 6), explorados na seção 2.4. Begalinova et al. (2020) e Veiga et al. (2015) realizaram experimentos semelhantes ao apresentado neste trabalho, utilizando de técnicas de aprendizado de máquina para identificar escorregamento. Comparando a performance dos resultados para objetos com formato e características próxima ao apresentado neste trabalho, visto uma comparação mais

justa, Veiga et al. (2015) atingiu 97,95% na métrica f1-score utilizando a SVM, para o treinamento por objeto com o bloco de madeira (*box*). Já para Begalinova et al. (2020), o resultado atingido foi de 92% no método LTSM e 69% no método HMM (métrica f1-score), porém, utilizando uma garrafa (único objeto). Portanto, o sistema desenvolvido por Veiga et al. (2015) obteve resultados mais satisfatórios em relação aos métodos propostos por Begalinova et al. (2020), se sobressaindo em um desempenho de 5,95 pontos percentuais a mais.

Como pode ser observado na Tabela 6, nossa SNN supera o método LTSM em até 8 pontos percentuais na métrica f1-score e o HMM com 31 pontos percentuais. Esse baixo desempenho do HMM pode ser atribuído principalmente à classificação das classes de escorregamento e à dificuldade de aprender a distinguir entre as duas classes, dado um número desequilibrado de amostras de treinamento. Embora o LTSM tenha sido capaz de reconhecer eventos de escorregamento com alta precisão, ou seja, 0,93, em comparação a SNN atingiu precisão de 1,0 no conjunto de dados de teste. Já comparando ao resultado da SVM (de Veiga et al. (2015)) com o sistema proposto neste trabalho, foi obtido um desempenho de 2,05 pontos percentuais a mais com a SNN na métrica f1-score. Portanto, os resultados obtidos indicam que a arquitetura proposta obteve boa performance, sendo possível validar a utilização de redes neurais pulsadas para classificação de escorregamento em mãos protéticas.

Tabela 6 – Desempenho da SNN e abordagens da literatura.

Model	Precisão	Sensibilidade	F1-Score	AUC
GENTILE (GENTILE et al., 2020)	-	-	-	0,999
SVM (VEIGA et al., 2015)	-	-	0,9795	-
LTSM (BEGALINOVA et al., 2020)	0,93	0,91	0,92	-
HMM (BEGALINOVA et al., 2020)	0,71	0,76	0,69	-
Este trabalho	1,0	1,0	1,0	1,0

Embora para treinar uma rede neural seja importante ter muitos dados de treinamento, nesta abordagem estamos limitados pela quantidade de dados de treinamento, apenas um experimento com 30 deslizamentos em apenas um objeto e um sensor. Porém, uma contribuição significativa deste trabalho é fornecer uma maneira biologicamente plausível para validar a abordagem de detecção de toque e deslizamento proposta, treinando uma SNN. Além disso, em comparação com esses processos convencionais projetados em computador, nossos métodos podem ser usados para aplicações embarcadas com maior sucesso e baixo custo de implementação de *hardware*.

## 6 CONCLUSÕES E CONSIDERAÇÕES FINAIS

Neste trabalho, uma solução biologicamente plausível para detecção de toque e deslizamento são apresentados. A performance obtida indicam que a arquitetura proposta obteve resultados promissores, sendo possível validar a utilização de redes neurais pulsadas para classificação de toque e escorregamento em mãos protéticas. A técnica funciona transferindo uma compreensão biológica da percepção do toque em uma rede neural pulsada, que é então treinada para executar a classificação dos eventos. Este método tem a vantagem da utilização de SNN, o que permite a construção de *hardwares* neuromórficos de baixo consumo de energia comparado a processadores tradicionais, com potencial para uso em aplicações robóticas. Além disso, outra vantagem é fornecer uma resposta neuromórfica que seja inteligível pelo cérebro, o que permitiria estimulação sensorial do paciente fornecendo um *feedback* tátil. Ainda, apesar de uma resposta neuromórfica, pode ser empregue no controle das próteses, auxiliando no manuseio de objetos pelos pacientes.

Considerando o potencial que tecnologias neuromórficas apresentam para a construção de sistemas táteis, é importante que sejam desenvolvidos métodos de processamento que permitam expandir as capacidades perceptuais, aprimorando o controle das próteses. Nesse sentido, as abordagens neuromórficas se tornam interessantes para a área por alguns motivos, como a eficiência computacional que permite a uma redução do consumo de energia, um aspecto importante considerando a portabilidade do sistema como um todo. Também, a utilização das propriedades computacionais dos pulsos se tornam eficientes do ponto de vista de serem assíncronos e esparsos, formando padrões espaço-temporais que permitem ser decodificados e classificados para diferentes tipos de estímulos. Além disso, a codificação de informações táteis em pulsos promove um aspecto interessante para a restauração de sensações táteis naturais por meio da estimulação dos nervos periféricos preservados após uma amputação.

Assim, destaca-se a relevância deste tópico que constitui um caminho para o estado da arte no desenvolvimento de tecnologias táteis. O desenvolvimento de sensores táteis, métodos de processamento e controle são vitais na construção de próteses mais sofisticadas. Espera-se, que as tecnologia evolua a ponto de que sejam desenvolvidas próteses que possibilitam a integração completa ao corpo, permitindo novamente a percepção de estímulos táteis para pessoas amputadas. Desta forma, acredita-se que as abordagens neuromórficas apresentam perspectivas promissoras na evolução para este objetivo. Assim sendo, os resultados deste trabalho constituem um passo importante nesta direção, complementando estudos que envolvem o processamento de sinais táteis eletrônicos na detecção de toque e escorregamento em mãos protéticas utilizando SNN.

Como trabalhos futuros, consideraremos fazer novos experimentos para coletar mais dados incorporando diferentes objetos e uma maior quantidade de sensores. Podem ser desenvolvidas SNN que combinem a detecção de toque e escorregamento em apenas uma rede, bem como redes mais complexas que envolvam sensores em todos os dedos da mão. Além disso, pretende-se construir as redes neurais desenvolvidas em *hardwares* neuromórficos customizados em silício, que é o objetivo final desta pesquisa, bem como a simulação no *hardware* Intel Loihi.

## REFERÊNCIAS

ABADI, Mart et al. TensorFlow: A System for Large-Scale Machine Learning. In: 12TH USENIX symposium on operating systems design and implementation (OSDI 16). [S.l.: s.n.], 2016. p. 265–283.

ALBU-SCHÄFFER, Alin; HADDADIN, Sami; OTT, Ch; STEMMER, Andreas; WIMBÖCK, Thomas; HIRZINGER, Gerhard. The DLR lightweight robot: design and control concepts for robots in human environments. **Industrial Robot: an international journal**, Emerald Group Publishing Limited, 2007.

ARSALAN, Muhammad; SANTRA, Avik; ISSAKOV, Vadim. RadarSNN: A Resource Efficient Gesture Sensing System Based on mm-Wave Radar. **IEEE Transactions on Microwave Theory and Techniques**, IEEE, v. 70, n. 4, p. 2451–2461, 2022. ISSN 15579670.

BEDNAR, James A. Topographica: Building and analyzing map-level simulations from Python, C/C++, MATLAB, NEST, or NEURON components. **Frontiers in Neuroinformatics**, v. 3, MAR, p. 1–9, 2009. ISSN 16625196.

BEGALINOVA, Ainur; KING, Ross D.; LENNOX, Barry; BATISTA-NAVARRO, Riza. Self-supervised learning of object slippage: An LSTM model trained on low-cost tactile sensors. **Proceedings - 4th IEEE International Conference on Robotic Computing, IRC 2020**, p. 191–196, 2020.

BEKOLAY, Trevor; BERGSTRA, James; HUNSBERGER, Eric; DEWOLF, Travis; STEWART, Terrence C.; RASMUSSEN, Daniel; CHOO, Xuan; VOELKER, Aaron Russell; ELIASMITH, Chris. Nengo: A Python tool for building large-scale functional brain models. **Frontiers in Neuroinformatics**, v. 7, JAN, p. 1–13, 2014. ISSN 16625196.

BOCCOLINI, Fernando. **Reabilitação: amputados, amputações e próteses**. Edição: Robe Editora. 2. ed. São Paulo: [s.n.], 2000.

BOFF, Kenneth R; KAUFMAN, Lloyd; THOMAS, James P. **Handbook of perception and human performance**. [S.l.]: Wiley New York, 1986. v. 1.

BORETIUS, Tim; BADIA, Jordi; PASCUAL-FONT, Aran; SCHUETTLER, Martin; NAVARRO, Xavier; YOSHIDA, Ken; STIEGLITZ, Thomas. A transverse intrafascicular multichannel electrode (TIME) to interface with the peripheral nerve. **Biosensors and Bioelectronics**, Elsevier B.V., v. 26, n. 1, p. 62–69, 2010. ISSN 09565663.

CHAUDHURI, Avijit. **Fundamentals of sensory perception**. [S.l.]: Oxford University Press, 2011.



DAHIYA, Ravinder S; VALLE, Maurizio. **Robotic tactile sensing: technologies and system**. [S.l.]: Springer, 2013.

DAVIES, Mike et al. Loihi: A Neuromorphic Manycore Processor with On-Chip Learning. **IEEE Micro**, v. 38, n. 1, p. 82–99, 2018. ISSN 02721732.

DAVISON, Andrew P.; BRÜDERLE, Daniel; EPPLER, Jochen; KREMKOW, Jens; MULLER, Eilif; PECEVSKI, Dejan; PERRINET, Laurent; YGER, Pierre. PyNN: A common interface for neuronal network simulators. **Frontiers in Neuroinformatics**, v. 2, JAN, p. 1–10, 2009. ISSN 16625196.

DEGNAN, Brian; MARR, Bo; HASLER, Jennifer. Assessing Trends in Performance per Watt for Signal Processing Applications. **IEEE Transactions on Very Large Scale Integration (VLSI) Systems**, v. 24, n. 1, p. 58–66, 2016.

DEWOLF, Travis; JAWORSKI, Pawel; ELIASMITH, Chris. Nengo and low-power AI hardware for robust, embedded neurorobotics. **Frontiers in Neuroinformatics**, Frontiers Media SA, v. 14, p. 568359, 2020.

DIEHL, Peter U; NEIL, Daniel; BINAS, Jonathan; COOK, Matthew; LIU, Shih-Chii; PFEIFFER, Michael. Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing. In: 2015 International Joint Conference on Neural Networks (IJCNN). [S.l.: s.n.], 2015. p. 1–8.

DORLAN, W A N (Ed.). **Dicionario medico ilustrado**. 28. ed. SaoPaulo: Manole, 1999.

DUTTA, Sangya; KUMAR, Vinay; SHUKLA, Aditya; MOHAPATRA, Nihar R; GANGULY, Udayan. Leaky integrate and fire neuron by charge-discharge dynamics in floating-body MOSFET. **Scientific reports**, Nature Publishing Group, v. 7, n. 1, p. 1–7, 2017.

ELIASMITH, Chris; ANDERSON, Charles H. **Neural engineering: Computation, representation, and dynamics in neurobiological systems**. [S.l.]: MIT press, 2003.

EPPLER, Jochen Martin; HELIAS, Moritz; MULLER, Eilif; DIESMANN, Markus; GEWALTIG, Marc Oliver. PyNEST: A convenient interface to the NEST simulator. **Frontiers in Neuroinformatics**, v. 2, JAN, p. 1–12, 2009. ISSN 16625196.

GENTILE, Cosimo; CORDELLA, Francesca; RODRIGUES, Cesar Ramos; ZOLLO, Loredana. Touch-and-slippage detection algorithm for prosthetic hands. **Mechatronics**, Elsevier Ltd, v. 70, September 2019, p. 102402, 2020. ISSN 09574158.

GOODMAN, Dan F.M.; BRETTE, Romain. The brian simulator. **Frontiers in Neuroscience**, v. 3, SEP, p. 192–197, 2009. ISSN 16624548.

HAN, Jianhui; LI, Zhaolin; ZHENG, Weimin; ZHANG, Youhui. Hardware implementation of spiking neural networks on FPGA. **Tsinghua Science and Technology**, TUP, v. 25, n. 4, p. 479–486, 2020.

HASLER, Jennifer; MARR, Bo. Finding a roadmap to achieve large neuromorphic hardware systems. **Frontiers in neuroscience**, Frontiers Media SA, v. 7, p. 118, 2013.

HINES, Michael L.; DAVISON, Andrew P.; MULLER, Eilif. NEURON and Python. **Frontiers in Neuroinformatics**, v. 3, JAN, p. 1–12, 2009. ISSN 16625196.

HODGKIN, A L; HUXLEY, A F. A quantitative description of ion currents and its applications to conduction and excitation in nerve membranes. **Journal of Physiology**, v. 117, n. 1, p. 500–544, 1952.

HOPNER, Sebastian *et al.* Dynamic Power Management for Neuromorphic Many-Core Systems. **IEEE Transactions on Circuits and Systems I: Regular Papers**, v. 66, n. 8, p. 2973–2986, 2019. ISSN 15580806.

HOSSIN, Mohammad; SULAIMAN, Md Nasir. A review on evaluation metrics for data classification evaluations. **International journal of data mining & knowledge management process**, Academy & Industry Research Collaboration Center (AIRCC), v. 5, n. 2, p. 1, 2015.

JAMALI, Nawid; SAMMUT, Claude. Material classification by tactile sensing using surface textures. **Proceedings - IEEE International Conference on Robotics and Automation**, IEEE, p. 2336–2341, 2010. ISSN 10504729.

JOHANSSON, Roland S; VALLBO, Ake B. Tactile sensibility in the human hand: relative and absolute densities of four types of mechanoreceptive units in glabrous skin. **The Journal of physiology**, Wiley Online Library, v. 286, n. 1, p. 283–300, 1979.

JOHANSSON, Roland S.; FLANAGAN, J. Randall. Coding and use of tactile signals from the fingertips in object manipulation tasks. **Nature Reviews Neuroscience**, v. 10, n. 5, p. 345–359, 2009. ISSN 1471003X.

JOHNSON, K. O.; LAMB, G. D. Neural mechanisms of spatial tactile discrimination: neural patterns evoked by braille-like dot patterns in the monkey. **The Journal of Physiology**, v. 310, n. 1, p. 117–144, 1981. ISSN 14697793.

JONES, Lynette. **Haptics**. [S.l.]: MIT press, 2018.

KANDEL, Eric R; SCHWARTZ, James H; JESSELL, Thomas M; SIEGELBAUM, Steven; HUDSPETH, A James; MACK, Sarah *et al.* **Principles of neural science**. [S.l.]: McGraw-hill New York, 2000. v. 4.

KASABOV, Nikola; DHOBLE, Kshitij; NUNTALID, Nuttapod; INDIVERI, Giacomo. Dynamic evolving spiking neural networks for on-line spatio- and spectro-temporal pattern recognition. **Neural Networks**, Elsevier Ltd, v. 41, n. 1995, p. 188–201, 2013. ISSN 08936080.

KHALSA, Partap S.; FRIEDMAN, Robert M.; SRINIVASAN, Mandayam A.; LAMOTTE, Robert H. Encoding of shape and orientation of objects indented into the monkey fingerpad by populations of slowly and rapidly adapting mechanoreceptors. **Journal of Neurophysiology**, v. 79, n. 6, p. 3238–3251, 1998. ISSN 00223077.

KIM, Elmer K.; WELLNITZ, Scott A.; BOURDON, Sarah M.; LUMPKIN, Ellen A.; GERLING, Gregory J. Force sensor in simulated skin and neural model mimic tactile SAI afferent spiking response to ramp and hold stimuli. **Journal of NeuroEngineering and Rehabilitation**, v. 9, n. 1, p. 1–14, 2012. ISSN 17430003.

KIM, Sung Soo; MIHALAS, Stefan; RUSSELL, Alexander; DONG, Yi; BENSMAIA, Sliman J. Does Afferent Heterogeneity Matter in Conveying Tactile Feedback Through Peripheral Nerve Stimulation? **IEEE Transactions on Neural Systems and Rehabilitation Engineering**, v. 19, n. 5, p. 514–520, 2011.

KIM, Sung Soo; SRIPATI, Arun P; BENSMAIA, Sliman J. Predicting the timing of spikes evoked by tactile stimulation of the hand. **Journal of neurophysiology**, American Physiological Society Bethesda, MD, v. 104, n. 3, p. 1484–1496, 2010.

KIM, Sung Soo; SRIPATI, Arun P.; VOGELSTEIN, R. Jacob; ARMIGER, Robert S.; RUSSELL, Alexander F.; BENSMAIA, Sliman J. Conveying tactile feedback in sensorized hand neuroprostheses using a biofidelic model of mechanotransduction. **IEEE Transactions on Biomedical Circuits and Systems**, v. 3, n. 6, p. 398–404, 2009. ISSN 19324545.

KNIBESTÖL, M; VALLBO, Åke B. Single unit analysis of mechanoreceptor activity from the human glabrous skin. **Acta Physiologica Scandinavica**, Wiley Online Library, v. 80, n. 2, p. 178–195, 1970.

LEE, Wang Wei et al. A neuro-inspired artificial peripheral nervous system for scalable electronic skins. **Science Robotics**, v. 4, n. 32, 2019. ISSN 24709476.

LIU, Shih Chii; DELBRUCK, Tobi. Neuromorphic sensory systems. **Current Opinion in Neurobiology**, Elsevier Ltd, v. 20, n. 3, p. 288–295, 2010. ISSN 09594388.

LUCAROTTI, Chiara; ODDO, Calogero Maria; VITIELLO, Nicola; CARROZZA, Maria Chiara. Synthetic and bio-artificial tactile sensing: A review. **Sensors (Switzerland)**, v. 13, n. 2, p. 1435–1466, 2013. ISSN 14248220.

LUIS, Francisco; MONCAYO, Gil. Biomimetic sensory feedback through peripheral nerve stimulation improves dexterous use of a bionic hand, 2019.

MCDONALD, Cody L.; WESTCOTT-MCCOY, Sarah; WEAVER, Marcia R.; HAAGSMA, Juanita; KARTIN, Deborah. Global prevalence of traumatic non-fatal limb amputation. **Prosthetics and Orthotics International**, 2020. ISSN 17461553.

MELCHIORRI, Claudio. Slip detection and control using tactile and force sensors. **IEEE/ASME Transactions on Mechatronics**, v. 5, n. 3, p. 235–243, 2000. ISSN 10834435.

NAJARIAN, Siamak; DARGAHI, Javad; MEHRIZI, Ali Abouei. **Artificial tactile sensing in biomedical engineering**. [S.l.]: McGraw-Hill Education, 2009.

NGUYEN, Duy-Anh; TRAN, Xuan-Tu; IACOPI, Francesca. A review of algorithms and hardware implementations for spiking neural networks. **Journal of Low Power Electronics and Applications**, MDPI, v. 11, n. 2, p. 23, 2021.

ODDO, Calogero Maria et al. Intra-neural stimulation elicits discrimination of textural features by artificial fingertip in intact and amputee humans. **eLife**, v. 5, MARCH2016, p. 1–27, 2016. ISSN 2050084X.

OSBORN, Luke E; DRAGOMIR, Andrei; BETTHAUSER, Joseph L; HUNT, Christopher L; NGUYEN, Harrison H; KALIKI, Rahul R; THAKOR, Nitish V. Prosthesis with neuromorphic multilayered e-dermis perceives touch and pain. **Science robotics**, American Association for the Advancement of Science, v. 3, n. 19, eaat3818, 2018.

PATRO, SGOPAL; SAHU, Kishore Kumar. Normalization: A preprocessing stage. **arXiv preprint arXiv:1503.06462**, 2015.

PAUGAM-MOISY, H elene; BOHTE, Sander M. Computing with spiking neuron networks. **Handbook of natural computing**, v. 1, p. 1–47, 2012.

PHILLIPS, J R; JOHANSSON, R S; JOHNSON, K O. Representation of braille characters in human nerve fibres. **Experimental Brain Research**, Springer, v. 81, n. 3, p. 589–592, 1990.

PRACH, Anna; CABIBIHAN, John-john; THAKOR, Nitish V; BERNSTEIN, Dennis S. Pareto-Front Analysis of a Monotonic PI Control Law for Slip Suppression in a Robotic Manipulator, p. 2728–2733, 2017.

PURVES D., et al. **Neuroscience**. 3rd. [S.l.]: Sinauer Associates Inc., 2004.

RASCHKA, Sebastian. Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning, 2018.

RASMUSSEN, Daniel. NengoDL: Combining Deep Learning and Neuromorphic Modelling Methods. **Neuroinformatics**, Neuroinformatics, v. 17, n. 4, p. 611–628, 2019. ISSN 15590089.

RASOULI, Mahdi; CHEN, Yi; BASU, Arindam; KUKREJA, Sunil L.; THAKOR, Nitish V. An extreme learning machine-based neuromorphic tactile sensing system for texture recognition. **IEEE Transactions on Biomedical Circuits and Systems**, IEEE, v. 12, n. 2, p. 313–325, 2018. ISSN 19324545.

RUSSELL, Stuart J. **Artificial intelligence a modern approach**. [S.l.]: Pearson Education, Inc., 2010.

SAAL, Hannes P.; BENSMAIA, Sliman J. Biomimetic approaches to bionic touch through a peripheral nerve interface. **Neuropsychologia**, Elsevier, v. 79, p. 344–353, 2015. ISSN 18733514.

SEVERINO, Antônio Joaquim. **Metodologia do trabalho científico**. [S.l.]: Cortez editora, 2017.

SICILIANO, Bruno; KHATIB, Oussama; KRÖGER, Torsten. **Springer handbook of robotics**. [S.l.]: Springer, 2008. v. 200.

SILVA, Edna Lúcia da. MENEZES, Estera Muszkat. **Metodologia da pesquisa e elaboração de dissertação**, v. 3, 2001.

SINGH, Bharat; KUMAR, Rajesh; PRATAP, Vinay. **Reinforcement learning in robotic applications : a comprehensive survey**. [S.l.]: Springer Netherlands, 2022. v. 55, p. 945–990. ISBN 1046202109997.

SKEDUNG, Lisa; ARVIDSSON, Martin; CHUNG, Jun Young; STAFFORD, Christopher M.; BERGLUND, Birgitta; RUTLAND, Mark W. Feeling small: Exploring the tactile perception limits. **Scientific Reports**, v. 3, 2013. ISSN 20452322.

SRINIVASAN, M. A.; WHITEHOUSE, J. M.; LAMOTTE, R. H. **Tactile detection of slip: Surface microgeometry and peripheral neural codes**. v. 63. [S.l.: s.n.], 1990. p. 1323–1332.

SURESH, Aneesha K.; SAAL, Hannes P.; BENSMAIA, Sliman J. Edge orientation signals in tactile afferents of macaques. **Journal of Neurophysiology**, v. 116, n. 6, p. 2647–2655, 2016. ISSN 15221598.

TIWANA, Mohsin I.; REDMOND, Stephen J.; LOVELL, Nigel H. A review of tactile sensing technologies with applications in biomedical engineering. **Sensors and Actuators, A: Physical**, Elsevier B.V., v. 179, p. 17–31, 2012. ISSN 09244247.

VALLBO, A. B.; JOHANSSON, R. S. Properties of cutaneous mechanoreceptors in the human hand related to touch sensation. **Human Neurobiology**, v. 3, n. 1, p. 3–14, 1984. ISSN 07219075.

VEIGA, Filipe; VAN HOOFF, Herke; PETERS, Jan; HERMANS, Tucker. Stabilizing novel objects by learning to predict tactile slip. **IEEE International Conference on Intelligent Robots and Systems**, 2015-Decem, p. 5065–5072, 2015. ISSN 21530866.

VIRTANEN, Pauli *et al.* SciPy 1.0: fundamental algorithms for scientific computing in Python. **Nature methods**, Nature Publishing Group, v. 17, n. 3, p. 261–272, 2020.

VOELKER, Aaron Russell. Dynamical Systems in Spiking Neuromorphic Hardware, p. 235, 2019.

WEI, Lee Wang. **A neuromorphic approach to tactile perception**. 2016. Tese (Doutorado) – National University of Singapore (Singapore).

WEINSTEIN, Sidney. Intensive and extensive aspects of tactile sensitivity as a function of body part, sex and laterality. **The skin senses**, Springfield, Ill: Thomas, 1968.

YANG, Zhitao; HUANG, Yucong; ZHU, Jianghan; YE, Terry Tao. Analog circuit implementation of LIF and STDP models for spiking neural networks. In: PROCEEDINGS of the 2020 on Great Lakes Symposium on VLSI. [S.l.: s.n.], 2020. p. 469–474.

YI, Zhengkun; ZHANG, Yilei; PETERS, Jan. Biomimetic tactile sensors and signal processing with spike trains: A review. **Sensors and Actuators, A: Physical**, Elsevier B.V., v. 269, p. 41–52, 2018. ISSN 09244247.

YOUSEF, Hanna; BOUKALLEL, Mehdi; ALTHOEFER, Kaspar. Tactile sensing for dexterous in-hand manipulation in robotics - A review. **Sensors and Actuators, A: Physical**, Elsevier B.V., v. 167, n. 2, p. 171–187, 2011. ISSN 09244247.

ZHENGKUN, Y I. Biomimetic Tactile Sensor and Spike Train Processing for Surface Roughness Discrimination and Active Exploration, 2017.

ZOU, Fangyu; SHEN, Li; JIE, Zequn; ZHANG, Weizhong; LIU, Wei. A Sufficient Condition for Convergences of Adam and RMSProp. In: PROCEEDINGS of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). [S.l.: s.n.], 2019.

## APÊNDICE A – CÁLCULOS DE CONSUMO DE ENERGIA

Uma das principais motivações para usar SNNs é o potencial de economia significativa de energia em relação às técnicas padrão. Portanto, é útil poder estimar quanta energia seria usada por um modelo em diferentes dispositivos, para que possamos ter uma ideia de como diferentes parâmetros de modelo/dispositivo afetam o uso de energia antes de realizar uma implantação completa. Para isso, foi utilizado como base os cálculos de consumo de energia realizados pelo KerasSpiking<sup>1</sup>, uma ferramenta do Nengo que permite o treinamento e simulação de redes neurais pulsadas utilizando o Keras, bem como, providencia o cálculo de consumo estimado de energia em diferentes *hardwares*.

É importante ter em mente que o uso real de energia dependerá muito dos detalhes específicos da implementação subjacente de *software* e *hardware*. Os números fornecidos devem ser considerados apenas como estimativas aproximadas e dependem de várias suposições:

- Especificações do dispositivo: Para estimar a energia usada por um modelo em um determinado dispositivo, precisamos saber quanta energia é usada por operação sináptica (*energy per synop*) e pela atualização de neurônios (*energy per neuron*). Contamos com dados publicados para esses números ((DEGNAN; MARR; HASLER, 2016), (DAVIES *et al.*, 2018) e (HOPPNER *et al.*, 2019)). Os números de energia na prática podem diferir significativamente dos resultados publicados.
- Sobrecarga: Não é considerado nenhuma sobrecarga nas estimativas de energia (por exemplo, o custo de transferência de dados dentro e fora de um dispositivo). Estimamos apenas o uso de energia de cálculos de modelo interno (operações sinápticas e atualizações de neurônios). Na prática, a sobrecarga pode contribuir significativamente para o uso de energia de um modelo.
- Implementação de pulsos: Assumimos que o modelo que está sendo estimado foi totalmente convertido em uma implementação de pulsos ao estimar o uso de energia em um dispositivo de pulsos.

Os cálculos do consumo de energia consideram a energia consumida em cada camada pelas conexões (*synop energy*), ou seja, o número de conexões de todos os elementos de entrada para todas as unidades de ativação. A Equação (7) representa o cálculo de *synop energy*.  $E_{synop}$  é a energia por operação sináptica.  $N_{conn}$  é o número de conexões.  $Rate$  é a média de frequência de disparos dos neurônios.  $dt$  é o período de simulação e  $steps$  é a quantidade *time steps* utilizada por classificação.

<sup>1</sup> <https://www.nengo.ai/keras-spiking/>

$$synop_{energy} = E_{synop} * N_{conn} * rate * dt * steps \quad (7)$$

Também, para cada camada, é considerada a energia gasta pelos neurônios, ou seja, o número de atualizações de neurônios por *time step* realizados (*neuron energy*). A Equação (9) representa o cálculo de *neuron energy*.  $E_{neuron}$  é a energia por neurônio.  $N_{neuron}$  é o número de neurônios na camada.

$$neuron_{energy} = E_{neuron} * N_{neuron} * steps \quad (8)$$

Desta forma, o consumo total de energia por camada é a soma da energia consumida pelas conexões e atualizações dos neurônios:

$$energy\_layer_{total} = synop_{energy} + neuron_{energy} \quad (9)$$

Por fim, o consumo total da rede é a soma da energia consumida por camada. A equação (10) apresenta o cálculo do consumo total.

$$total_{energy} = energy\_layer1_{total} + energy\_layer2_{total} + \dots \quad (10)$$

Os cálculos de energia podem ser realizados para diferentes dispositivos, dentre eles:

- CPU: Por padrão se estima a energia da CPU Intel i7-4960X. É assumindo que as operações são realizadas no MAC. O valor utilizado de energia por *synop/neuron* é de  $8,6 \times 10^{-9}$  J (DEGNAN; MARR; HASLER, 2016).
- ARM: Por padrão se estima a energia ARM Cortex-A, na qual é assumindo que as operações são realizadas no MAC. O valor utilizado de energia por *synop/neuron* é de  $0,9 \times 10^{-9}$  J (DEGNAN; MARR; HASLER, 2016).
- Loihi: Estima a energia no *hardware* neuromórfico Intel Loihi. O valor utilizado de energia por *synop* é de  $1 \times 10^{-12}$ , bem como, a energia por *neuron* é de  $81 \times 10^{-12}$  (DAVIES et al., 2018).
- SpiNNaker and SpiNNaker2: Estima a energia no *hardware* neuromórfico SpiNNaker ou SpiNNaker2. O valor utilizado de energia por *synop* é de  $450 \times 10^{-12}$ , bem como, a energia por *neuron* é de  $2,19 \times 10^{-9}$  (HOPPNER et al., 2019), para o SpiNNaker2 (utilizado nos cálculos).

Além disso, conhecendo os valores de energia por operação em outros processadores que não sejam os citados, os cálculos podem ser realizados de igual maneira. Para mais detalhes, pode ser acessada a documentação do Nengo<sup>2</sup>.

<sup>2</sup> <https://www.nengo.ai>



## APÊNDICE B – PUBLICAÇÕES

### B.1 ARTIGOS EM CONGRESSOS

- **J. Follmann**, C. Gentile, F. Cordella, L. Zollo, **C. R. Rodrigues**.: “Slippage Classification in Prosthetic Hands with a Spiking Neural Network”, IX Latin American Congress on Biomedical Engineering and XXVIII Brazilian Congress on Biomedical Engineering, 2022.