UNIVERSIDADE FEDERAL DO CEARÁ

***CAMPUS* SOBRAL**

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA E COMPUTAÇÃO

MESTRADO ACADÊMICO EM ENGENHARIA ELÉTRICA E COMPUTAÇÃO

FLÁVIO VASCONCELOS DOS SANTOS

FUNÇÕES KERNEL TENSORIAIS BASEADAS EM TENSORES-NÚCLEO APLICADAS À CLASSIFICAÇÃO DE MOVIMENTO DE MÃOS

SOBRAL

2023

FLÁVIO VASCONCELOS DOS SANTOS

FUNÇÕES KERNEL TENSORIAIS BASEADAS EM TENSORES-NÚCLEO APLICADAS À CLASSIFICAÇÃO DE MOVIMENTO DE MÃOS

Dissertação apresentada ao Curso de Mestrado Acadêmico em Engenharia Elétrica e Computação do Programa de Pós-Graduação em Engenharia Elétrica e Computação do *Campus* Sobral da Universidade Federal do Ceará, como requisito parcial à obtenção do título de mestre em Engenharia Elétrica e Computação. Área de Concentração: Engenharia Elétrica e Computação

Orientador: Prof. Dr. Carlos Alexandre Rolim Fernandes

SOBRAL

2023

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Sistema de Bibliotecas
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

D762f dos Santos, Flávio.
Funções kernel tensoriais baseadas em tensores-núcleo aplicadas à classificação de movimento de mãos / Flávio dos Santos. – 2023.
70 f. : il. color.

Dissertação (mestrado) – Universidade Federal do Ceará, Campus de Sobral, Programa de Pós-Graduação em Engenharia Elétrica e de Computação, Sobral, 2023.
Orientação: Prof. Dr. Carlos Alexandre Rolim Fernandes.

1. SVM. 2. Função kernel. 3. HOSVD. 4. Tensor. 5. Aprendizado tensorial. I. Título.
CDD 621.3

FLÁVIO VASCONCELOS DOS SANTOS

FUNÇÕES KERNEL TENSORIAIS BASEADAS EM TENSORES-NÚCLEO APLICADAS À CLASSIFICAÇÃO DE MOVIMENTO DE MÃOS

Dissertação apresentada ao Curso de Mestrado Acadêmico em Engenharia Elétrica e Computação do Programa de Pós-Graduação em Engenharia Elétrica e Computação do *Campus* Sobral da Universidade Federal do Ceará, como requisito parcial à obtenção do título de mestre em Engenharia Elétrica e Computação. Área de Concentração: Engenharia Elétrica e Computação

Aprovada em:

BANCA EXAMINADORA

Prof. Dr. Carlos Alexandre Rolim Fernandes (Orientador)
Universidade Federal do Ceará (UFC)

Prof. Dr. Iális Cavalcante de Paula Júnior
Universidade Federal do Ceará (UFC)

Prof. Dr. André Lima Ferrer de Almeida
Universidade Federal do Ceará (UFC)

Aos meus pais, que acreditaram em mim.

AGRADECIMENTOS

Gostaria de agradecer primeiramente ao meu orientador, Prof. Dr. Carlos Alexandre, por me apoiar, em vários aspectos, desde a época de minha graduação, sem seu apoio com certeza não estaria aqui.

Agradeço também aos meus pais, que sempre apoiaram minha educação apesar de toda a dificuldade.

À minha namorada e melhor amiga Maria Ingrid, que está ao meu lado em todos os momentos.

Ao meu amigo Aron Souza, que mesmo distante continua me apoiando.

Agradeço aos meus amigos que fiz durante minha jornada na Universidade Federal do Ceará, meus agradecimentos vão em especial a André Rodrigues, que presenciou minha jornada e esteve comigo desde meu primeiro dia no Campus.

Ao Prof. Me. David Coelho, que sempre me auxiliou quando precisei e também aos professores dos cursos de Engenharia Elétrica e Engenharia da Computação do Campus UFC-Sobral, que me ajudaram a chegar até aqui.

Aos meus amigos da Comensais, por essa amizade que já dura mais de 12 anos.

“Onde estiver, seja lá como for, tenha fé, porque até no lixão nasce flor.”

(Racionais MC’s)

RESUMO

Os métodos de *kernel* e a Máquina de Vetores de Suporte (SVM) tornaram-se muito populares em Aprendizado de Máquina (AM). No entanto, quando dados multidimensionais são usados, as funções *kernel* clássicas baseadas em vetores devem vetorizar os dados de entrada, o que quebra a estrutura tensorial original, levando à perda de desempenho. Para evitar esse problema, as funções *kernel* tensoriais podem ser usadas. No presente trabalho, são apresentadas três novas funções de *kernel* tensoriais. Os métodos propostos são baseados nos tensores-núcleo da Higher-Order Singular Value Decomposition (HOSVD) e da Tensor-Train Decomposition (TTD).. Dois dos métodos apresentados são funções *kernel* rápidas que ignoram as matrizes de fatores dessas decomposições tensoriais, aliviando a carga de complexidade de tempo. As técnicas apresentadas foram avaliadas na classificação de seis movimentos diferentes das mãos. Para isso, foi desenvolvido o protótipo de uma "luva inteligente" de baixo custo com acelerômetros e giroscópios acoplados, gerando amostras de entrada tensoriais com dimensões referentes aos sensores, canais e atributos. Os experimentos mostraram um bom desempenho das técnicas propostas quando comparadas com funções de *kernel* tensorial do estado da arte.

Palavras-chave: SVM, função kernel, HOSVD, tensor, aprendizado tensorial, tensor-núcleo.

ABSTRACT

Kernel methods and Support Vector Machine (SVM) became very popular in machine learning. However, when multidimensional data are used, the classical vector-based kernel functions must vectorize the inputs, which breaks down the original tensor structure, leading to performance loss. To avoid this problem, tensor kernel functions can be used. In the present work, three novel tensor kernel functions are presented. The proposed methods are based on the core tensors of the Higher-Order Singular Value Decomposition (HOSVD) and Tensor-Train Decomposition (TTD). Two of the presented methods are fast kernel functions that ignore the factor matrices of these tensor decompositions, alleviating the time complexity burden. The presented techniques were evaluated in the classification of five different hand movements. For this purpose, the prototype of a low-cost "smart glove" was developed with accelerometers and gyroscopes was developed, generating tensor input samples with modes related to sensors, channels and features. The experiments showed a good performance of the proposed techniques when compared with state-of-art tensor kernel functions.

Keywords: SVM, kernel function, HOSVD, tensor, tensor learning, core tensor.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AM	Aprendizado de Máquina
RP	Reconhecimento de Padrões
SVM	Máquina de Vetores de Suporte
STM	Support Tucker Machines
HOSVD	Higher-Order Singular Value Decomposition
TTD	Tensor-Train Decomposition
DuSK	Dual Structure-preserving Kernel
PARAFAC	PARAllel FACtors model
fMRI	Ressonância Magnética Funcional
KTPLS	kernel kensor Partial Least Squares
KTCCA	Kernel Tensor Canonical Correlation Analysis
ECoG	Eletrocorticografia
MMK	Multi-way Multi-level Kernel
TDAH	Transtorno do Deficit de Atenção com Hiperatividade
HIV	Acquired Immunodeficiency Virus
FAKSETT	Fast Kernel Subspace Estimation based on Tensor Train decomposition
EMG	Eletromiograma
IHM	Interface Homem-Máquina
MLP	Multilayer perceptron
KNN	K-Nearest Neighbors
RBF	Função Gaussiana de Base Radial
H-TCK	HOSVD Tensor-Core Kernel
F-TCK	Fast Tensor-Core Kernel
TF-TCK	TTD Fast Tensor-Core Kernel
MPCA	Análise Multilinear de Componentes Principais - Multilinear Principal Component Analysis
CPD	Canonical Polyadic Decomposition
CANDECOMP	Canonical Decomposition
ALS	Alternating Least Square
SVD	Singular Value Decomposition

TT-SVD	Tensor-Train Singular Value Decomposition
SNR	Signal-to-Noise
SLC	Clock Serial
SDA	Dados Seriais
USB	Barramento Serial Universal

LISTA DE SÍMBOLOS

$\mathbb{R}$	Conjunto dos números reais
$.\circ.$	Produto externo
$\langle .,. \rangle$	Produto interno
$.\mathscr{X} \times_n .$	Produto modo-n
$. \times_m^n .$	Contração entre dois tensores nas dimensões com índices n e m
$\|\|.\|\|_F$	Norma de Frobenius
$\text{vec}(.)$	Vetorização

SUMÁRIO

1 INTRODUÇÃO

Em um cenário onde a quantidade de dados gerados diariamente é colossal, a tarefa de extrair informações significativas e relevantes dessa grande quantidade de dados tornou-se uma tarefa cada vez mais desafiadora. É nesse contexto que o Aprendizado de Máquina (AM), mais especificamente o Reconhecimento de Padrões (RP), surge como uma ferramenta poderosa para desvendar padrões e ideias ocultas nos dados. Ao aproveitar algoritmos inteligentes e técnicas avançadas, o RP capacita as máquinas a aprenderem a partir dos dados, identificar tendências e tomar decisões autônomas. Com aplicações em áreas que vão desde a medicina até a segurança computacional, o RP está revolucionando a forma como compreendemos e interagimos com o mundo ao nosso redor. Ao abordar a identificação e interpretação de regularidades e estruturas nos dados, o RP permite a compreensão de fenômenos complexos e a criação de modelos preditivos eficazes (BISHOP, 2006).

Uma das principais ferramentas no âmbito do RP são os classificadores. Os classificadores são *softwares* que realizam a diferenciação automática de classes através de atributos. Um dos mais conhecidos classificadores é a Máquina de Vetores de Suporte (SVM), que consiste em um classificador supervisionado e binário sendo um dos mais utilizados atualmente devido à sua robustez e velocidade de processamento, podendo ser usado também em problemas de regressão. Dentre os usos da SVM destacam-se principalmente seu uso em problemas que exigem uma separação não linear (CORTES; VAPNIK, 1995), problemas de alta dimensionalidade (JOACHIMS, 1998) e onde há um desequilíbrio de quantidade de amostras entre as classes (AKBANI *et al.*, 2004).

O SVM pode ser expresso como um problema de otimização de duas formas diferentes: a primal e a dual. Ambas as formas consistem basicamente em gerar um hiperplano que que separa as classes, buscando maximizar a distância entre as amostras mais próximas de cada classe. A forma primal do SVM é mais utilizada em problemas linearmente separáveis com uma pequena quantidade de dados (BURGES, 1998). Seu objetivo é otimizar diretamente os pesos do hiperplano de separação e o viés (*bias*) enquanto mantém as restrições (CRISTIANINI *et al.*, 2000).

Por outro lado, a forma dual do SVM faz uso dos coeficientes de Lagrange (também chamados de multiplicadores de Lagrange), que são introduzidos para expressar as restrições do problema de otimização. Esses coeficientes são associados às amostras de treinamento e quantificam a importância relativa de cada amostra na determinação do hiperplano de separação.

Resolver o problema dual envolve otimizar esses coeficientes para maximizar a margem entre as classes, sujeito a certas condições (CRISTIANINI *et al.*, 2000). Isso permite a utilização da função *kernel*, que manipula os dados de maneira que problemas antes não-linearmente separáveis se tornem linearmente separáveis.

Uma função *kernel* é uma medida de similaridade entre dois vetores de entrada que executa implicitamente um transformação não linear nos dados de entrada. O uso de funções *kernel* no SVM aumentam significativamente a capacidade de classificação deste método. Devido a isto, o SVM é amplamente usado em conjunto com as funções *kernel*, em sua forma dual (CORTES; VAPNIK, 1995).

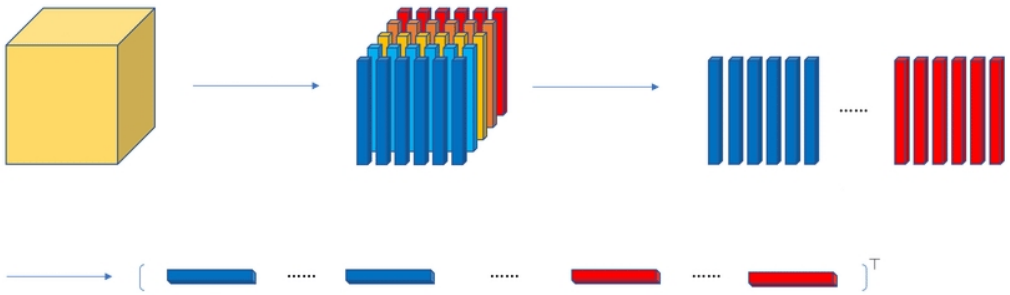
Outra vantagem do uso da forma dual do SVM é sua fácil adaptação ao uso de dados que possuem estruturas com dimensões superiores, os chamados dados multidimensionais ou tensoriais. Um tensor consiste em uma estrutura matemática que generaliza escalares, vetores e matrizes para dimensões superiores (PEIXOTO, 2022). Os chamados métodos de aprendizagem tensorial têm sido utilizados para evitar a vetorização dos dados de entrada e consequentemente o desmantelo da estrutura dos dados. Na aprendizagem tensorial, a estrutura multidimensional dos dados é mantida, o que ajuda a melhorar o desempenho em tarefas de aprendizagem (PANAGAKIS *et al.*, 2021).

Há várias vantagens para no uso de dados tensoriais para classificação. Naturalmente, alguns tipos de dados já possuem uma estrutura tensorial, como imagens coloridas (tensores ordem três) e vídeos (tensores ordem 4). Ao aplicar métodos de classificação vetorial em dados tensoriais é necessário vetorizar esses dados, o que pode levar à perda de informações, quebra da estrutura original dos dados e problemas de *overfitting* (LIU *et al.*, 2022).

De fato, classicamente, classificadores recebem dados de entrada vetoriais (com uma dimensão) ou matriciais (com duas dimensões). Por outro lado, quando os dados de entrada em um classificador são tensoriais, faz-se necessário uma quebra da estrutura dos dados para serem usado em como entrada de classificadores. Sendo assim, sua estrutura original é modificada. Um exemplo deste processo é mostrado na Figura 1, onde a estrutura de um tensor é quebrada e apesar de ser funcional, esse processo leva, em geral, a uma perda de performance (EVENBLY, 2022).

Outra desvantagem desse processo é o aumento excessivo da quantidade de atributos que pode causar a maldição da dimensionalidade, que ocorre quando a quantidade de atributos é maior que a quantidade de amostras, aumentando a complexidade e reduzindo a performance

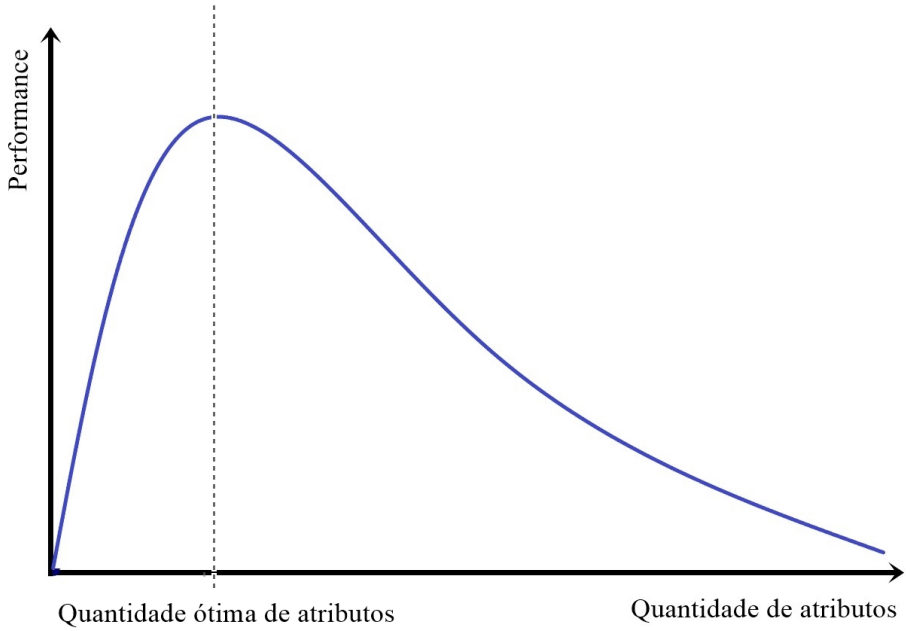
Figura 1 – Processo de vetorização de um tensor de ordem 3.



Fonte: Modificado de (CAI *et al.*, 2021).

de classificação (LATHAUWER *et al.*, 1994). A Figura 2 mostra, em um exemplo hipotético genérico, como se comporta o processo de classificação de acordo com a dimensionalidade dos atributos utilizados.

Figura 2 – Maldição da dimensionalidade em um exemplo fictício.



Fonte: Produzida pelo autor.

Os métodos de aprendizagem tensorial permitem que tais problemas possam ser contornados ao preservar a estrutura tensorial dos dados. De fato, a abordagem de classificação baseada em tensores pode reduzir significativamente o número de atributos necessários para treinar o modelo, fornecer uma interpretação física dos atributos e reter as informações espaciais e espectrais dos dados. Isso pode tornar a classificação mais eficiente e precisa (ERTAM; AYDıN,

2017; MAKANTASIS *et al.*, 2018).

Em particular, a SVM em sua forma dual pode ser adaptado de forma relativamente simples ao uso de dados de entrada tensoriais, ao contrário da forma primal da SVM, que necessita de significativas modificações para ser usado com dados tensoriais. Podemos citar como exemplo as Support Tucker Machines (STM), que consistem na generalização tensorial do SVM na forma primal (MA *et al.*, 2017).

No caso do SVM dual, uma função *kernel* tensorial realiza uma transformação diretamente nos dados de entrada e utiliza os dados transformados como entrada do SVM em sua forma dual, com isso, a estrutura tensorial é preservada e não há perda de informação estrutural. Sendo assim, o custo computacional para o cálculo dos *kernels* tensoriais tende a ser elevado necessitando de um maior esforço para desenvolver métodos mais rápidos e precisos.

Este trabalho consiste no desenvolvimento de três novas funções *kernel* tensoriais. Os métodos propostos são baseados nos tensores-núcleo da Higher-Order Singular Value Decomposition (HOSVD) e da Tensor-Train Decomposition (TTD). Dois dos métodos apresentados são funções *kernel* rápidas que usam apenas os tensores-núcleo das decomposições acima citadas, ignorando matrizes fatoriais dessas decomposições tensoriais, aliviando a carga de complexidade de tempo. As técnicas apresentadas foram avaliadas na classificação de seis movimentos diferentes das mãos. Para tanto, foi desenvolvida uma "luva inteligente" de baixo custo com acelerômetros e giroscópios acoplados, gerando amostras de entrada na forma tensoriais com modos relacionados a sensores, canais e atributos. Os experimentos mostraram um bom desempenho das técnicas propostas quando comparadas com funções *kernel* tensorial do estado da arte.

1.1 Estado da arte

1.1.1 Funções Kernel Tensoriais

O estudo de processamento de dados multilineares impactam diretamente a forma com que lidamos com grandes quantidade de dados. A classificação de dados tensoriais se aprimorou ao longo dos anos e alguns trabalhos propuseram funções *kernels* tensoriais que preservam a estrutura destes dados. Um destes trabalhos é descrito em (HE *et al.*, 2014), onde o autor desenvolve o Dual Structure-preserving Kernel (DuSK) que utiliza PARAllel FACtors model (PARAFAC), um tipo de decomposição tensorial. A função *kernel* DuSK nada mais é que uma extensão das funções *kernels* convencionais vetoriais para o espaço tensorial. Os autores

realizaram testes utilizando sinais tridimensionais de Ressonância Magnética Funcional (fMRI) de diferentes doenças, obtendo bons resultados de acurácia no processo de classificação com o SVM, entretanto, com um custo computacional relativamente alto.

Em (ZHAO *et al.*, 2013b), os autores propõe duas funções *kernel* tensoriais: kernel kensor Partial Least Squares (KTPLS) e Kernel Tensor Canonical Correlation Analysis (KTCCA), que podem ser aplicados tanto em problemas não-lineares de classificação quanto de regressão. Foram utilizados dados de movimentos provenientes de Eletrocorticografia (ECoG) com bons resultados, chegando até uma média de 98% de acurácia para o KTCCA. Estes métodos também foram utilizados em (ZHAO *et al.*, 2013a).

No artigo (SIGNORETTO *et al.*, 2011), o autor desenvolve uma função *kernel* tensorial não-paramétrica baseada em matrizes de valor singular gerada por HOSVD, um tipo de decomposição tensorial que preserva informações da estrutura tensorial dos dados. Essa função *kernel* é calculada utilizando o ângulo entre os subespaços gerados pelas matrizes. A técnica não utiliza o tensor-núcleo gerado pela HOSVD, que consiste em um tensor de menor complexidade gerado por essa decomposição tensorial. Por fim, o autor realizou testes de classificação utilizando imagens obtendo resultados satisfatórios.

Em (HE *et al.*, 2017), os autores propõem o método Multi-way Multi-level Kernel (MMK), que utiliza decomposição PARAFAC como método de decomposição dos dados preservando suas características tensoriais. Para validação do método, os autores utilizam neuroimagens reais para classificação de diferentes tipos de doenças, como por exemplo: Alzheimer, Transtorno do Deficit de Atenção com Hiperatividade (TDAH) e Acquired Immunodeficiency Virus (HIV).

Por fim, o método Fast Kernel Subspace Estimation based on Tensor Train decomposition (FAKSETT), desenvolvido em (KARMOUDA *et al.*, 2021), é uma função *kernel* tensorial baseado em TTD. O FAKSETT utiliza um método alternativo para cálculo dos subespaços do método de (SIGNORETTO *et al.*, 2011), melhorando o tempo de processamento. São utilizados os subespaços gerados pelo tensor núcleo a partir de TTD, um tipo de decomposição tensorial, com objetivo de melhorar a performance do método em termos de processamento.

1.1.2 Reconhecimento de gestos de mão usando sensores

O reconhecimento de gestos de mão se tornou uma área muito explorada nos últimos anos, tendo como uma das suas principais aplicações o controle de periféricos (YUAN *et al.*,

2022; NIELSEN *et al.*, 2020).

Visando a classificação de movimentos de mão, em (GIJSBERTS *et al.*, 2014) o autor combina acelerômetros e sensores de Eletromiograma (EMG) de superfície para criar um conjunto mais robusto de sensores para classificação de movimentos de mão utilizando a base de dados NinaPro. De modo semelhante, em (PIZZOLATO *et al.*, 2017), também são utilizados sensores EMG para controle de próteses juntamente com técnicas de AM. Em (MACHANGPA J. W.; CHINGTHAM, 2018), o autor controla uma cadeira de rodas a partir de sinais gerados por sensores MPU-6050 e outros sensores auxiliares com isso possibilitando uma melhor mobilidade de pessoas tetraplégicas. Finalmente, em (PRADO, 2020), o autor usa sinais EMGs para criação de uma Interface Homem-Máquina (IHM) que controla um editor de texto adaptado.

Como exemplo de trabalhos que usam sensores giroscópios, acelerômetros e magnetômetros na área de RP podemos citar (GUPTA, 2016), onde são utilizados os giroscópios e acelerômetros presentes em um *smartphone*, conseguindo uma acurácia de classificação bastante elevada.

Um componente que possui acelerômetro e giroscópio em conjunto é o MPU-6050. A principal vantagem desse sensor é seu baixo custo e a possibilidade de utilização de mais de um desses componentes, sendo muito usado na construção de protótipos, como em (YUDHANA, 2019), onde sensores do tipo *flex* são utilizados de maneira complementar juntamente com sensores MPU-6050. Também pode-se encontrar exemplos desses sensores sendo usados em jogos de realidade virtual (LEE S.; PARK, 2017). Outro exemplo de uso de diferentes sensores utilizados em conjuntos é dado em (ZHANG, 2011). Mais uma variação do uso de sensores se dá em (FAIDALLAH, 2014), onde são utilizados sensores EMG e sensores *flex*.

No ano de 2020 o autor publicou, como segundo autor, o artigo (FRANCISCO *et al.*, 2020), em que foram utilizados movimentos de mão utilizando sinais captados por sensores EMG para realizar a classificação com o vários classificadores, sendo eles: SVM, Multilayer perceptron (MLP), K-Nearest Neighbors (KNN). Foram obtidos bons resultados com o classificador SVM utilizando o *kernel* vetorial Função Gaussiana de Base Radial (RBF), chegando até uma taxa de 93,13% de acurácia.

1.2 Contribuições

Nessa dissertação, três novas funções *kernel* tensoriais são propostas. O primeiro, denotado por HOSVD Tensor-Core Kernel (H-TCK), é inspirado na função tensor *kernel* de

(SIGNORETTO *et al.*, 2011), que utiliza as matrizes de fatores do HOSVD. No entanto, o tensor-núcleo do HOSVD não é usada em (SIGNORETTO *et al.*, 2011). A ideia principal da função H-TCK proposta é introduzir um termo adicional na função *kernel* que leve em consideração o tensor-núcleo do HOSVD, juntamente com o *kernel* RBF. Este termo adicional do H-TCK introduz informações discriminantes relevantes que são ignoradas pela função *kernel* do (SIGNORETTO *et al.*, 2011).

A segunda função *kernel* tensorial proposta, denotada por Fast Tensor-Core Kernel (F-TCK), pode ser vista como uma simplificação do H-TCK que usa apenas o termo correspondente aos tensores-núcleo do HOSVD para calcular a função do *kernel*. Em contraste com o H-TCK e a função *kernel* do (SIGNORETTO *et al.*, 2011), o F-TCK não utiliza as matrizes de fator unitário do HOSVD. Ignorar essas matrizes de fatores simplifica significativamente o cálculo da função do *kernel*, aliviando a carga de complexidade de tempo.

A terceira função *kernel* proposta, denotada por TTD Fast Tensor-Core Kernel (TF-TCK), é aplicada ao caso de tensores de entrada de terceira ordem, pois nesse caso, a TTD gera apenas um tensor-núcleo. O TF-TCK é semelhante ao F-TCK, mas usa o tensor-núcleo da TTD em vez do HOSVD. De fato, à semelhança do F-TCK, por razões de complexidade, o TF-TCK não utiliza as matrizes de fatores do TTD, o que simplifica significativamente o cálculo da função *kernel*. Além disso, a utilização do TTD faz com que o TF-TCK tenha um tempo de processamento menor e uma melhor capacidade de discriminação dos dados, quando comparado ao F-TCK.

As funções propostas foram avaliadas por meio de experimentos que testaram sua eficiência na classificação dos movimentos de mãos. Para isso, foi desenvolvida uma "luva inteligente" de baixo custo equipada com dois módulos MPU-6050. A luva capta diferentes variações angulares e de aceleração medidas por acelerômetros e giroscópios, totalizando 12 canais de comunicação.

Uma base de dados foi construída a partir dos sinais captados pela luva inteligente. Foram utilizados cinco tipos diferentes de movimentos, que foram utilizados para treinar e testar uma SVM com diversas funções *kernel* tensoriais. A natureza multidimensional dos dados gerados pela luva inteligente é explorada para formar entradas tensoriais. Cada movimento realizado foi organizado em um tensor de ordem 3 com dimensões correspondentes a sensores, canais e características. Os experimentos mostraram que as funções *kernel* tensoriais propostas forneceram boa precisão e tempo de processamento, quando comparadas com funções *kernel*

tensoriais do estado da arte, especialmente o TF-TCK.

A “luva inteligente” é um sistema de captação de dados que possui a vantagem de ter um baixo custo de produção. De fato, com cerca de R$30,00 (cotado em janeiro de 2023) é possível construir todo o sistema de captação, enquanto um sistema diferente, como o que poderia ser visto em (RAWAT *et al.*, 2016), era encontrado por volta de $149 no ano de 2016.

Neste trabalho, foram realizados vários testes para comparação entre diferentes *kernels* tensoriais do estado da arte na linguagem Python 3 utilizando a base de dados gerada a partir da luva com sensores.

As principais contribuições desta dissertação são citas a seguir:

- Desenvolvimento de duas novas funções *kernel* tensoriais baseadas em HOSVD;
- O desenvolvimento de uma nova função *kernel* tensorial baseadas em TTD;
- Uso de multicanais e multisensores em classificação de movimento de mãos;
- A construção de um sistema de captação de movimentos de mão de baixo custo;
- Classificação da base de dados tensorial criado pelo autor utilizando *kernels* tensoriais no classificador SVM em sua forma dual.

1.3 Divisão do Trabalho

O restante deste trabalho é divido da seguinte maneira:

- No Capítulo 2, será descrita a base teórica necessária para dar prosseguimento aos capítulos seguintes;
- O Capítulo 3 apresenta o desenvolvimento teórico das novas funções *kernels* propostas;
- No Capítulo 4, será descrita a construção do sistema de captação dos dados e da base de dados utilizada nos testes de comparação dos *kernels* utilizados;
- No Capítulo 5, serão apresentados os resultados de classificação obtidos com as diferentes funções *kernels* testadas e a base de dados descrita no Capítulo 3;
- Por fim, no Capítulo 6 será apresentada a conclusão e as perspectivas futuras.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo é dedicado à fundamentação teórica relacionadas à álgebra multilinear, decomposições tensoriais, ao classificador SVM em sua forma dual e também às funções *kernel*. Será necessário o entendimento teórico destes quatro pilares para esta dissertação. Na Seção 2.1, será apresentado um apanhado de informações e definições básicas para o entendimento da álgebra multilinear. A Seção 2.2 apresentará as decomposições tensoriais utilizadas nessa dissertação. A Seção 2.3 apresentará alguns conceitos básicos acerca da classificação supervisionada. Na seção 2.4, será feita uma definição do classificador SVM, suas formas primal e dual, e será definido a função *kernel*. Por fim, na Seção 2.5, será apresentada a Análise Multilinear de Componentes Principais - Multilinear Principal Component Analysis (MPCA).

2.1 Álgebra multilinear

A álgebra multilinear ou álgebra tensorial se refere ao estudo de arranjos com um número de dimensões maior ou igual a três. Vetores e matrizes, apesar de terem dimensões menores, também podem ser considerados tensores de ordem um e dois, respectivamente. Nessa seção, serão abordados aspectos básicos de notação e operações relacionadas a tensores.

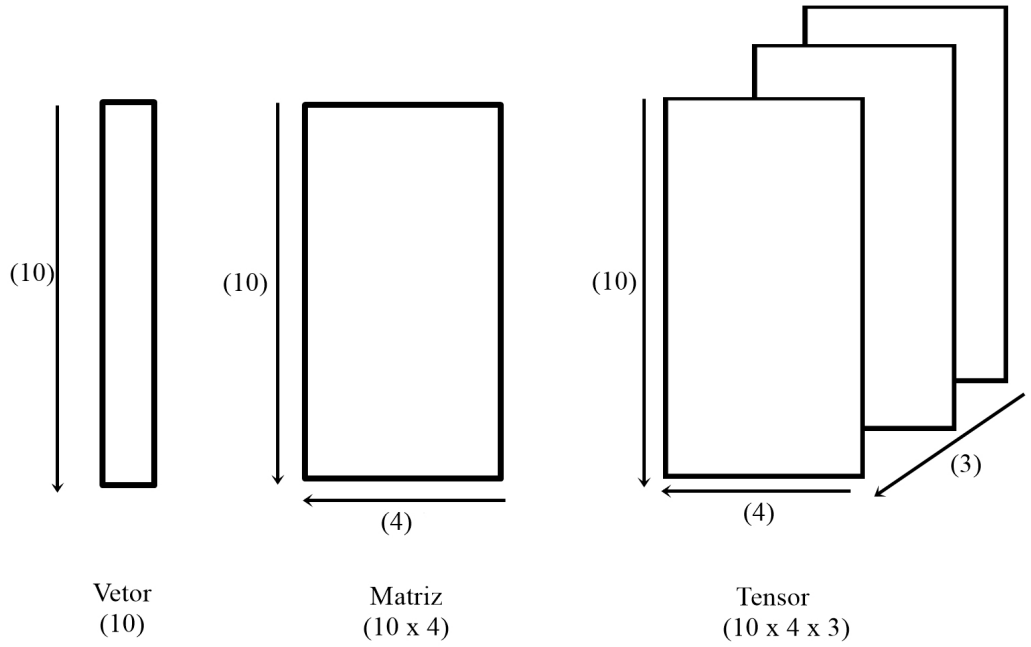
Tensores são generalizações de vetores e matrizes para ordem superiores. Podemos então afirmar que a ordem, ou quantidade de dimensões, de um tensor refere-se à quantidade de índices que possui. Um exemplo hipotético de um vetor de 10 elementos, uma matriz de dimensões 10×4 e um tensor de ordem $N = 3$ com dimensões $10 \times 4 \times 3$ são demonstrados na Figura 3.

As seguintes notações serão usadas no restante desta dissertação: escalares serão representados por letras minúsculas em itálico (x), vetores serão representados por letras minúsculas em negrito e itálico ($\boldsymbol{x}$), matrizes serão representadas por letras maiúsculas em negrito e itálico ($\boldsymbol{X}$) e tensores serão representados por letras caligráficas maiúsculas ($\mathscr{X}$).

Um elemento típico de um vetor $\boldsymbol{x} \in \mathbb{R}^{I_1}$, de uma matriz $\boldsymbol{X} \in \mathbb{R}^{I_1 \times I_2}$ e de um tensor de N-ésima ordem $\mathscr{X} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$ são descritos respectivamente por x_{i_1}, x_{i_1,i_2} e $x_{i_1,\ldots,i_N}$, tal que $i_n = 1,\ldots,I_N$ e $n = 1,\ldots,N$.

Definição 2.1 (Produto externo). O produto externo entre dois vetores $\boldsymbol{a} \in \mathbb{R}^I$ e $\boldsymbol{b} \in \mathbb{R}^J$ é dado

Figura 3 – Representação de um vetor, uma matriz e um tensor hipotéticos.



Fonte: Elaborado pelo autor.

por $\boldsymbol{a} \circ \boldsymbol{b} \in \mathbb{R}^{I \times J}$ e pode ser escrito como:

$$\boldsymbol{a} \circ \boldsymbol{b} = \boldsymbol{a}\boldsymbol{b}^T. \tag{2.1}$$

Definição 2.2 (Produto interno). O produto interno entre dois vetores $\boldsymbol{a} \in \mathbb{R}^I$ e $\boldsymbol{b} \in \mathbb{R}^I$ é um escalar definido por $\langle \boldsymbol{a}, \boldsymbol{b} \rangle \in \mathbb{R}^1$ e pode ser escrito como:

$$\langle \boldsymbol{a}, \boldsymbol{b} \rangle = \sum_{i=1}^{I} a_i b_i. \tag{2.2}$$

Definição 2.3 (Produto modo-n). O produto modo-n entre um tensor $\mathscr{X} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$ e uma matriz $\boldsymbol{M} \in \mathbb{R}^{P_n \times I_n}$, é dado por $\mathscr{C} = \mathscr{X} \times_n \boldsymbol{M} \in \mathbb{R}^{I_1 \times \ldots \times I_{n-1} \times P_n \times I_{n+1} \times \ldots \times I_N}$ e representa a rotação na n-ésima ordem de um tensor, definido por:

$$c_{i_1,\ldots,i_{n-1},p_n,i_{n+1},\ldots,i_N} = \sum_{i_n=1}^{I_n} m_{p_n,i_n} x_{i_1,\ldots,i_n,\ldots,i_N}. \tag{2.3}$$

Definição 2.4 (Contração). A operação de contração entre dois tensores $\mathscr{X} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$ e $\mathscr{Y} \in \mathbb{R}^{J_1 \times \ldots \times J_M}$ que compartilham uma dimensão em comum, por exemplo: $I_n = J_m = K$, é um tensor $\mathscr{Z} = \mathscr{X} \times_m^n \mathscr{Y} \in \mathbb{R}^{I_1 \times \ldots \times I_{n-1} \times J_1 \times \ldots \times J_{m-1} \times J_{m+1} \times \ldots J_M \times I_{n+1} \times \ldots \times I_N}$, onde seus elementos são definidos por:

$$z_{i_1,\ldots,i_{p-1},\ldots,j_1,\ldots,j_{q-1},\ldots,j_M,\ldots,i_N} = \sum_{k=1}^{K} x_{i_1,\ldots,i_{p-1},k,\ldots,i_N} y_{j_1,\ldots,j_{q-1},k,\ldots,j_M}, \tag{2.4}$$

em que $1 \leq p \leq N$ e $1 \leq q \leq M$.

Definição 2.5 (Norma de Frobenius). A norma de Frobenius (LOAN; GOLUB, 1996) de um tensor $\mathscr{X}$ de N-ésima ordem é dada por:

$$||\mathscr{X}||_F = \sqrt{\sum_{i_1=1}^{I_1} \cdots \sum_{i_N=1}^{I_N} |x_{i_1 \ldots i_N}|^2}. \tag{2.5}$$

Definição 2.6 (Vetorização). A operação de vetorização transforma um tensor $\mathscr{X} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$ em um vetor $\boldsymbol{x} \in \mathbb{R}^{I_1 I_2 \ldots I_{N-1} I_N}$ que pode ser escrito como:

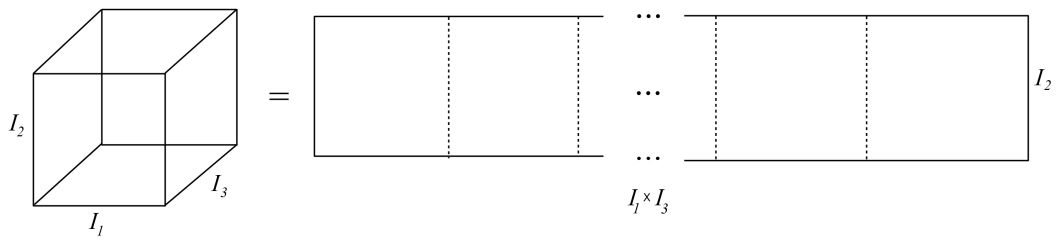
$$\boldsymbol{x} = [\text{vec}(\mathscr{X})], \tag{2.6}$$

e seus j-ésimo elemento é dados por:

$$j = i_1 + \sum_{n=2}^{N} (i_n - 1) \prod_{v=1}^{n-1} I_v. \tag{2.7}$$

Definição 2.7 (Desdobramento de tensor (KOLDA; BADER, 2009)). O desdobramento de um tensor $\mathscr{X} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$ é o processo de transformação de um tensor em uma matriz (*unfolding*). O tensor $\mathscr{X}$ é equivalente à matriz $\boldsymbol{X}_{[n]}$ em que n é a dimensão do tensor sobre qual o tensor será desdobrado, como mostra o exemplo da Figura 4, em que um tensor de ordem $N = 3$ é desdobrado na dimensão $n = I_2$ gerando a matriz $\boldsymbol{X}_{[2]} \in \mathbb{R}^{I_2 \times I_1 I_3}$.

Figura 4 – Desdobramento de um tensor de ordem 3 ao longo de I_2.



Fonte: Elaborado pelo autor.

De modo geral, um tensor de ordem N em que realiza-se a operação de desdobramento ao longo da dimensão $n = I_n$ gera uma matriz $\boldsymbol{X}_{[n]} \in \mathbb{R}^{I_n \times I_1 I_2 \ldots I_{n-1} I_{n+1} \ldots I_{N-1} I_N}$.

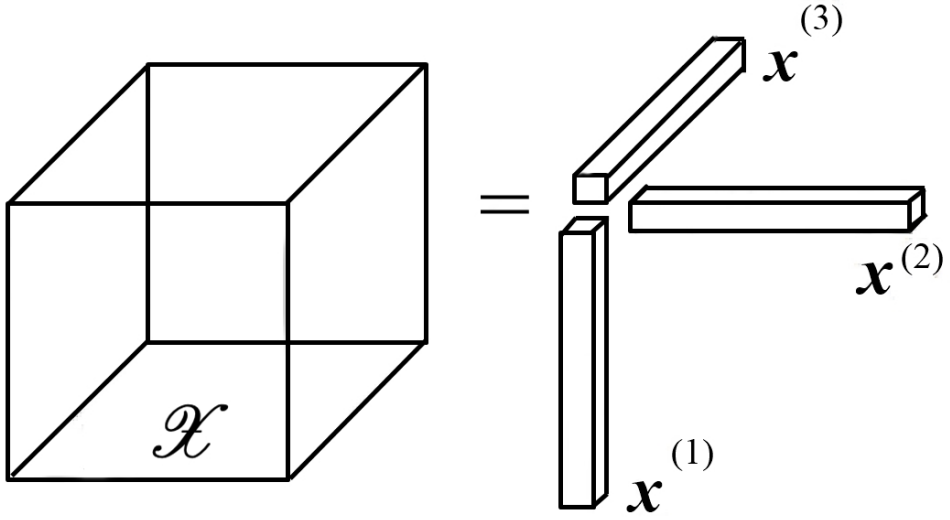
Definição 2.8 (Tensor de posto 1 (KOLDA; BADER, 2009)). Um tensor $\mathscr{X} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$ é considerado um tensor de posto 1 quando o mesmo pode ser representado por N vetores $\boldsymbol{x}^{(1)} \in$

$\mathbb{R}^{I_1}, \boldsymbol{x}^{(2)} \in \mathbb{R}^{I_2}, \ldots, \boldsymbol{x}^{(N)} \in \mathbb{R}^{I_N}$ da seguinte forma:

$$\mathscr{X} = \sum_{n=1}^{N} \boldsymbol{x}^{(1)} \circ \ldots \circ \boldsymbol{x}^{(N)} \in \mathbb{R}^{I_1 \times \ldots \times I_N}. \tag{2.8}$$

A operação descrita em (2.8) pode ser vista graficamente na Figura 5.

Figura 5 – Representação de um tensor de ordem 3 e posto 1.



Fonte: Elaborado pelo autor.

Definição 2.9 (Posto de um tensor). O posto R de um tensor $\mathscr{X} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$, é a quantidade mínima de tensores de posto 1 que podem expressar $\mathscr{X}$ em uma combinação linear.

Definição 2.10 (Normalização l^2 (GRADSHTEYN; RYZHIK, 2000)). A normalização l^2 é definida como a divisão de um vetor $\boldsymbol{x} \in R^I$ pela sua norma l^2, da seguinte forma:

$$\boldsymbol{x}_{norm} = \frac{\boldsymbol{x}}{\sqrt{\sum_{i=1}^{I} x_i^2}}. \tag{2.9}$$

2.2 Decomposições Tensoriais

Nesta Seção serão detalhadas as principais decomposições tensoriais usadas nesse trabalho.

2.2.1 PARAllel FACtors model (PARAFAC)

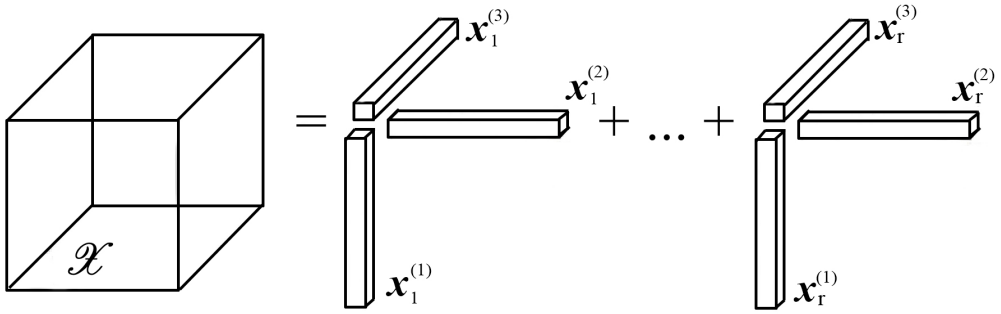
A decomposição PARAFAC, também conhecida como Canonical Polyadic Decomposition (CPD) ou Canonical Decomposition (CANDECOMP), é uma técnica de análise multilinear

que permite decompor um tensor $\mathscr{X} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$ em uma combinação linear de R tensores de posto 1, da seguinte maneira:

$$\mathscr{X} = \sum_{r=1}^{R} \boldsymbol{x}_r^{(1)} \circ \cdots \circ \boldsymbol{x}_r^{(N)} \in \mathbb{R}^{I_1 \times \ldots \times I_N}, \tag{2.10}$$

em que R é o posto do tensor $\mathscr{X}$ e $\boldsymbol{x}_r^{(n)} \in \mathbb{R}^{I_n}$, tal que $1 \leq n \leq N$ e $1 \leq r \leq R$, são os fatores da decomposição PARAFAC. Pode-se ver uma demonstração gráfica da CPD de um tensor de ordem 3 na Figura 6 onde é vista a soma de R produtos externos.

Figura 6 – Decomposição PARAFAC aplicada em um tensor de ordem 3 e posto R.



Fonte: Modificado de (KOLDA; BADER, 2009).

A PARAFAC foi originalmente proposta por (HITCHCOCK, 1927), posteriormente desenvolvida por (HARSHMAN *et al.*, 1970) e aprimorada em (KOLDA; BADER, 2009). A solução da decomposição PARAFAC é geralmente obtida através de métodos de otimização iterativos, como o método de Alternating Least Square (ALS), que foi proposto por (HARSHMAN *et al.*, 1970).

Além das vantagens de representar dados multidimensionais, outra propriedade importante da decomposição PARAFAC é sua unicidade essencial. Sob certas circunstâncias, a decomposição PARAFAC é única, exceto por permutação e escalonamento dos fatores (KRUSKAL, 1977; STEGEMAN; SIDIROPOULOS, 2007). Isso significa que, mesmo que os dados sejam representados por diferentes combinações de fatores, a decomposição PARAFAC pode ser usada para recuperar os mesmos fatores latentes.

2.2.2 Decomposição de Tucker

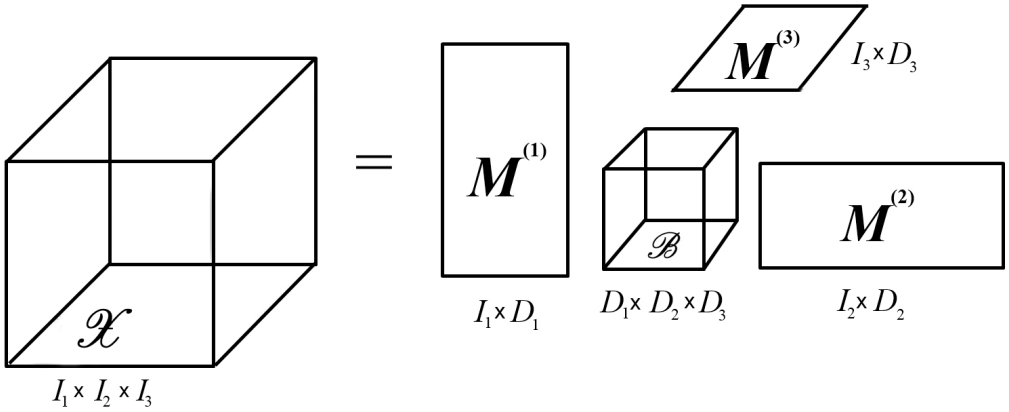
A Decomposição de Tucker é uma técnica matemática que permite decompor um tensor em componentes matriciais ao longo de cada dimensão e um tensor-núcleo. A decomposição é chamada de Tucker em homenagem ao matemático Lawrence Tucker, que a desenvolveu em 1966 (TUCKER, 1966).

O tensor-núcleo que compõe a decomposição Tucker preserva características importantes do tensor original. As componentes matriciais, ou fatores, são dispostas ao longo de cada dimensão do tensor original. Na decomposição Tucker, um tensor $\mathscr{X} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$ pode ser expresso da seguinte maneira:

$$\mathscr{X} = \mathscr{B} \times_1 \boldsymbol{M}^{(1)} \ldots \times_N \boldsymbol{M}^{(N)}, \tag{2.11}$$

onde $\mathscr{B}$ é o tensor núcleo gerado pela decomposição e $\boldsymbol{M}^{(1)} \in \mathbb{R}^{I_1 \times D_1}, \ldots, \boldsymbol{M}^{(N)} \in \mathbb{R}^{I_N \times D_N}$ são as matrizes de fatores ao longo da dimensão $1, \ldots, N$ respectivamente. A Figura 7 mostra um exemplo de Tucker aplicada a um tensor $\mathscr{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, é perceptível que o tensor núcleo $\mathscr{B} \in \mathbb{R}^{D_1 \times D_2 \times D_3}$ compartilha uma dimensão com cada matriz $\boldsymbol{M}^{(N)} \in \mathbb{R}^{I_N \times D_N}$.

Figura 7 – Decomposição Tucker aplicada em um tensor de ordem 3.



Fonte: Elaborado pelo autor

A decomposição de Tucker é amplamente utilizada em aplicações de processamento de imagens, processamento de sinais, mineração de dados, análise de dados financeiros e outras aplicações que requerem a representação eficiente de dados multidimensionais (CHEN *et al.*, 2019). Além disso, a decomposição de Tucker é uma técnica computacionalmente eficiente, pois permite a compressão de grandes quantidades de dados sem perda significativa de informação. Atualmente, existem várias variantes da decomposição de Tucker, incluindo a HOSVD, TTD, entre outras (KOLDA; BADER, 2009).

2.2.2.1 Decomposição de Valores Singulares de Ordem Superior (HOSVD)

Um caso especial da decomposição de Tucker é a HOSVD, que consiste na generalização do Singular Value Decomposition (SVD) para tensores (KOLDA; BADER, 2009). Pode-se dizer que a HOSVD é uma variante da decomposição Tucker que possui matrizes fatores

unitárias. Uma matriz unitária possui colunas ortogonais e com normas iguais a 1. A HOSVD de um tensor $\mathscr{X} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$ pode ser expressa por:

$$\mathscr{X} = \mathscr{G} \times_1 \boldsymbol{M}^{(1)} \ldots \times_N \boldsymbol{M}^{(N)}, \tag{2.12}$$

onde $\boldsymbol{M}^{(n)} \in \mathbb{R}^{I_n \times R_n}$, tal que $n = 1, ..., N$, são as matrizes unitárias contendo os vetores singulares e $\mathscr{G} \in \mathbb{R}^{R_1 \times \ldots \times R_N}$ é o tensor núcleo.

Apesar das semelhanças da HOSVD com a decomposição de Tucker, existem algumas diferenças, por exemplo: a HOSVD produz uma representação única do tensor. Já na decomposição de Tucker, não há garantia de que a decomposição seja única (KOLDA; BADER, 2009).

A HOSVD é uma das decomposições tensoriais mais importantes, é comumente utilizada em RP para extração de informações relevantes de dados multidimensionais (LATHAUWER *et al.*, 2000; LATHAUWER *et al.*, 1994; LATHAUWER; VANDEWALLE, 2004). Além disso, ao contrário da SVD, cujas matrizes de valores singulares são diagonais, o tensor central do HOSVD não é diagonal. Ao contrário, o tensor núcleo $\mathscr{G}$ é totalmente ortogonal e ordenado (LATHAUWER *et al.*, 2000).

2.2.3 Decomposição Tensor-Train (TTD)

Uma importante decomposição tensorial que está presente nesta dissertação é a TTD. Nessa decomposição, há uma representação de um tensor $\mathscr{X} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$ em $N-2$ tensores núcleo e duas matrizes núcleo.

A TTD pode ser estimada a partir de Tensor-Train Singular Value Decomposition (TT-SVD) que usa n SVD sequenciais (OSELEDETS, 2011). A TTD é definida como:

$$\mathscr{X} = \boldsymbol{G}_1 \times_1^2 \mathscr{G}_2 \times_1^3 \mathscr{G}_3 \times_1^4 \ldots \times_1^{N-1} \mathscr{G}_{N-1} \times_1^N \boldsymbol{G}^N, \tag{2.13}$$

onde $\mathscr{G}_n \in \mathbb{R}^{R_{n-1} \times I_n \times R_n}$, para $n = 2, ..., N-1$, $\boldsymbol{G}_1 \in \mathbb{R}^{I_1 \times R_1}$ e $\boldsymbol{G}_N \in \mathbb{R}^{R_{N-1} \times I_N}$, sendo $R_1, \ldots, R_N$ os postos da TTD, dados por:

$$R_n = \min\left(\prod_{p=1}^{n} R_p, \prod_{p=1+1}^{N} R_p\right), \tag{2.14}$$

para $1 \leq n \leq N$.

A TTD é uma aproximação que oferece uma vantagem sobre algumas outras decomposições: é de fácil implementação e não requer nenhuma recursão (OSELEDETS, 2011). A

TTD é mais genérica que a HOSVD e pode ser estimado a partir de Decomposições de Valores Singulares Tensor-Train (TT-SVD), usando n SVDs sequenciais, o que torna a complexidade de tempo do TTD menor que a do HOSVD (OSELEDETS, 2011).

2.3 Conceitos Básicos Sobre Classificação Supervisionada

Em AM, a classificação refere-se ao processo de treino, teste e validação imposta a um conjunto de dados, afim de definir um modelo que diferencie, automaticamente, diferentes classes. Uma classe refere-se a um conjunto de conceitos ou objetos que possam serem consideradas do mesmo tipo, amostras da mesma classe normalmente compartilham características específicas, como forma, cor, textura, entre outras (BACKES A. R.; JUNIOR, 2019).

A classificação é uma tarefa supervisionada de AM que trabalha com amostras de dados com classes bem conhecidas. Ao receber um conjunto de dados, um classificador se adéqua ao dados para que seja possível prever futuras amostras não-rotuladas. A classificação supervisionada pode auxiliar a compreensão humana a cerca de dados de alta complexidade (ALPAYDIN, 2010).

Para validar uma classificação supervisionada é necessário uma divisão dos dados entre treino e teste. Os dados de treino, como o nome afirma, são responsáveis pelo treinamento do classificador. Na fase de treinamento os rótulos da cada amostra são oferecidos ao classificador, diferente da fase de testes, onde os rótulos são omitidos. Na fase de teste é verificada a qualidade de treinamento do classificador. Em muitos casos essa aferição é realizada obtendo a acurácia de classificação, que nada mais é que a quantidade correta de amostras classificadas dividido pela quantidade total de amostras oferecidas no conjunto de dados de teste (KOHAVI, 2001).

Outras métricas também são comumente usadas, como o *recall* e o F1-*score*. O *recall* ou sensibilidade, mede a proporção de verdadeiros positivos em relação ao total de exemplos reais da classe positiva. Já o F1-*score* é uma métrica que combina precisão e o *recall* em uma única pontuação. É especialmente útil quando se deseja encontrar um equilíbrio entre essas duas métricas, uma vez que leva em consideração ambos os falsos positivos e os falsos negativos (TAHA; HANBURY, 2015). Se considerarmos p como sendo a precisão e r sendo o valor do *recall*, fórmula do F1-*score* será a seguinte:

$$\mathrm{F1}_{\mathrm{score}} = 2\frac{sr}{s+r}. \tag{2.15}$$

2.4 Máquina de Vetores de Suporte (SVM)

A SVM é um modelo de AM supervisionado usado para classificação e regressão. Desenvolvido nos anos 90 por (CORTES; VAPNIK, 1995), a ideia principal da SVM é encontrar um hiperplano que separe as amostras de diferentes classes no espaço de características gerado. O hiperplano é escolhido de modo a maximizar a distância entre as amostras mais próximas de cada classe, chamadas de vetores de suporte. A SVM é um classificador binário, ou seja consegue diferenciar apenas duas classes por vez.

A SVM possui duas formas: a primal e a dual. A forma primal da SVM é formulada como uma função objetivo quadrática, sujeita a restrições lineares. A solução ótima é encontrada por meio de métodos de otimização convexa, como o método de gradiente conjugado, dos Método dos Pontos Interiores, método do Gradiente Projetado (BISHOP, 2006; CORTES; VAPNIK, 1995).

2.4.1 Forma primal do SVM

Se considerarmos um problema de classificação real binária com as classes $y = 1$ e $y = -1$, a equação do hiperplano que separa as classes é dada por:

$$f(x) = \boldsymbol{w}^T x + b = 0, \tag{2.16}$$

onde $\boldsymbol{w} \in \mathbb{R}^P$ é o vetor de pesos e b é o termo de viés.

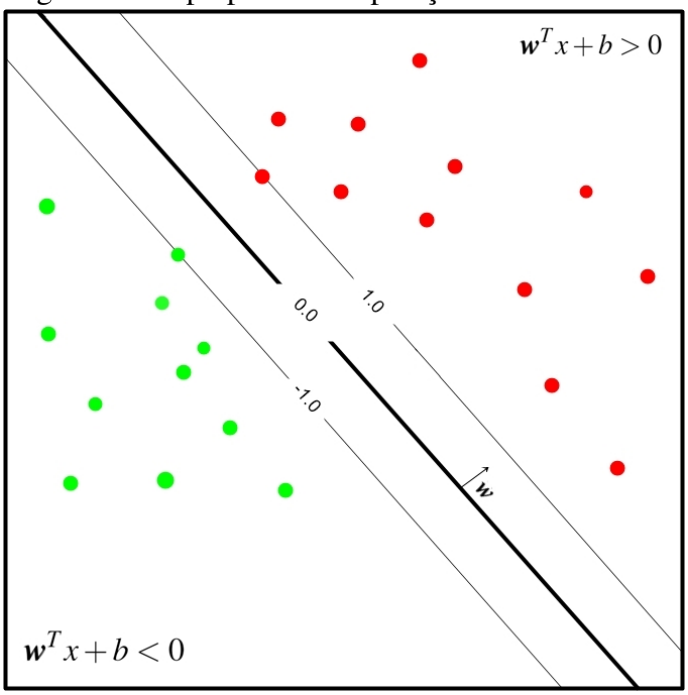
De acordo com a Figura 8, o vetor de pesos $\boldsymbol{w}$ é ortogonal ao plano de separação das classes. Como afirmado antes, o hiperplano é gerado buscando maximizar a distância entre as amostras de cada classe mais próximas, gerando assim dois vetores de suporte sendo eles: $\boldsymbol{w}^T x + b = -1$ para a classe $y = -1$, representados pelos círculos verdes e $\boldsymbol{w}^T x + b = 1$ para a classe $y = 1$, representados pelos círculos vermelhos.

2.4.2 Forma dual da SVM

O problema dual do SVM para problemas de classificação binária pode ser resolvido minimizando a seguinte função:

$$\min_{\alpha} \sum_{i=1}^{P} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{P} \alpha_i \alpha_j y_i y_j \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle, \tag{2.17}$$

Figura 8 – Hiperplano de separação do SVM.



Fonte: Elaborado pelo Autor.

sujeita, às seguintes restrições:

$$\begin{cases} 0 \leq \alpha_i \leq C, \quad \forall_i = 1,\ldots,P \\ \sum_{i=1}^{P} \alpha_i y_i = 0, \end{cases} \tag{2.18}$$

onde P é o número de amostras, C é a constante de relaxamento, α_i e α_j representam os coeficientes de Lagrange, $\langle .,. \rangle$ representa o produto interno entre os vetores de entrada $\boldsymbol{x}_i \in \mathbb{R}^P$ e $\boldsymbol{x}_j \in \mathbb{R}^P$ e y_i e y_j são os rótulos de amostra de cada entrada.

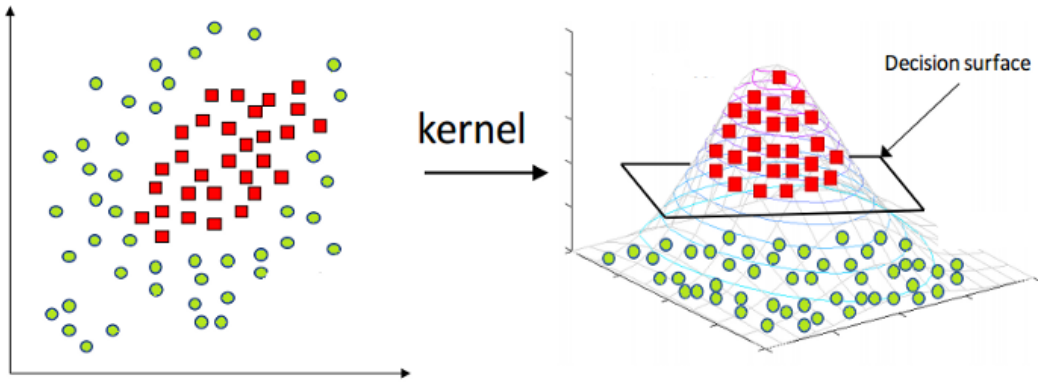
O problema dual do SVM permite a aplicação de funções *kernel* para transformação não-linear do espaço de características, o que permite a separação de classes que não são linearmente separáveis, como visto na Figura 9. A utilização de funções *kernel* equivale a mapear as amostras de entrada para um espaço de características de dimensão superior, onde é mais fácil encontrar um hiperplano de separação linear. A forma dual do SVM é particularmente útil nesse caso, porque a aplicação de funções *kernel* não é diretamente possível na forma primal (BISHOP, 2006).

Podemos então adequar (2.17) para o caso em que há o mapeamento dos dados de entrada através da função *kernel* da seguinte maneira:

$$\min_{\alpha} \sum_{i=1}^{P} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{P} \alpha_i \alpha_j y_i y_j k(\boldsymbol{x}_i, \boldsymbol{x}_j), \tag{2.19}$$

que está sujeita às restrições de (2.18) e sendo $k(.,.)$ a função *kernel* aplicada aos dados de entrada.

Figura 9 – Função *Kernel* aplicada.



Fonte: (MEDIUM, 2019)

Quando são utilizados dados tensoriais como entrada, o uso de funções *kernel* baseadas em vetores requer a vetorização dos dados de entrada, o que quebra a estrutura dos dados e, geralmente, leva à perda de desempenho podendo também causar a maldição da dimensionalidade (YANG *et al.*, 2020). Para evitar esse problema de vetorização, as funções *kernel* tensoriais podem ser usadas, permitindo o uso de entradas tensoriais sem o processo de vetorização, preservando a estrutura multidimensional dos dados (CORTES *et al.*, 2009).

A forma dual do classificador SVM para dados tensoriais pode ser representada por:

$$\max_{\alpha} \sum_{i=1}^{P} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{P} \alpha_i \alpha_j y_i y_j k(\mathscr{X}_i, \mathscr{X}_j), \tag{2.20}$$

sujeita às restrições:

$$\begin{cases} 0 \leq \alpha_i \leq C, \quad \forall_i = 1, \ldots, P \\ \sum_{i=1}^{P} \alpha_i y_i = 0. \end{cases} \tag{2.21}$$

onde $k(.,.)$ é a função *kernel* aplicada às amostras de entrada $\mathscr{X}_i$ e $\mathscr{X}_j$ (CRISTIANINI *et al.*, 2000).

Nota-se então que a única diferença entre a versão vetorial e a tensorial da forma dual do SVM é a função *kernel*, como visto em (2.19) e (2.20). No Capítulo 4 será feita uma revisão bibliográfica das principais funções *kernel* do estado da arte.

2.5 Análise Multilinear de Componentes Principais (MPCA)

Análise Multilinear de Componentes Principais (Multilinear Principal Component Analysis - MPCA) pode ser descrito como uma generalização da PCA para dimensões maiores que 3. É uma técnica utilizada para identificar os componentes principais de um conjunto de

dados. A MPCA permite a análise de conjuntos de dados multidimensionais, fazendo com que seja possível a eliminação de componentes correlacionados. Resumidamente, a MPCA elimina atributos correlacionados reduzindo assim a dimensionalidade original do tensor (LATHAUWER *et al.*, 2009).

A MPCA aplicada a um tensor $\mathscr{X} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$ gera um novo tensor $\mathscr{Y} \in \mathbb{R}^{D_1 \times \ldots \times D_N}$ onde $(D_N < I_N)$, em que $\mathscr{Y}$ detém a maior parte da variância de $\mathscr{X}$ (WANG *et al.*, 2017).

O principal objetivo do MPCA é encontrar as matrizes de transformação $\widetilde{\boldsymbol{U}}^{(n)} \in \mathbb{R}^{I_n \times P_N}$, $n = 1, 2, \ldots, N$, que projetam o tensor original no subespaço $\mathbb{R}^{D_1 \times \ldots \times D_N}$ $(D_N < I_N)$ (WANG *et al.*, 2017):

$$\mathscr{Y} = \mathscr{X} \times_1 \widetilde{\boldsymbol{U}}^{(1)T} \times_2 \widetilde{\boldsymbol{U}}^{(2)T} \ldots \times_N \widetilde{\boldsymbol{U}}^{(N)T}. \tag{2.22}$$

3 FUNÇÕES KERNEL PROPOSTAS

Nesta Seção, são apresentadas funções *kernel* tensoriais existentes no estado da arte, bem como as funções *kernel* tensoriais propostas nesta dissertação.

3.1 Funções *kernel* do estado da arte

Há diversas funções *kernel* tensoriais existentes na literatura (HE *et al.*, 2014; ZHAO *et al.*, 2013b; ZHAO *et al.*, 2013a; KARMOUDA *et al.*, 2021; SIGNORETTO *et al.*, 2011). A seguir, algumas dessas funções *kernel* serão apresentadas. A função DuSK (HE *et al.*, 2014) consiste na aplicação de uma técnica chamada mapeamento dual-tensorial. O DuSK usa os fatores do tensor de entrada obtidos através da decomposição PARAFAC juntamente com a função *kernel* RBF padrão, da seguinte maneira:

$$k(\mathscr{X}_i, \mathscr{X}_j) = \sum_{r_1=1}^{R_1} \sum_{r_2=1}^{R_2} \prod_{n=1}^{N} k(\boldsymbol{a}_{r_1}^{(n)}, \boldsymbol{a}_{r_2}^{(n)}), \tag{3.1}$$

onde $\boldsymbol{a}_{r_1}^{(n)}$ e $\boldsymbol{a}_{r_2}^{(n)}$ são os fatores da decomposição PARAFAC de $\mathscr{X}_i$ e $\mathscr{X}_j$, respectivamente, como em (2.10), e $k(\cdot,\cdot)$ é a função *kernel* RBF padrão. Esta decomposição se baseia nos fatores da decomposição PARAFAC, juntamente com o *kernel* RBF vetorial, para extrair a medida de similaridade entre os tensores de entrada. Mais especificamente, cada objeto tensorial é representado como uma soma de tensores de posto-1 no espaço original e mapeado no espaço de características do produto tensorial. Os resultados desse trabalho demonstraram que preservar a estrutura tensorial dos dados melhora significativamente a performance da função *kernel* quando usada em uma tarefa de classificação.

Em (SIGNORETTO *et al.*, 2011), uma função *kernel* tensorial é proposta, usando os fatores do HOSVD dos tensores de entrada. Considerando que as HOSVDs dos tensores de entrada são dadas por:

$$\mathscr{X}_i = \mathscr{W} \times_1 \boldsymbol{U}^{(1)} \times_2 \cdots \times_N \boldsymbol{U}^{(N)}, \tag{3.2}$$

$$\mathscr{X}_j = \mathscr{Y} \times_1 \boldsymbol{V}^{(1)} \times_2 \cdots \times_N \boldsymbol{V}^{(N)}, \tag{3.3}$$

a função *kernel* tensorial proposta em (SIGNORETTO *et al.*, 2011) é definida como:

$$k(\mathscr{X}_i, \mathscr{X}_j) = \prod_{n=1}^{N} k_n(\boldsymbol{U}^{(n)}, \boldsymbol{V}^{(n)}), \tag{3.4}$$

sendo:

$$k_n(\boldsymbol{U}^{(n)},\boldsymbol{V}^{(n)}) = \exp(-\gamma \sin^2(\theta_n)), \tag{3.5}$$

onde γ é um escalar e θ_n é o ângulo entre os subespaços gerados por $\boldsymbol{U}^{(n)}$ e $\boldsymbol{V}^{(n)}$, obtido de:

$$\sin^2(\theta_n) = 2||\boldsymbol{U}^{(n)}\boldsymbol{U}^{(n)^T} - \boldsymbol{V}^{(n)}\boldsymbol{V}^{(n)^T}||_F^2. \tag{3.6}$$

Esta técnica foi usada no *dataset* UCF11: um conjunto de dados que contém 1600 videoclipes pertencentes a 11 ações humanas, tais como: mergulhar, saltar trampolim, caminhar, atirar, etc. O autor de (KARMOUDA *et al.*, 2021) também utilizou os dados de Yale B que contém 2414 imagens faciais frontais de diferentes pessoas. Os resultados mostraram que a acurácia do *kernel* de (SIGNORETTO *et al.*, 2011) se assemelha ao FAKSETT, mas a velocidade de processamento chega a ser 7 vezes menor que a de (SIGNORETTO *et al.*, 2011).

A função *kernel* FAKSETT (KARMOUDA *et al.*, 2021) é similar ao método proposto em (SIGNORETTO *et al.*, 2011), entretanto, é usada a TTD ao invés de HOSVD para calcular o ângulo entre os subespaços. Em particular, o método FAKSETT usa as matrizes $\boldsymbol{G}^{(1)}$ e $\boldsymbol{G}^{(N)}$, e a matriz desdobrada modo-2 de $\mathscr{G}^{(n)}$, para $n = 2,...,N-1$, obtida da TTD em (2.13), para calcular o ângulo entre os subespaços.

Ao realizar testes de classificação com dados de vídeo e imagens, os autores mostram que o *kernel* FAKSETT atinge uma precisão de classificação muito semelhante ao método do estado da arte de (SIGNORETTO *et al.*, 2011), mas com um tempo de execução consideravelmente reduzido. Os testes indicaram tempos de processamentos do *kernel* até 3,7% menores que o de (SIGNORETTO *et al.*, 2011).

3.1.1 HOSVD Tensor-Core Kernel (H-TCK)

Nesta seção, três novas funções *kernel* tensoriais são propostas. A primeira, chamada de H-TCK é baseada na função *kernel* de (SIGNORETTO *et al.*, 2011), mas com um termo adicional que leva em consideração o tensor-núcleo da HOSVD. A segunda função *kernel* tensorial, denotada por F-TCK, é uma simplificação do H-TCK que usa apenas os tensores-núcleo da HOSVD para cálculo da função *kernel*. A terceira função *kernel*, chamada de TF-TCK, é similar ao F-TCK, mas usa os tensores-núcleo da TTD ao invés da HOSVD.

O H-TCK proposto é inspirado pelo *kernel* de (SIGNORETTO *et al.*, 2011), que usa o ângulo entre o subespaço gerado entre as matrizes de fatores do HOSVD. Entretanto, os

tensores-núcleo $\mathscr{W} \in \mathbb{R}^{I_1 \times \cdots \times R_N}$ e $\mathscr{Y} \in \mathbb{R}^{R_1 \times \ldots \cdots \times R_N}$ da HOSVD em (3.2) e (3.3) não são usados em (SIGNORETTO *et al.*, 2011). A ideia principal da função H-TCK é introduzir um termo adicional à função *kernel* para levar em consideração a o tensor-núcleo da HOSVD.

O H-TCK é definida por:

$$k(\mathscr{X}_i, \mathscr{X}_j) = \exp(-2\gamma||\mathscr{W} - \mathscr{Y}||_F^2) \prod_{n=1}^{N} k_n(\boldsymbol{U}^{(n)}, \boldsymbol{V}^{(n)}), \tag{3.7}$$

onde $k_n(\boldsymbol{U}^{(n)}, \boldsymbol{V}^{(n)})$ é definida em (3.5) e (3.6), $\mathscr{W} \in \mathbb{R}^{R_1 \times \cdots \times R_N}$ e $\mathscr{Y} \in \mathbb{R}^{R_1 \times \ldots \cdots \times R_N}$ são os tensores-núcleo da HOSVD de $\mathscr{X}_i$ e $\mathscr{X}_j$, respectivamente, e γ é um escalar.

Comparando (3.7) e (3.4), pode ser visto que a única diferença entre essas duas funções *kernel* tensoriais é o termo $\exp(-2\gamma||\mathscr{W} - \mathscr{Y}||_F^2)$ que usa o tensor-núcleo do HOSVD, junto com o *kernel* RBF padrão. Esse termo adicional introduz informações relevantes do tensor de entrada que é ignorada no *kernel* de (SIGNORETTO *et al.*, 2011). De fato, o tensor-núcleo da HOSVD pode prover informações relevantes do tensor de dados. Porém, a quantidade de processamentos computacionais tende a aumentar, devido ao termo adicional acrescentado ao *kernel*.

3.1.2 Fast Tensor-Core Kernel (F-TCK)

A segunda função *kernel* tensorial proposta pode ser vista como uma simplificação do H-TCK que usa apenas os termos correspondentes dos tensores-núcleos $\mathscr{W} \in \mathbb{R}^{R_1 \times \cdots \times R_N}$ e $\mathscr{Y} \in \mathbb{R}^{R_1 \times \cdots \times R_N}$ da HOSVD para cálculo da função *kernel*.

A função F-TCK proposta não usa as matrizes de fatores $\boldsymbol{U}^{(n)}$ e $\boldsymbol{V}^{(n)}$, para $n = 1, \ldots, N$, do HOSVD, devido às razões de complexidade. De fato, ignorar essas matrizes de fatores simplificam consideravelmente o custo computacional da função *kernel*, aliviando a carga de complexidade. O F-TCK proposto é dado por:

$$K(\mathscr{X}_i, \mathscr{X}_j) = \exp(-2\gamma||\mathscr{W} - \mathscr{Y}||_F^2), \tag{3.8}$$

onde γ é um parâmetro escalar.

Note que a normal de Frobenius $||\mathscr{W} - \mathscr{Y}||_F^2$ em (3.8) é equivalente a $||\boldsymbol{w} - \boldsymbol{y}||^2$, onde $||\cdot||^2$ representa a norma vetorial, $\boldsymbol{w}$ e $\boldsymbol{y}$ são vetorizações de $\mathscr{W}$ e $\mathscr{Y}$. Em outras palavras, a norma de Frobenius em (3.8) é equivalente a realizar a vetorização de nos tensores-núcleo antes da aplicação do *kernel* RBF padrão. No entanto, essa operação não quebra a estrutura tensorial

dos dados de entrada, pois não é aplicado diretamente sobre $\mathscr{X}_i$ e $\mathscr{X}_j$. De fato, usar os tensores-núcleo dos dados tensoriais de entrada assegura a preservação estrutural dos dados, como esses tensores-núcleo $\mathscr{W}$ e $\mathscr{Y}$ dependem da estrutura dos tensores de entrada $\mathscr{X}_i$ e $\mathscr{X}_j$. Vale lembrar que o tensor-núcleo de um HOSVD é totalmente ortogonal e ordenado (LATHAUWER *et al.*, 2000).

Em outras palavras, usar apenas o tensor-núcleo do HOSVD na expressão da função *kernel* preserva a informação estrutural multidimensional dos dados de entrada enquanto reduz a complexidade computacional ao ignorar o uso das matrizes de fator unitário.

3.1.3 TTD Fast Tensor-Core Kernel (TF-TCK)

A terceira função *kernel* proposta é aplicada para o caso de um tensor de entrada de ordem três. Nesse caso, a TTD do tensor de entrada $\mathscr{X}_i \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ e $\mathscr{X}_j \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ é expressa como:

$$\mathscr{X}_i = \boldsymbol{G}^{(1)} \times_1^2 \mathscr{G}^{(2)} \times_1^3 \boldsymbol{G}^{(3)}, \tag{3.9}$$

$$\mathscr{X}_j = \boldsymbol{H}^{(1)} \times_1^2 \mathscr{H}^{(2)} \times_1^3 \boldsymbol{H}^{(3)}, \tag{3.10}$$

onde $\mathscr{G}^{(2)} \in \mathbb{R}^{R_1 \times I_2 \times R_2}$ e $\mathscr{H}^{(2)} \in \mathbb{R}^{R_1 \times I_2 \times R_2}$ são os tensores núcleo da TDD, e $\boldsymbol{G}^{(1)} \in \mathbb{R}^{I_1 \times R_1}$, $\boldsymbol{G}^{(3)} \in \mathbb{R}^{R_2 \times I_3}$, $\boldsymbol{H}^{(1)} \in \mathbb{R}^{I_1 \times R_1}$ e $\boldsymbol{H}^{(3)} \in \mathbb{R}^{R_2 \times I_3}$ são as matrizes de fatores.

A função *kernel* proposta TF-TCK é similar à F-TCK, entretanto, utiliza os tensores núcleo $\mathscr{G}^{(2)}$ e $\mathscr{H}^{(2)}$ da TTD ao invés dos tensores núcleo da HOSVD, da seguinte maneira:

$$K(\mathscr{X}_i, \mathscr{X}_j) = \exp(-2\gamma||\mathscr{G}^{(2)} - \mathscr{H}^{(2)}||_F^2), \tag{3.11}$$

onde γ é um parâmetro escalar.

Similar à TCK, devido a motivos de complexidade, a TF-TCK não utiliza as matrizes de fatores $\boldsymbol{G}^{(1)}$, $\boldsymbol{G}^{(3)}$, $\boldsymbol{H}^{(1)}$ e $\boldsymbol{H}^{(3)}$, que simplifica significativamente o custo computacional da função *kernel*. O uso do tensor núcleo da TTD também preserva a estrutura multidimensional dos dados ao mesmo tempo que reduz a complexidade computacional ao ignorar as matrizes de fatores.

A TTD é mais genérica que a HOSVD e pode ser estimada por SVDs sequenciais (OSELEDETS, 2011). Essas propriedades podem proporcionar ao TF-TCK um tempo de processamento menor e uma melhor capacidade de discriminar os dados, quando comparado ao F-TCK.

4 SISTEMA DE AQUISIÇÃO E CLASSIFICAÇÃO PROPOSTO

Neste capítulo, é apresentado o sistema de captação e classificação de movimentos de mão proposto e também o processo realizado para obtenção da base de dados usada para classificação. A Seção 4.1 possui informações sobre os sensores e componentes utilizados no sistema de captação. Na Seção 4.3, é mostrada a construção e o resultado final do sistema. Por fim, na Seção 4.4, são demonstrados os passos necessário para a obtenção da base de dados utilizado na fase de testes.

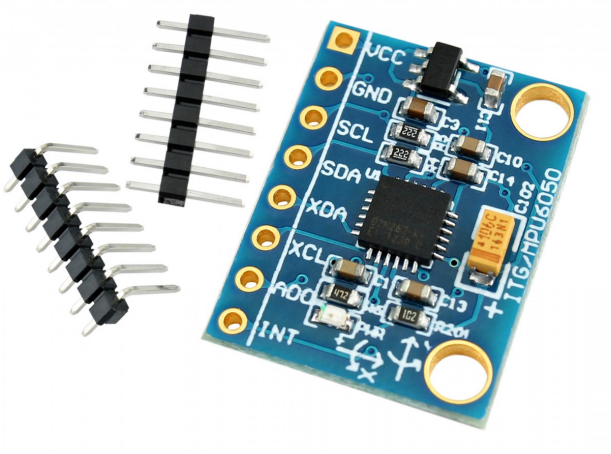
4.1 Sensores

Esta seção apresenta uma breve descrição de acelerômetros e giroscópios, sensores necessários para desenvolvimento do sistema de captação usado nessa dissertação. Como mencionado na Seção 1.1, o componente MPU-6050 é composto por sensores acelerômetro e giroscópio, esse conjunto de sensores é bastante utilizado pela sua facilidade de configuração, rápida conexão e valor acessível.

O MPU-6050, como visto na Figura 10, consiste em um componente eletrônico não-invasivo ou seja, não é necessário nenhum tipo de procedimento cirúrgico no usuário para seu uso. Outra característica deste componente é ser triaxial, possuindo três graus de liberdade, cada um referente às dimensões físicas existentes, isso permite monitorar sua posição e seu deslocamento com maior precisão. Também possui razão Signal-to-Noise (SNR) relativamente baixa, tornando-se uma excelente ferramenta para extração de características para classificação (INVENSENSE, 2013). Em alguns casos, esse componente também pode ser encontrado no mercado com a nomenclatura de GY-521.

Um acelerômetro é um sensor que capta a aceleração relativa de seu movimento (HEWITT, 2011), já o giroscópio, consegue medir o ângulo relativo do dispositivo em que esteja posicionado. Esse tipo de tecnologia é possível utilizando cristais piezoelétricos, piezoresistivos ou piezocapacitivos (WEBSTER, 1999). Cada eixo do acelerômetro do MPU-6050 independe do outro, fazendo com que a aceleração e a posição do componente sejam mensuradas. Uma ilustração dos graus de liberdade do componentes é mostrada na Figura 11. Um único MPU-6050 conta com um acelerômetro de três eixos e um giroscópio de três eixos, totalizando seis canais de comunicação. O MPU-6050 também possui controle de sensibilidade de até três níveis diferentes e trabalha com protocolo I^2C que é compatível com boa parte dos microcontroladores disponíveis

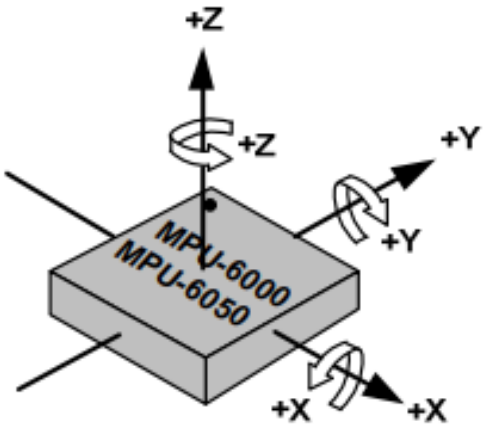
Figura 10 – MPU-6050



Fonte: usinainfo.com.br

no mercado (INVENSENSE, 2013).

Figura 11 – Eixos de atuação do MPU-6050.



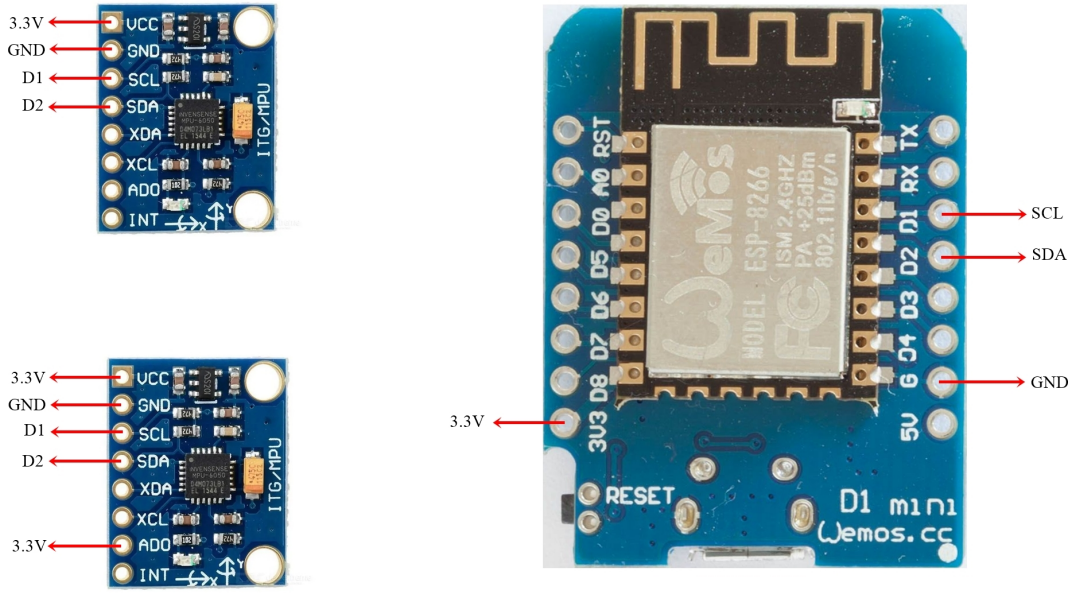
Fonte: (INVENSENSE, 2013)

4.2 Construção da luva inteligente

Para a construção do sistema de captação, dois MPU-6050 foram utilizados em conjunto com o placa Wemos D1 mini (ESP8266), pois a mesma possui conexão I^2C, conectividade com a plataforma Arduino e compatibilidade com a linguagem *Python 3* (ESPRESSIF, 2013). A conexão I^2C possibilita o uso de dois MPU-6050, sendo necessário apenas adicionar um nível de tensão à entrada AD0 de um dos componentes. O esquemático do circuito é mostrado na Figura 12. As portas Clock Serial (SLC) e Dados Seriais (SDA) são relacionadas à conexão I^2C, onde a SDA é responsável pela transmissão bidirecional de dados e a SCL cria um sinal

de sincronização que dita o inicio e fim de uma transmissão de dados (SEMICONDUCTORS, 2000).

Figura 12 – Esquemático do circuito do sistema de captação.



Fonte: Produzida pelo autor

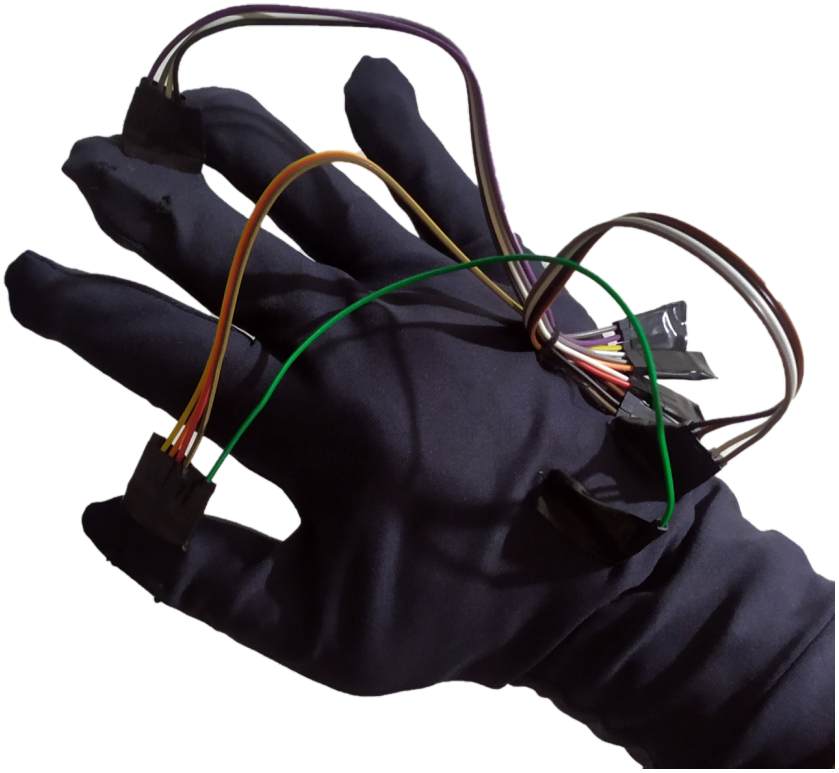
Após a montagem do circuito, o sistema foi conectado à plataforma Arduino para a transmissão do *software* criado para captura dos sinais gerados pelos sensores. A comunicação foi realizada de forma serial em uma porta Barramento Serial Universal (USB).

O sistema foi projetado para ser construído para captação de movimentos de mão, sendo assim, a opção mais viável para tal foi acoplar os componentes em uma luva de tecido, permitindo assim o mapeamento de cada movimento pelos sensores acelerômetro e giroscópio.

A confecção da “luva inteligente” foi realizada utilizando algodão devido sua características dielétricas (HEMSTREET, 1982). A Figura 13 mostra a luva com os componentes já acoplados. Foram escolhidos o polegar e o dedo médio para para fixação dos sensores. O polegar opositor tem uma maior mobilidade e possibilidade de rotação que os outros dedos, permitindo uma variedade maior de movimentos, enquanto o dedo médio está mais ao centro da mão e possibilita uma média entre os movimentos dos outros quatro dedos. Uma luva extra foi confeccionada para que os componentes não entrem em contato direto com a pele do usuário.

A utilização de *hardware open source* torna o sistema de baixo custo, a Tabela 1 mostra os valores de importação de cada item utilizado para construção da luva (cotação de Março de 2023).

Figura 13 – Sistema de captação.



Fonte: Produzida pelo autor

Tabela 1 – Orçamento do sistema de captação.

Item	Quantidade	Valor unitário(R$)	Valor Total(R$)
Jumper	16	0,13	2,08
Luva de algodão	2	5,00	10,00
ESP8266	1	9,96	12,07
MPU-6050	2	5,98	11,96
			TOTAL = 36,11

Fonte: elaborado pelo autor (2023).

4.3 Captação da base de dados tensorial

Uma base de dados tensorial foi criado realizando seis diferentes tipos de movimentos de mão por três pessoas. Cada movimento foi repetido cem vezes para cada participante, somando assim 300 amostras para cada movimento e 1800 amostras no total. No âmbito da classificação, cada movimento será considerada uma classe diferente. A Tabela 2 informa dados das pessoas que participaram do processo de criação da base de dados.

Cada amostra possui duração de 2,8 segundos, captando 500 pontos por cada um dos 12 canais de comunicação, totalizando 6000 pontos por movimento. A Figura 14 mostra os 12 sinais captados pelos sensores em uma amostra de movimento. Os seis primeiros canais são referentes ao sensor do dedo médio, já os últimos seis são correspondentes ao sensor do polegar.

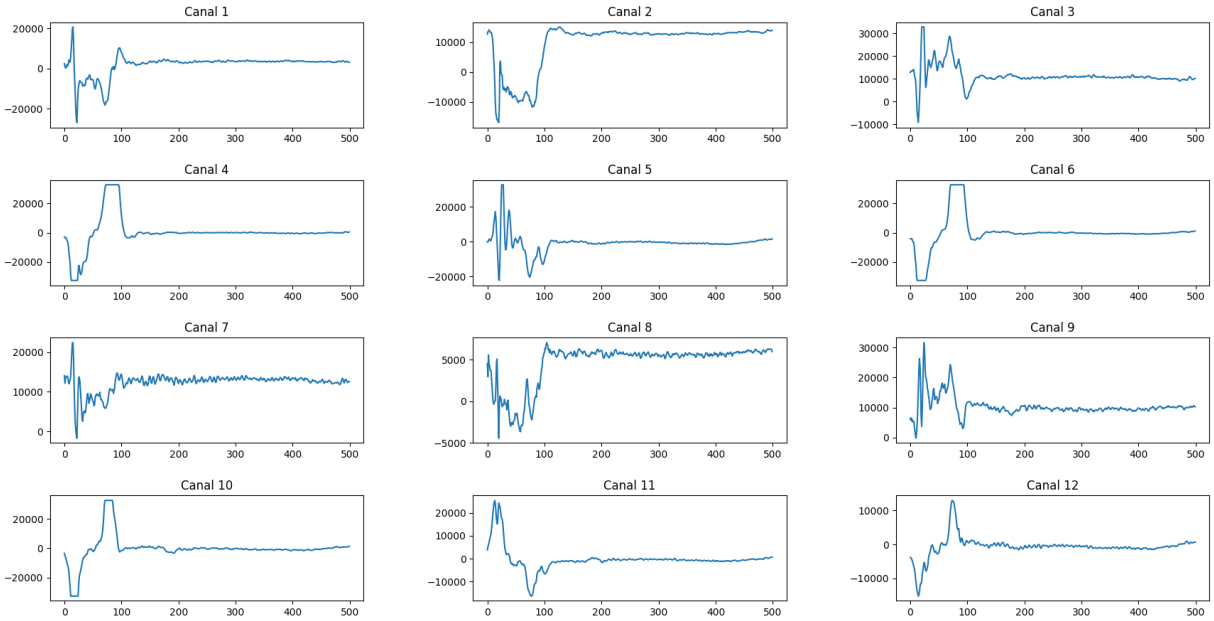
Tabela 2 – Dados dos participantes.

	Idade	Sexo
Participante 1	27	M
Participante 2	20	F
Participante 3	77	M

Fonte: elaborado pelo autor (2023)

A Tabela 3 mostra características dos dados salvos.

Figura 14 – Visualização de um movimento de mão a partir dos 12 canais de comunicação.



Fonte: produzido pelo autor (2023)

Tabela 3 – Resumo das características da matriz de dados.

Característica	Valor
Número de participantes	3
Movimentos distintos (classes)	6
Canais de comunicação	12
Movimentos por classe	100
Pontos coletados por canal	500
Taxa de amostragem média (Hz)	2150

Fonte: elaborado pelo autor (2023).

Ao final, a base de dados tensorial foi disposto com dimensões $2 \times 6 \times 500 \times 1800$, referente à quantidade de sensores, canais de comunicação, quantidade de pontos por canais e quantidade de amostras. A natureza tensorial dos dados da própria luva inteligente, que possui

múltiplos sensores, cada um com múltiplos canais, sendo que cada canal de cada sensor captura múltiplas amostras, gerando assim informação tridimensional do tipo sensor $\times$ canal $\times$ amostra. A base de dados pode ser acessada a partir do link: (*BASE DE DADOS*). O algoritmo de captação foi feito na linguagem Python 3.9.

As 300 primeiras amostras da base de dados são pertinentes ao movimento “abrir a mão”, as amostras 301 à 600 representam o movimento “para baixo", do intervalo 601 ao 900 representam o movimento “para cima", as amostra 901 à 1200 representam o movimento “fechar a mão”, o intervalo 1201 ao 1500 são referentes ao movimento “polegar pra cima”, por fim, as 300 últimas amostras representam o movimento “lateral”.

A conexão do microcontrolador com o computador foi feita em uma taxa de 115,2kHz utilizando conexão USB. Os sensores 1 e 2 ocuparam o endereço 0x68 e 0x69 da conexão I^2C, respectivamente. O critério para início da captura de movimentos foi feito através de limiar de amplitude. O canal 4 foi definido como o gatilho para que o sinal seja capturado. Por ser um canal correspondente ao sensor de acelerômetro, o mesmo permanece com valor próximo de zero quando em repouso, a partir do momento que o valor sai do intervalo $-2000 < x < 2000$ o algorítimo inicia a coleta de dados durante cerca de 2,8 segundos.

4.3.1 Escolha dos movimentos

A escolha dos movimento realizado foi realizada de maneira que os sensores acoplados nos dedos polegar e médio sejam movimentados o máximo possível, permitindo que o sensor forneça informações suficientes sobre o movimento. A simplicidade do movimento também foi um ponto considerado, uma vez que um movimento de simples execução possibilita uma maior inclusão de pessoas que podem utilizar o sistema. A descrição de cada movimento escolhido segue abaixo:

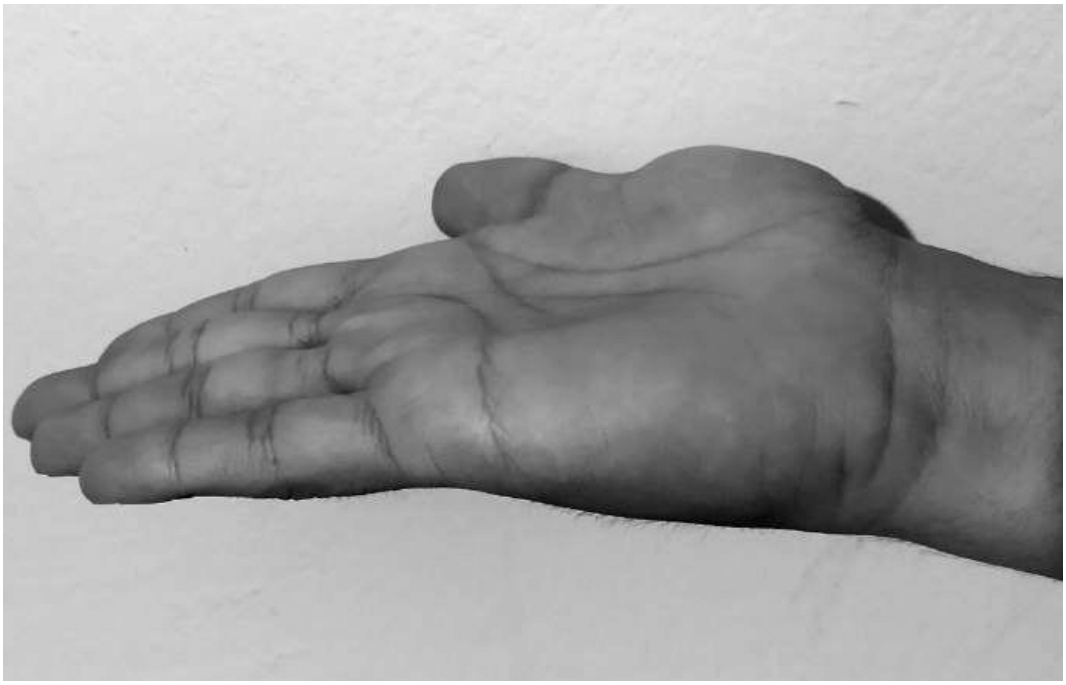
- **Movimento para cima:** consiste em movimentar toda a mão para cima, a palma da mão deve estar voltada para cima ;
- **Movimento para baixo:** consiste em movimentar toda a mão para baixo com os dedos juntos, a palma da mão deve estar voltada para cima;
- **Fechar a mão:** com a palma da mão para cima, o usuário deve fechar a mão, tomando cuidado para movimentar o polegar;
- **Abrir a mão:** o usuário deve abrir a mão, o movimento é semelhante a um espasmo e feito com a palma da mão para cima;

- **Lateral:** o usuário permanece com os dedos unidos e move, lateralmente, os dedos em direção a si;
- **Polegar para cima**: com a palma da mão para baixo o usuário deve levar todos os dedos à palma da mão com exceção do polegar, isso deve ser realizado enquanto a mão é rotacionada, o polegar deve estar firme apontando para cima.

Os movimentos foram alternados entre força e velocidade de execução intencionalmente para que o classificador possa ser treinado de forma generalizada para identificar essas diferenças na execução do movimento.

Para uma melhor padronização de captação do movimento, foi definido que a mão do usuário deverá estar na posição horizontal, voltada para cima, para cada início de movimento, tal como ilustrada na Figura 15. Em seguida o movimento deve estar correspondente ao que se vê na Figura 16 e por fim a mão deve retornar a posição inicial.

Figura 15 – Posição inicial da mão para os movimentos.



Fonte: produzido pelo autor (2023)

4.4 Sistema de classificação

O sistema de classificação foi construído em Python 3.9 utilizando, em sua maior parte, de *frameworks* de código aberto, tais como: numpy (HARRIS, 2020), scikit-learn (PEDREGOSA, 2011), matplotlib (HUNTER, 2007) e tensorly (KOSSAIFI *et al.*, 2019).

Figura 16 – Movimentos de mão escolhidos.

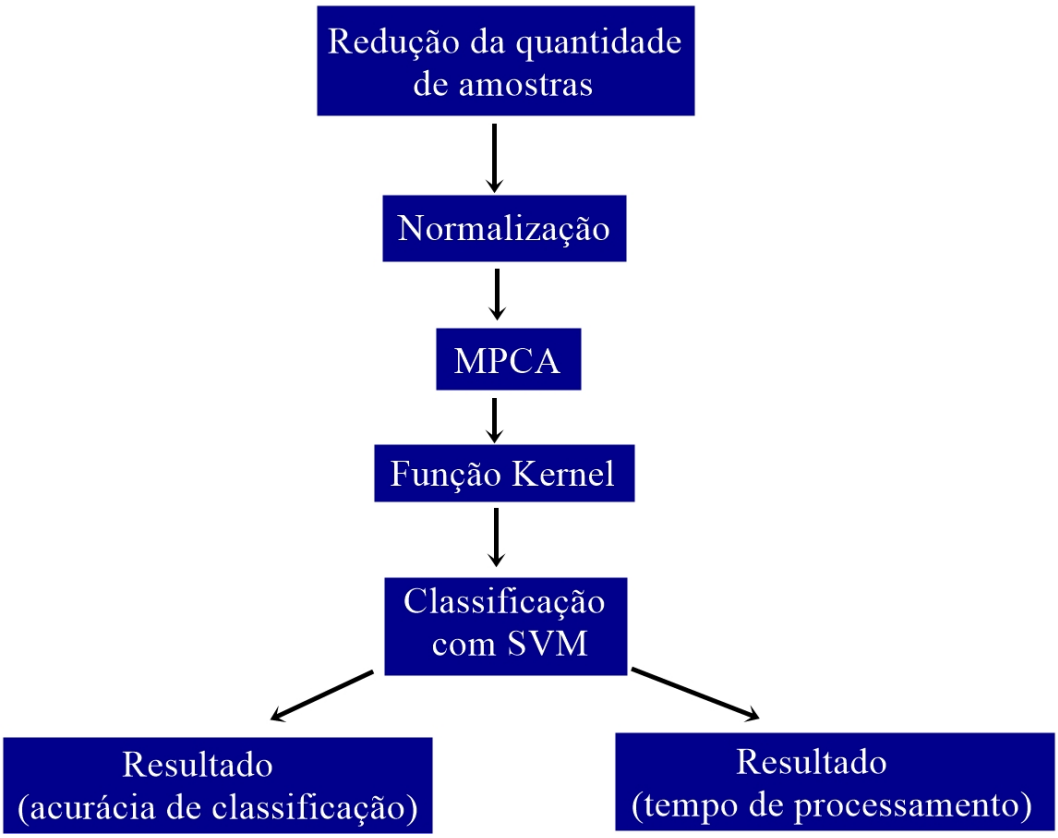
(a) Cima (b) Fechar (c) Abrir

(d) Baixo (e) Lateral (f) Polegar para cima

Fonte: produzido pelo autor (2023).

A para a classificação, foi utilizado o classificador SVM em sua forma dual, usando várias funções *kernel* do estado da arte, juntamente com as funções *kernel* propostas nesta dissertação. Para pré-processamento dos dados, foi utilizada a normalização l^2 e MPCA para redução de dimensionalidade dos atributos e melhora de performance de classificação. Mais detalhes sobre sobre o processo de classificação pode ser encontrados no Capítulo 5. As etapas do sistema de classificação são descritas na Figura 17.

Figura 17 – Sistema de classificação proposto.



Fonte: produzido pelo autor (2023)

5 TESTES E RESULTADOS

Esse capítulo apresenta os resultados obtidos na classificação da base de dados tensorial de movimentos de mãos introduzida na Seção 4.3. O SVM para dados de entrada tensoriais descrito na subseção 2.4.2 é usado juntamente com as funções *kernel* tensoriais propostas e diferentes funções *kernels* tensoriais do estado da arte. Além das três funções *kernel* apresentadas, foram testadas as seguintes funções *kernel* tensoriais: o *kernel* tensorial de (SIGNORETTO *et al.*, 2011), denotado aqui por *kernel* HOSVD, o FAKSETT (KARMOUDA *et al.*, 2021) e o DuSK (HE *et al.*, 2014).

5.1 Ajuste dos Hiperparâmetros

Afim de encontrar a melhor configuração para cada função *kernel*, foi desenvolvida uma metologia para alcançar o melhor cenário para cada técnica. Para todos os testes, a validação cruzada *K-fold* com $K = 10$ foi usada, o posto de todos os tensores de entrada foi fixado em 3 e os dados foram processados com normalização l^2. Os testes foram realizados com as seis classes, sendo elas: “abrir a mão”, “fechar a mão”, “movimento para cima”, “ movimento para baixo”, “movimento lateral” e “polegar para cima”.

O primeiro teste consistiu em encontrar os valores ótimos dos hiperparâmetros γ da função *kernel* e C do SVM. Para isso, o intervalo dos hiperparâmetros testados foram $0.01 \leq C \leq 100$ e $0.01 \leq \gamma \leq 5$, com 20 valores igualmente espaçados para cada um. Para reduzir o tempo de processamento, a quantidade de amostras por classe foi reduzida para 60 e foi aplicado MPCA nos dados ao longo da terceira dimensão (dimensão com maior índice), onde a variância explicada foi fixada em $0,95$. Os valores dos hiperparâmetros que forneceram as melhores acurácias estão listados na Tabela 4.

Tabela 4 – Hiperparâmetros ajustados.

Kernel	γ	C
DuSK	1,84	5,27
F-TCK	4,20	15,8
TF-TCK	1,05	5,27
H-TCK	0,26	5,27
Kernel HOSVD	0,26	5,27
FAKSETT	1,05	5,27

Fonte: elaborado pelo autor (2023).

Com esses hiperparâmetros, foi realizado o segundo teste, que consistiu em encontrar

a quantidade ótima de componentes de MPCA de cada função *kernel* ao longo de cada dimensão da base de dados tensorial (denotadas por I_1, I_2 e I_3). A quantidade de componentes de MPCA que forneceu a melhor acurácia para cada função *kernel* é vista na Tabela 5 respectivamente, onde o intervalo das componentes que foram testados foi de $2 \leq R_2 \leq 6$ e $2 \leq R_3 \leq 200$. Não houve redução de dimensionalidade para R_1 pois, caso houvesse, os dados de entrada se tornariam matriciais para $R_1 = 1$, o que fugiria do escopo desta dissertação. Para esse teste também foram usadas apenas 60 amostras.

Tabela 5 – Número de componentes de MPCA para cada função *kernel* ao longo de cada dimensão do tensor.

Técnica	R_1	R_2	R_3
DuSK	2	5	6
F-TCK	2	5	34
TF-TCK	2	5	29
H-TCK	2	5	2
Kernel HOSVD	2	4	4
FAKSETT	2	5	30

Fonte: elaborado pelo autor (2023).

Observa-se que o MPCA levou a uma redução massiva no número de entradas para todas as funções *kernel* testadas. Por exemplo, o MPCA permitiu uma redução no número de tensores de entrada de $2 \times 6 \times 500 = 6.000$ para $2 \times 5 \times 2 = 20$, para o H-TCK. Para todas as funções *kernel* tensoriais, a redução da dimensionalidade foi particularmente significativamente I_3.

5.2 Acurácia das Funções *Kernel* Tensoriais

Com os hiperâmetros e a quantidade de componentes de MPCA selecionados, pode-se então analisar o melhor caso para cada função *kernel* tensorial. A Tabela 6 mostra a acurácia média de cada técnica para uma validação cruzada *K-Fold* com $K = 10$ utilizando os hiperparâmetros da Tabela 4 e as componentes de MPCA da Tabela 5.

As técnicas baseadas em TTD possuem as maiores acurácias médias. A FAKSETT com 93,33% e a TF-TCK com 93,05%. O FAKSETT possui uma acurácia ligeiramente superior pois utiliza as matrizes de fatores da TTD, enquanto o TF-TCK usa apenas os tensores-núcleo da decomposição. Por outro lado, as funções baseadas em HOSVD seguem um padrão interessante: a maior acurácia das três funções *kernel* tensoriais baseados em HOSVD é a H-TCK com 85,06%. Como visto anteriormente, essa função *kernel* conta com as matrizes de fatores e o tensor-núcleo

Tabela 6 – Acurácia média das técnicas.

Kernel	Acurácia
DuSK	84,06%
F-TCK	82,50%
TF-TCK	93,06%
H-TCK	85,06%
Kernel HOSVD	84,27%
FAKSETT	93,33%

Fonte: elaborado pelo autor (2023).

da HOSVD, o *kernel* HOSVD de (SIGNORETTO *et al.*, 2011) possui uma acurácia um pouco menor (84,27%) por utilizar apenas as matrizes de fatores. A F-TCK, por utilizar apenas o tensor-núcleo da decomposição HOSVD, possui a menor acurácia média de todas as técnicas, com 82,50%. Por fim, o *kernel* DuSK possui uma acurácia apenas superior a do F-TCK, com 84,06%.

Sendo assim, as duas funções *kernel* baseadas em TTD forneceram os melhores resultados, mostrando que usar a TTD para cálculo das funções *kernel* é mais eficiente do que usar a decomposição HOSVD e PARAFAC. Como mencionado anteriormente, o TTD é mais genérico que a HOSVD e a PARAFAC, isso fornece às funções *kernel* baseadas em TTD uma melhor capacidade de discriminar os dados. Além disso, a utilização apenas do termo relativo aos tensores-núcleo da TTD mostrou-se uma escolha eficaz, já que a TF-TCK possui acurácia média praticamente igual a de FAKSETT, sendo a melhor técnica proposta em termos de acurácia.

A Tabela 7 mostra o desvio padrão das acurácias obtidas a partir da validação cruzada utilizada. A função *kernel* TF-TCK obteve o menor valor de desvio padrão de acurácia (0,034), a metade do valor de FAKSETT que, apesar de ter a maior acurácia média, possui o maior desvio padrão (0,068). Por outro lado, apesar de possuir a menor acurácia, a F-TCK possui o segundo menor desvio padrão (0,054), sendo o menor entre as funções que utilizam a HOSVD.

Tabela 7 – Desvio padrão das funções *kernel* tensoriais.

Kernel	Desvio padrão
DuSK	0,061
F-TCK	0,054
TF-TCK	0,034
H-TCK	0,057
Kernel HOSVD	0,068
FAKSETT	0,068

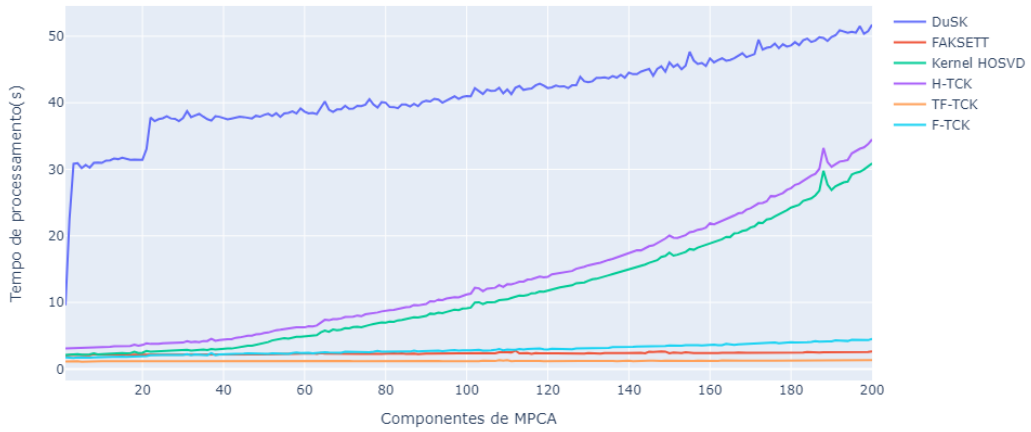
Fonte: elaborado pelo autor (2023).

5.3 Tempo de Processamento das Funções *Kernel* Tensoriais

Uma análise do comportamento do tempo de processamento das funções *kernel* tensoriais é mostrada na Figura 18 onde o tempo de processamento obtido por cada função *kernel* tensorial de acordo com a quantidade de componentes de MPCA na dimensão I_3. Os tempos apresentados nos testes correspondem ao tempo total (em segundos) necessário para realizar o processamento, treinamento e testes dos dados. Os testes foram realizados em Python 3.9 com a seguinte configuração computacional: CPU Intel(R) Core(TM) i7-9750H CPU @ 2,60GHz até 4,5GHz, 32GB de RAM *dual channel*, GPU NVIDIA GeForce RTX 2060 e SSD M.2 com velocidade de leitura de 2400MB/s.

Percebe-se que para os *kernels* que utilizam as decomposições HOSVD e PARAFAC, o tempo de processamento tende a aumentar exponencialmente, entretanto, as técnicas que utilizam a decomposição TTD o tempo de processamento aumenta de forma mais lenta. Isso mostra que essa decomposição se torna mais vantajosa ao uso para grandes quantidades de dados. Uma exceção a isso é a função *kernel* F-TCK, que permanece com um tempo de processamento semelhante às técnicas que utilizam a TTD. Isso se deve ao fato do F-TCK utilizar apenas o tensor-núcleo da HOSVD, otimizando o tempo de processamento. O posto de cada tensor de entrada foi fixado em 3 e devido ao grande requisito computacional, essa análise foi realizada utilizando 5 classes e 60 amostras para cada classe.

Figura 18 – Tempo de processamento de classificação de acordo com a quantidade de componentes de MPCA na dimensão I_3 para 5 classes.



Fonte: produzido pelo autor (2023).

Essa análise indica que a TTD se torna mais vantajosa ao uso para grandes quantidades de dados. Uma exceção a isso é a função *kernel* F-TCK, que permanece com um tempo

de processamento semelhante às técnicas que utilizam a TTD. Isso se deve ao fato do F-TCK utilizar apenas o tensor-núcleo da HOSVD, otimizando o tempo de processamento.

Partiremos agora para a análise de tempo de processamento dos testes realizados utilizando os hiperparâmetros das Tabela 4 e as componentes de MPCA da Tabela 5. A Tabela 8 mostra o tempo de processamento de cada função *kernel* tensorial.

Tabela 8 – Tempo de processamento em segundos - usando os hiperparâmetros das Tabelas 4 e as componentes de MPCA da Tabela 5.

Kernel	Tempo de processamento (s)
DuSK	485,63
F-TCK	46,43
TF-TCK	45,33
H-TCK	96,15
Kernel HOSVD	54,90
FAKSETT	78,83

Fonte: elaborado pelo autor (2023).

A partir dos dados da Tabela 8 vemos que as funções *kernels* tensoriais que apresentam o menor tempo de processamento são os que utilizam apenas os termos tensoriais de suas respectivas decomposições, sendo eles o F-TCK e o TF-TCK, esse último sendo 2,4% mais rápido que o F-TCK, 112,1% mais rápido que o H-TCK, 73,90% mais rápido que a FAKSETT e 30,0% mais rápido que o *kernel* HOSVD.

Como era de se esperar, o tempo de processamento do H-TCK é superior ao do *kernel* HOSVD, pois a expressão do H-TCK possui um termo adicional quando comparada ao *kernel* HOSVD, o mesmo vale para a função FAKSETT, que possui um termo adicional em relação à TF-TCK. Isso mostra que usar apenas o tensor-núcleo da HOSVD reduz consideravelmente o tempo de processamento. Além disso, o DuSK forneceu, de longe, o maior tempo de processamento, sendo cerca de 5 vezes mais lento que a segunda técnica mais lenta.

Com isso, pode-se concluir que o TF-TCK apresentou bons resultados em termos de precisão e os melhores resultados em tempo de processamento. O TF-TCK possui as melhores métricas das três funções *kernel* tensoriais propostas, com uma acurácia média de 93,06%, sendo a segunda melhor acurácia no geral, possui o menor desvio padrão de todas as funções *kernel* analisadas com 0,034 e o menor tempo de processamento entre todas as técnicas testadas com 45,33s.

5.4 Tabelas de Confusão - Melhor Caso

A Tabela 9 mostra a matriz de confusão obtida com a função *kernel* TF-TCK proposta usando os hiperparâmetros das Tabelas 4 e as componentes de MPCA da Tabela 5.

Tabela 9 – Tabela de confusão do melhor caso usando TF-TCK.

	Abrir	Baixo	Cima	Fechar	Lateral	Polegar para cima	Predição (%)
Abrir	282	17	0	1	0	0	94,00
Baixo	12	273	1	13	0	1	91,00
Cima	0	7	289	4	0	0	96,33
Fechar	0	13	10	275	1	1	91,67
Lateral	2	6	0	6	285	1	95,00
Polegar para cima	0	8	0	19	2	271	90,33
							MÉDIA = 93,06

Fonte: elaborado pelo autor (2023).

Pode-se notar que o movimento “cima” e “lateral” proporcionaram as maiores acurácias, com 96,33 % 95,00% de amostras corretas, respectivamente. Por outro lado, a classe “polegar” apresentou a pior acurácia (90,33%), com a maior parte das amostras erroneamente classificadas como “mão para baixo” e “fechar a mão”.

5.5 Análise do TF-TCK para as 5 melhores classes

A Tabela 9 indica que a pior classe foi a “polegar para cima”, que apresentou acurácia de 90,33%. Faremos agora uma análise das 5 melhores classes para a função *kernel* TF-TCK. Para isso, a metodologia anterior foi replicada, onde foi feita uma otimização dos hiperpâmetros e da quantidade de componentes de MPCA usadas. A Tabela 10 indica a quantidade de componentes de MPCA usadas ao longo de cada dimensão do tensor e os hiperparâmetros usados nesse experimento.

Tabela 10 – Número de componentes de MPCA e Hiperpâmetros ajustados para a função *kernel* TF-TCK usando 5 classes.

Dimensão	Quantidade de componentes	Hiperparâmetro	Valor
I_1	2	γ	0,26
I_2	5	C	47,3
I_3	26	-	-

Fonte: elaborado pelo autor (2023).

A Tabela 11 mostra a Tabela de confusão para as 5 melhores classes do TF-TCK.

Tabela 11 – Tabela de confusão do melhor caso com 5 classes usando TF-TCK.

	Abrir	Baixo	Cima	Fechar	Lateral	Predição (%)
Abrir	291	9	0	0	3	97,00
Baixo	15	279	1	4	1	93,00
Cima	0	8	288	4	0	96,00
Fechar	0	11	10	278	1	92,77
Lateral	3	5	0	0	292	97,33
						MÉDIA = 95,22

Fonte: elaborado pelo autor (2023).

No caso das 5 melhores classes, a acurácia de classificação atinge os 95,22%. Nesse caso, as classes “Abrir” e “Lateral” proporcionam a maior taxa de acertos, com 97,00% e 97,33%, respectivamente. Já a pior classe é a “fechar a mão” com 92,77%, com a maioria das confusões ocorridas com as classes “mão para baixo” e “mão para cima”.

5.6 Outras Métricas - Melhor Caso

Por fim, os dados da Tabela 12 mostram o F1-*score* e ao *recall* do *kernel* TF-TCK, no seu melhor caso. Por se tratar de um problema multiclasses, ambas as métricas foram o obtidas usando uma média não ponderada das métricas individuais de cada classes.

Tabela 12 – Métricas de F1-*score* e *recall* para o *kernel* TF-TCK.

Métrica	5 classes	6 classes
F1-*score*	95,22 %	93,13%
Recall	95,20%	93,06%

Fonte: elaborado pelo autor (2023).

Os resultados da Tabela 12 estão iguais ou bem próximos do valor de acurácia para o *kernel* TF-TCK tanto para 5 como para 6 classes, isso indica que a média de erro entre as classes estão, de certa forma, balanceados.

6 CONCLUSÕES

Esta dissertação apresentou estudos acerca de aprendizado tensorial usando dados captados pelo próprio autor, foram desenvolvidas três novas funções *kernel* tensoriais, cada uma utilizando técnicas diferentes e proporcionando vantagens tanto em capacidade de classificação quanto na velocidade de processamento quando comparadas às funções *kernel* do estado da arte.

Uma "luva inteligente" foi desenvolvida para captura de movimento de mãos, a luva consiste em um protótipo de baixo custo com sensores acoplados, capaz de transmitir informações de aceleração e de giro nas três dimensões. Com a luva, foi construída da base de dados tensorial a partir de movimentos de mãos, a base de dados possui seis movimentos, cada um representando um movimento de mão, a mecânica tridimensional do movimento captado pelos dois sensores acoplados na "luva inteligente" permitiu organizar os dados de maneira naturalmente tensorial.

A primeira função *kernel* desenvolvida denotada por F-TCK foi inspirada na função *kernel* desenvolvida em (SIGNORETTO *et al.*, 2011). Entretanto, diferente da técnica desenvolvida em (SIGNORETTO *et al.*, 2011), a F-TCK utiliza-se apenas dos tensores-núcleo da HOSVD, esse fato fez com que essa técnica tivesse um dos melhores resultados de tempo de processamento quando comparado às demais funções *kernel* usadas nessa dissertação. Quando comparados aos *kernels* da literatura testados, o tempo de processamento da F-TCK chega a ser 18,24% menor que o de (SIGNORETTO *et al.*, 2011) e 70,00% mais rápido que o *kernel* de (KARMOUDA *et al.*, 2021) e 10,5 vezes mais rápido que o *kernel* de (HE *et al.*, 2014). Já em termos de acurácia o F-TCK obteve a menor acurácia, com 82,50%.

A segunda função *kernel* chamada de H-TCK, também inspirada na função *kernel* de (SIGNORETTO *et al.*, 2011), apresentou resultados de tempo de processamento maiores que a F-TCK e o *kernel* HOSVD pois possui o cálculo de termos adicionais em comparação à essas duas técnicas. A H-TCK teve um tempo de processamento 42,9% maior que a técnica de (SIGNORETTO *et al.*, 2011) e 18,0% maior que a FAKSETT, mas obteve um tempo menor que *kernel* de (HE *et al.*, 2014). Já em termos de acurácia essa função possui as melhores métricas para as técnicas que tem como base a HOSVD, mostrando que a combinação do tensor-núcleo e as matrizes de fatores do HOSVD são uma boa opção para discretização dos dados tensoriais, ficando atrás apenas das técnicas baseadas em TTD.

Por último temos a função *kernel* TF-TCK, inspirada em (KARMOUDA *et al.*, 2021), com ele foram obtidos o segundo melhore resultado em termos de acurácia, sendo até 9,00%

melhor que o DuSK, 8,79% melhor que o *kernel* HOSVD e apenas 0,27% pior que FAKSETT. Já o tempo de processamento do TF-TCK é o melhor em comparação as outras 5 técnicas testadas. Sendo assim a utilização dos tensores-núcleo do TTD mantém uma alta acurácia de classificação e reduz o tempo de processamento, ela preserva a estrutura dos dados e fornece características que seriam perdidas o processo de vetorização dos dados de entrada fosse realizado.

Esses resultados de classificação utilizando aprendizado tensorial se mostraram bastantes promissores. Futuramente esses métodos devem ser testados em diferentes bases de dados, com diferentes propósitos, como classificação de imagens e vídeos. Há também a possibilidade explorar as funções em tensores de ordem superior a 3. Outra importante perspectiva é variar o posto dos tensores de entrada aplicadas às decomposições tensoriais. Também há a possibilidade de explorar outros tipos de decomposições tensoriais, como a *Low-Rank Tensor Decomposition*. Outra perspectiva é adicionar diferentes tipos de sensores na "luva inteligente" e realizar aplicações práticas.

REFERÊNCIAS

AKBANI, R.; KWEK, S.; JAPKOWICZ, N. Applying support vector machines to imbalanced datasets. **European conference on machine learning**, Springer, p. 39–50, 2004.

ALPAYDIN, E. Introduction to machine learning. **MIT press**, 2010.

BACKES A. R.; JUNIOR, J. J. d. M. S. **Introdução à visão computacional usando Matlab**. [*S. l.*]: Alta Books Editora, 2019.

BISHOP, C. M. **Pattern recognition and machine learning**. [*S. l.*]: Springer, 2006.

BURGES, C. A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery, v. 2, n. 2, p. 121–167, 1998.

CAI, B.; ZHANG, J.; SUN, W. **Heterogeneous Tensor Mixture Models in High Dimensions**. 2021.

CHEN, B.; SUN, T.; ZHOU, Z.; ZENG, Y.; CAO, L. Nonnegative tensor completion via low-rank tucker decomposition: Model and algorithm. **IEEE Access**, v. 7, p. 95903–95914, 2019.

CORTES, C.; MOHRI, M.; SYED, U. Learning from multiway data: simple and efficient tensor regression. **Machine Learning**, Springer, v. 74, n. 3, p. 255–283, 2009.

CORTES, C.; VAPNIK, V. Support-vector networks. **Machine learning**, Springer, v. 20, n. 3, p. 273–297, 1995.

CRISTIANINI, N.; SHAWE-TAYLOR, J. *et al.* **An introduction to support vector machines and other kernel-based learning methods**. [*S. l.*]: Cambridge university press, 2000.

ERTAM, F.; AYDıN, G. Data classification with deep learning using tensorflow. In: **2017 International Conference on Computer Science and Engineering (UBMK)**. [*S. l.*: *s. n.*], 2017. p. 755–758.

ESPRESSIF SYSTEMS. **ESPRESSIF SMART CONNECTIVITY PLATFORM: ESP8266**. Shangai, 2013.

EVENBLY, G. Number-state preserving tensor networks as classifiers for supervised learning. **Frontiers in Physics**, v. 10, 2022.

FAIDALLAH, E. M. e. a. Control and modeling a robot arm via emg and flex signals. **15th International Workshop on Research and Education in Mechatronics (REM)**, 2014.

FRANCISCO, d. S.; FLáVIO, d. S.; ALEXANDRE, C. A. C. Classification of hand movements from emg signals for people with motor disabilities. **IEEE Latin America Transactions**, v. 18, n. 11, p. 2019–2026, 2020.

GIJSBERTS, A.; ATZORI, M.; CASTELLINI, C.; MüLLER, H.; CAPUTO, B. Movement error rate for evaluation of machine learning methods for semg-based hand movement classification. **IEEE Transactions on Neural Systems and Rehabilitation Engineering**, v. 22, n. 4, p. 735–744, 2014.

GRADSHTEYN, I.; RYZHIK, I. Table of integrals, series, and products 6th edn (new york: Academic). 2000.

GUPTA, H. P. e. a. A continuous hand gestures recognition technique for human-machine interaction using accelerometer and gyroscope sensors. **IEEE Sensors Journal**, v. 16, n. 16, p. 6425–6432, 2016.

HARRIS, t. C. R. Array programming with NumPy. **Nature**, Springer Science and Business Media LLC, v. 585, n. 7825, p. 357–362, set. 2020. Disponível em: https://doi.org/10.1038/s41586-020-2649-2.

HARSHMAN, R. A. *et al.* Foundations of the parafac procedure: Models and conditions for an"explanatory"multimodal factor analysis. University of California at Los Angeles Los Angeles, 1970.

HE, L.; KONG, X.; YU, P. S.; YANG, X.; RAGIN, A. B.; HAO, Z. Dusk: A dual structure-preserving kernel for supervised tensor learning with applications to neuroimages. In: SIAM. **Proceedings of the 2014 SIAM International Conference on Data Mining**. [*S. l.*], 2014. p. 127–135.

HE, L.; LU, C.-T.; DING, H.; WANG, S.; SHEN, L.; YU, P. S.; RAGIN, A. B. Multi-way multi-level kernel modeling for neuroimaging classification. In: **Proceedings of the IEEE conference on computer vision and pattern recognition**. [*S. l.*: *s. n.*], 2017. p. 356–364.

HEMSTREET, J. M. Dielectric constant of cotton. journal of electrostatics. Bulletin of Mathematical Biophysics, v. 13, n. 3, p. 345–353, 1982.

HEWITT, P. G. **Física Conceitual**. 2^a edição. [*S. l.*]: Bookman, 2011.

HITCHCOCK, F. L. The expression of a tensor or a polyadic as a sum of products. **Journal of Mathematics and Physics**, Wiley Online Library, v. 6, n. 1-4, p. 164–189, 1927.

HUNTER, J. D. Matplotlib: A 2d graphics environment. **Computing in Science & Engineering**, IEEE COMPUTER SOC, v. 9, n. 3, p. 90–95, 2007.

INVENSENSE. **MPU-6000 and MPU-6050 Product Specification Revision 3.4**. Sunnyvale, 2013.

JOACHIMS, T. Text categorization with support vector machines: Learning with many relevant features. **European conference on machine learning**, Springer, p. 137–142, 1998.

KARMOUDA, O.; BOULANGER, J.; BOYER, R. Speeding up of kernel-based learning for high-order tensors. In: IEEE. **ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)**. [*S. l.*], 2021. p. 2905–2909.

KOHAVI, R. A study of cross-validation and bootstrap for accuracy estimation and model selection. v. 14, 03 2001.

KOLDA, T. G.; BADER, B. W. Tensor decompositions and applications. **SIAM review**, SIAM, v. 51, n. 3, p. 455–500, 2009.

KOSSAIFI, J.; PANAGAKIS, Y.; ANANDKUMAR, A.; PANTIC, M. Tensorly: Tensor learning in python. **Journal of Machine Learning Research**, v. 20, n. 26, p. 1–6, 2019. Disponível em: http://jmlr.org/papers/v20/18-277.html.

KRUSKAL, J. B. Three-way arrays: rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics. **Linear algebra and its applications**, Elsevier, v. 18, n. 2, p. 95–138, 1977.

LATHAUWER, L. D.; MOOR, B. D.; VANDEWALLE, J. Blind source separation by higher-order singular value decomposition. In: **Proc. EUSIPCO**. [*S. l.*: *s. n.*], 1994. v. 1, p. 175–178.

LATHAUWER, L. D.; MOOR, B. D.; VANDEWALLE, J. On the best rank-1 and rank-(r 1, r 2,..., rn) approximation of higher-order tensors. **SIAM journal on Matrix Analysis and Applications**, SIAM, v. 21, n. 4, p. 1324–1342, 2000.

LATHAUWER, L. D.; MOOR, B. D.; VANDEWALLE, J. Tensor decompositions and applications. **SIAM review**, SIAM, v. 51, n. 3, p. 455–500, 2009.

LATHAUWER, L. D.; VANDEWALLE, J. Dimensionality reduction in higher-order signal processing and rank-(r1, r2,. . . , rn) reduction in multilinear algebra. **Linear Algebra and its Applications**, Elsevier, v. 391, p. 31–55, 2004.

LEE S.; PARK, K. L. J. K. User study of vr basic controller and data glove as hand gesture inputs in vr games. **2017 International Symposium on Ubiquitous Virtual Reality (ISUVR)**, 2017.

LIU, Y.; LIU, J.; LONG, Z.; ZHU, C. Statistical tensor classification. In: ____. **Tensor Computation for Data Analysis**. Cham: Springer International Publishing, 2022. p. 199–217. ISBN 978-3-030-74386-4. Disponível em: https://doi.org/10.1007/978-3-030-74386-4_8.

LOAN, C. F. V.; GOLUB, G. Matrix computations (johns hopkins studies in mathematical sciences). **Matrix Computations**, v. 5, 1996.

MA, L.; HU, Y.; ZHANGR, Y. Support Tensor Machines based bubble defect detection of Lithium-ion polymer cell sheets. **Engineering Letters**, v. 25, n. 1, p. 304–337, 2017.

MACHANGPA J. W.; CHINGTHAM, T. S. Head gesture controlled wheelchair for quadriplegic patients. Procedia Computer Science, v. 132, p. 342–351, 2018.

MAKANTASIS, K.; DOULAMIS, A. D.; DOULAMIS, N. D.; NIKITAKIS, A. Tensor-based classification models for hyperspectral data analysis. **IEEE Transactions on Geoscience and Remote Sensing**, v. 56, n. 12, p. 6884–6898, 2018.

MEDIUM. **Kernel Trick in SVM**: Kernel trick is used to classify non-linear to linear using some mathematical concepts provided by mercer's theorem. 2019. Disponível em: https://medium.com/analytics-vidhya/how-to-classify-non-linear-data-to-linear-data-bb2df1a6b781. Acesso em: 15 set. 2020.

NIELSEN, S.; NELLEMANN, L. J.; LARSEN, L. B.; STEC, K. The social acceptability of peripheral interaction with 3d gestures in a simulated setting. In: SPRINGER. **International Conference on Human-Computer Interaction**. [*S. l.*], 2020. p. 57–69.

OSELEDETS, I. V. Tensor-train decomposition. **SIAM Journal on Scientific Computing**, SIAM, v. 33, n. 5, p. 2295–2317, 2011.

PANAGAKIS, Y.; KOSSAIFI, J.; CHRYSOS, G. G.; OLDFIELD, J.; NICOLAOU, M. A.; ANANDKUMAR, A.; ZAFEIRIOU, S. Tensor methods in computer vision and deep learning. **arXiv preprint arXiv:2107.03436**, 2021.

PEDREGOSA, F. e. a. Scikit-learn: Machine learning in Python. **Journal of Machine Learning Research**, v. 12, p. 2825–2830, 2011.

PEIXOTO, A. A. T. Theoretical and applied contributions on tensor learning. 2022.

PIZZOLATO, S.; TAGLIAPIETRA, L.; COGNOLATO, M.; REGGIANI, M.; MüLLER, H.; ATZORI, M. Comparison of six electromyography acquisition setups on hand movement classification tasks. **PLOS ONE**, Public Library of Science, v. 12, n. 10, p. 1–17, 10 2017. Disponível em: https://doi.org/10.1371/journal.pone.0186132.

PRADO, F. J. J. **Engenharia de Computação e Tecnologias Assistivas: Recursos de Acessibilidade Ao Computador para Pessoas Com Deficiência Motora**. Dissertação (Mestrado em Engenharia Elétrica e de Computação) – Universidade Federal do Ceará - Campus Sobral, Sobral, 2020.

RAWAT, S.; VATS, S.; KUMAR, P. Evaluating and exploring the myo armband. In: **2016 International Conference System Modeling Advancement in Research Trends (SMART)**. [*S. l.*: *s. n.*], 2016. p. 115–120.

SEMICONDUCTORS, P. The I^2C bus specification version 2.1. Philips Semiconductors, 2000.

SIGNORETTO, M.; LATHAUWER, L. D.; SUYKENS, J. A. A kernel-based framework to tensorial data analysis. **Neural networks**, Elsevier, v. 24, n. 8, p. 861–874, 2011.

STEGEMAN, A.; SIDIROPOULOS, N. D. On kruskal's uniqueness condition for the candecomp/parafac decomposition. **Linear Algebra and its applications**, Elsevier, v. 420, n. 2-3, p. 540–552, 2007.

TAHA, A. A.; HANBURY, A. Metrics for evaluating 3d medical image segmentation: analysis, selection, and tool. **BMC Med Imaging**, Aug 2015. ISSN 2634-1964.

TUCKER, L. R. Some mathematical notes on three-mode factor analysis. **Psychometrika**, Springer, v. 31, n. 3, p. 279–311, 1966.

WANG, T.; WEI, X.; JIA, L.; ZHAI, G. Weak fault detection of rail vehicle suspension system based on mpca. In: IEEE. **2017 29th Chinese Control And Decision Conference (CCDC)**. [*S. l.*], 2017. p. 2091–2096.

WEBSTER, J. G. **The Measurement Instrumentation and Sensor Handbook**. [*S. l.*]: CRC Press, 1999.

YANG, H.; LI, B.; LI, Y.; LI, H.; LI, S. A novel hybrid convolutional neural network for stock price prediction. In: IEEE. **2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)**. [*S. l.*], 2020. p. 2307–2312.

YUAN, M.; WEI, S.; ZHAO, J.; SUN, M. A systematic survey on human behavior recognition methods. **SN Computer Science**, Springer, v. 3, n. 1, p. 1–20, 2022.

YUDHANA, A. Flex sensors and mpu6050 sensors responses on smart glove for sign language translation. **IOP Conf. Ser.: Mater. Sci.Eng. 403**, 2019.

ZHANG, X. . e. a. A framework for hand gesture recognition based on accelerometer and emg sensors. **IEEE Transactions on Systems, Man, and Cybernetics- Part A: Systems and Humans**, v. 41, n. 6, p. 1064–1076, 2011.

ZHAO, Q.; ZHOU, G.; ADALI, T.; ZHANG, L.; CICHOCKI, A. Kernel-based tensor partial least squares for reconstruction of limb movements. In: IEEE. **2013 IEEE International Conference on Acoustics, Speech and Signal Processing**. [*S. l.*], 2013. p. 3577–3581.

ZHAO, Q.; ZHOU, G.; ADALI, T.; ZHANG, L.; CICHOCKI, A. Kernelization of tensor-based models for multiway data analysis: Processing of multidimensional structured data. **IEEE Signal Processing Magazine**, IEEE, v. 30, n. 4, p. 137–148, 2013.

APÊNDICE A – ARTIGO COM COLABORAÇÃO DO AUTOR

Classification of Hand Movements from EMG Signals for People with Motor Disabilities

Francisco J. Prado Júnior, Flávio V. dos Santos and C. Alexandre R. Fernandes

***Resumo*—People with disabilities correspond to about 25% of the Brazilian population. A great part of these people have physical impairments that difficult the use computer peripherals. This article presents the development of a system for detection of hand movements through the acquisition and classification of electromyographic (EMG) signals using machine learning techniques. The purpose of the proposed system is to be used by people with disabilities to control an adapted text editor. The signals are capture by surface EMG electrodes and used to the detect 4 different hand movements. In addition, a database with 3200 EMG signals generated by the hand movements was created, made by one user diagnosed with cerebral palsy and another user without diagnosed motor disabilities. Several tests were carried out, showing the good accuracy of the proposed system, with a success classification rate of 96% to 98%.**

***Index Terms*—EMG signals, assistive technology, human-machine interface, machine learning, text editor.**

I. INTRODUÇÃO

UM ser humano é considerado como Pessoa Com Deficiência (PcD) quando possui algum impedimento de longo prazo, podendo ser de natureza física, mental, intelectual ou sensorial, fazendo com que essa pessoa seja impedida de exercer atividades cotidianas, impossibilitando assim uma vida social em igualdade com as demais pessoas [1]. Segundo dados da Organização Mundial da Saúde (OMS), a população mundial contém cerca de 14% de pessoas com algum tipo de deficiência, com 80% destas pessoas vivendo em países subdesenvolvidos, sendo em sua maioria crianças e adolescentes com menos de 18 anos [2]. Só no Brasil, estima-se cerca de 46 milhões de PcD, o que equivale a 25% de sua população. Dessas, 38,8% conseguem algum grau de escolaridade básica e apenas 6,7% conseguem o atingir o nível superior [3].

As dificuldades que as PcD encontram em suas vidas podem ser caracterizadas como tudo aquilo que impõe limites ou impeça a sua participação na sociedade e o uso de seus direitos. Dentre as principais barreiras encontradas pelas PcD, listadas no Estatuto da PcD [3], se encontram as "Barreiras tecnológicas", que estão relacionadas às dificuldades ou impedimentos ao acesso da pessoa com deficiência às tecnologias. Todas essas barreiras, atreladas às dificuldades físicas e mentais das pessoas com deficiência, geram a exclusão social das mesmas [3].

Neste contexto, as chamadas Tecnologias Assistivas (TAs) [4], [5] possuem um papel fundamental. As TAs são um conjunto de serviços e ferramentas para auxiliar às PcD e, em especial, a categoria "Recursos de acessibilidade ao computador" tem possibilitado o uso de computadores na Educação Inclusiva (EI). De fato, para que o computador seja usado adequadamente na educação inclusiva, é necessário que haja uma Interface Homem-Máquina (IHM) que possibilite ao usuário com deficiência física motora nos membros superiores controlar suas ferramentas sem maiores dificuldades.

Muitos pesquisadores vêm trabalhando para desenvolver soluções inovadoras para o uso do computador pelas PcD, desde a confecção de periféricos, como mouses e teclados adaptados [6], até o uso de técnicas de aprendizagem de máquina (AM) para a classificação de sinais biológicos (sinais produzidos pelo corpo humano) para a realização do controle de máquinas. Um exemplo de sinal biológico são os sinais eletromiográficos (EMG), que estão sendo constantemente usados para a criação de novas interfaces com diversas utilizações, como controle de próteses e cadeiras de rodas motorizadas [7]

A proposta deste trabalho é desenvolver um sistema para detecção de movimentos das mãos através da aquisição e classificação de sinais biológicos EMG usando técnicas de AM. O objetivo do sistema proposto é ser usado por PcD para controlar um editor de texto adaptado. Os sinais são capturados por um sistema de aquisição que possui 2 pares de eletrodos de EMG de superfície, duas placas de aquisição *open source* (Shield EKG/EMG Olimex) e uma placa Arduino. Algumas técnicas de AM são usadas para a classifiação dos sinais EMG correspondentes a quatro movimentos diferentes da mão. São testados 3 diferentes classificadores, em conjunto com a técnica Análise de Componentes Principais (PCA): Máquina de Vetores de Suporte (SVM), *K-Nearest Neighbors* (KNN) e *Multilayer Perceptron* (MLP).

Como atributos para o vetor de características, foram testadas diversas combinações de atributos envolvendo Transformada de Fourier (TF), Transformada de *Wavelet* (TW) e parâmetros diversos no domínio do tempo e frequência. Vale a pena mencionar que foi desenvolvido um editor de texto adaptado, cuja funcionalidade dispensa o uso do teclado convencional, para que o usuário consiga digitar palavras usando apenas o sistema de detecção de movimentos desenvolvido. O editor de texto necessita apenas de três movimentos de mão para funcionar. Desta forma, pode-se escolher os 3 movimentos que forneceram os melhores resultados, dentre os 4 movimentos detectados pelo sistema proposto.

Ao final do trabalho, são apresentados os resultados obtidos

F. J. P. Junior is with the Electrical and Computer Engineering Graduate Program, Universidade Federal do Ceará, Ceará, Brasil, e-mail: juniorcvnnn@gmail.com

F. V. dos Santos and C. A. R. Fernandes are with the Department Computer Engineering, Universidade Federal do Ceará, Ceará, Brasil, e-mail: flaviotub@hotmail.com, alexandrefernandes@ufc.br.

Os autores deste trabalho gostariam de agradecer à Fundação Cearense de Apoio ao Desenvolvimento Científico e Tecnológico (FUNCAP/CE/Brasil) e ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq/Brasil - processo n. 304053/2016-3) pelo financiamento parcial deste trabalho.

na realização dos testes com o objetivo de medir da eficiência do sistema desenvolvido no reconhecimento dos movimentos realizados pelos usuários. Para este fim, foi construído um banco de dados com 3200 sinais EMG emitidos pelos músculos do antebraço produzidos pela execução de movimentos de mão de dois usuários, sendo um dos usuários uma PcD com deficiência motora nos braços e um sem deficiência diagnosticada. Os testes apresentados são divididos em duas partes: testes *offline*, realizados com a base de dados (fases de treinamento e validação), e testes *online*, realizados com sinais não contidos na base de dados, processados em tempo real (fase de testes).

Pode-se resumir as contribuições e vantagens do sistema proposto no presente trabalho da seguinte forma:

- Baixo custo, pois utiliza placas de aquisição de baixo custo e *open source*, bem como eletrodos reutilizáveis e cabos de baixo custo.
- Não invasivo, pois usa apenas sensores EMG de superfície em braceletes elásticos, sem necessidade de uso de introdução de agulhas no músculo do usuário.
- Integração ao editor de texto adaptado.
- Testes realizados em tempo real, assim como usando amplo banco de dados com 3 diferentes classificadores e diversas configurações de atributos.
- Boa acurácia.

O restante do presente artigo se divide nas seguintes partes. Na Seção II, é apresentada uma revisão do estado da arte relacionada à temática deste artigo. Na Seção III, será detalhado o sistema de detecção de movimentos de mão proposto. Na Seção IV, é apresentado o banco de dados de amostras de sinais EMG. Na Seção IV, são fornecidos os resultados dos testes realizados, enquanto na Seção VI são apresentadas as conclusões e perspectivas do presente trabalho.

II. Trabalhos Relacionados

A seguir, serão apresentados trabalhos que utilizam sinais biológicos e técnicas de AM para o desenvolvimento de periféricos de acesso ao computador acessível às PcDs.

Em [8], propõe-se uma interface voltada para auxiliar pessoas com paralisia cerebral (PC). A interface baseia-se em *Emotiv Epoc* e usa sinais cerebrais captados por um "capacete" posto na cabeça do usuário. Neste trabalho, é proposto um *software* que foi desenvolvido para interagir com impulsos cerebrais. Uma das metas abordadas foi a análise emocional do usuário.

Já em [9], avalia-se o controle de próteses de pernas em pacientes amputados. Para isso, é utilizada a classificação de sinais EMG produzidos pela movimentação do joelho do usuário. Para a classificação, foi utilizado como classificador uma rede neural MLP, cujos dados de treino foram extraídos de uma pessoa sadia enquanto caminhava. A rede já treinada recebe sinais do usuário real (pessoa amputada) e efetua o controle de sua prótese.

No trabalho [10], avalia-se o uso de um dispositivo de controle chamado MYO, composto por um bracelete com 8 pares de eletrodos, um transmissor sem fio e uma central de processamento. O dispositivo é colocado no braço do usuário e reconhece três tipos de movimentos distintos. Esse mesmo dispositivo (MYO) foi utilizado no artigo [11], em que se registra a classificação de 6 movimentos de mão utilizando redes neurais e TW. A proposta é que a classificação desses sinais sirva para o controle de robôs.

Em [12], propôs-se uma IHM para auxílio de pessoas com membros superiores amputados no controle de próteses mecânicas. A IHM usa sinais EMG captados por 8 pares de eletrodos bipolares. Para a captação também usou-se o MYO posicionado na região do cotoco (extremidade do membro amputado). Foi utilizado o *Linear Discriminant Analysis* (LDA) como classificador para reconhecimento de 5 movimentos simples de braço, sendo 82,37% a melhor taxa de acerto encontrada.

Em [13], observa-se o uso de MLP e *Convolutional Neural Network* (CNN) para a classificação de sinais EMG e EEG. Nesse trabalho avalia-se o uso de treino cruzado da CNN usando a resposta do treino dos sinais EMG para o EEG e vise-versa, obteve-se nesse experimento uma taxa máxima de acerto de 93,82% de acerto para os sinais EEG e 85,12% para os sinais EMG.

O trabalho [14] apresenta uma IHM baseada em sinais EMG faciais (fEMG) e eletroencefalografia (EEG). Os sinais fEMG foram captados através de dois pares de eletrodos bipolares posicionados nos músculos laterais faciais. Já os sinais EEG foram captados por um par de eletrodos posicionados na região superior da cabeça do usuário. Nesse trabalho, conseguiu-se a classificação de 4 movimentos faciais com uma taxa máxima de acertos de 76,7%, além da detecção de sinais EEG com máxima precisão de 84,5%.

Em [15], utilizou-se uma rede neural linear com sete neurônios para classificação de sete movimentos de mão. Uma das diferenças deste trabalho em relação a [9] é o fato de, além do uso de sinais EMG, também serem usados sinais mecanomiográficos (MMG), que são sinais sonoros produzidos pelas unidades motoras durante a contração muscular. Para os sinais MMG, foram utilizados três microfones sensoriais, ambos posicionados no músculo inferior do antebraço.

Já em [16], destaca-se uma interface utilizando um *personal digital assistant* (PDA) para controle de uma cadeira de rodas motorizada. O controle se dá por meio de um apontador de cabeça e do auxílio de sinais EMG, EEG, *electrooculography* (EOG - sinais elétricos produzidos pelo movimento dos olhos) e vídeo-oculografia (VOG), sendo os dois primeiros captados por eletrodos de superfície e os dois últimos por meio de uma câmera. Além da movimentação da cadeira, o PDA também pode indicar expressões particulares do usuário.

No artigo [7], foi proposta uma IHM baseada em sinais EMG e EOG para controle de um editor de texto. O sinal EMG é captado através de um dispositivo semelhante a uma tiara que fica posicionada na região superior da cabeça e capta o sinal EMG facial. Já o sinal EOG é captado através de 5 eletrodos de superfície localizados próximo aos olhos. No editor proposto, as letras ficam dispostas na tela do computador e, ao detectar um sinal EOG, o sistema troca a seleção de caractere, enquanto que, ao detectar o sinal EMG, ele escreve o caractere selecionado.

Em [17], propõe-se o uso de sinais EMG para o controle de

um robô de monitoramento doméstico. O sinal EMG é extraído do antebraço por um par de eletrodos e, após a captação e processamento, o sinal é classificado com o auxílio de uma rede neural artificial, para então ser convertido em comandos de ações do robô.

No artigo [18], destaca-se a utilização de SVM para a classificação de sete movimentos de mão. O sinal classificado foi captado por quatro eletrodos de superfície localizados próximos à articulação do antebraço. Também em [19], avalia-se a eficácia do algoritmo SVM para classificar sinais EMG de seis movimentos de braço, para controle de um braço mecânico. Os sinais obtidos para esse experimento foram extraídos do bíceps e antebraço, utilizando seis pares de eletrodos de superfície.

Em [20], é feito um comparativo entre o SVM e o KNN na classificação de sinais EMG. Para isso, foi utilizada uma base de dados retirada de UCI *Machine Learning Repository*. Essa base de dados contém sinais de seis movimentos de mão, captados através de um sistema de dois canais de aquisição. Constatou-se nesse estudo que o classificador SVM apresentou maior acurácia em relação ao classificador KNN.

Em [21], foi apresentada a utilização dos sinais EOG para controle das funcionalidades do mouse "mover o cursor" e "realizar cliques". Utilizam-se quatro eletrodos de captação e um eletrodo de referência para a captação dos sinais. Para a classificação foram utilizados valores de limiarização, de acordo com o potencial captado pelos eletrodos.

III. Sistema de Detecção de Movimentos de Mão

Nesta seção, é apresentado o sistema para detecção de movimentos das mãos proposto, que foi pensado e desenvolvido com o objetivo de oferecer uma nova opção de controle do computador pelas PcD motora.

A. Visão Geral

O sistema proposto neste trabalho foi desenvolvido para amenizar as dificuldades que as PcD motora nos membros superiores possuem para manusear o *mouse* e teclado convencionais. Pensou-se então em um dispositivo que utilize a classificação de sinais EMG com eletrodos de superfície para o controle do computador, aplicando-se apenas movimentos de mão para seu funcionamento. O processo de funcionamento deste sistema se divide em diversas etapas, que são mostradas na Fig. 1 e explicadas na sequência.

B. Sistema de aquisição e pré-processamento

Foram utilizadas duas placas de aquisição *Olimex Shield EKG/EMG*. Trata-se de uma placa opensource de aquisição de sinais de eletrocardiograma (ECG) e EMG. Esta placa possui um sistema de blindagem eletromagnética que diminui consideravelmente a interferência de sinais ruidosos externos e possui um potenciômetro regulável, calibrado para sinais ECG/EMG e compatível com várias versões do Arduino 3V e 5V, incluindo o Arduino UNO, que é utilizado na aquisição do sinal [22].

Os sinais EMG foram captados usando dois pares de eletrodos passivos bipolares superficiais *Shield-EKG-EMG-PA*,

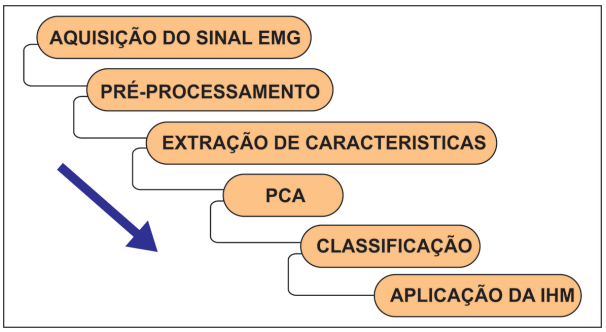


Figura 1. Etapas do sistema para detecção de movimentos das mão

ilustrados na Fig. 2. Além dos dois pares de eletrodos de captação, usaram-se ainda dois eletrodos de referência. Optou-se por esse tipo de eletrodo pelo fato de usarem braceletes elásticos para fixar os eletrodos nos pontos desejáveis ao invés de gel condutores, possibilitando a sua reutilização.

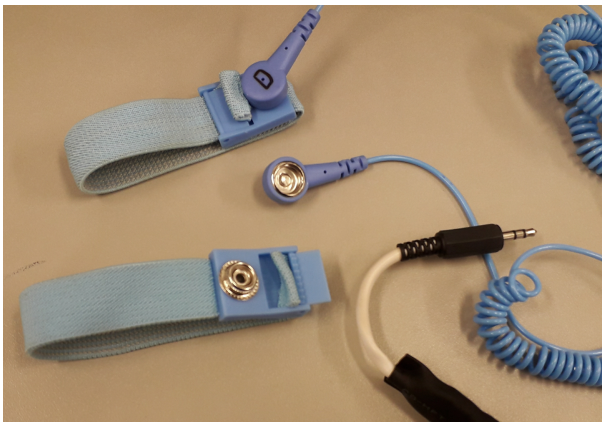


Figura 2. Eletrodos passivos bipolares superficiais.

Os eletrodos de captação foram posicionados nas extremidades dos músculos superiores e inferiores do antebraço direito. Através de estudos e testes prévios descobriu-se que esses eram os músculos que mais reagiam aos movimentos escolhidos. Já os eletrodos de referência foram colocados próximos ao tornozelo da perna esquerda, região indicada pelo fabricante da placa de aquisição.

Cada par de eletrodos foi ligado a uma das duas placas de aquisição, captando simultaneamente dois sinais: canal 1 (Ch1) e canal 2 (Ch2), em que Ch1 capta a diferença de potencial causada pela contração/relaxamento do músculo superior do antebraço, e o Ch2 representa a diferença de potencial causada pela contração/relaxamento do músculo inferior do antebraço. Na Fig. 3, é ilustrado o esquema eletrodos-placas de aquisição, com a disposição dos eletrodos na parte superior e inferior do mesmo antebraço, enquanto na Fig. 4 é mostrada uma foto do posicionamento dos eletrodos no antebraço esquerdo.

As placas de aquisição *Olimex Shield EKG/EMG* recebem os sinais obtidos pelos eletrodos e realizam um pré-processamento destes sinais, através de operações analógicas de amplificação e filtragem. A placa de aquisição envia os

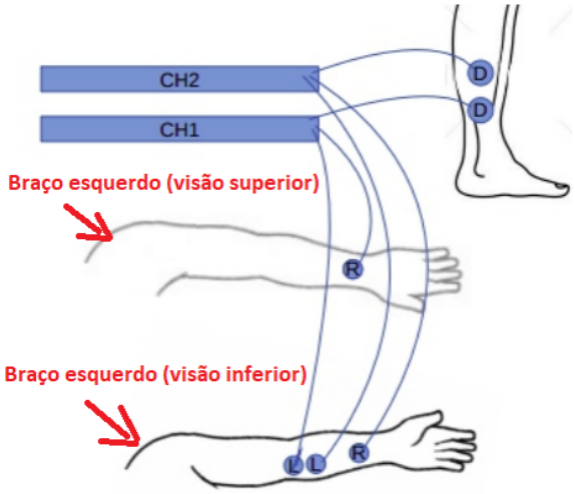


Figura 3. Esquema eletrodos-placas de aquisição.

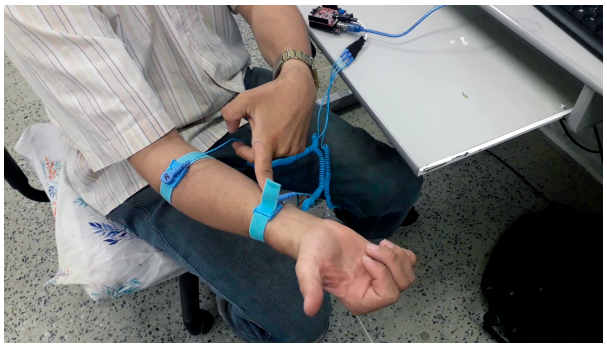


Figura 4. Posicionamento dos eletrodos.

dados ao Arduino que realiza a transferência do sinal pré-processado para o computador.

Na Fig. 5, é mostrada a representação gráfica das amostras de sinal do movimento "Fechado" captadas pelos dois canais no domínio do tempo. O movimento "Fechado" está ilustrado na Fig. 6.b. Os quatro movimentos usados na base de dados serão detalhados na Seção V.

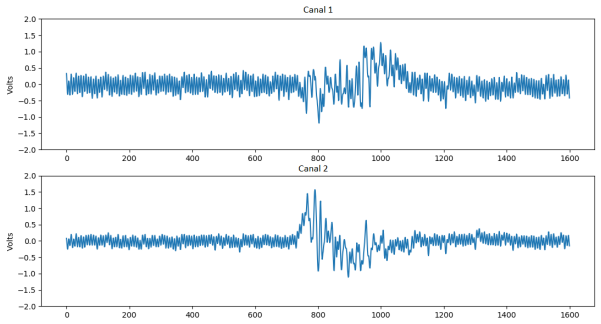


Figura 5. Amostra de sinais captado pelo movimento "Fechado" no domínio do tempo.

Na Fig. 7, é mostrado o módulo da TF dos sinais referente à Fig. 5. Como se pode observar por estas figuras, o movimento

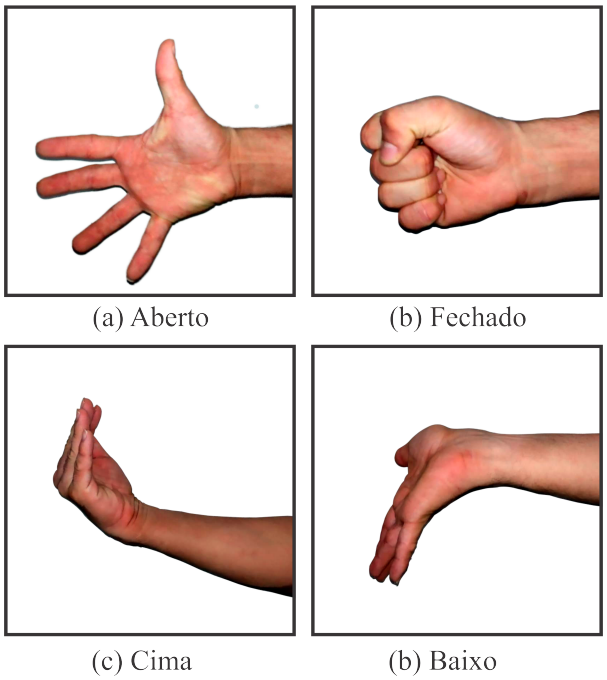


Figura 6. Movimentos de mão escolhidos.

"Fechado" excita tanto os músculos da região superior do antebraço, captados pelo Ch1, quanto os músculos da região inferior, captados pelo Ch2, com maior intensidade na região inferior.

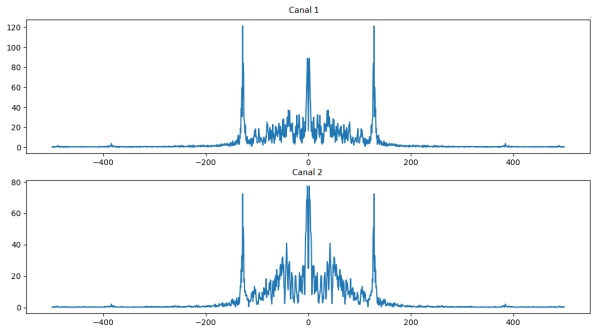


Figura 7. Amostra de sinais captado pelo movimento "Fechado" no domínio da frequência.

C. *Extração de características*

A primeira etapa do processo de classificação dos sinais EMG é a extração de características das amostras obtidas. Determinou-se 4 grupos de atributos que seriam extraídos de cada amostra para teste, são eles:

- Grupo 1 (G1): Nesse grupo, calculou-se o desvio padrão janelado da TF. Neste caso, a TF é dividida em vinte partes iguais e não sobrepostas, para então se calcular o desvio padrão do módulo de cada um desses intervalos. Extraíram-se desse grupo 40 atributos (20 atributos por canal).

- Grupo 2 (G2): Usou-se nesse grupo a TW. Selecionaram-se as baixas frequências da decomposição de quarta ordem da TW, gerando-se desse procedimento, 200 atributos (100 para cada canal).
- Grupo 3 (G3): Nesse grupo, a extração dos atributos deu-se de maneira semelhante ao grupo anterior, a diferença é que usaram-se os valores de baixa frequência da sexta decomposição da TW, extraindo 50 atributos (25 por canal).
- Grupo 4 (G4): Nesse grupo, utilizaram-se diversas operações para a extração dos atributos, são elas: média do sinal (tempo e frequência), valor máximo do sinal (tempo e frequência), assimetria (tempo e frequência), *zero cross rating* (tempo), curtose (tempo e frequência) e componente de frequência em qual a maior amplitude ocorre (frequência). No total, extraiu-se 20 atributos (10 por canal).

Como será detalhado na Seção V, durante as fases de treinamento e validação, foram realizados diversos testes com diferentes combinações destes atributos para se escolher o melhor vetor de características.

D. Análise de Componentes Principais (PCA)

A alta dimensionalidade do vetor de características acima descrito pode causar uma dispersão dos dados e o conhecido problema da "maldição da dimensionalidade". Para evitar esse problema, a técnica PCA é usada para redução de dimensionalidade. Além de evitar esses problemas, o PCA também diminui o tempo de processamento. Durante as fases de treinamento e validação, foram realizados diversos testes com diferentes números de componentes do PCA para se avaliar o melhor valor para este parâmetro.

E. Classificador

A última etapa do sistema de detecção de movimentos de mão proposto é o algoritmo de classificação. Para isso, escolheu-se trabalhar com 3 classificadores: KNN, MLP e SVM. Esses classificadores já foram testados no contexto de sinais EMG, tal como mencionado na Seção II [20], [9], [13], [18], [19].

Como será visto na Seção V, a técnica SVM proporcionou os melhores resultados. Por este motivo, a maior parte dos resultados da simulação foram gerados usando este método. O classificador SVM realiza a separação das classes por meio de hiperplanos que são otimizados para gerar a maior distância possível entre as classes. Várias simulações foram realizadas a fim de comparar diferentes *kernels* de SVM e parâmetros de penalidade.

F. Editor de Texto Adaptado

Além dos problemas relacionados ao uso do mouse, muitas PcD motora nos membros superiores possuem dificuldades para usar o teclado convencional, por conta da localização de algumas teclas e também pela exatidão necessária em pressionar a tecla com o caractere desejado. Como aplicação do sistema de detecção de movimentos de mão, optou-se então pelo desenvolvimento de um editor de texto adaptado. O editor possui uma interface simples semelhante a um editor de texto convencional, sendo desenvolvido na linguagem Java. Neste editor de texto, no lugar do teclado convencional para a escrita de caracteres, utiliza-se 3 movimentos de mão para tal ação, executando 3 comandos: passar caractere, escrever caractere e deletar caractere. Definiu-se seu funcionamento da seguinte maneira:

- Passar caractere: passa as letras em ordem alfabética e é ativado pelo movimento "Aberto" (Fig. 6.a).
- Escrever caractere: fixa o caractere escolhido e é ativado pelo movimento "Cima" (Fig. 6.c).
- Apagar caractere: Apaga ultimo caractere escrito e é ativado pelo movimento "Baixo" (Fig. 6.d).

Como será detalhado na Seção V, foram utilizados os 3 movimentos do banco de dados que obtiveram melhores taxas de acerto na classificação, sendo descartado o movimento com pior acurácia. O desenvolvimento e validação deste editor de texto adaptado será detalhado em um trabalho futuro.

IV. Banco De Dados

Para treinamento e validação dos classificadores, foi contruído um banco de dados com um grande número de amostras de sinais EMG relativos a 4 tipos de movimentos de mão movimentos feitos por uma PcD e uma pessoa sem deficiência. Cada movimento é descrito na Tabela I, enquanto na Fig. 6 são ilustrados os 4 movimentos. Deve-se considerar que a posição de repouso, posição inicial para cada movimento, é a mão relaxada com a palma da mão apontada para cima.

Tabela I
Descrição dos movimentos de mão

Movimento	Descrição	Representação
Aberto	Extensão dos dedos e abertura da palma da mão.	Fig. 6.a
Fechado	Fechamento total do punho, fazendo pressão com os dedos.	Fig. 6.b
Cima	Flexão de punho para cima e extensão dos dedos para cima.	Fig. 6.c
Baixo	Flexão de punho para baixo e extensão dos dedos para baixo	Fig. 6.d

A coleta inicia-se com a realização de um dos movimentos mostrados na Fig. 6. Quando é realizado o movimento, acontece uma contração do músculo superior e inferior do antebraço gerando um pico de sinal EMG. A duração das contrações e relaxamentos é de aproximadamente 1,6 segundo para cada movimento realizado. Quando um pico de tensão é detectado, uma amostra do sinal com 3200 valores é escrita em um arquivo de texto utilizando o pico como centro do sinal (1599 amostras antes e 1600 depois), com taxa de transferência de 1kHz. O número de valores de cada amostra é equivalente aos valores captados por Ch1 e Ch2 durante o intervalo de 1,6 segundos. Após a captação dos valores recebidos, é escrito o rótulo do movimento, um valor inteiro de 1 a 4 representando cada um dos movimentos.

Foram coletadas amostras de 2 participantes: Participante 1, diagnosticado com PC e primeiro autor deste artigo, e Participante 2, sem deficiências motoras diagnosticadas. O Particiante 1 possui o seguinte perfil: sexo mascuino, 30 anos, escolaridade superior completo, com diagnóstico de PC [1]. Já o Participante 2 possui seguinte perfil: sexo masculino, 25 anos, escolaridade superior icompleto, sem deficiências motoras. Cada participante realizou um total de 1600 movimentos, sendo 400 amostras de cada movimento, gerando um banco de dados com 3200 amostras. A base de dados está disponibilizada no repositório "ZENODO", através do *link*: https://zenodo.org/record/3834919#.Xxuqy55KjIU.

V. Resultados

Nesta seção são apresentados os resultados dos testes realizados para a classificação das amostras coletadas. Os resultados apresentados são divididos em duas partes: resultados *offline*, realizados com a base de dados (fases de treinamento e validação), e resultados *online*, realizados com sinais não contidos na base de dados e processados em tempo real (fase de testes).

A. Resultado dos testes com a base de dados

Os primeiros resultados apresentados possuem o objetivo de avaliar o desempenho de 3 diferentes classificadores (SVM, MLP e KNN). Para tanto, foram realizadas algumas simulações preliminares com o intuito se definir uma configuração base para os testes dos classificadores. Os resultados destas simulações preliminares, omitidos por simplicidade, mostraram que os atributos do grupo G1 (desvio padrão janelado da TF) e o PCA com 35 componentes forneceram bons resultados para todos os classificadores. Desta forma, para a padronização dos resultados dos classificadores, esta configuração de atributos e PCA foi utilizada. Ademais, nesta etapa, foi usado o método de validação cruzada *k-Fold*, com $k = 3$ (fases de treinamento e validação).

Para cada classificador, foram realizados vários testes para se encontrar os melhores parâmetros de cada técnica. No caso do SVM, testaram-se os *kernels* linear, polinomial, Gaussiano (RBF - *radial basis function*) e sigmoide, variou-se o parâmetro do *kernel*, a constante de relaxamento e a técnica usada para caso multiusuário (*one-vs-one* e *one-vs-all*). No caso do MLP, variou-se a função de ativação, o número de camadas e o número de neurônios por camada. Já no caso do KNN, variou-se o parâmetro *k*. Além disso, no KNN, usou-se a técnica *k-d tree* para melhorar o desempenho do classificador.

Os parâmetros que forneceram os melhores resultados de acurácia para cada classificador são mostrados na Tabela II, para o Participante 1 (com deficiência), e na Tabela III, para o Participante 2 (sem deficiência), enquanto as correspondentes taxas de acerto são apresentadas na Tabela IV, para o melhor caso de cada classificador.

Pode-se perceber pela Tabela IV que a técnica SVM forneceu a melhor taxa de acerto para ambos os participantes. Deve-se destacar ainda que se obteve uma melhor taxa de acerto com o Participante 2 (93,13%) do que com o Participante 1 (89,55%). Isto se deve ao fato de o participante 1 ter espasmos musculares involuntários constantes, o que afeta as características extraídas. Desta forma, usando como critério de seleção a taxa de acerto, escolheu-se o classificador SVM para realizar o restante dos testes.

[1]CID10: G80 - Paralisia cerebral e F82 - Transtorno específico do desenvolvimento motor

Tabela II
Parâmetros de configuração usados para os classificadores (Participante 1)

Movimento	Parâmetros
SVM	*kernel* Gausiano, a = 0; 25, C = 7 e *one-vs-one*
MLP	2 camadas ocultas, função de ativação sigmoide e 4 neurônios por camada.
KNN	K = 3 (com *k-d tree*)

Tabela III
Parâmetros de configuração usados para os classificadores (Participante 2)

Movimento	Parâmetros
SVM	*kernel* Gausiano, a = 0;00055, C = 10 e one-vs-one
MLP	2 camadas ocultas, função de ativação sigmoide e 4 neurônios por camada.
KNN	K = 3 (com *k-d tree*)

Tabela IV
Taxa de acerto dos classificadores

Movimento	Participante 1	Participante 2
SVM	89,55%	93,13%
MLP	79,1%	73,44%
KNN	72,48%	72,39%

Os próximos resultados avaliam quais os melhores atributos a serem utilizados, bem como o melhor valor para o número de componentes do PCA, com o SVM. Foram testadas as seguintes 11 combinações de atributos: G1, G2, G3, G4, G1-G2, G1-G3, G1-G4, G2-G4, G3-G4, G1-G2-G4 e G1-G3-G4. Para cada combinação de atributos, foram testados diversos valores do número de componentes do PCA, para ambos os participantes.

São mostrados na Tabela V os melhores resultados dos testes realizados para as amostras dos dois participantes. Em cada tabela, mostrou-se as 3 maiores taxas de acerto obtidas quando se variou o número de componentes do PCA.

Tabela V
Taxas de acerto nos melhores casos

Participante 1		Participante 2	
Grupo 2 e 4 (70 atributos)		Grupo 1 e 2 (240 atributos)	
Nº comp. PCA	Taxa	Nº comp. PCA	Taxa
10	91,10%	10	91,37%
20	95,38%	20	95,44%
30	97,12%	30	96,06%

A maior taxa de acerto encontrada para o Participante 1

foi de 97,12%, com 30 componentes de PCA e os atributos dos Grupos 2 e 4, enquanto que a maior taxa de acerto do Participante 2 foi 96,06%, com 30 componentes de PCA e com os atributos dos Grupos 1 e 2. Percebe-se que a taxa de acerto máxima do Participante 1 é ligeiramente maior do que a do Participante 2. Isto mostra que o ajuste fino dos atributos e do número de componentes do PCA conseguiu cancelar o efeito negativo que os espasmos musculares do Participante 1 causaram nas primeiras simulações. Ademais, pode-se observar que ambos os participantes obtiveram taxas de acerto bastante elevadas, acima de 96%.

São mostradas nas Tabelas VI e VII as tabelas de confusão com os números de acertos e erros de cada movimento, para os melhores casos dos Participantes 1 e 2, respectivamente. Ademais, nas Tabelas VI e VII, são mostradas as taxas de acerto para cada classe de movimento, para ambos os participantes. Pode-se concluir a partir destas tabelas que, para os dois participantes, o movimento "Fechar" obteve o pior desempenho, com 94% de acerto. Pode-se ainda observar que o pior desempenho deste movimento se deu pelo fato de estar sendo confundido com o movimento "Baixo", no caso do Participante 1, e com o movimento "Aberto" do Participante 2.

Para efeitos de comparação, as taxas de acerto alcançadas em [12] e [13], que também realizam classificação de movimentos usando sinais EMG, foram 82,37 % e 88,55 %, respectivamente. Pode-se então deduzir que a taxa de acerto encontrada no presente trabalho é satisfatória, pois é significativamente superior às encontradas em [12] e [13], mesmo que utilizando um número menor de eletrodos.

Tabela VI
TABELA DE CONFUSÃO PARA O MELHOR RESULTADO (PARTICIPANTE 1)

	Aberto	Baixo	Cima	Fechar	Taxa de acerto
Aberto	395	3	0	2	98,75%
Baixo	1	393	1	5	98,25%
Cima	2	1	390	7	97,50%
Fechar	4	13	7	376	94,00%
Média					97,12%

Tabela VII
TABELA DE CONFUSÃO PARA O MELHOR RESULTADO (PARTICIPANTE 2)

	Aberto	Baixo	Cima	Fechar	Taxa de acerto
Aberto	383	1	1	15	95,75%
Baixo	3	389	2	6	97,25%
Cima	1	6	389	4	97,25%
Fechar	15	1	8	376	94,00%
Média					96,06%

B. Resultado dos testes de classificação em tempo real

Para os testes em tempo real (*online*), ou seja, com dados processados em tempo real, foram usados sinais não contidos na base de dados (fase de testes). Cada um dos 2 participantes realizou 400 movimentos, com 100 repetições para cada movimento apresentado na Fig. 6. Usou-se o mesmo classificador e a mesma configuração de atributos e PCA do caso com melhor taxa de acerto na fase de valiação (usando a base de dados), tanto para as amostras do participante 1 quanto para as amostras do participante 2, ou seja, foram usadas a mesmas configurações utilizadas nas Tabelas VI e VII, respectivamente.

Nas Tabelas VIII e IX, são mostradas as tabelas de confusão com os números de acertos e erros de cada movimento nos testes *online* dos dois participantes, juntamente com as taxas de acerto para cada movimento. A partir destas tabelas, pode-se observar que, tal como nos testes com a base de dados, o movimento "Fechar" obteve o pior desempenho, para os dois participantes. Deve-se ainda ressaltar que as taxas de acerto médias obtidas nos testes em tempo real (81,75% e 82,75%) é significativamente menor do que aquelas obtidas com a base de dados. Este resultado é esperado, visto que os classificadores, o PCA e os atributos não foram configurados usando estes sinais. Ademais, vale ressaltar que o tempo de resposta para cada comando ficou entre 0,9 s e 1,8 s.

Tabela VIII
TABELA DE CONFUSÃO PARA O TESTE EM TEMPO REAL (PARTICIPANTE 1)

	Aberto	Baixo	Cima	Fechar	Taxa de acerto
Aberto	89	0	3	8	89,00%
Baixo	4	79	5	12	79,00%
Cima	2	1	86	11	86,00%
Fechar	5	10	12	73	73,00 %
Média					81,75%

Tabela IX
TABELA DE CONFUSÃO PARA O TESTE EM TEMPO REAL (PARTICIPANTE 2))

	Aberto	Baixo	Cima	Fechar	Taxa de acerto
Aberto	84	2	0	14	84,00%
Baixo	2	87	1	10	87,00%
Cima	1	1	88	10	88,00%
Fechar	9	12	7	72	72,00%
Média					82,75%

É válido ainda mencionar, como justificativa de uma possível aplicação, que, durante os testes em tempo real, o sistema de detecção de movimentos estava integrado ao editor de texto adaptado, com os movimentos "Aberto", "Cima" e "Baixo" sendo escolhidos devido às suas taxas de acerto serem maiores do que o movimento "Fechado". Foram feitos ainda, para efeitos de visualização, pequenos testes que resumem-se em escrever palavras curtas, escolhidas de forma aleatória, visto que o objetivo era apenas testar a viabilidade da sistema para controle de um editor de texto, substituindo o uso do teclado convencional por movimentos de mão. Ambos os participantes lograram sucesso em escrever as palavras com movimentos de mão, sem grandes dificuldades.

VI. CONCLUSÃO

Neste trabalho, foi apresentado um sistema para detecção de

movimentos de mãos usando sinais EMG e técnicas de AM, possibilitando a classificação de até 4 movimentos de mão. O objetivo do sistema proposto é ser usado como periférico de acesso ao computador por PcD para controlar um editor de texto adaptado. Foram realizados testes com um banco de dados com 3200 sinais EMG, bem como testes em tempo real, com um usuário diagnosticado com PC e outro usuário sem deficiências motoras diagnosticadas, usando atributos no domínio do tempo, frequência e TW.

Os testes mostraram que o sistema de classificação proposto forneceu boas taxas de acerto, com o classificador SVM obtendo o melhor resultado, sendo usado juntamente com a técnica PCA. Nos testes com a base de dados, foi obtida uma acurácia de 96% a 97%, enquanto que os teste em tempo real forneceram uma acurácia de 82% a 83%.

Em trabalhos futuros, pretende-se incrementar o sistema de aquisição de sinais usando acelerômetros e giroscópios, bem como serão realizados testes com um número maior de usuários com PC. Serão também testadas técnicas de aprendizagem profunda (*deep learning*) para a classificação dos movimentos. Ademais, serão feitas melhorias no editor de textos, tais como a introdução de um dicionário de palavras e um sistema de auto completude, bem como será feito um estudo envolvendo o desenvolvimento e a validação do editor de texto.

REFERÊNCIAS

[1] C. d. D. Brasil, Lei no 13.146, de 6 de julho de 2015. institui a lei brasileira de inclusão da pessoa com deficiência (estatuto da pessoa com deficiência). Diário Oficial da União, 2015.

[2] W. H. Organization, et al. "World report on disability 2011". [S.l.]: *World Health Organization*, 2011.

[3] I. B. G. E.Brasil, "Instituto Brasileiro de geografia e Estatística". *Censo demográfico*, v. 2010, 2010.

[4] M. R. C. Maciel, "Portadores de deficiência: a questão da inclusão social". *São Paulo em perspectiva, SciELO Brasil*, v. 14, n. 2, p. 51–56, 2000.

[5] Hussey, S. M.; Cook, A. M. "Assistive Technologies: Principles and Practice". St. Louis, MO: Mosby. Year Book, Inc, 1995.

[6] C. R. C. da Silva, A. M. Torres, J. L. T. Nogueira and L. C. S. Motta, Hardware adaptado para apoio ao desenvolvimento de portadores de síndrome de down. 2007.

[7] H. S. Dhillon, R. Singla, N. S. Rekhi and R. Jha, "EOG and EMG based virtual keyboard: A brain-computer interface," 2009, *2nd IEEE International Conference on Computer Science and Information Technology*, Beijing, 2009, pp. 259-262.

[8] García Ramírez, A.R.; Da Silva, J.F.; Savall, A.C.R.; Catecati, T.; Ferreira, M.G.G. "User's Emotions and Usability Study of a Brain-Computer Interface Applied to People with Cerebral Palsy". *Technologies*. Jun, 2018.

[9] R. U. Ferreira, A. F. da Rocha, C. C. Jr, G. A. Borges, F. A. O. Nascimento and W. H. Veneziano," Reconhecimento de padrões de sinais de emg para controle de prótese de perna," In *SN. XI Congresso Brasileiro de Biomecânica*. [S.l.], 2005. p. 1–5.

[10] A. Patel, J. Ramsay, M. Imtiaz and Y. Lu, "EMG-based Human Machine Interface Control,in " *12th International Conference on Human System Interaction (HSI), Richmond, USA, 2019, VA, pp. 127-131.*

[11] D. Oh, and Jo, Y. Emg-based hand gesture classification by scale average wavelet transform and cnn., p. 533–538, 2019.

[12] O. W. Samuel et al., "Intelligent EMG Pattern Recognition Control Method for Upper-Limb Multifunctional Prostheses: Advances, Current Challenges, and Future Prospects". *IEEE Access, vol. 7*, pp. 10150-10165, 2019, doi: 10.1109/ACCESS.2019.2891350.

[13] J. J. Bird, J. Kobylarz, D. R. Faria, A. Ekárt and E. P. Ribeiro, "Cross-Domain MLP and CNN Transfer Learning for Biological Signal Processing: EEG and EMG". *IEEE Access, vol. 8*, pp. 54789-54801, 2020, doi: 10.1109/ACCESS.2020.2979074.

[14] K. Chuysud and Y. Punsawad," Hybrid EEG-fEMG based Human-Machine Interface for Communication and Control Applications", in *16th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, pp. 1-5. IEEE, 2019.

[15] T. G. Amaral, O. P. Dias, A. Wolczowski and V. Fernão Pires, "Neural network based identification of hand movements using biomedical signals in,"*IEEE 16th International Conference on Intelligent Engineering Systems (INES)*, Lisbon, 2012, pp. 125-129.

[16] R. L. Silva, "Desenvolvimento de uma interface homem-máquina aplicada a uma cadeira de rodas robótica por meio de PDA". Universidade Federal do Espírito Santo, 2007.

[17] J. Morón, T. DiProva, J. R. Cochrane, I. S. Ahn and Y. Lu, "EMG-based hand gesture control system for robotics, in" *IEEE 61st International Midwest Symposium on Circuits and Systems (MWSCAS)*, Canada, 2018, Windsor, ON, pp. 664-667.

[18] M.Yoshikawa, M. Mikawa and K. Tanaka," Real-time hand motion estimation using emg signals with support vector machines", in *IEEE. 2006 SICE-ICASE International Joint Conference*. 2006. [S.l.]. p. 593–598, .

[19] L. Liao, Y. Tseng, H. Chiang and W. Wang, "EMG-based Control Scheme with SVM Classifier for Assistive Robot Arm," in *International Automatic Control Conference (CACS),* Taoyuan, 2018, pp. 1-5.

[20] Y. Paul, V. Goyal and R. A. Jaswal, Comparative analysis between svm & knn classifier for emg signal classification on elementary time domain features, in *IEEE. 4th International Conference on Signal Processing, Computing and Control (ISPCC)*. 2017, [S.l.]. p. 169–175.

[21] H. Caetano and R. Souza, "Eletrooculografia utilizada em interface homem-máquina como uma ferramenta de tecnologia assistiva." Núcleo de Tecnologias Assistivas UFU, 2015.

[22] E.Olimex," Shield manual, olimex, bulgaria (2014)". Disponível: https://www. olimex. com/Products/Duino/Shields/SHIELD-EKGEMG, 2015.

Francisco José Prado Junior possui graduação em Engenharia de Computação (2017) e Mestrando em Engenharia Elétrica e de Computação pela Universidade Federal do Ceará (UFC). Possui experiência em Educação Inclusiva, desenvolvimento de Tecnologias Assistivas e desenvolvimento de hardware e software.

Flávio Vasconcelos dos Santos é estudante do último semestre de Engenharia Elétrica pela Universidade Federal do Ceará (UFC - Campus Sobral). Possui conhecimento em machine learning, processamento digital de sinais, reconhecimentos de padrões e transmissão de ondas eletromagnéticas, além de experiência no tema de tecnologias assistivas.

Carlos Alexandre Rolim Fernandes possui graduação em Engenharia Elétrica pela Universidade Federal do Ceará - UFC (2003), mestrado pela UFC (2005) em Engenharia de Teleinformática, *Master 2 Recherche* pela *Université de Nice - Sophia Antipolis* - UNSA/França (2005) e doutorado em cotutela pela UNSA/FR e UFC (2009), na área de processamento de sinais. Foi professor substituto da UNSA/França em 2008/2009 e realizou pós-doutorado na UFC de jul/09 a fev/2010 na mesma área. Desde mar/10 é professor associado do curso de Engenharia da Computação do Campus Sobral da UFC. É fundador e ex-coordenador do Programa de Pós-Graduação em Engenharia Elétrica e de Computação (PPGEEC) do Campus Sobral da UFC. Atualmente, é membro do PPGEEC/UFC Sobral e do Programa de Pós-Graduação em Engenharia de Teleinformártica (PPGETI/UFC). É coordenador do Grupo de Tecnologias Assistivas e Educacionais (Grupo TAE), um grupo de pesquisa e extensão do Campus Sobral da UFC com diversos projetos concluídos e em andamento nas área de tecnologias assistivas para para pessoas com deficiência e na área de tecnologias educacionais, em parceria com diversas instituições da sociedade. Seus principais temas de pesquisa envolvem tecnologias assistivas, processamento de sinais, reconhecimento de padrões, álgebra multilinear (tensorial), sistemas não lineares, comunicações sem fio, etc.