Bruno Santoro Carlos

SmartRacket

São Paulo
2016

Bruno Santoro Carlos

SmartRacket

Monografia apresentada à Escola Politécnica da Universidade de São Paulo para a obtenção do título de Engenheiro Eletricista.

Área de Concentração:
Departamento de Engenharia de Computação e Sistemas Digitais – PCS

Orientador: Professor Doutor Bruno de Carvalho Albertini

São Paulo
2016

Este trabalho é dedicado à Deus.

Agradecimentos

Os agradecimentos são direcionados ao Prof. Dr. Bruno de Carvalho Albertini por todo apoio, dedicação, paciência e ensinamentos transmitidos durante esse ano de realização do trabalho.

Agradecimentos também ao meu pai - João Carlos, à minha mãe - Elisabeth Carlos, à e minha irmã - Viviane Carlos, por todo suporte, amor e carinho direcionados a mim durante todos os meus anos de faculdade, sem os quais eu jamais teria conseguido alcançar essa conquista.

Por fim a meus amigos pelas orações que me deram forças para finalizar esse trabalho e pela compreensão da minha ausência durante esse ano e a todos os colaboradores diretos e indiretos na execução desse projeto.

"Quanto a mim, estou a ponto de ser imolado
e o instante da minha libertação se aproxima.
Combati o bom combate, terminei a minha carreira,
guardei a fé. Resta-me agora receber a coroa da
justiça, que o Senhor, justo Juiz, me dará naquele
dia, e não somente a mim, mas a todos aqueles que
aguardam com amor a Sua aparição.
(Bíblia Sagrada, II Timóteo 4, 6-8)

Resumo

O objetivo que se quis alcançar com este projeto foi a análise dos movimentos da raquete de um tenista amador ou profissional em seus treinos e jogos, a fim de trazer informações relevantes ao esportista, sobre seus pontos fortes e fracos. Obteve-se dados coletados por um sensor desenvolvido pelo próprio aluno e, a partir da análise desses dados, construiu-se uma rede neural capaz de reconhecer as rebatidas de um jogador de tênis. Vale ressaltar que a rede neural foi treinada para três tipos de movimentos: batida de direita, batida de esquerda e outros (voleio de direita e de esquerda). Com a premissa de que é possível reconhecer as jogadas de um tenista através de rede neural confirmada, o resultado desta pesquisa foi transformado em um produto. Modelou-se uma caixa para proteger o sensor criado e, ao mesmo tempo, ser encaixada em uma raquete de tênis. Criou-se, também, um *software* capaz de ler os dados coletados pelo sensor e mostrá-los ao usuário em uma forma amigável e de fácil entendimento.

Palavras-chave: Sensor. Rede neural. Machine learning. Tênis.

Abstract

The main goal of this project was the analysis of the racket moves of a amateur or pro tennis player at their game or practice session. Data were collected by a sensor developed by the student and it was developed a neural network able to recognize how a tennis player hits was constructed. It is noteworthy that the neural network was trained for three types of movements: forehand, backhand and others (right and left volley). The result of this research has been transformed into a product once confirmed the premise that it is possible recognize the moves of a tennis player using neural network. A box was designed (modeled up a box) to protect the sensor and, at the same time, to fit into (be embedded in) a tennis racket. A software was also created to read the data collected by the sensor and to show them to the user in a friendly and easy-to-understand way.

Keywords: Sensor. Neural network. Machine learning. Tennis.

Lista de ilustrações

Figura 1 – Arquitetura de *IoT* 15
Figura 2 – Força média no impacto de esquerdas curta e longa 18
Figura 3 – Estrutura conceitual de uma RNA 19
Figura 4 – Arduino Nano original 22
Figura 5 – Arduino Nano CH430g 22
Figura 6 – IMU 9DOF LSM9DS0 24
Figura 7 – RTC DS3231 26
Figura 8 – Módulo microSD 27
Figura 9 – Arduino IDE 28
Figura 10 – Atom 29
Figura 11 – WEKA 30
Figura 12 – Ligações dos componentes ao Arduino 31
Figura 13 – Protótipo conectado a *protoboard* 31
Figura 14 – Protótipo ligado a raquete 32
Figura 15 – Sensor criado pelo autor 33
Figura 16 – Gráfico do módulo da aceleração de um movimento isolado 35
Figura 17 – Gráfico do módulo da velocidade de um movimento isolado 35
Figura 18 – Normalização dos valores 39
Figura 19 – Resultado do treino da RNA utilizando o programa WEKA 41
Figura 20 – Matriz de confusão 41
Figura 21 – Servidor para análise dos dados 42
Figura 22 – Servidor para análise dos dados 43

Lista de tabelas

Tabela 1 – Explicação do formato de armazenamento definido dos dados coletados pelo sensor 22
Tabela 2 – Características do IMU 9DOF LSM9DS0 25
Tabela 3 – Comparação dos classificadores J48, BayesNet e *MultilayerPerceptron* . 40

Lista de abreviaturas e siglas

CITI-USP	Centro Interdisciplinar em Tecnologias Interativas da Universidade de São Paulo
DOF	*Nine Degrees of Freedom*
IMU	*Inertial Measurement Unit*
IDE	*Integrated Development Environment*
IoT	Internet of Things
RNA	Rede Neural Artificial
RTC	*Real Time Clock*
USB-B	*Universal Serial Bus* do tipo B
UTC	*Coordinated Universal Time*
WEKA	*Waikato Environment for Knowledge Analysis*

Lista de símbolos

dps	*Degrees per second*
g	Unidade de aceleração da gravidade terrestre
G	Gauss
kB	Quilobyte
kgf	Quilograma-força
mAh	Miliampère-hora
N	Newton
V	Volt

Sumário

	Introdução	**13**
1	**REVISÃO DA LITERATURA**	**16**
1.1	**Concorrentes**	**19**
2	**MATERIAIS E MÉTODOS**	**21**
2.1	**Materiais**	**21**
2.1.1	Hardware	21
2.1.1.1	Sensor	21
2.1.1.1.1	Arduino CH430g	21
2.1.1.1.2	IMU 9DOF LSM9DS0	23
2.1.1.1.3	RTC DS3132	25
2.1.1.1.4	Módulo microSD	26
2.1.1.2	Materiais auxiliares	26
2.1.2	Software	27
2.1.2.1	Software do Arduino	27
2.1.2.2	Software de análise dos dados	28
2.1.2.3	WEKA	29
2.2	**Métodos**	**30**
2.2.1	Hardware	30
2.2.2	Software	33
2.2.2.1	Software do Arduino	33
2.2.2.2	Software de análise dos dados	34
2.2.2.2.1	Identificação do início e do fim do movimento	34
2.2.2.2.2	Padronizar a entrada de dados	38
2.2.2.2.3	Reconhecimento do movimento	38
3	**RESULTADOS**	**41**
4	**CONCLUSÃO**	**44**
4.1	**Discussão**	**44**
4.2	**Trabalhos futuros**	**45**
	REFERÊNCIAS	**46**

Introdução

O tênis é um esporte em que os movimentos do jogador são essenciais para um alto desempenho. Como se pode ver em (BALBINOTTI, 2009), a aprendizagem dos golpes básicos do esporte são abordadas como fundamentos desde o primeiro contato com o tênis.

Os golpes básicos estão presentes no jogo de qualquer tenista e ele decide quando usá-los durante o jogo. Ainda de acordo com (BALBINOTTI, 2009), tem-se uma descrição de cada um dos movimentos:

1. Direita (*Forehand*) - Não possui uma tradução direta do inglês. Esse movimento é conhecido como direita e é o principal golpe de um tenista. O movimento começa no mesmo lado do braço que o jogador segura a raquete e termina no lado oposto. Ele é igualmente importante em situações de ataque e defesa, porém é normalmente utilizado em momentos que definirão quem é o ganhador do ponto.

2. Esquerda (*Backhand*) - Também não possui tradução direta do inglês. É conhecido como esquerda e é o segundo golpe mais utilizado pelos tenistas. O movimento é oposto ao golpe de direita, ou seja, começa no lado oposto ao braço que o jogador segura a raquete e termina no outro lado. Normalmente, não é tão sólido quanto a direita e por isso é um golpe utilizado mais em situações de defesa.

3. Voleio (*Volley*) - O tenista faz o contato com a bola antes dela tocar no chão do seu lado da quadra. É usado quando o jogador está próximo a rede e tem como finalidade diminuir o tempo de resposta do adversário e, por consequência, colocá-lo em uma situação de defesa.

4. *Slice* - Nesse movimento, parece que o tenista está "cortando" a bola, ou seja, ele aumenta o tempo de contato das cordas da raquete com a bola de tênis, transferindo, assim, mais momento angular para ela. Após o golpe acontecer e depois do contato com o chão, a bola não se eleva a uma altura ideia para o adversário golpeá-la. Esse golpe é popularmente conhecido como "deixadinha".

5. *Lob* - Golpe característico de defesa do jogador. Quando o adversário está próximo à rede, o tenista golpeia a bola de tênis, fazendo um movimento de baixo para cima com a finalidade de encobrir o adversário, fazendo com que ele se afasta da rede.

6. Saque (*Serve*) - Golpe utilizado no começo de cada ponto disputado. O tenista arremessa a bola para cima e a golpeia antes dela tocar no chão. O objetivo é acertar uma área delimitada da metade da quadra do adversário.

7. *Smash* - A movimentação desse golpe é muito similar ao saque. Uma diferença é que esse golpe acontece durante a disputa do ponto e não no começo dele, como é o caso do saque. Outra diferença é que o tenista não tem área delimitada para acertar, basta ele golpear a bola no interior da quadra adversária.

A motivação para esse estudo se decorre do fato do tênis ser um esporte fundamentado em repetições, uma vez que os treinos são sessões de rebater bolas incessantemente. Um treino comum para um tenista amador é fazer cem repetições de cada golpe básico. Assim, a mecânica para cada um dos golpes se torna cada vez mais natural para o tenista. Por consequência disso, realizar um golpe de maneira mais correta possível é essencial não somente para melhorar o desempenho do esportista, mas também para evitar que esse sofra alguma lesão por repetir movimentos errados. Como foi citado em (SCHLEGEL, 2014), "a forma como os tenistas passaram a golpear a bola é um dos motivos para as contusões mais freqüentes no quadril e na virilha em atletas ligados ao esporte", corroborando com as afirmações anteriores.

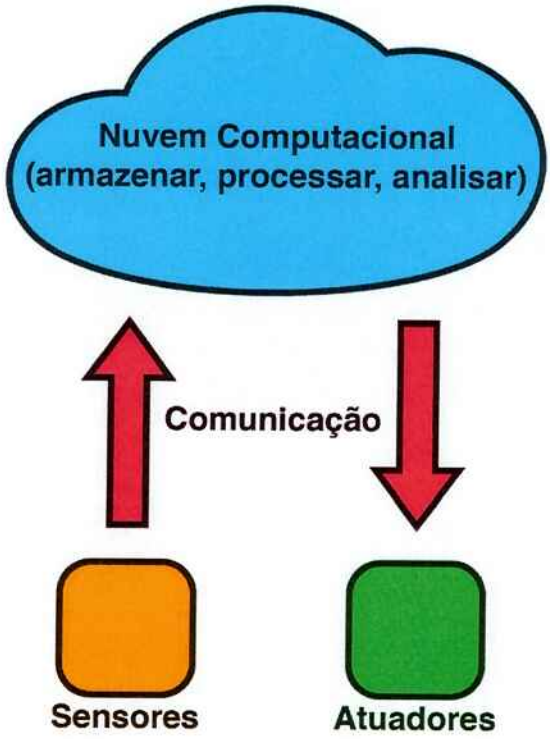
Em outros esportes, encontram-se casos em que a tecnologia é utilizada para melhorar aspectos do treino e, consequentemente, do jogo. Dois exemplos podem ser citados. No basquete, utiliza-se a análise dos movimentos de um atleta a fim de melhorar seu desempenho, prever lesões ou otimizar os treinos. Utilizam-se para isso câmeras e vestíveis que coletam informações de um jogador, que serão analisadas para obter os benefícios citados (NGUYEN et al., 2015). Outro exemplo ilustrativo é abordado em (GHASEMZADEH et al., 2009) que é um estudo sobre o *swing* de um golfista e como a análise desse movimento ajuda a aprimorar a qualidade de seus treino e jogo. A arquitetura de coleta de dados dos artigos citados segue o modelo apresentado em (XIA et al., 2012) de *Internet of Things* (IoT) que se trata de um conjunto de sensores que fazem medições esporádicas ou quando acontecem eventos predeterminados e mandam esses dados para uma nuvem computacional que será responsável por armazenar, processar, analisar e transmiti-los (Figura 1).

Esse estudo buscou responder as seguintes perguntas:

1. É possível desenvolver um sensor capaz de coletar dados suficientes dos golpes de um tenista? Esses dados são conclusivos?
2. Supondo que a coleta tenha gerado dados conclusivos, é possível, a partir desses dados, classificar um movimento e diferenciá-lo de outro? É possível comparar esse resultado com um padrão pré estabelecido? Existe um padrão ou varia de tenista para tenista?

Este trabalho teve como objetivos a montagem e programação de um dispositivo capaz de coletar dados de um tenista, bem como a implementação de um software que

Figura 1 – Arquitetura de *IoT*



Fonte: Autor.

interprete essas informações e que permita uma leitura simples ao usuário. Baseando-se na arquitetura de IoT, foi colocado um sensor, que mede aceleração, velocidade angular e direção do movimento do tenista, em um local fixo da raquete enquanto ele praticava o esporte. A partir desses dados, foram feitas análises utilizando o conceito de Rede Neural Artificial (RNA) para mapear os golpes de um tenista e fazer uma comparação com um movimento padrão para o qual a RNA foi treinada. Como resultados, obtiveram-se a classificação do movimento (direita, esquerda, voleio etc.) bem como a porcentagem do quão perto o movimento realizado pelo tenista está do movimento padrão. Essas informações foram transmitidas ao usuário de forma visual através do gráfico da variação do módulo da aceleração dos golpes realizados, junto com a porcentagem que indica o quão próximo cada golpe está do golpe referência.

1 Revisão da literatura

Na literatura formal, encontrou-se um artigo que analisa o *swing* de golfistas e mostra como os resultados obtidos podem influenciar positivamente o treino e o "jogo de um jogador" (Nesta monografia, define-se "jogo" como o estilo que o jogador apresenta ao praticar um esporte). Fazendo um paralelo com o estudo desta pesquisa, aproveitou-se a experiência que os autores do artigo citado demonstraram para alcançar o objetivo proposto neste trabalho, ou seja, o que foi utilizado como coletor de informação, como foi agregado esse sensor ao taco golfe, quais dados são relevantes e quais não são e erros cometidos no processo. Também como um auxílio, estudou-se os resultados para prever o que seria obtido no final deste trabalho. O artigo é o (GHASEMZADEH et al., 2009).

Tendo esboçado um caminho a seguir, procurou-se pesquisas para validar as duas questões principais que viabilizam a estratégia escolhida:

1. É possível utilizar algum sensor para captar o impacto de uma bola de tênis na raquete?
2. Existe alguma técnica capaz de diferenciar um movimento de outro do mesmo tenista?

Essas questões foram respondidas respectivamente pelo artigo (WU et al., 2001) e pelo conceito apresentado em (PATTERSON, 1996) e (CARVALHO, 2016).

Artigo "Sport Training Using Body Sensor Networks: A Statistical Approach to Measure Wrist Rotation for Golf Swing"

Este artigo discorre sobre como o a análise do treino de um atleta é benéfico para seu desenvolvimento e tem como objetivo investigar o desenvolvimento de um sistema de treino assistido para golfistas novatos. Assim como no tênis, no golfe um movimento consistente é essencial para uma boa tacada. Não apenas a movimentação dos braços deve ser analisada, mas o *swing* do corpo todo. No *paper* é proposto um sistema para isso.

Foram colocados 5 sensores criados pelos autores: dois embaixo de cada ombro, um na mão de apoio da tacada e dois no taco (um no meio e outro na ponta oposta à mão do jogador). Esses sensores são compostos de um microcontrolador - responsável por armazenar e processar as informações coletadas -, um transmissor de rádio frequência, um acelerômetro com três graus de liberdade e um giroscópio com dois graus de liberdade.

Utilizou-se a técnica de processamento de sinais nos dados coletados para se criar um modelo quantitativo. Tomando esse modelo como referência, conseguiu-se analisar a porcentagem de acerto que cada tacada obtive.

A pesquisa descrita neste documento baseia-se no artigo de golfe. Seguiram-se os mesmos passos de coleta com sensores colocados na raquete de tênis, porém não foi possível a utilização da técnica de comparação com um movimento referência já que as rebatidas de um tenista variam muito, principalmente entre amadores e profissionais, e este projeto visa a identificação de movimentos de qualquer tenista. No caso desse trabalho, utilizou-se a técnica de RNA.

Artigo "Comparison of Ball-and-Racquet Impact Force Between Two Tennis Backhand Stroke Techniques"

Este documento apresenta um estudo da força no impacto de golpes de esquerda de tenistas amadores e profissionais. Os movimentos descritos são enquadrados em duas categorias: longa esquerda (*long backswing*) e curta esquerda (*short backswing*).

Como o estudo deste projeto de conclusão de curso necessita saber a máxima força que um sensor precisa aguentar em uma raquetada, desconsideraram-se os dados dos tenistas amadores e as informações coletadas dos movimentos longos de esquerda de profissionais, já que esses apresentaram forças médias inferiores.

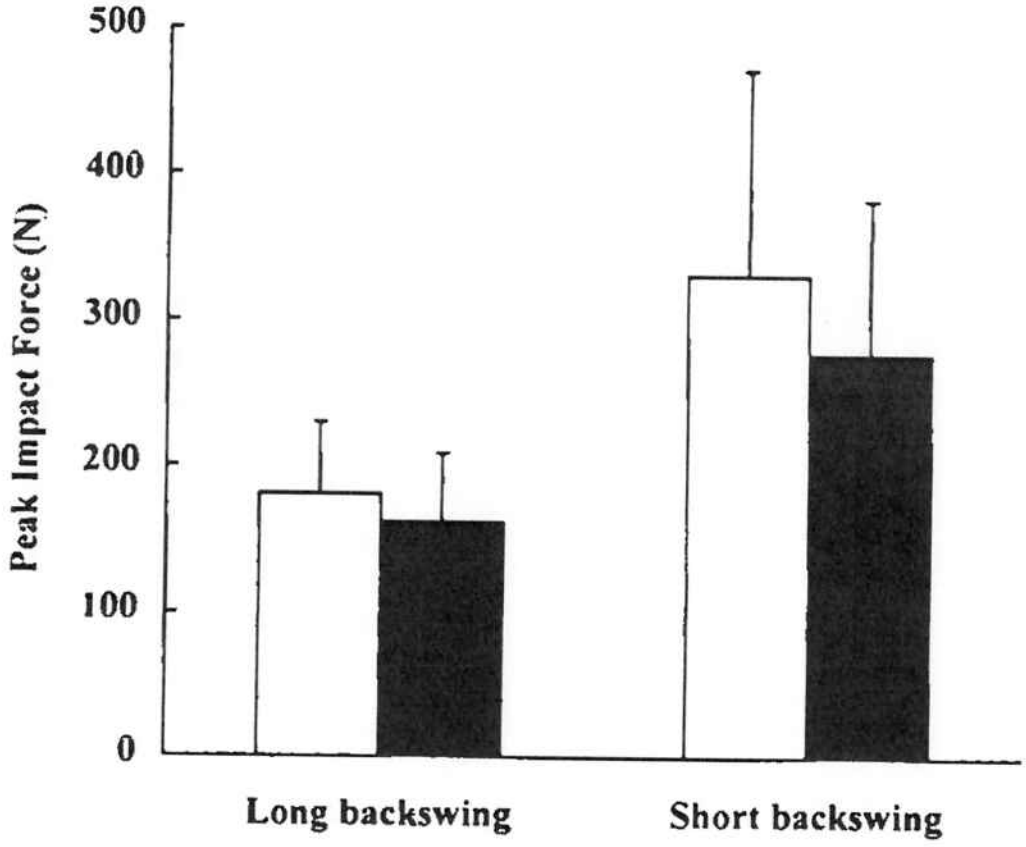
Como pode ser visto na Figura 2, a máxima força média de impacto é de 477,7 N - dados da esquerda curta de um profissional: $(330,0 \pm 147,7)$ N -, que é equivalente a 48,711843 kgf. Esse resultado na literatura respondeu a primeira pergunta que foi a motivação para esta pesquisa, pois existem sensores, como acelerômetros, capazes de trabalhar nesta faixa.

Redes Neurais Artificiais

A confirmação da segunda pergunta sobre a existência de uma técnica capaz de diferenciar rebatidas do mesmo tenista, apresentada no começo deste capítulo, veio com o conceito apresentado nesta seção.

O modelo de Redes Neurais Artificiais (RNAs) é inspirado no funcionamento da estrutura neural de organismos inteligentes. O sistema nervoso é composto de bilhões de células responsáveis pelo funcionamento do corpo, os neurônios. Esses têm um terminal de entrada (dendritos), responsável pela entrada do estímulo que chega de fora, e um terminal de saída (axônios), responsável pela saída do estímulo para outro neurônio ou outra célula. Os neurônios estão conectados entre si e fazem a locomoção do impulso de um lugar para

Figura 2 – Força média no impacto de esquerdas curta e longa



Fonte: (WU et al., 2001)

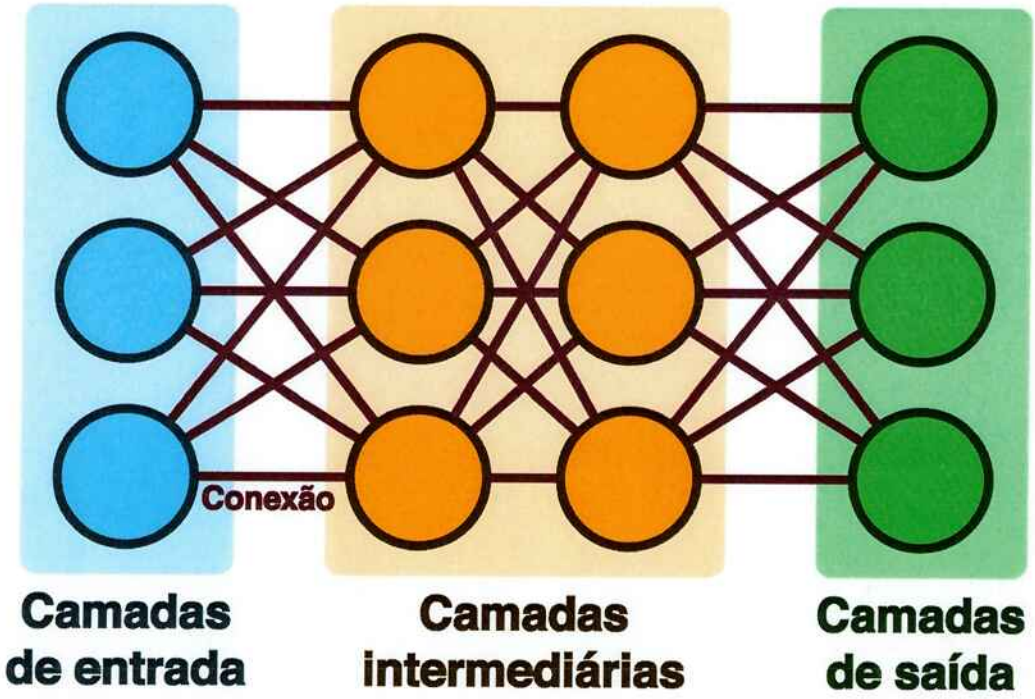
outro formando uma rede de neurônios.

Como pode ser visto na Figura 3, as RNAs possuem unidades responsáveis pelo processamento dos dados internos a elas, fazendo o papel dos neurônios. Essas unidades estão conectadas entre si por conexões e essas têm pesos diferentes, representando os dendritos e axônios. O funcionamento segue quatro passos:

1. Sinais são recebidos nas conexões de entrada
2. Os sinais são multiplicados pelos pesos das conexões
3. Somam-se todos os valores já multiplicados
4. Se a soma passar de um mínimo, a saída envia um sinal específico

Essa arquitetura pode ser treinada. Carrega-se, na camada de entrada, dados em um formato específico e indicando o que eles representam. A RNA cuida de criar n camadas intermediárias com conexões e pesos específicos. Como saída, tem-se os dados classificados e uma rede neural treinada para identificar padrões de casos específicos do problema. Outro

Figura 3 – Estrutura conceitual de uma RNA



Fonte: Autor.

dado fundamental para esse projeto é a porcentagem de quão perto um dado está dos dados que foram utilizados para treinar a RNA, ou seja, o quanto um dado se aproxima de um dado referência.

Essa foi a abordagem escolhida para este trabalho, ou seja, coletar dados de cada movimento desejado e entrar com essas informações em uma RNA para capacitá-la à classificar os golpes para os quais ela foi treinada.

Como este conceito é bastante conhecido na academia e não é foco deste trabalho desenvolver RNAs, mas sim aplicá-las, indica-se a referência (PATTERSON, 1996).

1.1 Concorrentes

Existem duas soluções no mercado que entregam resultados parecidos com os objetivos desse trabalho: Babolat PLAY e *Smart Tennis Sensor* da SONY.

A Babolat é uma marca consolidada no tênis. Ela oferece produtos para tenistas, como roupas, equipamentos e acessórios. Uma linha de raquetes que ela oferece é a PLAY, que é composta de três tipos de raquetes para que o jogador possa escolher a que melhor que adeque ao seu estilo de jogo. Todas as raquetes PLAY possuem sensores espalhados por elas e são capazes de se conectar via *Bluetooth* com uma aplicação móvel desenvolvida pela Babolat. Os dados coletados por esses sensores são mostrados para o usuário através

desse aplicativo (BABOLAT, 2016).

Já a SONY é uma empresa consolidada no mercado de tecnologia. Dentre sua variedade de produtos, ela oferece o *Smart Tennis Sensor*, um sensor capaz de ser acoplado em alguns modelos de raquetes e que é capaz de enviar informações de seus sensores para uma aplicação desenvolvida por ela. Também se conecta através de *Bluetooth* com o celular (SONY, 2016).

Ambas as soluções oferecem funcionalidades similares, como:

1. Quantidade de golpes de direita, esquerda, voleios, saque e *smash* em cada treino/jogo.
2. Potência em cada golpe realizado pelo tenista.
3. Quantidade de efeito aplicado na bola pelo jogador.
4. Velocidade do saque.
5. Mapa de calor do local de impacto da raquete com a bola.
6. Histórico de todas as informações coletadas pelo sensor para que o usuário possa acompanhar seu progresso.
7. Possibilidade de compartilhar essas informações com seus amigos.

A diferença entre as soluções é que a Babolat exige uma raquete específica e a SONY possui um sensor que pode ser acoplado em vários modelos de raquetes.

A solução da Babolat exige que o jogador compre outra raquete, mesmo que ele já possua outras, caso queira ter a funcionalidade oferecida pela linha PLAY. O modelo da raquete é muito pessoal ao estilo do tenista e a Babolat PLAY exige que o jogador escolha uma das três raquetes da linha se ele quiser monitorar seu jogo com o aplicativo. Além desses contra, o aplicativo não foi traduzido para o português e o preço é elevado se comparado com outras raquetes no mercado. Na loja oficial da Babolat no Mercado Livre, a raquete Babolat *Pure Drive* PLAY custa R$ 2619,90.

A solução da SONY é mais adaptável por se tratar de um sensor que pode ser acoplado no cabo da raquete. Mesmo assim, são apenas quatro marcar de raquetes que são compatíveis com o sensor. O produto tem um custo menor (US$ 149,99 no site oficial), porém não está disponível no Brasil. O jogador interessado teria que importar e isso pode encarecer o produto. Além disso, o aplicativo não está disponível em nenhuma loja brasileira de aplicativos (Play Store, App Store e Windows Store) nem foi traduzido para o português.

Nenhuma das duas oferece a comparação de seus movimentos com um movimento referência. Essa é a funcionalidade que se procurou realizar com esse projeto.

2 Materiais e métodos

A seção 2.1 é dividida em duas partes distintas, porém complementares: *hardware*, que descreve o motivo da utilização e a função de cada componente físico usado na construção do sensor, e *software*, que justifica, da mesma maneira, a utilização das ferramentas usadas na implementação dos programas auxiliares desenvolvidos.

2.1 Materiais

2.1.1 Hardware

O dispositivo é composto de um microcontrolador, que é o controlador de todos os módulos adjacentes utilizados, e de três sensores: acelerômetro, giroscópio e magnetômetro. O acelerômetro é responsável por capturar os três eixos da aceleração do movimento. O giroscópio faz a mesma coisa para a velocidade de rotação e o magnetômetro exerce o mesmo papel para o campo magnético. Além dos sensores citados, o dispositivo possui um *Real Time Clock* (RTC), que captura o *timestamp* no momento da medição, e um módulo microSD, que será o armazenador dos dados coletados. Portanto, o protótipo criado é responsável por capturar a aceleração, a velocidade angular e a direção dos movimentos da raquete de tênis e será acoplado na parte debaixo do seu cabo. Por decorrência disso, uma característica fundamental do protótipo é seu tamanho reduzido.

Uma vez capturados os nove dados - os três eixos de cada sensor, esses e o *timestamp* são armazenados, através do módulo microSD, em um cartão de memória e a informação tem o formato dd/MM/AAAA HH:mm:ss:ccc a_x a_y a_z g_x g_y g_z m_x m_y m_z, que está explicado na Tabela 1. Vale ressaltar que o *timestamp* será guardado em um cartão no formato *Universal Time Coordinated* (UTC).

Para a construção do dispositivo citado foram utilizados componentes de *hardware* e alguns materiais auxiliares. Todos esses módulos são descritos nesta subseção.

2.1.1.1 Sensor

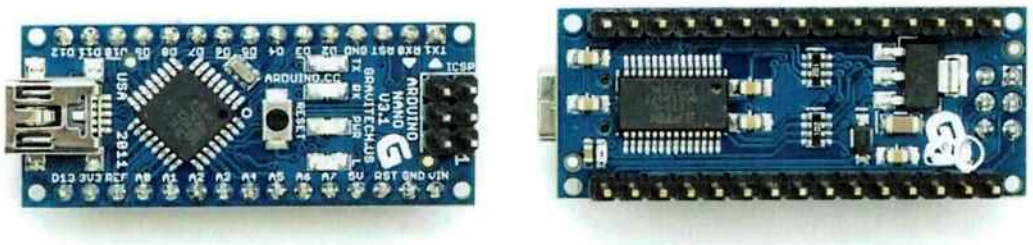
2.1.1.1.1 Arduino CH430g

O microcontrolador escolhido foi um clone chinês do Arduino Nano conhecido como Arduino Nano CH430g. Ele segue a mesma arquitetura de um Arduino Nano original baseado na placa ATmega328 (Arduino Nano 3.x), tem exatamente as mesmas características e possui a mesma aparência, como se pode verificar nas Figuras 4 e 5.

Tabela 1 – Explicação do formato de armazenamento definido dos dados coletados pelo sensor

Código	Explicação
dd	dia do mês (01 – 31)
MM	mês (01 – 12)
AAAA	ano
HH	hora (00 – 23)
mm	minuto (00 – 59)
ss	segundo (00 – 59)
ccc	milissegundo (000 – 999)
a_x a_y a_z	três eixos da aceleração em g
g_x g_y g_z	três eixos do giroscópio em dps
m_x m_y m_z	três eixos do magnetômetro em G

Figura 4 – Arduino Nano original



(a) Vista de cima (b) Vista de baixo

Fonte: (ARDUINO, 2016a)

Figura 5 – Arduino Nano CH430g

(a) Vista de cima (b) Vista de baixo

Fonte: (MECHATRONICS, 2016)

O Arduino Nano escolhido possui entradas digitais e analógicas, podendo trabalhar com elas de forma simultânea. Pode ser alimentado por uma entrada *Universal Serial Bus* do tipo B (USB-B) ou pelos pinos de alimentação de $6 - 20$ V desregulado ou 5 V regulado. O modelo utilizado (ATmega328) possui uma memória de 32 kB. Para se codificar programas para o Arduino, pode-se usar o *software* da própria marca Arduino que será abordado com mais detalhes na subseção 2.1.2 (ARDUINO, 2016a).

O protótipo exige dois aspectos do microcontrolador: tamanho reduzido para poder ser encaixado na parte inferior do cabo de uma raquete de tênis e capacidade suficiente de processamento.

Uma pesquisa de mercado trouxe como resultados diversos microcontroladores, como Arduino Nano v3, Teensy 2.0, Teensy++ 2.0, MSP430 LaunchPad, Pinguino PIC32. Reduziu-se o número de opções avaliando as dimensões desses dispositivos.

Como o protótipo fará operações de leitura e escrita em um cartão de memória e aquisição de dados de um sensor, o processamento não foi decisivo na escolha, já que todos os microcontroladores selecionados anteriormente tinham capacidade de realizar essas tarefas de acordo com exemplos de códigos dos próprios fabricantes de cada componente. Portanto, levou-se em consideração mais dois aspectos: preço e disponibilidade.

O Arduino Nano CH430g é um microcontrolador de fácil acesso no Brasil. Inclusive, como experiência do autor, foi o único microcontrolador, do resultado de toda a pesquisa realizada, encontrado em São Paulo. O preço também foi levado em conta já que se pagaria cerca de cinco vezes mais no Arduino italiano original. O custo do Arduino Nano clone foi R$ 23,99 + frete no Mercado Livre, e o valor do Arduino Nano italiano é, aproximadamente, R$ 120,00 + impostos.

Estes quatro fatores foram utilizados para a escolha do microcontrolador, apesar de o poder de processamento não ter sido decisivo.

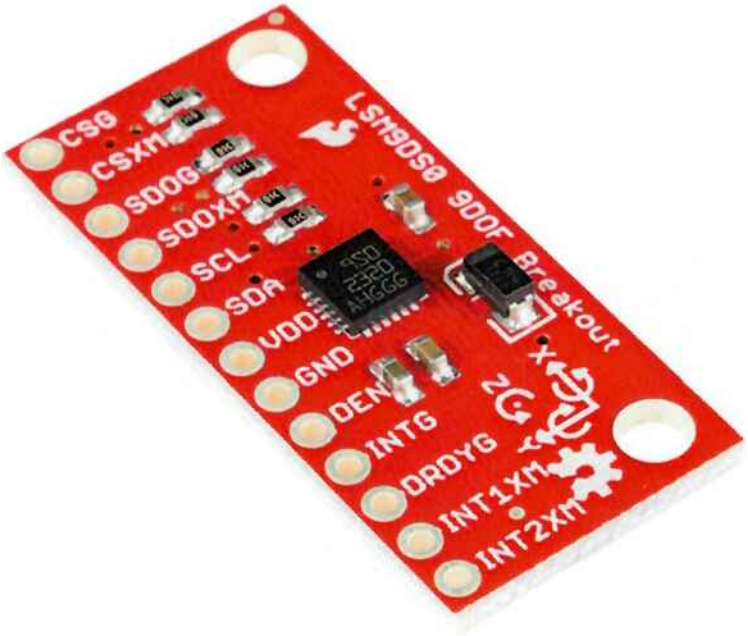
2.1.1.1.2 IMU 9DOF LSM9DS0

Com o intuito de reduzir o tamanho do dispositivo, decidiu-se usar um *Inertial Measurement Unit* (IMU) para os sensores de aceleração, taxa angular e campo magnético.

IMUs são dispositivos eletrônicos que combinam um acelerômetro, um giroscópio e, às vezes, um magnetômetro. Eles são capazes de capturar os valores dos três eixos de liberdade de cada um desses sensores que ele possui, ou seja, em uma só placa eletrônica, têm-se três sensores combinados. Essa característica se enquadra à exigência do protótipo deste documento, que é apresentar tamanho reduzido.

A escolha de um IMU pode se tornar muito complexa pela enorme quantidade de opções e combinações que se pode obter. Seguiu-se um guia de compras de IMUs da

Figura 6 – IMU 9DOF LSM9DS0



Fonte: (ELECTRONICS, 2016b)

SparkFun Electronics, uma fabricante americana do Colorado de microcontroladores e placas de circuito impresso, que podem ser encontradas em (ELECTRONICS, 2016a) e é descrito nesta subseção.

A primeira decisão é escolher um IMU com ou sem magnetômetro. Os IMUs que possuem magnetômetro geralmente são conhecidos como 9DOF ou *Nine Degrees of Freedom*, ou seja, conseguem coletar os três eixos da aceleração, os três da velocidade angular e os três do campo magnético, totalizando nove graus de liberdade.

O próximo passo foi a escolha de dois parâmetros dos sensores internos do IMU:

1. Faixa de medição: é a faixa para a qual um determinado sensor consegue coletar dados. Caso a informação capturada extrapole a faixa, o sensor lê como o valor máximo ou mínimo da faixa. Um exemplo para melhor compreensão é o acelerômetro. Existem acelerômetros com a faixa de ± 2 g. Isso significa que o sensor é capaz de coletar acelerações entre -2 g e $+2$ g. Caso extrapole para mais, como uma aceleração de 6 g, o acelerômetro medirá $+2$ g. Caso um acelerômetro possua uma faixa mais ampla, ele pode ser configurado para trabalhar em faixas menores.

2. Eixo: é a quantidade de graus de liberdade que cada sensor possui. Levando em consideração o exemplo acima, um acelerômetro pode ser capaz de coletar as acelerações em um, dois ou três eixos.

Tabela 2 – Características do IMU 9DOF LSM9DS0

Faixa	Comunicação	Eixo	Alimentação	Bônus
acel.: ±2, 4, 6, 8, 16 g giro.: ±245, 500, 2000 dps mag.: ±2, 4, 8, 16 G	SPI e I^2C	acel.: 3 giro.: 3 mag.: 3	2,4 – 3,6 V	Compatível com Arduino

No momento da escolha, não se sabia ao certo se o magnetômetro seria necessário para o projeto. Optou-se pelos modelos 9DOF porque se viesse a precisar das informações do campo magnético, não seria necessário gastar com outro dispositivo. Essas placas de circuito impresso não são encontradas facilmente e possuem um preço elevado em comparação com os outros módulos do projeto. Para o próximo passo da escolha da faixa e eixos de liberdade, levou-se em consideração a pesquisa comentada em (WU et al., 2001). Como o protótipo precisa localizar a exata posição da raquete do tenista, escolheu-se sempre trabalhar com três eixos para cada sensor.

Procurando na lista de IMUs da *SparkFun Electronics* com essas características apresentadas acima, optou-se pelo IMU 9DOF LSM9DS0 (Figura 6), cujas características estão resumidas na Tabela 2.

2.1.1.1.3 RTC DS3132

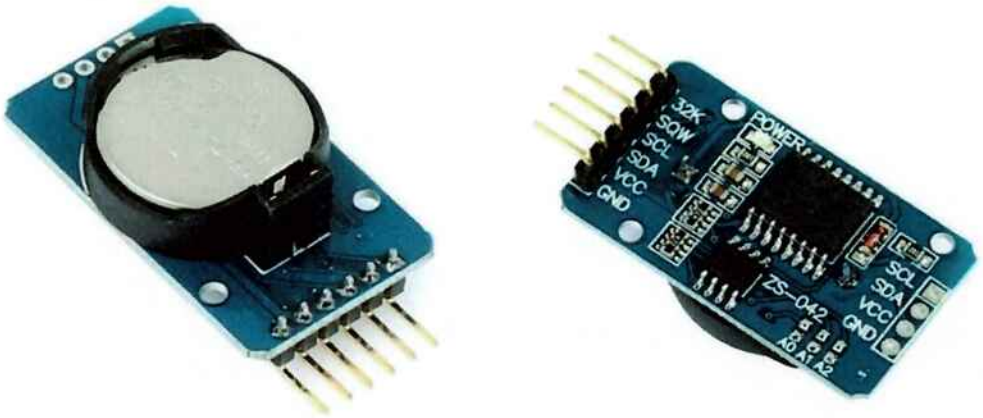
O tempo é um fator muito importante para a análise dos dados deste projeto. Por exemplo, a taxa de variação da aceleração é analisada e, sem o tempo como referência, esses dados são inconclusivos.

O próprio Arduino escolhido possui uma maneira de se extrair o tempo. A lógica é que se dispara um contador a partir do momento em que o programa carregado nele começa a funcionar. Pode-se ter acesso a esse contado através da função *millis()* da biblioteca padrão do Arduino. Como o próprio nome do método diz, ele retorna os milissegundos decorridos a partir do momento em que o programa começou a rodar. Pode-se tirar horas, minutos e segundos do valor da função *millis()* fazendo consecutivas divisões por sessenta (transformação de milissegundo para segundo, de segundo para minutos, de minutos para horas).

Essa abordagem não pode ser escolhida neste projeto por dois motivos. O primeiro é a perda da contagem em caso de falta de energia. Se por algum motivo o Arduino para de ser alimentado, ao ser religado, o contador dos milissegundos zera e a contagem de tempo decorrido é perdida. O segundo motivo é que o Arduino não é capaz de gravar a data atual, ou seja, não é possível se ter um histórico dos dados coletados.

Portanto, procurou-se uma solução diferente do padrão do Arduino. Escolheu-se, então, utilizar um *Real Time Clock* (RTC). O RTC é um módulo que pode ser usado

Figura 7 – RTC DS3231



(a) Vista de cima (b) Vista de baixo

Fonte: (FILIPEFLOP, 2016)

com o Arduino e tem a função de indicar para o microcontrolador a data e horário atuais. Diferente do Arduino, o RTC é alimentado por uma bateria, além da alimentação do módulo em si, que possibilita ao componente estar sempre contabilizando o horário.

A escolha do RTC levou em consideração o tamanho, a fim de minimizar as dimensões do protótipo, e a disponibilidade/preço (custo/benefício). Escolheu-se, por esses motivos, o RTC DS3231 que pode ser visto na Figura 7. Esse RTC contabiliza os meses com diferentes números de dias e ano bissexto automaticamente (FILIPEFLOP, 2016).

2.1.1.1.4 Módulo microSD

Como foi discutido na introdução da seção 2.1, o componente escolhido para armazenamento dos dados coletados foi um módulo microSD. Ele possui uma entrada para um cartão microSD. Pode-se ter acesso a esse cartão, utilizando uma biblioteca padrão do Arduino, e, assim, escrever e ler arquivos de texto contidos no SD.

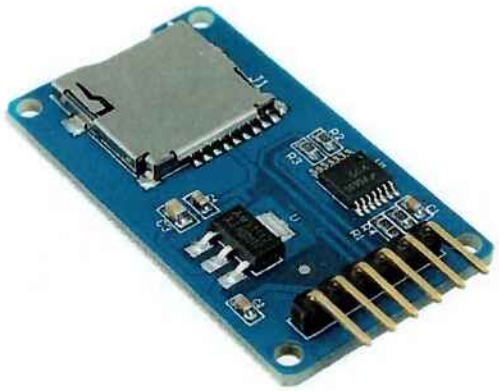
O critério de escolha foi similar ao utilizado para o RTC. Levou-se em consideração o tamanho e a disponibilidade/preço. Procurando no mercado, foi encontrado o módulo microSD ilustrado na Figura 8.

2.1.1.2 Materiais auxiliares

Além dos componentes de *hardware* descritos na subseção 2.1.1.1, utilizou-se materiais auxiliares na construção do protótipo.

Para a alimentação dos componentes, utilizou-se uma bateria de lítio com capacidade de 280 mAh e uma voltagem de 3,7 V. Essa bateria é recarregável e, para carregá-la, foi

Figura 8 – Módulo microSD



Fonte: (ARDUINO, 2016b)

usado um componente USB de recarga. Este módulo USB fica a parte do protótipo. Os dados de voltagem e capacidade da bateria são compatíveis com o Arduino selecionado e suficientes para o projeto, por isto se escolheu esta bateria.

Utilizou um botão *on/off* para ligar e cortar a alimentação do protótipo, ou seja, ligar e desligar o produto. Vale ressaltar aqui que o RTC está sempre ligado direto na bateria para não perder a contagem da data e horário.

Para fazer as ligações entre os componentes, foram utilizados fios de cobre encapados muito finos conhecidos como "cabelo de anjo". São fios de espessura muito pequena e alta flexibilidade.

2.1.2 Software

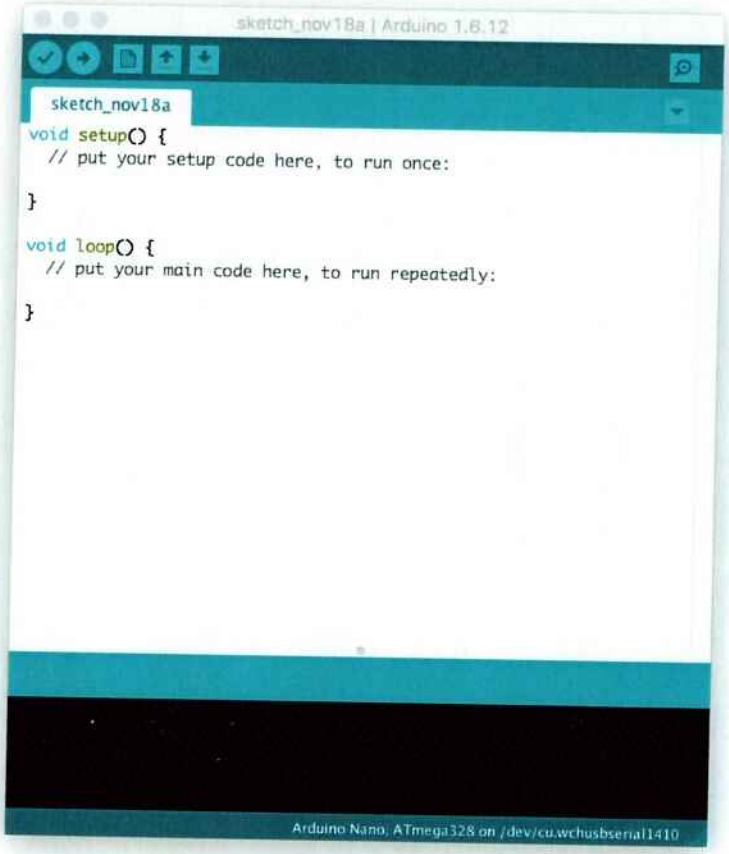
Foram codificados dois programas principais. Um foi carregado no Arduino e o outro é responsável pela análise dos dados coletados. Esta subseção se divide em dois itens que citam as ferramentas que possibilitaram o desenvolvimento desses *softwares*.

2.1.2.1 Software do Arduino

Para codificar o programa descrito nesta subseção, foi utilizado uma *Integrated Development Environment* (IDE) chamada Arduino IDE[1] (Figura 9). IDE é um ambiente de desenvolvimento que permite a criação e compilação de códigos para a plataforma que ela dá suporte. Ela foi desenvolvida pela empresa responsável pelo Arduino. A linguagem

[1] Acesso para *download* da ferramenta através do site https://www.arduino.cc/en/Main/Software/

Figura 9 – Arduino IDE



Fonte: Autor.

aceita por esta IDE é a linguagem de programação do Arduino, baseada na Wiring e também desenvolvida pela empresa responsável pelo microcontrolador (ARDUINO, 2016c).

Foram utilizadas três bibliotecas de terceiros, normalmente, desenvolvidas pelos fabricantes dos módulos conectados ao Arduino. As bibliotecas *RTCLib.h*, *SD.h* e *SFE_LSM9DS0.h* permitem acesso e controle dos componentes RTC, módulo microSD e IMU LSM9DS0 respectivamente.

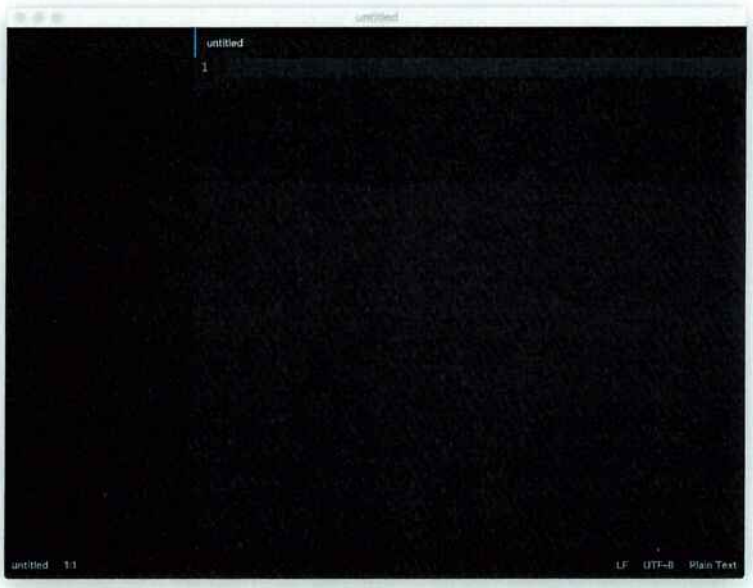
2.1.2.2 Software de análise dos dados

Como IDE, foi utilizado o Atom[2] (Figura 10).

Foram utilizadas a linguagem *Python* versão 2.7, a biblioteca padrão e duas bibliotecas de terceiros. Uma delas é do próprio *Python*, a biblioteca *math*, que permite realizar operações matemáticas mais complexas como módulo, potências e raízes de um

[2] Acesso para *download* da ferramenta através do site https://atom.io/

Figura 10 – Atom



Fonte: Autor.

número. A outra - biblioteca *python-weka-wrapper* - foi desenvolvida para se trabalhar com RNAs utilizando a linguagem *Python*. Essa última permite a criação de RNAs capazes de identificar valores para os quais ela foi treinada (REUTEMANN, 2014).

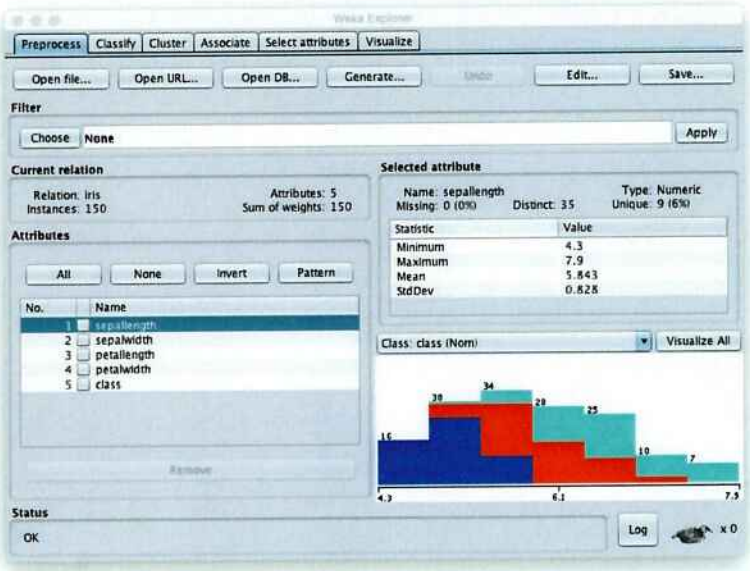
Foi justamente essa funcionalidade que foi usada na implementação do *software*, como será visto na subseção 2.2.2.2.

2.1.2.3 WEKA

O *Waikato Environment for Knowledge Analysis*, conhecido como WEKA, foi desenvolvido na *University of Waikato*, na Nova Zelândia. É um *software* que agrega algoritmos para se trabalhar com *machine learning*, ou seja, algoritmos que permitem que a máquina identificar padrões (WEKA, 2016).

A partir de análises computacionais e estatísticas, o WEKA é capaz do reconhecimento de padrões e classificações de dados, tendo como resultado final os nós e valores de seus pesos de uma RNA. Ou seja, o WEKA treina uma RNA a partir dos dados de entrada e, como saída, obtêm-se os nós e pesos para ser usado em algum *software* que implemente uma RNA.

Figura 11 – WEKA



Fonte: Autor.

2.2 Métodos

Essa seção segue a mesma divisão da seção 2.1 e descreve os procedimentos para a construção do sensor e o desenvolvimento dos *softwares*.

2.2.1 Hardware

A construção do sensor seguiu uma rotina que pode ser resumida por teste do funcionamento do dispositivo atual, conexão de algum módulo e o teste do funcionamento dessa junção.

Testou-se o funcionamento do Arduino sem nenhum componente agregado a ele, rodando um exemplo que vem junto com o IDE, o *Blink*. Esse exemplo apenas pisca o LED do Arduino. Depois se ligou o RTC ao microcontrolador e se rodou um exemplo da biblioteca *RTCLib.h*. Esse exemplo imprime no *prompt* a data e a hora da última compilação do programa e, a partir daí, imprime a data e a hora atualizadas a cada um segundo. Repetiu-se o mesmo processo para o módulo microSD, rodando um exemplo que cria um arquivo, escreve "Teste" nele, salva e fecha. Depois abre o mesmo arquivo, escreve "Teste" de novo, salva e fecha. Por fim, conectou-se o IMU e se rodou um exemplo de teste que imprime no *prompt* em "tempo real" os valores de todos os eixos de todos os sensores que ele possui.

As ligações foram feitas a partir dos *datasheets* de cada componente que podem ser resumidas pela Figura 12. Também na figura, nota-se a utilização de um botão. Esse