UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação

# Redes Neurais Artificiais Empregadas no Reconhecimento e Classificaçao de eventos em series temporais

**Drausio Linardi Rossi**

Monografia - MBA em Inteligência Artificial e Big Data

ICMC USP SÃO CARLOS

**Drausio Linardi Rossi**

# Redes Neurais Artificiais Empregadas no Reconhecimento e Classificaçao de eventos em series temporais

Monografia apresentada ao Departamento de Ciências de Computação do Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo - ICMC/USP, como parte dos requisitos para obtenção do título de Especialista em Inteligência Artificial e Big Data.

Área de concentração: Inteligência Artificial

Orientador: Advisor: Prof. Dr. Jó Ueyama

**Versão original**

**São Carlos**

**2024**

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi, ICMC/USP, com os dados fornecidos pelo(a) autor(a)

S856m

Linardi Rossi, Drausio
Redes Neurais Artificiais Empregadas no Reconhecimento e Classificaçao de eventos em series temporais / Drausio Linardi Rossi ; orientador Jó Ueyama ; co-orientador Caetano Mazzoni Ranieri. – São Carlos, 2024.
57 p. : il. (algumas color.) ; 30 cm.

Monografia (MBA em Inteligência Artificial e Big Data) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2024.

1. LaTeX. 2. abnTeX. 3. Classe USPSC. 4. Editoração de texto. 5. Normalização da documentação. 6. Tese. 7. Dissertação. 8. Documentos (elaboração). 9. Documentos eletrônicos. I. Ueyama, Jó, orient. II. Mazzoni Ranieri, Caetano , co-orient. II. Título.

**Drausio Linardi Rossi**

# Event Recognition in Time Series using Artificial Neural Networks

Monograph presented to the Departamento de Ciências de Computação do Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo - ICMC/USP, as part of the requirements for obtaining the title of Specialist in Artificial Intelligence and Big Data.

Concentration area: Artificial Intelligence

Advisor: Prof. Dr. Jó Ueyama

**Original version**

**São Carlos**

**2024**

*Este trabalho é dedicado aos alunos, professores e todos os funcionarios da USP, como uma contribuição*
*das Bibliotecas do Campus USP de São Carlos para o desenvolvimento*
*e disseminação da pesquisa científica da Universidade.*

## AGRADECIMENTOS

Deus.
À minha Esposa Marcia por toda sua dedicação e aos meus filhos Hiago e Heitor.
Aos meus orientadores Prof. Dr. Jó Ueyama e Dr. Caetano Mazzoni Ranieri pela atenção, amizade e excelente orientação.
Aos organizadores deste Curso, que me possibilitaram, mesmo que a distância, a ter a chance de ampliar os meus conhecimentos.
A minha gratidão e muito obrigado.

*"Dê-me, Senhor, agudeza para entender, capacidade para reter, método e faculdade para aprender, sutileza para interpretar, graça e abundância para falar. Dê-me, Senhor, acerto ao começar, direção ao progredir e perfeição ao concluir."*

*Santo Tomás de Aquino*

# RESUMO

Linardi Rossi, D. **Redes Neurais Artificiais Empregadas no Reconhecimento e Classificaçao de eventos em series temporais**. 2024. 57 p. Monografia (MBA em Inteligência Artificial e Big Data) - Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, 2024.

O presente trabalho tem como objetivo o estudo dos sinais gerados por sensores MEMs (Micro-Electro-Mechanical Systems), utilizar os sinais gerados e aquisitados por estes sensores atraves de um smartphone que foi colocado dentro de um veiculo que efetuou manobras especificas, e atraves da inteligencia artifical criar uma rede neural para classificar os tipos de eventos ou manobras que foram aquisitados.

**Palavras-chave**: MEMs. Inteligencia Artifical. Series Temporais. Redes Neurais. CNN. LSTM. Tese.

## ABSTRACT

Linardi Rossi, D. **Event Recognition in Time Series using Artificial Neural Networks**. 2024. 57 p. Monograph (MBA in Artificial Intelligence and Big Data) - Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, 2024.

This work aims to analyze the MEMs (Micro-Electro-Mechanical Systems), sensors and its signals, use the MEMs generated signals during specific vehicle maneuvers that were recorded by a Smartphone that was set into the vehicle, and use artificial intelligence and neural networks to classify the event types and maneuvers that were recorded.



**Keywords**: MEMs, Artifical Inteligence, Temporal Series, CNN, LSTM, Thesis.

## LISTA DE FIGURAS

## LISTA DE TABELAS

## SUMÁRIO

# 1 INTRODUÇÃO

Devido ao desenvolvimento de novos softwares e hardwares cada vez mais potentes e sistemas cada vez mais conectados, várias possibilidades são encontradas para a utilizando os sinais de sensores presentes nos smartfones (acelerômetros, giroscópios, GPS, etc.).

### 1.0.1 Justificativa e importância:

Um exemplo é a utilização de redes neurais para classificar situações de direção de automóveis baseadas em sinais de sensores de aceleração aquisitados de um smartfone.

Identificada uma situação específica de direção (frenagem, aceleração, curvas, etc.) será possível conectar outros tipos de sensores e ou equipamentos, bem como o próprio smartfone, que poderão utilizar estas informações para criar novas aplicações uteis ao usuário.

Exemplo podem ser, alarmes em caso de acidentes, one o smartfone pode, de maneira autonoma, efetuar uma chamada ao hospital ou ao departamento de polícia,

### 1.0.2 Objetivos:

O objetivo principal é criar uma rede neural, a ser executada por um aplicativo rodando em um smartfone, que seja capaz, através da utilização dos sinais dos sensores de aceleração e giroscópios, identificar situações de direção de um veículo.

Parado, acelerando, freando, curvas a esquerda/direita,

#### 1.0.2.1 Objetivos Específicos:

1. Estudar os sinais dso sensores MEMs aquisitados pelo smartfone: acelerômetros, giroscópio e campo magnético.
2. Associar estes sinais a situações especificas de direção.
3. Estudar redes neurais artificiais e como uma rede neural artificial pode ser treinada para reconhecer um evento específico.
4. Estudar como esta rede neural pode depois ser implementada em um aplicativo de celular ou em um equipamento embarcado onde os recursos computacionais são escassos e limitados quando comparados a um computador pessoal cluster no qual a rede neural foi primeiramente implementada.
5. Estudo e aprendizado: o estudo das tecnologias de aprendizado de maquinas, redes neurais, aprendizado profundo, tratamento de dados. A metodologia é analisar e estudar os dados disponíveis e aplicar inteligência computacional e realizar a classificação de situações de frenagem e aceleração bruscas. Os diferentes tipos de aprendizado: supervisionado, não supervisionado e por reforço, devem ser compreendidos e utilizados caso necessários.

6. Implementação do algoritmo: Serão estudados alguns algoritmos que sejam capazes de realizar os trabalhos de predição e seleção necessários. Dentre esses serão considerados inicialmente: Regressão Linear, Regressão Polinomial, K-means, Árvores de Decisão [2]. Com esse estudo será possível determinar qual ou quais algoritmos são melhores para a resolução desse problema.

7. Avaliação qualitativa e quantitativa: Para avaliarmos quantitativamente o resultado desse trabalho vamos utilizar métricas estatísticas já conhecidas como: Acurácia, F1 Score e Precisão.

8. Testes e Ajustes: Após a identificação dos possíveis ajustes, é importante que tais ajustes sejam de fato implementados e uma nova avaliação seja feita em um cenário real, sendo assim após os ajustes necessários para que consigamos uma melhor qualidade nos indicadores de avaliação,

9. Implementação do sistema: O algoritmo será implementado em um computador pessoal poderá futuramente ser executado em um aplicativo em um smartfone.

#### 1.0.3 Resultados e Impactos Esperados:

É esperado que o algoritmo consiga reconhecer/classificar as diferentes situações de direção do veículo, e com isso outros dispositivos/algoritmos possam utilizar esta informação.

# 2 FUNDAMENTAÇÃO TEÓRICA

## 2.1 Series temporais

Em matematica, uma seria temporal é um conjunto de numeros indexados (ou representados em um grafico) em ordem de tempo. Mais comumente, uma sequencia de tempo discreto é uma serie temporal constituida de dados acquisitados em um intervalo constante de tempo. Exemplos de series temporais sao dados de temperatura, pressao e aceleracao coletados de um sistema.

Uma serie temporal é frequentemente plotada em um grafico cujo eixo x representa uma definida unidade de tempo. As series temporais sao usadas em estatística, processamento digital de sinais, reconhecimento de padroes, matematica financeira, engenharia de controle, comunicacoes, etc.

A figura 1 mostra dados acquisitados de sensores de um smartphone colocado dentro de um veiculo. O veiculo realisa uma manobra na qual muda para uma a esquerda de maneira agressiva (Ferreira *et al.*, 2017).

## 2.2 Redes Neurais - Fundamentos

### 2.2.1 Perceptron

A forma mais basica de uma rede neural é o perceptron, mostrado na figura 2, que é a representacao artificial de um neuronio que possui tres componentes importantes: os dendritos, o corpo celular e o axônio. Este modelo foi proposto em 1958 por Frank Rosenblatt (Rosenblatt, 1958) , um psicólogo que desenvolveu o algoritmo para ajustar os pesos da rede e provou sua convergência.

É importante ressaltar a existencia de um peso (bias), que serve para aumentar os graus de liberdade, permitindo assim uma melhor adaptacao da funcao neuronal.

O perceptron é um classificador binario, que é capaz de possuir varias variaveis de entrada, ou seja, um vetor de entrada. Este neuronio artificial pode aprender atraves de diferentes valores de entrada (vetores de entrada), gerando apenas uma resposta binaria de classificacao.

O aprendizado é o processo pelo qual os parâmetros livres de uma rede neural são adaptados por meio de um processo de estímulo pelo ambiente. O tipo de aprendizado é definido pela maneira com que acontecem as mudanças nos parâmetros.

Toda a vez que ele é treinado, os pesos podem variar a gerar o mesmo resultado na saida. Isto nos leva a termos um problema com varias solucoes. Os componentes basicos do perceptron sao: entradas, pesos, funcao de ativacao e saida, onde os pesos sao os principais componentes. Vale salientar que o perceptron só funciona para problemas linearmente separáveis.


Figura 1 – Dados de sensores: acelerometros, giroscopio e magnetrometro plotados no tempo (Ferreira *et al.*, 2017).

### 2.2.2 Multi-Layer Perceptron Networks

Multi-Layer Perceptrons (MLP) sao caracterizadas pela presenca de pelo menos uma camada intermediaria, como mostrado na figura 3. Uma rede neural para ser considerada multi-layer, deve ter, no minimo, duas camadas. O algoritmo de aprendizado de uma MPL é chamado de algoritmo Back-Propagation.

MPLs superam as limitações práticas do Perceptron. O modelo de cada neurônio inclui uma função de ativação não linear e diferenciável e contém uma ou mais camadas escondidas/ocultas entre a camada de entrada e a camada de saída. MPLs possuem alto grau de conectividade

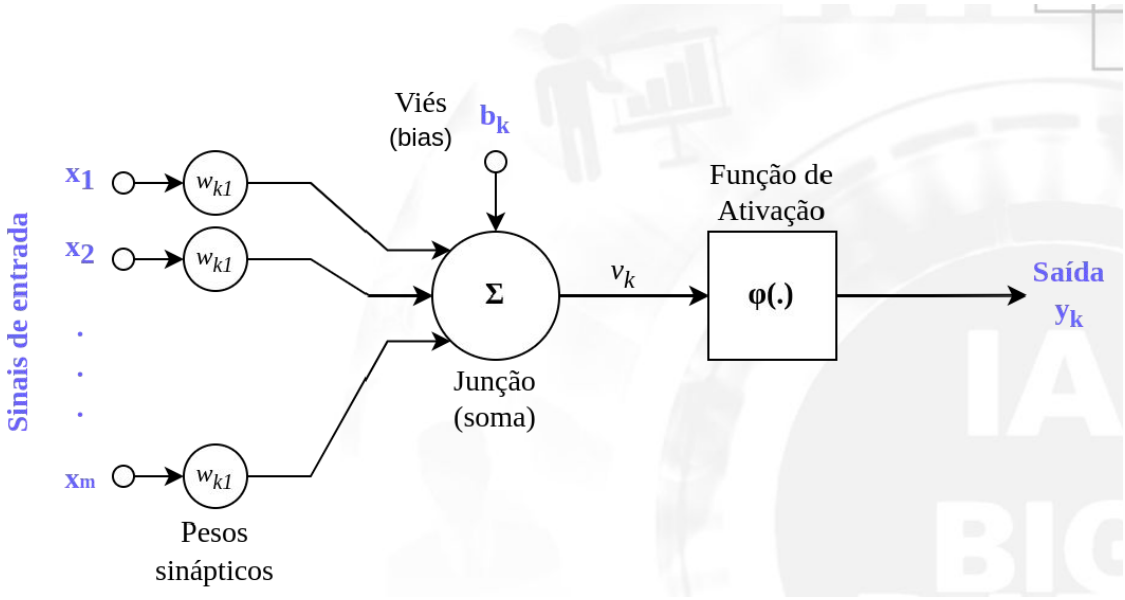


Figura 2 – Perceptron

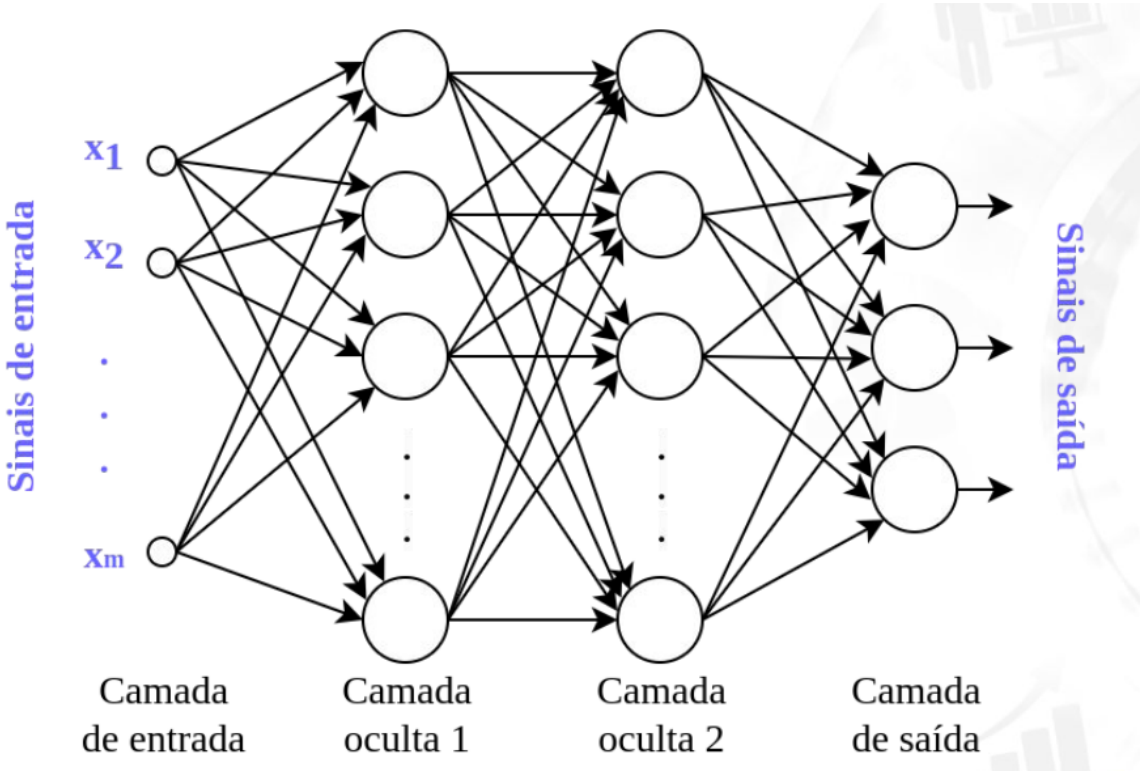


Figura 3 – Multilayer Perceptron (MLP)

### 2.2.3 Back-propagation

Para compreender o algoritmo de Back-propagation é necessario compreender as sucessivas fases que o compoem, que sao Forward phase e Backward phase.

Durante a Forward phase os pesos sao fixos e o sinal é propagado através da rede, camada por camada, até a saída. Mudanças só ocorrem nos potenciais de ativação e nas saídas dos neurônios da rede.

Ja o aprendizado é feito na Backward phase. O sinal de erro produzido comparando a saída desejada com a obtida. O erro é retropropagado através da rede, camada por camada, ajustando assim os pesos sinápticos da rede.

Na Backward phase, um calculo matematico é realizado para ajustar os pesos, podendo ser atraves de derivadas parciais, erro dos minimos quadrados ou entropia cruzada.

### 2.2.4 Deep Learning - Aprendizado profundo

Deep learning, ou aprendizado profundo, é o subconjunto de metodos de aprendizado de maquina baseado em redes neurais artificiais. O adjetivo "profundo"se deve ao fato do uso de multiplas camadas na rede. Os metodos usados podem ser supervisionados, semi-supervisionados e nao supervisionados (LeCun; Bengio; Hinton, 2015).

Architeturas de redes de aprendizado profundo, redes neurais recorrentes, redes neurais convolucionais e transformers, tem sido aplicadas em varios campos, como visao computacional, reconhecimento de fala, processamento de linguagem natural, traducao automatica, etc., onde os resultados obtidos sao comparaveis ou superiores, em alguns casos, aos resultados obtidos por humanos.

### 2.2.5 Redes Neurais Recorrentes

Uma Rede Neural Recorrente (RNNs) é um tipo de rede neural artificial que utiliza dados sequenciais ou series de dados temporais. Estes algoritmos de sao comumente usados, por exemplo, no processamento de video e processamento de linguatem. Assim como como redes feedforward e redes convolucionais, as redes neurais recorrented utilizam dados de treinamento para aprenderem. Elas sao distinguidas por sua "memoria"assim que utilizam informacoes de dados de entrada anteriores para influenciar a entrada e a saida atuais.

Enquanto redes neurais profundas tradicionais assumem que entradas e saidas sao independentes entre si, a saida de uma rede neural recorrente depende dos elementos passados da sequencia de dados que serve de entrada para essa rede (IBM, 2024), (Cala, 2019).

#### 2.2.5.1 Arquitetura de Redes Neurais Recorrentes

Redes neurais recorrentes bi-direcionais (BRNN): Esta é uma arquitetura variante de RNNs. Enquanto RNNs unidireionais podem somente se basear em entradas previas para realizar predicoes sobre o estado corrente, redes neurais recorrentes bidirecionais utilizam dados futuros para melhorar a sua acuracia.

Long short-term memory (LSTM): Esta é uma arquitetura comumente usada, que foi introduzida por Hochreiter e Juergen Schmidhuber como a solucao para acabar com o problema do gradiente (Hochreiter; Schmidhuber, 1997). Esta architetura endereca o problema de dependencias de long-term, isto é, se o estado previo que esta influenciando a predicao atual nao esta situado em um passado recente, o modelo da RNN nao sera capaz de predizer o estado atual acuradamente.

Gated recurrent units (GRUs): Esta arquitetura de RNN é similar a LSTMs. Foram introduzidas em 2014 por Kyunghyun Cho et al(Cho *et al.*, 2014). A GRU é como uma LSTM com um mechanismo de gating para introduzir ou desconsiderar algumas funcoes, mas nao possui um vetor de contexto ou um gate de saida, resultando em um numero menor de parametros quando comparada a uma LSTM. Ela tambem trabalha para solucionar o problema de memoria de short-term.

Conforme demonstrado em (Chung *et al.*, 2014) as redes LSTMs (Long Short-Term Memory) sao indicadas para a classificacao de eventos em series temporais por varias razoes:

Habilidate de capturar as dependencias de tempos longos: LSTMs sao projetadas para capturar dependencias de tempos longos em dados sequenciais, assim elas sao indicadas para a classificacao de eventos onde a relacao entre dados atraves do tempo sao importantes.

Capacidade de lidar com sequencias de tempo variavel: LSTMs podem lidar com sequencias de entrada de comprimento variavel, o que é comum em series temporais. onde o comprimento dos eventos pode variar.

Memoria de informacoes passadas: LSTMs possuem uma celula de memoria que pode armazenar informacoes passadas, possibilitando-as reter contextos importantes de dados passados contidos nas series temporais.

Sao robustas a ruidos e dados incompletos: LSTMs sao robustas ao ruido e a dados incompletos se sao comuns em series temporais, tornando-as adequadas para tratar dados reais.

Habilidade de aprender padroes complexos: LSTMs sao capases de aprender padroes complexos e relacionamentos dentro das series temporais, fazendo-as efetivas para capturar a dinamica intricada de series temporais.

No geral, as LSTMs sao muito adequadas par a classificacao em series temporais

devido a sua capacidade de capturar dependencias de long-term, tratar sequencias de tempo variaveis, relembrar informacoes passadas e aprender padroes complexos existentes nos dados.

Ja as CNNs (Convolution Neural Network) sao efetivas em capturar padroes em eventos espaciais e temporais. No contexto de dados gerados a partir de acelerometros, as CNNs podem aprender a extrair aspectos relevantes das series de dados. As camadas convolucionais da CNN podem processar eficientemente sequencias de dados geradas por sensores.

A combinacao CNN-LSTM pode ser beneficial para capturar tanto as dependencias locais quanto as globais em uma serie temporal. A CNN pode ser usada para a extracao de caracteristicas e reducao de dimensionalidade, enquanto a LSTM pode efetivamente modelar a dinamica temporal e as dependencias de long-term dentro das sequencias.

## 2.3 Micro-Electromechanical Systems MEMS

Sistemas micro-eletromecanicos (MEMS) sao componentes de tamanho microscopico que incorporam tanto eletronica quanto partes mecanicas moveis. MEMS sao feitos a partir de componentes que tem em torno de 1 a 100 micrometros. Usualmente consistem em uma unidate central de processamento (um circuito integrado como um microprocessador) e varios componentes que interagem com o meio ambiente, como microsensores (Gabriel *et al.*, 1988).

OS MEMS comecaram a ser fabricados usando tecnologias de fabricacao de semicondutores, normalmente usadas em eletronica. Isso inclui tecnicas como molding, plating, wet etching (KOH, TMAH) e dry etching (RIE and DRIE), electrical discharge machining (EDM), e outras tecnologias capazes de fabricar pequenos componentes (**??**). A figura **??** mostra um exemplo de MEMS utilizado no setor automotivo.

Existem 2 tipos basicos de tecnologia MEMS: capacitiva e resistiva. MEMS capacitivos sao compostos por placas que se movem ou elementos sensiveis, que mudam de capacitancia. MEMS resistivos sao controlados eletrostaticamente, e podem falhar por fadiga, pois seus componentes podem se deformar com o uso.

### 2.3.1 Acelerometros MEMS

Acelerometros MEMS capacitivos detectam a aceleracao explorando o movimento de uma massa sismica. O movimento dessa estrutura mecanica, que afasta e aproxima placas capacitivas, assim alterando a capacitancia do circuito ao qual o componente esta conectado. A figura 5 mostra um acelerometro MEMS, suas estruturas eletromecanicas assim como o diagrama do circuito em que é conectado.

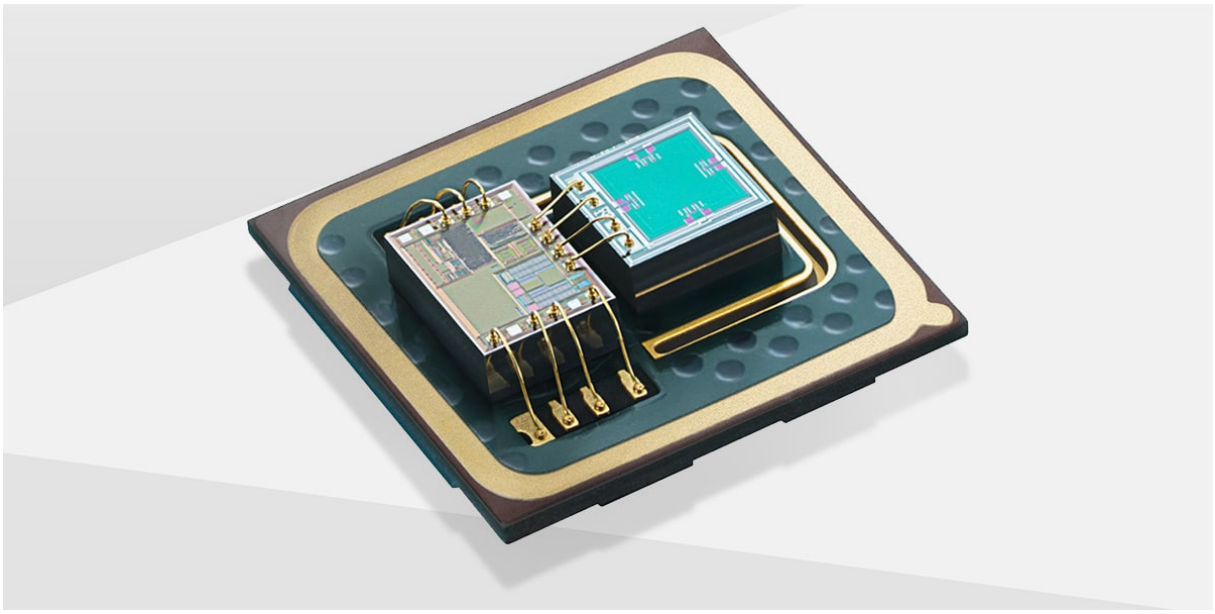

Figura 4 – Sensor MEMS

(Bosch, 2025)

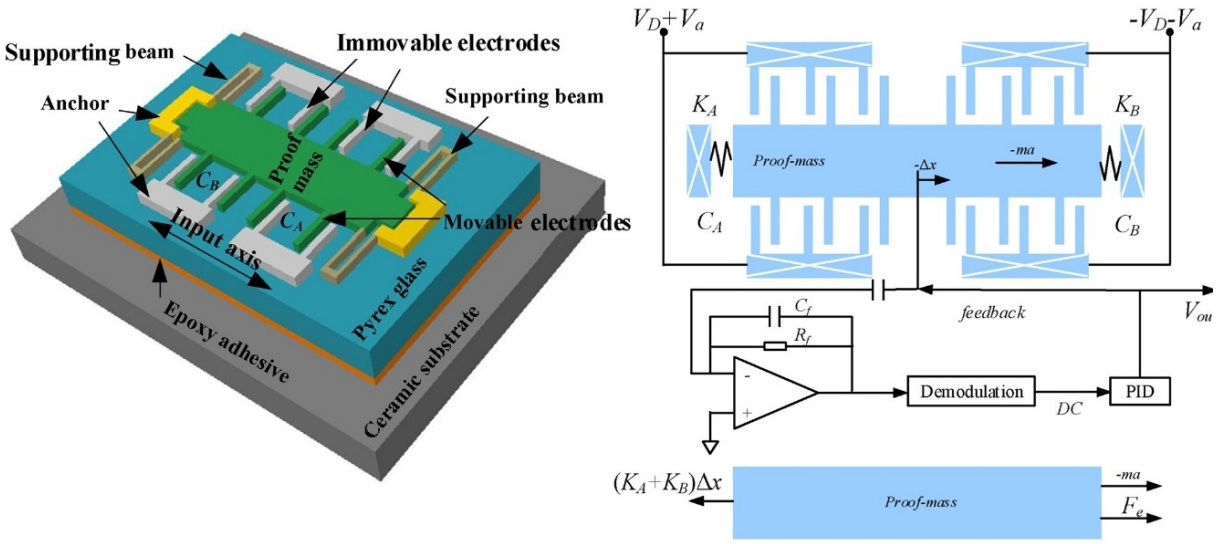


Figura 5 – Acelerometro MEMS

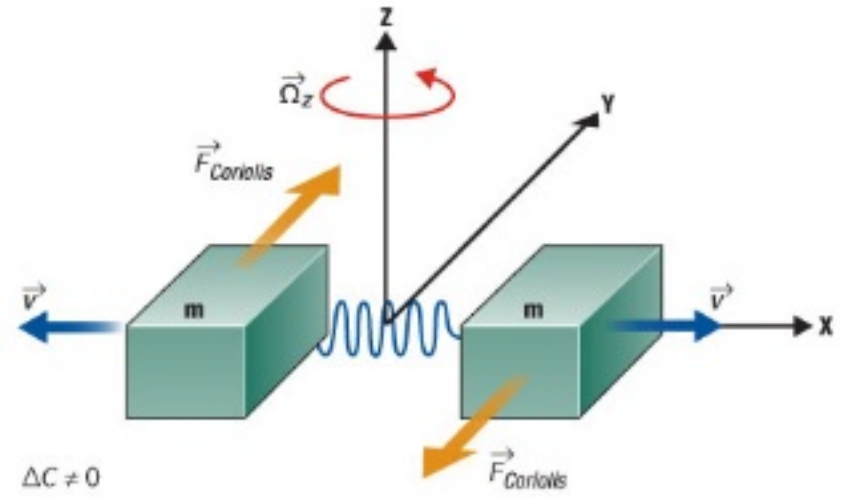


Figura 6 – Principio fisico de funcionamento do giroscopio MEMS

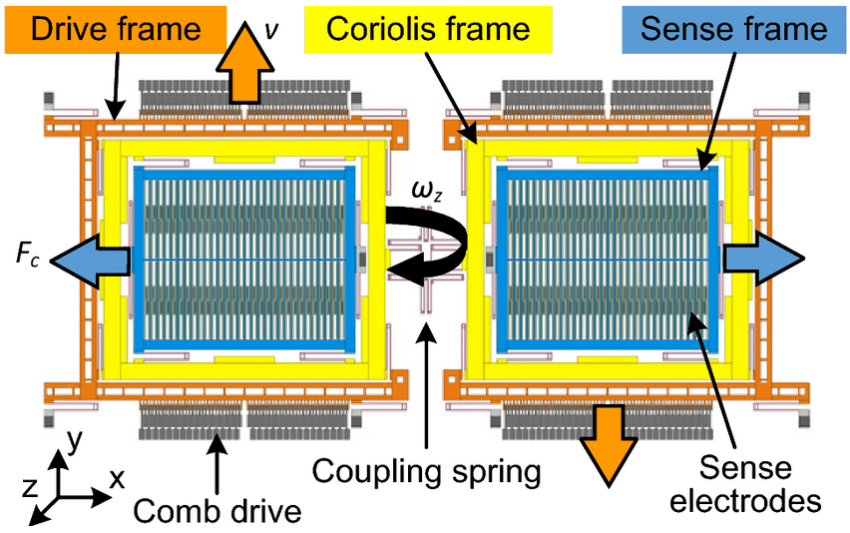


Figura 7 – Arquitetura da construcaodo do giroscopio MEMS

### 2.3.2 Giroscopio MEMS

Em sua configuracao mais comum, silício é usado para fabricar um par de massas, que sao submetidas a uma oscilacao, causada por campos eletricos. Estes campos eletricos oscilantes fazem com que as massas oscilem constantemente em direcoes opostas, comforme mostrado na figura 6. Quando uma velocidade angular é aplicada, o efeito de Coriolis afeta cada massa tambem em direcoes opostas, o que resulta em uma mudanca de capacitancia entre elas. Esta mudanca é o que principio do funcionamento do sensor. Quando as duas massas sao submetidas apenas a aceleracao linear, elas se movem na mesma direcao, nao ocasionando mudanca na capacitancia entre elas.

A fig mostra 7 a arquitetura da construcao de um giroscopio MEMS (Willers *et al.*, 2020)

# 3 METODOLOGIA

| **Tipo de manobra** | **Amostras** |
|---|---|
| Frenagem agressiva | 12 |
| Aceleracao agressiva | 12 |
| Curva agressiva a esquerda | 11 |
| Curva agressiva a direita | 11 |
| Troca de faixa agressiva a esquerda | 4 |
| Troca de faixa agressiva a direita | 5 |
| Eventos nao agressivos | 14 |
| Total | 69 |

Tabela 1 – Eventos e Amostras

## 3.1 Descricao dos Dados e Amostragem

O objetivo do trabalho é o de treinar uma rede neural a fim de reconhecer padroes de manobras de um veiculo, com base em sinais de sensores coletados de um smartfone.

A base deste trabalho sera a que foi apresentada em (Ferreira *et al.*, 2017), na qual coleta de dados foi realizada. Os mesmos dados serao utilizados para o treinamento e teste da rede neural.

Os dados foram coletados utilizando um Motorola XT1058 com Android versao 5.1; e um veiculo Hoda Civic 2011. Mais informacoes sobre a coleta de dados podem ser encontradas na descricao destes

Os dados serao organizados a partir dos eventos e sensores disponiveis.

## 3.2 Avaliacao do experimento

Como mostrado em (Ferreira *et al.*, 2017) os melhores resultados foram obtidos utilizando o acelerometro e o giroscopio, portanto inicialmente somente os dados gerados por estes 2 sensores serao utilizados.

Os dados de sensores utilizados serao os do acelerometro (Acc), acelerometro linear (LinAcc) e o giroscopio (Gyr). Conforme mostrado na seccao anterior, o acelerometro mede a aceleracao que foi aplicada no veiculo em metros por segundo ao quadrado (m/s2), incluindo a forca da gravidade. O giroscopio mede a taxa de rotacao em relacao aos eixos do veiculo em radianos por segundo (rad/s). Estes sensores fornecem uma serie temporal em 3 dimensoes (x,y e z) com uma precisao de naosegundos, relativos ao sistema de coordenadas do veiculo.

Os eixos dos sensores (x, y, z) tambem serao avaliados. No sistema de coordenadas definido pela norma ISO-885-2011, os eixos do veiculo definem as direcoes de velocidade, aceleracao e rotacao. Os eixos podem ser vistos na figura 8

Os dados estao distribuidos em grupos que includem os tres eixos, somente eixo x,

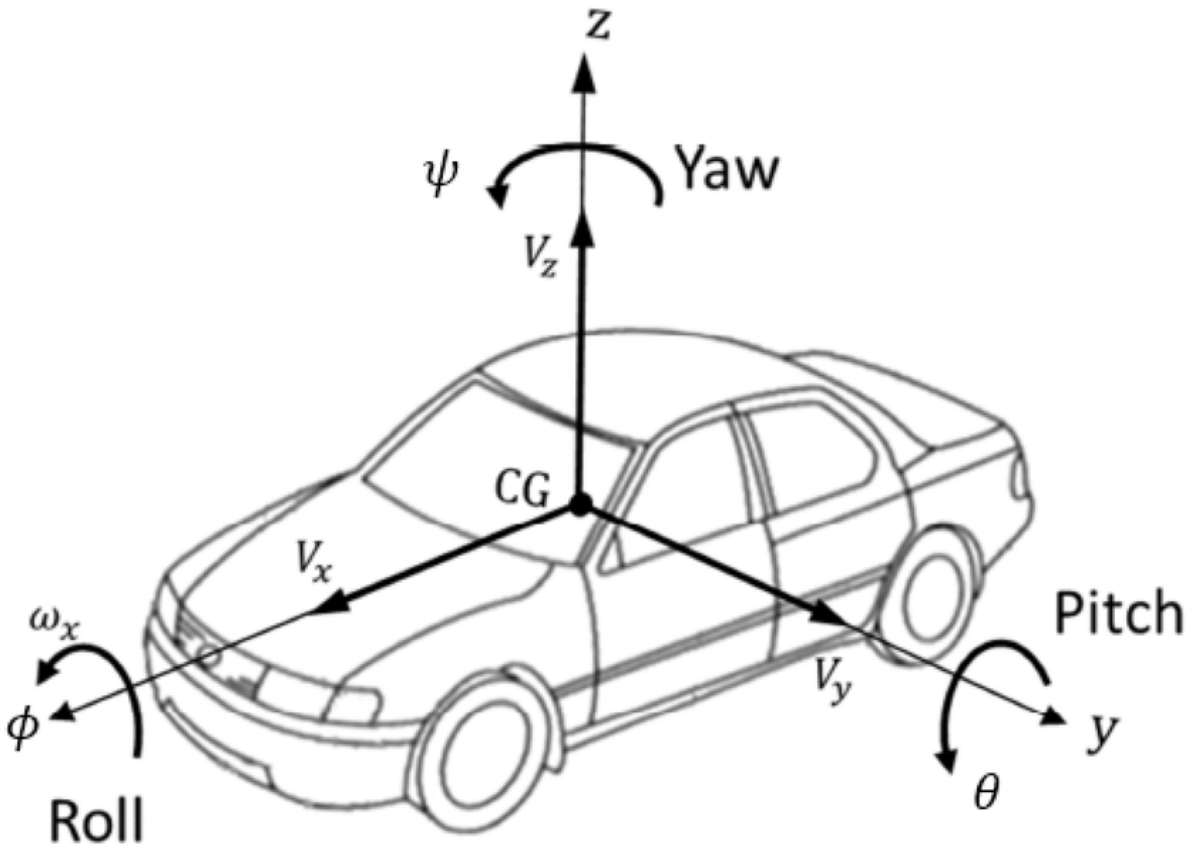


Figura 8 – Eixos do Veiculo de acordo com a norma ISO-885-2011

somente eixo y e somente eixo z. Por exemplo, os dados do acelerometro se originam dos data-sets com os tres eixos, acelerometro x, acelerometro y e acelerometro z.

Os dados brutos coletados sao compostos basicamente de valores de 3 eixos e um timestamp de precisao de nanosegundos, que indica o tempo em que a amostra foi coletada. Os dados brutos nao sao os dados de entrada para os classificadores da rede neural. Os dados serao agrupados em conjuntos de amostras de 1 segundo, compondo frames que depois originaral vetores de atributos. A medida que sao analisados, a janela de tempo é atualizada sobre a serie temporal. O tamanho da janela dependera do tempo que o evento de direcao foi realizado, como mostrado na tabela 1.

## 3.3 Procedimento de aquisicao de dados

Os dados foram coletados em um experimento real, onde um aplicativo executado smartphone com o sistema operacional Android foi utilisado. Durante o experimento um motorista executou manobras especificas. O tempos nos quais as manobras foram realisadas tambem foram gravados.

Ao todo, 4 viagens foram realisadas, cada uma com aproximadamente 13 minutos

de duracao. As condicoes do experimento foram: (i) o veiculo era um Honda Civic 2011; (ii) o smartphone era um Motorla XT1058 com Android versao 5.1; (iii) o smartphone foi fixado ao para-brisas do carro atraves de um suporte, e nao foi tocado, movimentado ou operado durante a acquisicao dos dados; (iv) a frequencia de aquisicao dos sensores variou entre 50 e 100Hz, dependendo do sensor; (v) dois motoristas, com mais de 15 anos de experiencia executaram as manobras; e (vi) as condicoes climatiacas eram favoraveis, com sol e pista asfaltada e seca.

O proposito foi o de estabelecer um conjunto de eventos que representao situacoes reais de direcao tais como frenagem, aceleracao, curvas e trocas de faixa, conforme descrito na tabela 1.

# 4 AVALIAÇÃO EXPERIMENTAL

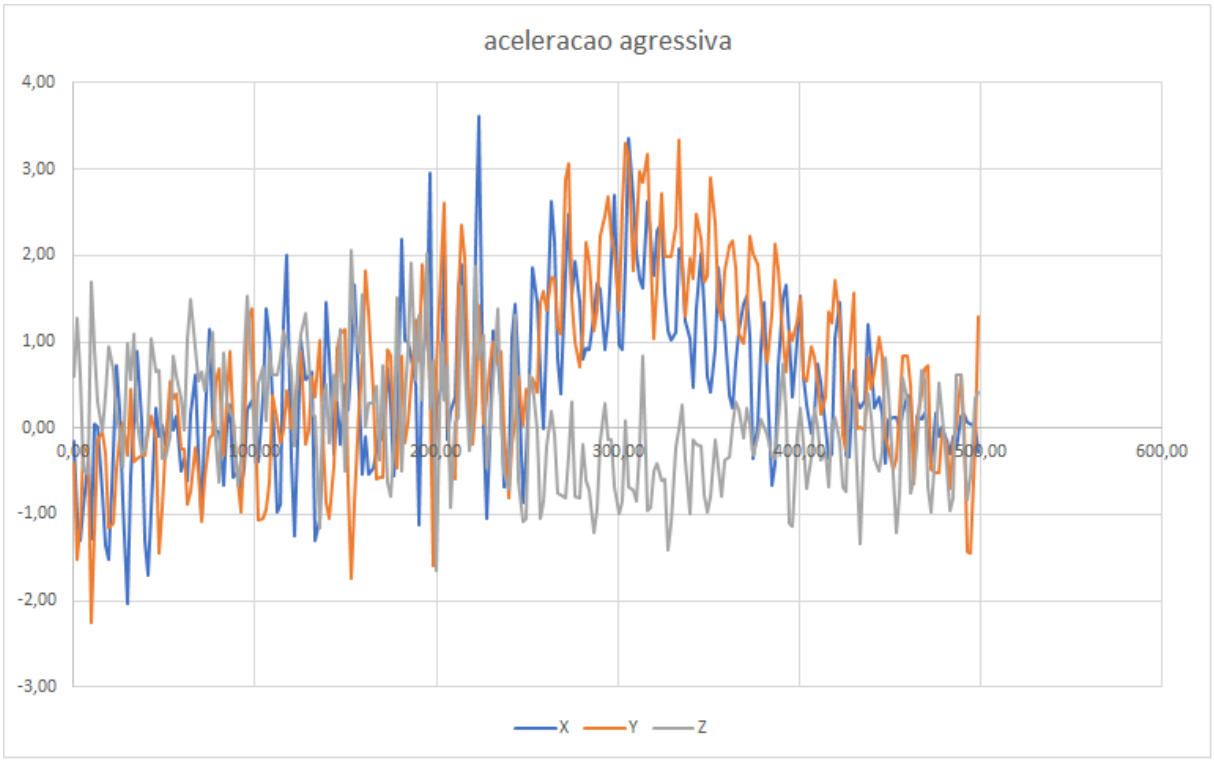


Figura 9 – Aceleracao agressiva - sensores acelerometros lineares

## 4.1 Análise dos dados:

Os dados estao sendo analisados conforme os eventos descritos na tabela 1.

Como mostrado na figura 8, durante as manobras de frenagem e aceleracao, os sinais dos acelerometros lineares, principalmente do eixo x, e do giroscopio, eixo y, sao os mais proeminentes nas medidas, como mostrado nas figuras 9 e 10

Durante as curvas, a direita e a esquerda, ambas realizadas de maneira agressiva, os sinais mais proeminentes sao os dos giroscopio, como podemos verificar nas figuras 11 e 12.

Os dados foram divididos pro eventos, conforme a tabela 1, onde cada data set possui uma duracao de 3 a 5 segundos, dependendo do tempo de cada evento.

Os dados amostrados foram disponibilizados em 4 data sets, onde varios eventos foram reunidos. Estes dados foram filtrados de acordo com as informacoes disponiveis para cada evento, conforme mostrado em tabelas como a tabela

## 4.2 Treinamento da rede neural

Os dados estão sendo trabalhados, e visualizados através da biblioteca NumPy e Matplotlib.

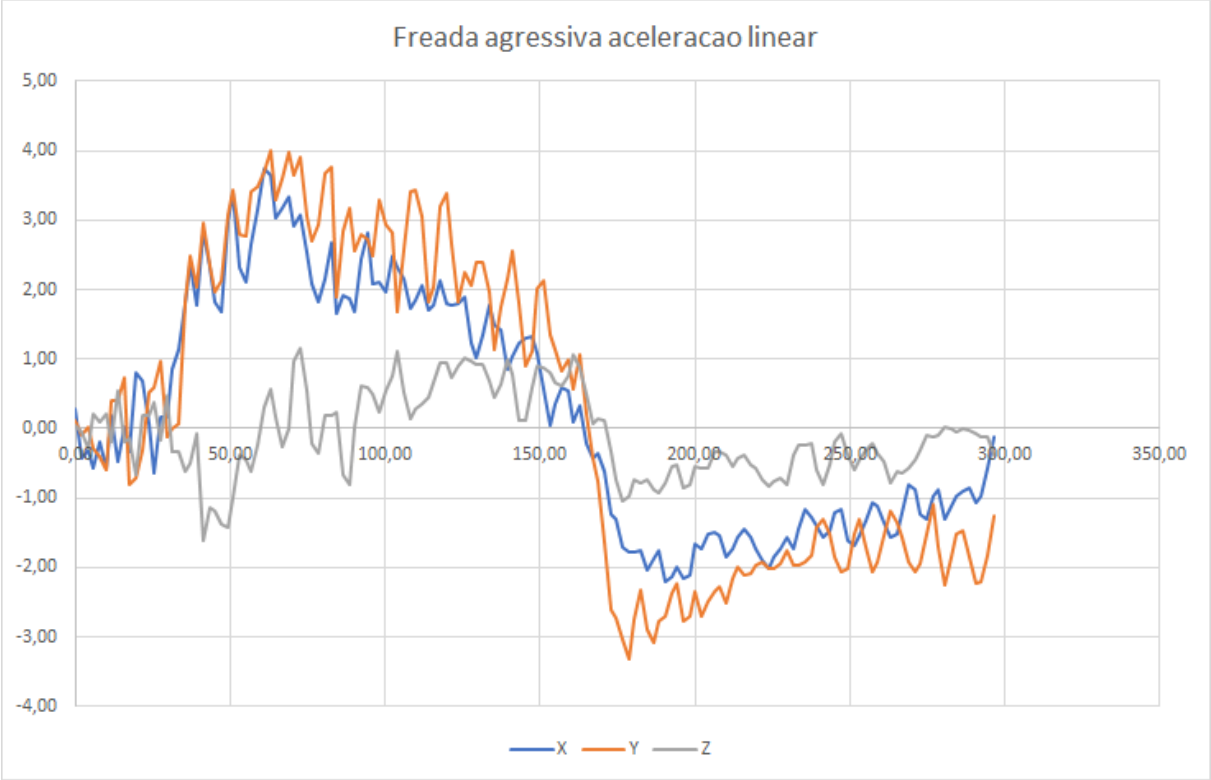


Figura 10 – Frenagem agressiva - sensores acelerometros lineares

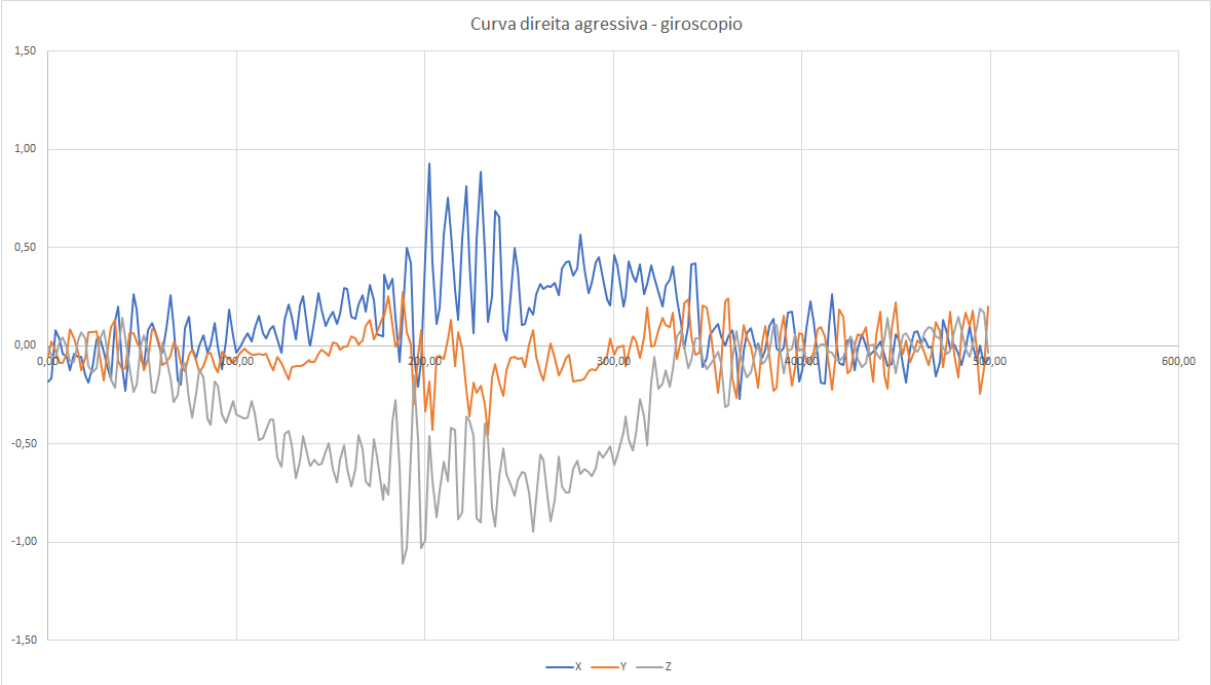


Figura 11 – Curva direita agressiva - sensores acelerometros

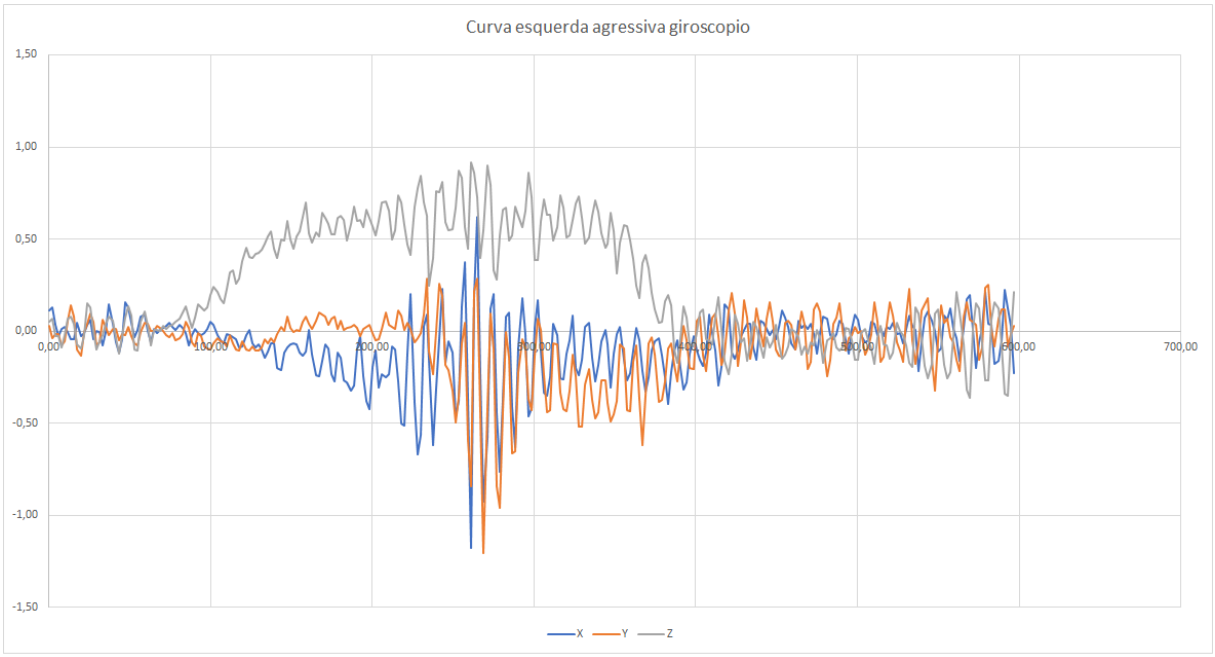


Figura 12 – Curva esquerda agressiva - sensores acelerometros

| Evento | Inicio[s] | Fim[s] |
|---|---|---|
| evento_nao_agressivo | 2,00 | 6,50 |
| curva_direita_agressiva | 19,50 | 23,50 |
| evento_nao_agressivo | 30,00 | 33,50 |
| curva_direita_agressiva | 95,00 | 98,00 |
| curva_esquerda_agressiva | 247,00 | 251,50 |
| curva_esquerda_agressiva | 348,70 | 352,30 |
| evento_nao_agressivo | 485,00 | 489,00 |
| curva_esquerda_agressiva | 496,00 | 499,50 |
| curva_direita_agressiva | 587,00 | 590,00 |
| curva_esquerda_agressiva | 750,00 | 753,80 |
| curva_direita_agressiva | 840,70 | 844,00 |
| curva_direita_agressiva | 980,00 | 983,20 |
| curva_esquerda_agressiva | 1087,40 | 1090,90 |
| troca_faixa_direita_agressiva | 1139,80 | 1142,00 |
| troca_faixa_direita_agressiva | 1201,00 | 1202,90 |
| troca_faixa_direita_agressiva | 1211,40 | 1213,50 |

Tabela 2 – Eventos - Inicio e Fim.

Os dados deverão ser lidos, e como diferentes escalas não são ideais para redes neurais, mas primeiro deverão ser normalizados, pois cada sensor tem um sinal e diferente (baseando somente em acelerômetros e giroscópio).

Após a preparação dos dados, o modelo de rede neural está sendo construído, e devera ser treinado com 80 porcento dos dados disponíveis, e testado com os 20 porcento restantes.

Os resultados e avaliações serão através dos testes de acurácia e perda (loss).

As atividades estão em andamento e serão apresentadas na conclusão e apresentação deste trabalho.

## 4.3 Tratamento dos dados

Durante a implementação, foi constatado que a quantidade de amostras para cada evento era insuficiente para que um aprendizado da rede fosse bem-sucedido.

O número total de eventos disponíveis consta em 65, sendo estes distribuídos conforme indicado na tabela 1. Foi decidido que todos os eventos agressivos seriam agrupados em uma única classe, treinando assim a rede neural a identificar um evento agressivo, sem que este seja especificado em um tipo específico de manobra.

Conforme mostrado na figura 13, os dados originais utilizados se encontravam desbalanceados, pois o tempo de amostragem de direção onde nenhum evento acontece superava em muito o tempo de amostragem dos eventos agressivos.

Para o balanceamento dos dados, um data frame contendo eventos repetidos foi construído para que o número tempo de amostragem de eventos agressivos fosse equiparável ao tempo de eventos não agressivos ou apenas um estado de direção normal.

O resultado dos dados balanceados e mostrado na figura 14

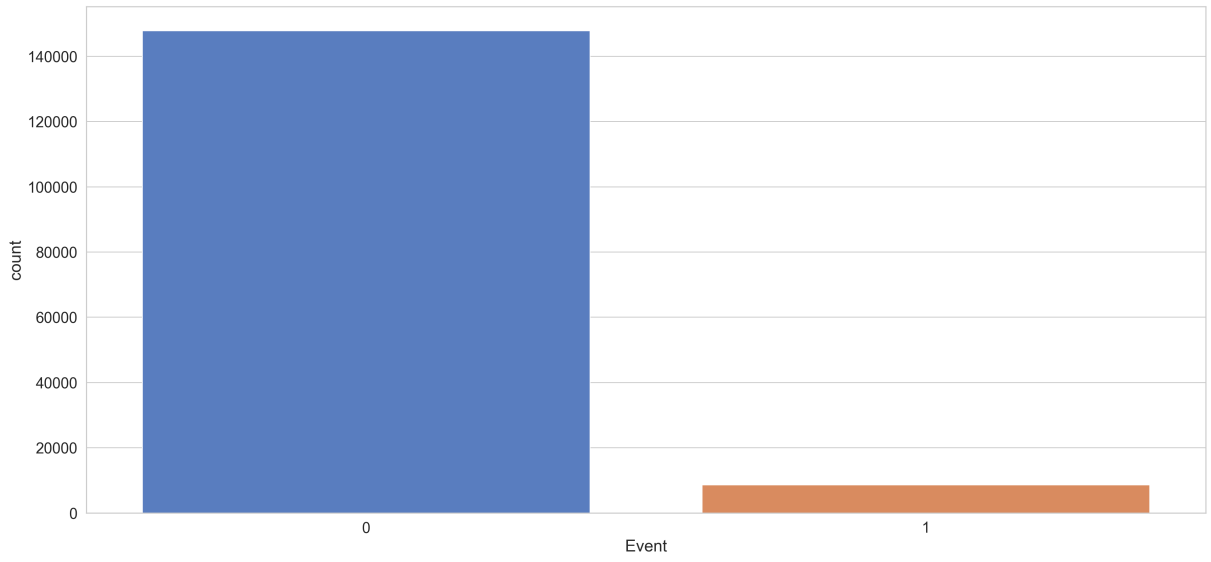


Figura 13 – Desbalanceamento de dados na classificacao de enventos: 1=Eventos agressivos, 0=Eventos nao agressivos

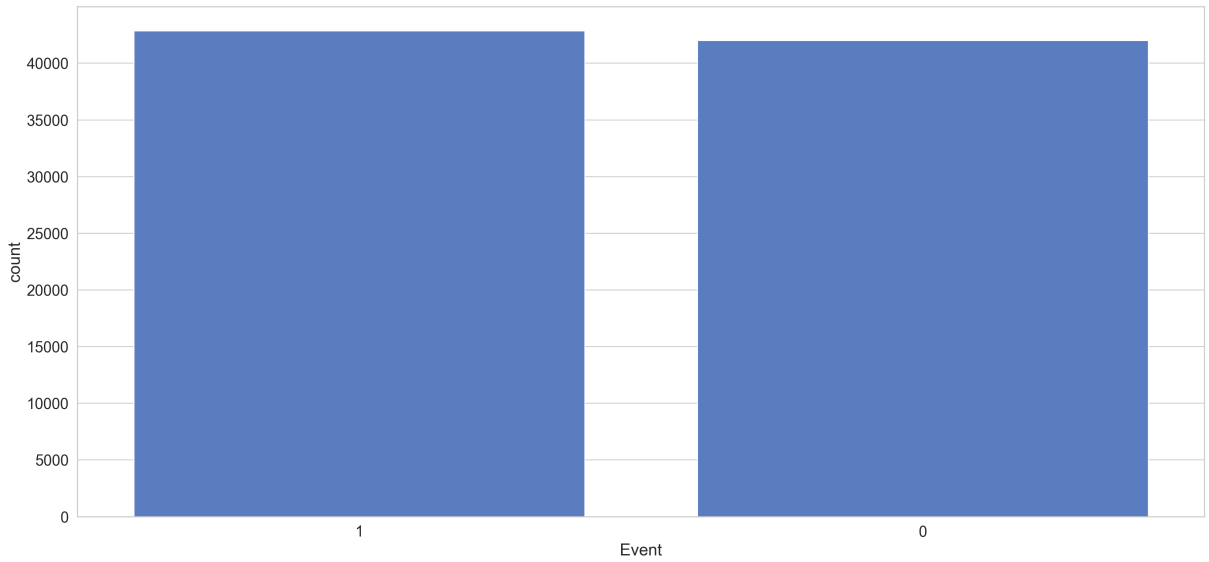


Figura 14 – Dados balanceados apos tratamento: 1=Eventos agressivos, 0=Eventos nao agressivos

## 4.4 Implementacao das redes neurais

### 4.4.1 Rede Neural LSTM

O primeiro teste foi feito com uma rede neural LSTM.

O código de implementação dessa rede:

```
model_LSTM = keras.Sequential()
model_LSTM.add(
    keras.layers.Bidirectional(
        keras.layers.LSTM(
        units=128,
        input_shape=[X_train.shape[1], X_train.shape[2]]
        )
    )
)
model_LSTM.add(keras.layers.Dropout(rate=0.5))
model_LSTM.add(keras.layers.Dense(units=128, activation='relu'))
model_LSTM.add(keras.layers.Dense(y_train.shape[1], activation='softm
model_LSTM.compile(loss='binary_crossentropy', optimizer='adam', met

# Modell bersicht anzeigen
model_LSTM.summary()
```

| Layer (type) | Output Shape | Param |
|---|---|---|
| bidirectional (Bidirectional) | (None, 256) | 135,168 |
| dropout (Dropout) | (None, 256) | 0 |
| dense (Dense) | (None, 128) | 32,896 |
| dense_1 (Dense) | (None, 2) | 258 |

Tabela 3 – Sumario do Modelo da Rede Neural LSTM

A tabela 3 mostra o sumario da rede implementada apos o treinamento.

#### 4.4.1.1 Resultados Rede Neural LSTM

A figura 15 mostra os resultados de treinamento e perda da rede neural.

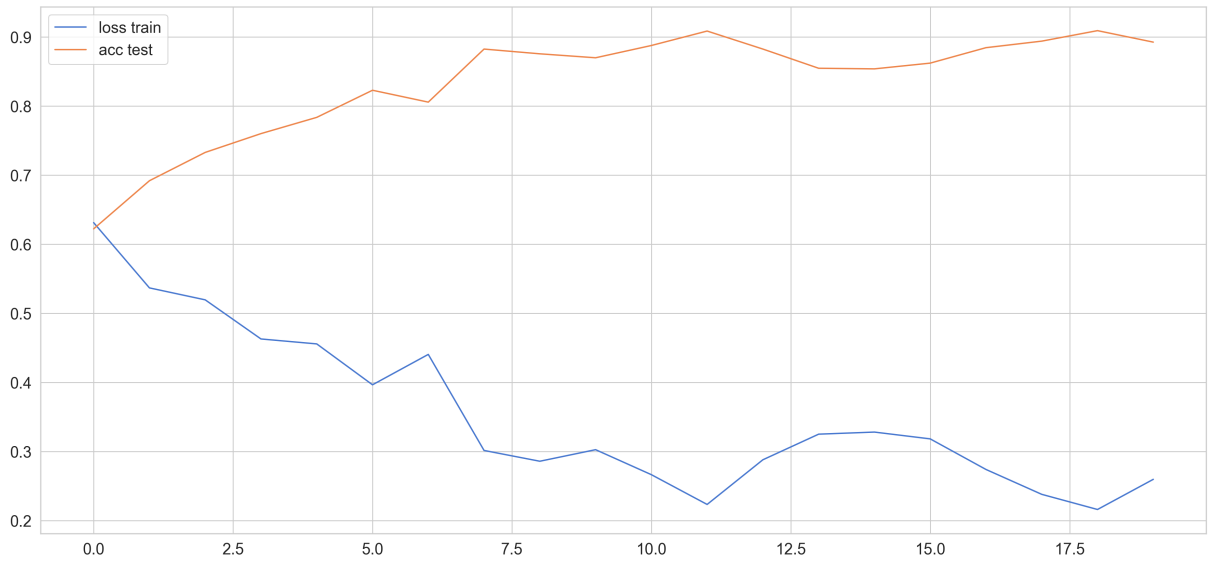


Figura 15 – Valores de perda e accertividade durante o treinamento da rede neral LSTM feito em 20 epocas e com $\text{batch}_size = 64$

A figura 16 mostra o resultado da matriz de confusão obtida através da LSTM.

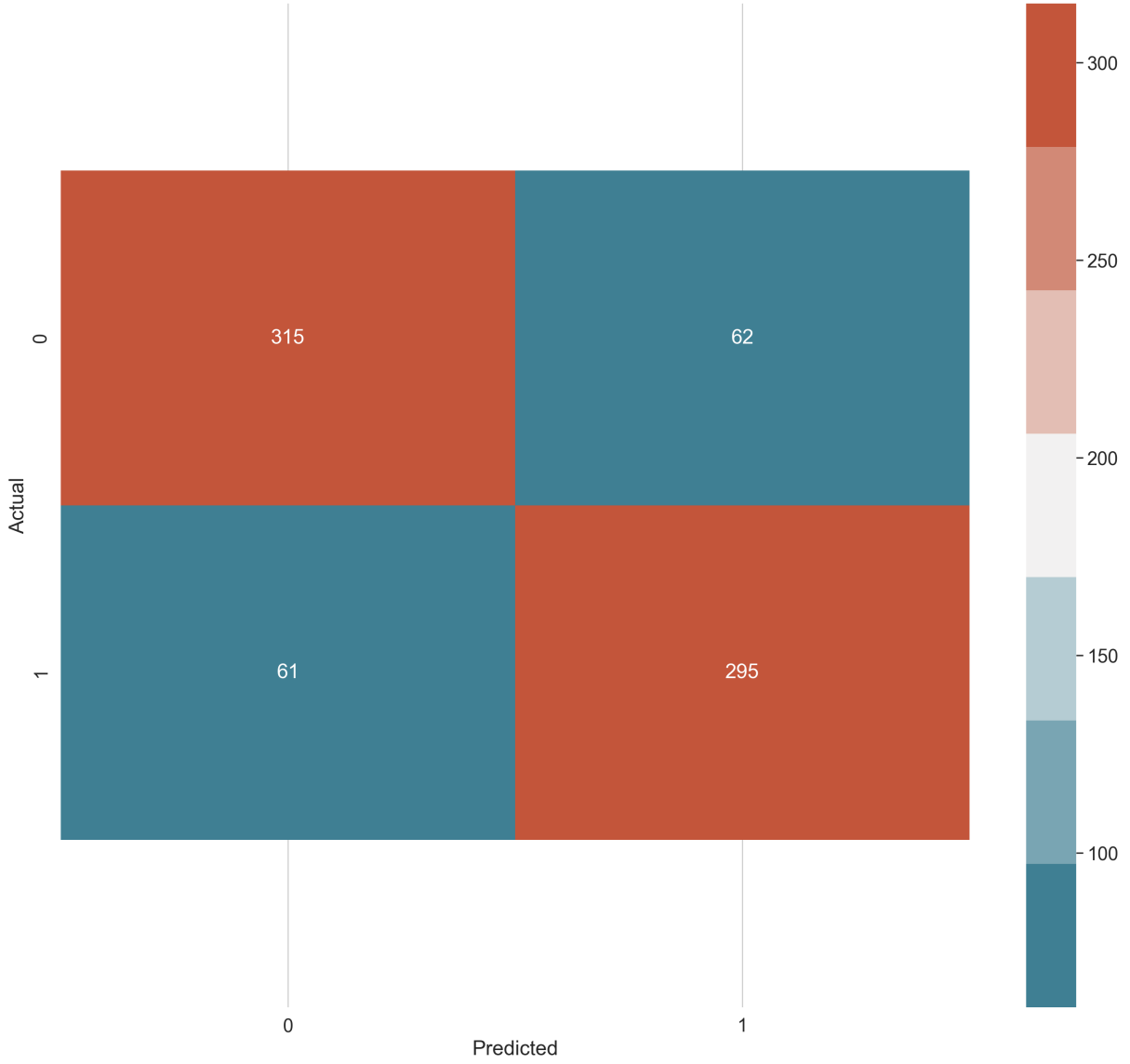


Figura 16 – Matriz de confusão resultante da LSTM

### 4.4.2 Rede Neural CNN-LSTM

O segundo teste foi feito com uma rede neural CNN-LSTM.

O código de implementação dessa rede:

```
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv1D, MaxPooling1D, Flatten, LSTM,
from tensorflow.keras.optimizers import Adam

# Daten erstellen
time_units = 200
```

```
num_features = 3

# Modell initialisieren
model_CNN_LSTM = Sequential()

# Erste CNN-Ebene
model_CNN_LSTM.add(Conv1D(32, 3, activation='relu', input_shape=(time_units,
model_CNN_LSTM.add(MaxPooling1D(2))

# Zweite CNN-Ebene
model_CNN_LSTM.add(Conv1D(64, 3, activation='relu'))
model_CNN_LSTM.add(MaxPooling1D(2))

# LSTM-Ebene
model_CNN_LSTM.add(LSTM(200, activation='relu'))

# Ausgabeebene
model_CNN_LSTM.add(Dense(2, activation='sigmoid'))

# Modell kompilieren
# Optimizer mit reduzierter Lernrate und Gradient Clipping
optimizer = Adam(learning_rate=0.00005, clipnorm=1.0) #0.00001
model_CNN_LSTM.compile(optimizer=optimizer, loss='binary_crossentropy', metri

# Modell bersicht anzeigen
model_CNN_LSTM.summary()
```

#### 4.4.2.1 Resultados Rede Neural CNN-LSTM

A figura 17 mostra os resultados de treinamento e perda da rede neural.

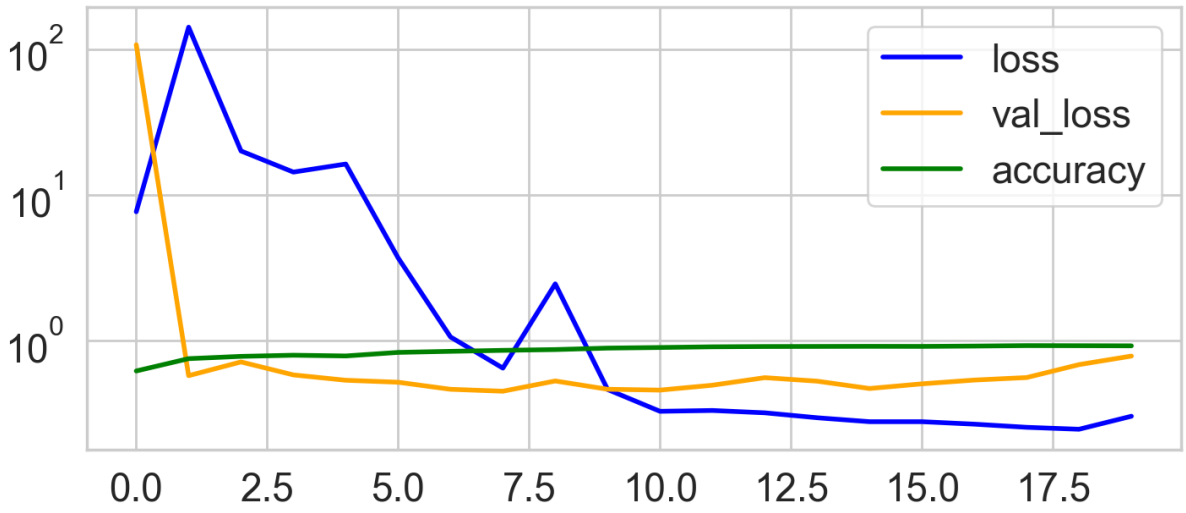


Figura 17 – Valores de perda e accertividade durante o treinamento da rede neral CNN-LSTM feito em 20 épocas e com batch$_size = 16$

A figura 18 mostra o resultado da matriz de confusão obtida atraves da CNN-LSTM.

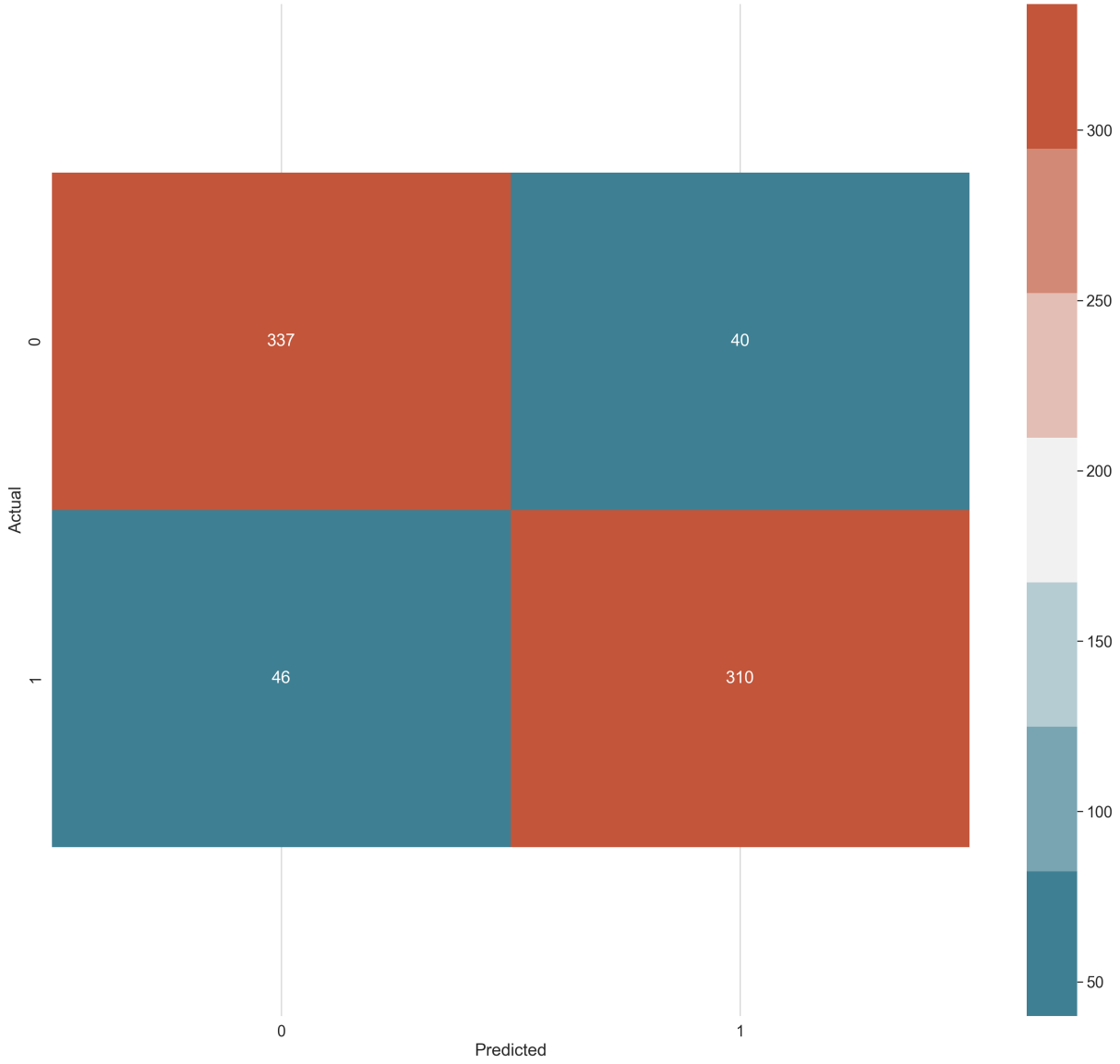


Figura 18 – Matriz de confusão resultante da CNN-LSTM

# 5 CONCLUSÕES

O tratamento dado aos dados demonstrou ter sumaria importância no desempenho da rede. Experimentos anteriores (não escritos neste documento) demonstravam que a rede era incapaz de classificar dados desbalanceados. A classificação dos eventos também deve ser revista, uma vez que a base de tempo descrita no data set esta sujeita a pequenos erros, o que influencia no resultado de aprendizagem e desempenho das redes.

Como esperado, a rede CNN-LSTM mostrou um melhor desempenho quando comparada a rede LSTM. A construção e o ajuste dos parâmetros das redes não são tarefas triviais. Os resultados podem e devem ser melhorados futuramente com um melhor refinamento da construção e ajuste dos parâmetros. Para isso, mais experimentos com diferentes combinações de parâmetros devem ser efetuados.

Ferramentas como o Copilot foram utilizadas no desenvolvimento deste trabalho, como parte do aprendizado sobre o tema.

Como somente os dados dos acelerômetros foram considerados, os dados do giroscópio também poderão trazer resultados melhores na identificação dos eventos.

Com a combinação do sinal dos acelerômetros mais giroscópio, uma tentativa de qualificar os eventos conforme descritos pela tabela 1 poderá ser feito.

A execução do algoritmo em um aplicativo de celular ou em um sistema embarcado que possui sensores semelhantes também é uma tarefa interessante que poderá trazer resultados promissores.

# REFERÊNCIAS

BOSCH. **Bosch MEMs Sensors**. 2025. Disponível em: https://www.bosch-mobility.com/de/loesungen/elektronische-bauelemente/mems-sensoren/.

CALA, L. L. **Recognition and Tracking of Vehicles in Highways using Deep Learning**. 2019. Tese (Doutorado) — Universidade de São Paulo, 2019.

CHO, K. *et al.* Learning phrase representations using rnn encoder-decoder for statistical machine translation. **arXiv preprint arXiv:1406.1078**, 2014.

CHUNG, J. *et al.* Empirical evaluation of gated recurrent neural networks on sequence modeling. 12 2014.

FERREIRA, J. *et al.* Driver behavior profiling: An investigation with different smartphone sensors and machine learning. **PLOS ONE**, v. 12, p. 1–16, 04 2017.

GABRIEL, K. *et al.* **Small Machines, Large Opportunities: A Report on the Emerging Field of Microdynamics : Report of the Workshop on Microelectromechanical Systems Research ; Sponsored by the National Science Foundation**. AT & T Bell Laboratories, 1988. Disponível em: https://books.google.de/books?id=7lKytgAACAAJ.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. **Neural Computation**, v. 9, n. 8, p. 1735–1780, 1997.

IBM. **What are recurrent neural networks?** 2024. Disponível em: https://www.ibm.com/topics/recurrent-neural-networks.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. **nature**, Nature Publishing Group, v. 521, n. 7553, p. 436, 2015.

ROSENBLATT, F. The perceptron: A probabilistic model for information storage and organization in the brain. **Psychological Review**, v. 65, n. 6, p. 386–408, 1958. ISSN 0033-295X. Disponível em: http://dx.doi.org/10.1037/h0042519.

WILLERS, O. *et al.* On the feasibility of deriving cryptographic keys from mems sensors. **Journal of Cryptographic Engineering**, v. 10, 04 2020.