UNIVERSIDADE FEDERAL DE SANTA CATARINA
CAMPUS ARARANGUÁ

Pedro Augusto Di Francia Rosso

DESENVOLVIMENTO E IMPLEMENTAÇÃO DE APLICATIVO DE SEGURANÇA PARA SISTEMAS DE ÁUDIO EM *SMARTPHONES*

Araranguá

2017

Pedro Augusto Di Francia Rosso

DESENVOLVIMENTO E IMPLEMENTAÇÃO DE APLICATIVO DE SEGURANÇA PARA SISTEMAS DE ÁUDIO EM *SMARTPHONES*

Trabalho de Conclusão de Curso submetido à Universidade Federal de Santa Catarina para a obtenção do Grau de Bacharel em Engenharia de Computação.
Orientador: Prof. Fabrício de Oliveira Ourique, Dr.

Araranguá

2017

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

Rosso, Pedro Augusto Di Francia
Desenvolvimento e Implementação de Aplicativo de Segurança para Sistemas de Áudio em Smartphones / Pedro Augusto Di Francia Rosso ; orientador, Fabrício de Oliveira Ourique, 2017.
94 p.

Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Santa Catarina, Campus Araranguá, Graduação em Engenharia de Computação, Araranguá, 2017.

Inclui referências.

1. Engenharia de Computação. 2. Reconhecimento de Padrões. 3. Sinais Sonoros . 4. K - Nearest Neighbor. 5. Smartphones. I. Ourique, Fabrício de Oliveira. II. Universidade Federal de Santa Catarina. Graduação em Engenharia de Computação. III. Título.

Pedro Augusto Di Francia Rosso

DESENVOLVIMENTO E IMPLEMENTAÇÃO DE APLICATIVO DE SEGURANÇA PARA SISTEMAS DE ÁUDIO EM *SMARTPHONES*

Este Trabalho de Conclusão de Curso foi julgado aprovado para a obtenção do Título de "Bacharel em Engenharia de Computação", e aprovado em sua forma final pela Universidade Federal de Santa Catarina.

Araranguá, 31 de julho 2017.

Profa. Eliane Pozzebon, Dra.
Coordenadora do Curso

Profª Eliane Pozzebon
Coordenadora do Curso de Graduação Engenharia de Computação
SIAPE: 1680881 / Portaria 061/2017
UFSC / Campus Araranguá

Banca Examinadora:

Prof. Fabrício de Oliveira Ourique, Dr.
Presidente

Prof. Anderson Luiz Fernandes Perez, Dr.

Prof. Alexandre Leopoldo Gonçalves, Dr.

À minha família, pelo inquestionável apoio e incentivo.

AGRADECIMENTOS

À Universidade Federal de Santa Catarina, pelas oportunidades disponibilizadas durante a realização da graduação. Ao professor e orientador Fabrício de Oliveira Ourique, por aceitar desenvolver este longo trabalho compartilhando valiosos conhecimentos no processo. Ao professor Anderson Luiz Fernandes Perez, por valiosas ideias e sugestões além de disponibilizar o espaço do Laboratório de Automação e Robótica Móvel para o desenvolvimento deste trabalho. Aos colegas do LARM, por sempre acreditarem no valor deste trabalho e auxiliar em seu desenvolvimento. Aos nobres colegas e professores que disponibilizaram seus automóveis para a aquisição de áudios valiosos. À meu pai, Pedro Rosso, pelas horas desprendidas na revisão da escrita e valiosas sugestões. À minha mãe, Telma Regina França Rosso, por entender e apoiar as decisões tomadas durante esses anos. À minha irmã Talita Di Francia Rosso, pelo apoio e incentivo. Ao meu irmão, Pedro Henrique Di Francia Rosso, pelas incontáveis horas de companhia no trajeto Criciúma-Araranguá, os tempos de jogo e/ou trabalho perdidos auxiliando-me na resolução dos desafios encontrados e por compactuar com as minhas ideias malucas. À Camila de Oliveira, que mesmo me achando um *nerd*, demonstrou fundamental compreensão e apoio durante toda graduação, especialmente do desenvolvimento deste trabalho. Ao amigo Luan Carlos Silva Casagrande, pelo auxílio na revisão do *abstract* e por não me deixar esquecer da importância da validação dos dados. Ao amigo Yuri Crotti, por acreditar neste trabalho desde do início. Aos demais amigos e colegas que me proporcionaram um enorme aprendizado durante todos esses anos.

Try not. Do. Or do not. There is no try.
(Yoda, Star Wars: Episode V - 1980)

RESUMO

A ascensão dos *smartphones* nos últimos anos tem motivado estudos sobre sua utilização nas diferentes faixas etárias da população. A maioria dos aplicativos disponíveis para *download* que visam o bem estar de seus utilizadores tem o foco na alimentação e na prática de exercícios físicos e muitos estimulam a utilização de fones de ouvido para que se tenha acesso às informações e/ou instruções. Além destes aplicativos, o uso de fones de ouvido no *smartphone* está associado ao consumo de mídia audiovisual e comunicação. Estes tipos de usos têm provocado um aumento significativo de transeuntes urbanos envoltos em sua bolha tecnológica, podendo afetar direta e indiretamente o trânsito por onde passam. Quando distraídos esses pedestre podem, involuntariamente, colocar sua vida em risco ao realizar uma ação. Neste trabalho foi desenvolvido um aplicativo que tem como objetivo alertar usuários de *smartphones* sobre sinais de perigo no trânsito, de modo a auxiliar pedestres e motoristas a evitarem acidentes. O método utilizado em tal aplicativo é a classificação continua dos sons que circundam o usuário e o algorítimo escolhido para definir se o som é perigoso ou não foi o *K - Nearest Neighbor*. Assim, se em execução no *smartphone*, o aplicativo interrompe a execução do áudio quando o som captado do ambiente for classificado como perigoso. Como a captura e classificação dos sons do ambiente ocorre em tempo real, o usuário de *smartphone* pode ouvi-los e, deste modo, dispor de alguns segundos para executar uma ação defensiva.

Palavras-chave: K-NN, *smartphone*, sinais sonoros, segurança, reconhecimento de padrões.

ABSTRACT

The rise of smartphones in recent years has motivated studies on their use in the different age groups of the population. Most of the available apps for download that aim the well-being of its users focus on nutrition and fitness and many encourage the use of headphones to give further instructions and/or information. In addition to these applications, the use of headphones on the smartphone is associated with the consumption of audiovisual media and communication. These types of uses have caused a significant increase of urban passers-by in their technological bubble, being able to directly and indirectly affect the transit through which they pass. When distracted these pedestrians may, unintentionally, put their life at risk while performing an action. In this work an application was developed that aims to alert users of smartphones about danger signs in traffic, in order to help pedestrians and drivers avoid accidents. The method used in such an application is the continuous classification of the sounds that surround the user and the algorithm chosen to define if the sound is dangerous or not, is the K - Nearest Neighbor. So if running on the smartphone, the application stops playing the audio when the sound captured from the environment is classified as dangerous. As the capture and classification of ambient sounds occurs in real time, the user of smartphone can hear them and thus have a few seconds to perform a defensive action.

Keywords: K-NN, smartphone, safety, sound signals, pattern recognition

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

LISTA DE EQUAÇÕES

SUMÁRIO

1 INTRODUÇÃO

A população brasileira foi estimada em 1º de julho de 2016 em cerca de 206 milhões de pessoas pelo Instituto Brasileiro de Geografia e Estatística (IBGE, 2016). Em junho de 2016 o Departamento Nacional de Trânsito (DENATRAN, 2016) revelou que existem 89 milhões de veículos automotores, incluindo carros, motocicletas, caminhões, ônibus, etc. Isso equivale aproximadamente a 1 veículo para cada 4 pessoas no Brasil.

Com as cidades repletas de veículos automotores e pessoas é frequente observar pedestres distraídos que, por reflexo e até por sorte, saem ilesos de situações que poderiam resultar em acidentes com riscos às suas vidas. Por outro lado, nem sempre fica claro aos motoristas qual será o comportamento de um pedestre em determinada situação, por mais concentrados no trânsito que estejam, o que também pode resultar em acidentes que podem envolver pedestres.

Devido a popularização da telefonia móvel, 121 milhões de pessoas com idade entre 10 e 59 anos possuem celular para uso pessoal, o que corresponde à 80,74% da população com mesma faixa etária em 2014 (IBGE, 2014).

Apesar dos dados do IBGE não apresentarem valores especificamente para *smartphones*, segundo pesquisa realizada pela Fundação Getúlio Vargas - SP (FVG-SP), com ajuda das empresas do ramo da telefonia, existem 168 milhões de *smartphones* no Brasil em 2016 (MEIRELLES, 2016). A Agência Nacional de Telecomunicações (ANATEL) em seu relatório de acompanhamento do setor de telecomunicações verificou um quantitativo 257 milhões de acessos no final de 2015, apontando ainda uma redução de 8,1% em relação ao final de 2014. A discrepância entre essas entidades bem com a contração na quantidade de acessos é explicada por fatores como a redução do valor de uso da rede móvel o que leva a cada usuário utilizar apenas um número, alterações no comportamento dos usuários, entre outros (ANATEL, 2016).

Quando observado o comportamento dos pedestres e/ou dos motoristas pode-se notar que é comum pessoas andarem e dirigirem enquanto utilizam seu *smartphone*. É mais preocupante quando esses indivíduos estão utilizando fones de ouvido, de forma à abdicar de dois sentidos (visão e audição) que são muito importantes para a sobrevivência. Os olhos e ouvidos dos seres humanos são responsáveis por alertar quando há uma situação de perigo eminente e sem eles extingue-se a noção do que acontece ao redor.

É difícil prever o comportamento de um pedestre que está atento e mais difícil ainda é saber como um pedestre que está focado em seu *smartphone* e/ou com seus fones de ouvido irá se comportar ou que ação ele irá executar. Ao analisar o relatório da empresa ERICSSON, é possível observar a quantidade de acessos a determinados aplicativos. A pesquisa realizada em 2014 com usuários americanos de *smartphones* com idade entre 15 e 69 anos. Os dados apontam que aproximadamente 53,1% de usuários realizam ligações 5 ou mais vezes ao dia (44% ligações de voz e 9,1% ligações via internet). Ao adicionar o percentual de usuários que utilizam entre 1 à 4 vezes ao dia alcança-se um total de 80,6% (21% ligações de voz e 6,5% ligações via internet). Outro dado relevante é que aproximadamente 44,1% dos usuários consomem conteúdo áudio-visual 5 ou mais vezes por dia (17,2% *streaming* de músicas e 27,9% é o valor combinado de assistir vídeo-clipes e/ou filmes e séries de TV). De forma análoga, ao contabilizar os dados de quem utiliza entre 1 à 4 vezes por dia esses aplicativos alcança-se um total de 84,5% (13,6% *streaming* de música e 25,8% é o valor combinado de assistir vídeo-clipes e/ou filmes e séries de TV) (ERICSSON, 2015).

Os dados apresentados pelo levantamento citado acima não permitem estimar a quantidade de minutos gastos em cada uma dessas tarefas diariamente. Contudo, é relevante o percentual de acesso repetidos a determinados tipos de aplicativos que envolvem os dois sentidos mencionados. Como uma das funções da audição é de alertar sobre situações de perigo, por exemplo, ao associar o som de uma buzina como um sinal de perigo eminente, toda vez que este som é reconhecido o corpo entra em estado de alerta. Porém, quando escuta-se música ou conversa-se ao telefone, não percebe-se esse tipo de sinal. Isto se torna ainda mais grave porque comumente se utilizam volumes elevados justamente para poder apreciar a música, a conversa ou até para fazer-se entender no universo barulhento das cidades,ficando-se mais vulnerável a sofrer acidentes. Dadas estas condições, é possível utilizar fones de ouvido e ainda assim ser "avisado"sobre os alertas de perigo?

De forma simplificada, o sistema auditivo humano é um mecanismo que transforma o som em pulsos elétricos, os quais são transmitidos para o nervo auditivo e, posteriormente, para o cérebro, possibilitando perceber os sons externos e analisar seus significados (SWENSSON; SWENSSON; SWENSSON, 2009). A buzina é um dispositivo que deverá emitir som contínuo e uniforme sem que seu espectro acústico varie substancialmente durante a sua execução (DENATRAN, 2002) e deve possuir máxima intensidade de som de 93dB para veículos fabricados a partir de 2002 como é apresentado pelo Conselho Nacional de Trânsito

(CONTRAN, 1998).

Dadas essas características, considera-se plausível pensar em um aplicativo capaz de analisar em tempo real os sons que circundam o usuário e emitir um alerta quando algum som classificado como perigoso é identificado.

1.1 OBJETIVOS

Afim de definir o escopo do projeto foi definido o objetivo geral e os objetivos específicos, estes são apresentados nas seções subsequentes.

1.1.1 Objetivo Geral

Desenvolver um aplicativo de segurança que seja capaz de alertar o usuário de *smartphone*, que esteja utilizando fones de ouvido, sobre um perigo eminente em tempo suficiente para que o mesmo possa executar uma ação que possa preservar a sua vida.

1.1.2 Objetivos Específicos

Para que o objetivo geral seja alcançado os seguintes objetivos específicos foram elencados:

1. Desenvolver um modelo que processe e reconheça sinais sonoros específicos.
2. Analisar o modelo desenvolvido.
3. Implementar o modelo analisado em aplicativo para *smartphones*.
4. Testar e validar o aplicativo em tempo real.

1.2 JUSTIFICATIVA

Uma retrospectiva publicada no *Injury Prevetion* faz a correlação entre acidentes de trânsito com morte ou ferimento nos Estados Unidos entre os anos de 2004 até 2011 e pedestres que utilizavam fones de ouvido, a pesquisa estimou 116 casos, onde 74% dos casos as pessoas estavam utilizando fones de ouvido e em 29% dos casos existe a menção

de que um som de alerta foi tocado antes da colisão (LICHENSTEIN et al., 2012).

O relatório técnico apresentado pelo *National Highway Traffic Safety Administration* (NHTSA) dos Estados Unidos em abril de 2016 aponta que em 2012 mais de 1500 pedestres nos Estados Unidos foram tratados em salas de emergência dos hospitais como resultado de acidentes de trânsito porque estavam conversando no telefone (SCOPATZ et al., 2016).

Um dos motivos levantados pelo NHTSA para a diferença no número de acidentes atribuídos a utilização de *smartphone* é que os boletins de ocorrência são realizadas por pessoas, nem sempre as que estão envolvidas, e a informação sobre o uso ou não de *smartphone* para conversação ou escutar música, pode ser perdido. A popularização dos *smartphones* também contribui para o aumento dos valores, visto que temos um crescimento de mais de 1,5 bilhões de usuários ativos entre 2010 e 2016 (NAKONO, 2016). Esta popularização fez com que houvesse mais atenção no comportamento dos pedestres. Ainda existem os acidentes que não estão diretamente ligados à pedestres, como por exemplo um veículo colidir em outro enquanto tenta desviar de um pedestre, neste caso o pedestre não faz parte da estatística.

1.3 METODOLOGIA

A presente pesquisa é classificada como aplicada, pois objetiva gerar conhecimentos para aplicação prática (SILVEIRA; CÓRDOVA, 2009), e tecnológica, pois insere-se no campo do conhecimento relativo ao projeto de artefatos e ao planejamento de sua realização, operação, ajuste, manutenção e monitoramento, à luz da ciência (BUNGE, 1985 apud Freitas Junior et al., 2014)

O trabalho foi desenvolvido em quatro etapas. Na etapa inicial foi realizada pesquisa bibliográfica exploratória com dois focos: o primeiro na aquisição e processamento de sinais sonoros e o segundo nas plataformas para *smartphones*, seu sistema operacional, recursos de *hardware* e *software*.

Na segunda etapa foram estudas das técnicas observadas na etapa anterior aplicando uma delas para o desenvolvimento do modelo que será a base do aplicativo.

Na terceira etapa, com as definições de *hardware* e de como seria o sistema, foi desenvolvido o aplicativo na plataforma escolhida.

Os resultados dos testes em tempo real, bem como os testes

realizados para a construção do modelo, apresentado na quarta etapa, está também será responsável por propôr futuras melhorias.

1.4 ORGANIZAÇÃO DO TRABALHO

As quatro etapas mencionadas na Seção 1.3 correspondem a 6 capítulos, onde o conteúdo é descriminado abaixo.

O Capítulo 1 apresenta a introdução, justificativa e objetivos do trabalho.

O Capítulo 2 apresenta uma breve revisão sobre classificação e explicita as informações sobre o classificador utilizado.

O Capítulo 3 explicita a solução (modelo) encontrada para o reconhecimento de sons de perigo e suas características.

O Capítulo 4 exibe conceitos sobre os *smartphones*, os recursos utilizados no desenvolvimento do aplicativo e o funcionamento do mesmo.

O Capítulo 5 apresenta os resultados obtidos através do modelo apresentado no Capítulo 3 e do aplicativo desenvolvido no Capítulo 4.

O Capítulo 6 descreve as considerações finais deste trabalho e propostas para trabalhos futuros.

2 CLASSIFICAÇÃO

Pode-se dizer que classificar é atribuir um valor, contido num conjunto discreto de classes, para um entrada desconhecida (KRAMER, 2013) ou que a função do classificador é mapear a classe da nova entrada em uma das possíveis classes por ele já definidas (FISCH; KALKOWSKI; SICK, 2014).

O objetivo, neste estudo, é atribuir classes (rótulos) para os sinais sonoros que circundam os usuários de *smartphone*. Contudo, existem diversos classificadores que podem satisfazer este objetivo e para entender o porquê do classificador *K - Nearest Neighbor* (K-NN) ser o selecionado algumas definições são apresentadas.

2.1 PARADIGMAS DE APRENDIZAGEM

O processo de aprendizado de um classificador típico pode ser dividido em duas fases: a criação de um modelo com base nos dados de treinamento e a predição dos dados desconhecidos de acordo com o modelo gerado (LIU; WU; ZHANG, 2016).

Conceitualmente existem dois paradigmas de aprendizagem: Supervisionado (conhecido como aprendizado com professor) e não-supervisionado (conhecido como aprendizado sem professor) (HAYKIN, 2001).

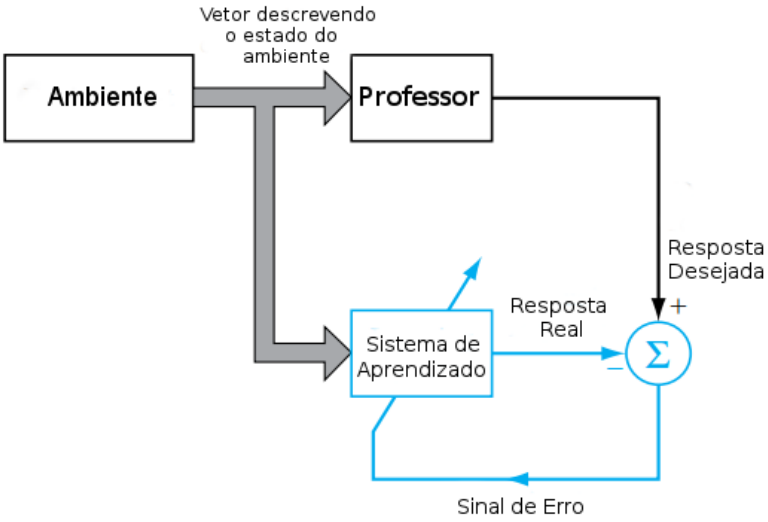
2.1.1 Supervisionado

O aprendizado supervisionado cria estruturas de conhecimento que auxiliam os algorítimos de classificação na atribuição das classes para as novas entradas, de acordo com as classes pré-definidas (NGUYEN; ARMITAGE, 2008). Esta estrutura de conhecimento pode ser expressada na forma de um conjunto com entradas e suas respectivas saídas que são utilizadas para o treinamento do sistema (HAYKIN, 2001).

O diagrama de blocos da Figura 1 apresenta uma forma conceitual do funcionamento do aprendizado supervisionado. O professor e o sistema de aprendizado são expostos a um vetor que descreve o estado do ambiente. Como o professor possui conhecimento prévio, ele fornece ao sistema de aprendizado o resultado esperado para aquela determinada entrada. A diferença entre a saída do professor e a saída

proveniente do sistema de aprendizado é chamada sinal de erro, de forma a possibilitar que o sistema seja realimentado e se modifique para no próxima iteração (HAYKIN, 2001).

Figura 1: Diagrama de Blocos do Aprendizado Supervisionado.



Fonte: Adaptado de Haykin (1999).

Um exemplo é um estudante receber a lista de exercícios e suas respectivas respostas e ter que descobrir como resolver outros exercícios no futuro. Em termos de algoritmos pode-se citar *Support Vector Machines*, algumas Redes Neurais Artificiais, *Classification Trees* entre outros (LOURIDAS; EBERT, 2016).

2.1.2 Não-Supervisionado

No aprendizado não-supervisionado a ausência do professor é suprida dando-se condições para que o sistema realize uma extração dos parâmetros do conjunto inicial (HAYKIN, 2001). Através das características extraídas são construídas representações (treinamento) que serão utilizadas na decisão sobre as futuras entradas (SURESH; SUNDA-

RARAJAN; SAVITHA, 2013).

O sistema deve absorver o máximo de informações possíveis sobre as características dos dados, assim formando grupos, conhecidos como *clusters*, com aqueles que estão muito próximos uns dos outros no espaço de dados (KRAMER, 2013). Ao contrário do aprendizado supervisionado, o não-supervisionado não possui uma solução, o sistema é que deverá encontrá-la. Algorítimos Genéticos, *K - Means Clustering*, algumas Redes Neurais Artificiais entre outros são exemplos de algorítimos deste tipo de aprendizado (LOURIDAS; EBERT, 2016).

2.2 CLASSES

Classificadores são amplamente utilizados em diversas aplicações. Comumente os problemas apresentados são multi-classe, i.e., a possível saída do classificador é um número discreto superior a 2. Um popular método para facilitar a resolução desses problemas é dividi-lo em subclasses onde é possível utilizar classificadores binários. Usualmente é mais simples construir um classificador que identifica uma entrada entre duas classes do que entre várias classes (GALAR et al., 2011).

Uma abordagem para utilizar esses classificadores como solução de problemas multi-classes é inicialmente realizar o treinamento dos classificadores binários e depois utilizar um combinador para agregar as saídas dos classificadores e então decidir a saída final do sistema. O passo mais importante neste tipo de abordagem é assegurar que os classificadores binários apresente uma alta exatidão (SHIRAISHI; FUKUMIZU, 2011).

Um conjunto de parâmetros é utilizado para definir uma classe. A distinção entre o conjunto de parâmetros é o que distingue uma classe da outra.

2.3 MÉTODOS DE AVALIAÇÃO

Optou-se por utilizar duas metodologias para a avaliação dos resultados: a primeira é percentual de acerto e a segunda é *Sensitivity* e *Specificity*.

2.3.1 Percentual de Acerto

O percentual de acerto é definido pela Equação 2.1, onde o resultado da classe é comparado com o valor esperado. Caso de seja distinto é atribuído 0 e caso esteja correto é atribuído 1.

$$Acerto(\%) = \frac{\sum_{i=0}^{N-1} C_i}{N} 100. \tag{2.1}$$

Onde, N é o número de dados da amostra, C_i é a o valor da comparação entre a saída e o valor esperado de cada dado presente na amostra.

2.3.2 *Sensitivity* e *Specificity*

Uma forma comum de caracterizar a exatidão do classificador é por meio da métrica *Sensitivity* e *Specificity* (ROGERS; GIROLAMI, 2011).

Ela é definida, segundo Nguyen e Armitage (2008) como:

- **Falso Negativo (FN)**: Percentual de membros da classe X que foram classificados incorretamente como não pertencentes a classe X.
- **Falso Positivo (FP)**: Percentual de membros de outras classes que foram classificados incorretamente como pertencentes a classe X.
- **Verdadeiro Negativo (TN)**: Percentual de membros de outras classes que foram corretamente classificados como não pertencentes a classe X.
- **Verdadeiro Positivo (TP)**: Percentual de membros da classe X que foram classificados corretamente como pertencentes a classe X.

Com base nesses parâmetros a *Sensitivity* é calculada pela Equação 2.2 e *Specificity* é calculada pela Equação 2.3 (ROGERS; GIROLAMI, 2011).

$$S_e = \frac{TP}{TP + FN}. \tag{2.2}$$

$$S_p = \frac{TN}{TN + FP}. \tag{2.3}$$

Sensitivity apresenta a proporção do que é realmente pertencente a classe X e a *Specificity* apresenta a proporção do que realmente não é pertencente a classe X.

2.3.3 Curvas *Receiver Operating Characteristic* - ROC

Com os dados obtidos na Subseção 2.3.2 pode-se calcular as curvas *Receiver Operating Characteristic* (ROC) que são uma medida de desempenho para sistemas com classificação binária (FERRIS et al., 2015). Os valores de *Sensitivity* e *Specificity* são calculados para variação de um parâmetro permitindo a visualização do desempenho relacionado a esta mudança. A *Sensitivity* é representada em função do complementar da *Specificity* $(1 - S_p)$ (ROGERS; GIROLAMI, 2011; FERRIS et al., 2015; SERRANO et al., 2010).

2.4 *K - NEAREST NEIGHBOR* - K-NN

O K-NN, ou K - vizinhos mais próximos, é um classificador popular devido a sua simplicidade, fácil compreensão, performance relativamente alta, habilidade de lidar com dados binários ou multi-classe, apresenta baixa identificação de erros, é utilizado em diversas aplicações e é baseado em um conjunto de treinamento (DENG et al., 2016; ROGERS; GIROLAMI, 2011; GUTIÉRREZ et al., 2016; GLOWACZ; GLOWACZ, 2016; YU et al., 2016).

A suposição comum para este tradicional algorítimo de aprendizagem é que para cada nova entrada ou valor desconhecido apenas um rótulo baseado no conjunto de treinamento é o resultado possível. Isso significa que para cada entrada uma única classe é atribuída, mesmo que existam duas os mais classes distintas, (LIU; WU; ZHANG, 2016).

Para classificar ele utiliza a ideia de que os vizinhos mais próximos a entrada desconhecida apresentam informações úteis sobre a mesma, (KRAMER, 2013; GUTIÉRREZ et al., 2016). Utilizando a similaridade, o K-NN calcula o valor da distância entre a entrada desconhecida e todos os dados contidos no conjunto de treinamento para então selecionar o K vizinhos mais próximos, (DENG et al., 2016).

A partir das classes desses vizinhos ele identifica qual é a classe

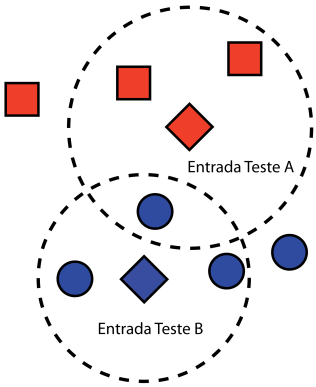
predominante e define a entrada desconhecida como pertencente à ela. Isso significa que cada entrada teste é comparada com todos os dados do treinamento, i.e., a distância é calculada para todos os pontos (GUTIÉRREZ et al., 2016; DENG et al., 2016; ROGERS; GIROLAMI, 2011; GLOWACZ; GLOWACZ, 2016; KRAMER, 2013; JIAN; CHEN, 2015; OZAY et al., 2016; LIU; WU; ZHANG, 2016).

Tipicamente utiliza-se a distância Euclidiana, que está definida na Equação 2.4, onde $d(x, y)$ é a distância entre as instâncias x e y e D é o número de atributos do problema, a classe é definida em uma votação onde vence a maioria (LIU; WU; ZHANG, 2016; GUTIÉRREZ et al., 2016).

$$d(x, y) = \sqrt{\sum_{i=1}^{D}(x_i - y_i)^2}. \tag{2.4}$$

Rogers e Girolami (2011) mostram que ao se considerar um cenário em que tem-se N objetos para o treinamento cada qual representado por um conjunto x_n com a classe c_n, para classificarmos um nova entrada x_{nova} com o K-NN é necessário inicialmente descobrir os K (onde K é o número de vizinhos) pontos do treinamento mais próximos da entrada x_{nova}. A classe c_{nova} será a mesma classe da maioria desses K vizinhos.

Figura 2: Esquema representando a operação do K-NN.



Fonte: Adaptado de Rogers e Girolami (2011).

A Figura 2 é um exemplo da utilização do K-NN, onde $K = 3$, as esferas e quadrados apresentam duas classes distintas e os diamantes

são duas entradas testes. Os 3 vizinhos mais próximos da Entrada Teste A são 2 quadrados e 1 esfera, logo o K-NN definirá que a classe da Entrada Teste A será quadrado. Da mesma forma os 3 vizinhos mais próximos da Entrada Teste B são esferas, logo a classe definida pelo K-NN será esfera.

A performance do K-NN está diretamente ligada ao valor de K. Deste modo a complexidade de tempo necessária para achar os K vizinhos é linear porque é proporcional ao tamanho do conjunto de treinamento, além do fato de que para classes desbalanceadas o K-NN possuem a tendência de escolher a classe dominante (LIU; WU; ZHANG, 2016; DENG et al., 2016).

Caso M seja o número de entradas testes e N o número de dados no conjunto de treinamento, o algorítimo requer MxN distâncias computacionais e M seleções de K instâncias a partir do vetor de N elementos. Quando os dados do treinamento e os dados de teste aumentam, a distância computacional aumenta quadraticamente (GUTIÉRREZ et al., 2016).

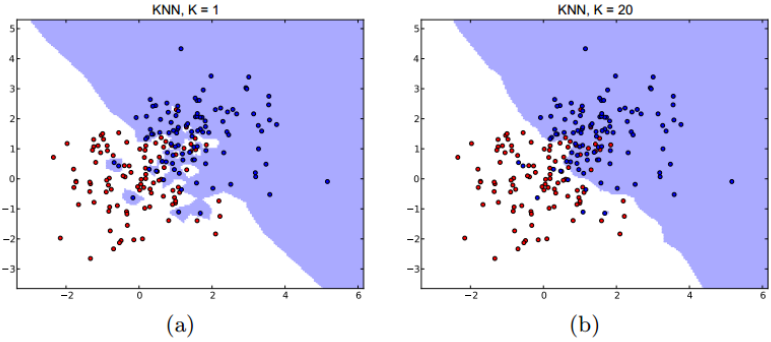
Tradicionalmente o K-NN é uma abordagem aplicável a diferentes tipos de conjunto de dados, contudo sua performance pode ser afetada em alguns casos especiais, tais como um pequeno número no conjunto de treinamento, existência de um desbalanceamento nas classes do conjunto treinamento, o conjunto de dados apresenta amostras com ruído e atributos com ruído (YU et al., 2016; KRAMER, 2013).

2.4.1 Vizinhos

Escolher corretamente o valor de K, ou seja, o quantidade de vizinhos que serão calculadas as distâncias, impacta no desempenho do classificador. Este problema é conhecido como Modelo de Seleção e existem técnicas, como validação cruzada, que podem ser empregadas para escolher o melhor o valor de K (KRAMER, 2013).

Usualmente utiliza-se valores de K ímpar como forma de evitar o empate na escolha da classe. Pode-se levar em conta, em caso de empate, a distância entre os pontos e a entrada, na forma de "pesos", i.e., cada distância possui um peso diferente na votação para desempate (ROGERS; GIROLAMI, 2011).

Figura 3: Comparação entre o número de vizinhos (K) em um espaço com duas dimensões.



Fonte: Adaptado de Kramer (2013).

O exemplo da Figura 3 apresenta um conjunto binário de classes. Nota-se que em (a) onde o valor de K é 1, não existe uma linha nítida a qual separa as duas classes e apresenta algumas "ilhas". Essas ilhas provavelmente são ruídos e podem ser superados aumentando o número de K, (ROGERS; GIROLAMI, 2011). Em (b) o valor de K é 20 e visualmente o conjunto de dados apresenta uma fronteira consistente entre as classes.

A fronteira, para este caso, significa que uma nova entrada será classificada com a mesma classe do lado em que ela se encontra, entretanto isso não significa que todos os pontos do mesmo lado pertencem a mesma classe.

3 EXTRAÇÃO DE PARÂMETROS

Desenvolver um aplicativo de segurança que utilize a interação com áudio, em *smartphones*, como gatilho para o acionamento possui a premissa de conhecer os tipos de sons que circundam o usuário. Desta maneira a primeira etapa para a sua construção foi desenvolver uma metodologia capaz de distinguir o qual som é normal e o qual é perigoso para o usuário.

Os áudios utilizados para validar a metodologia foram divididos em Sons de Ambiente, Sons de Perigo e Sons Mistos, onde o primeiro corresponde aos que não apresentam perigo para o usuário, o segundo descrevem os sons que apresentam perigo para o usuário, enquanto o terceiro são uma mistura dos dois primeiros, buscando uma melhor simulação do ambiente real.

O K-NN é o classificador que define qual é o tipo do som que está perto do usuário em determinado momento. Utilizando-se de uma classificação binária, ou seja, o som será Normal ou Perigo.

3.1 PARÂMETROS GERAIS DO K-NN

Ao utilizarmos o K-NN faz-se necessário que alguns parâmetros sejam definidos, que são: amostra, treinamento, classe, e K, onde:

- **Amostra:** é o áudio que será classificado;
- **Treinamento:** são os áudios que compõem as classes de referência utilizados;
- **Classe:** apresenta os valores relacionados aos áudios de referências. O resultado do classificador será um desses valores;
- ***K*:** é a quantidade de vizinhos mais próximos que a distância será avaliada com o intuito de definir a qual classe à amostra pertence.

3.2 CARACTERÍSTICAS DOS ÁUDIOS

Para que os resultados apresentados sejam consistentes, todos os áudios utilizados possuem as mesmas características, as quais são elencadas a seguir:

- **Codec:** WAV - Formato padrão de arquivos para áudio da *Microsoft* e IBM;
- **Taxa da Amostra:** 44100Hz;
- **Canais:** 1 (um) - Mono;
- **Bits por Amostra:** 16;
- **Compressão:** Sem compressão.

A relação de áudios com estas características utilizados e uma breve descrição é apresentada nas Tabelas 2 e 3 (Anexo A).

3.3 MÉTODO 1: ESPECTROGRAMA

O primeiro método é baseado no Espectrograma, que é uma representação da intensidade do sinal em diferentes bandas de frequência Bresolin (2003). Neste método é necessário definir os seguintes parâmetros: tempo, janela e DS (*Downsampling*), onde tempo e DS são responsáveis por realizar a normalização do sinal e a janela divide o sinal em seções com o tamanho pré-definidos.

3.3.1 Parâmetros do Espectrograma

1. **Tempo:** Um dispositivo de segurança deve avisar o usuário de forma a proporcionar ao mesmo um tempo hábil para efetuar uma ação de prevenção ou minimizar os efeitos colaterais de uma situação de perigo. Com o intuito de escolher o melhor e menor valor para utilizar na aplicação foram realizados os testes em *MATLab* com os tempos: 0,5; 1,0 e 1,5 segundos. Apesar de aparentemente esses tempos corresponderem a intervalos curtos no nosso dia-a-dia, quando se trata de segurança cada fração de segundo conta. Vale ressaltar que este não é o tempo total do dispositivo, i.e., este tempo da amostra somado ao tempo de processamento corresponde ao tempo necessário para o dispositivo identificar a situação de perigo e avisar o usuário.
2. **Janela:** é utilizado para dividir o áudio da amostra em N pontos de mesmo tamanho o quais serão analisados. Foram utilizados os seguintes valores para a janela:

- 128;
- 256;
- 512;
- 1024.

3. **DS:** é uma forma eficiente para retirar informações de grandes amostras. Ele divide a amostra em grades regulares podendo acarretar na perda de informações úteis Wang et al. (2014). Corresponde a quantidade de dados que serão utilizadas no processamento do áudio, e.g., um DS com valor 2 corresponde a utilização de um dado sim e outro não em toda a amostra. Os valores de DS estipulados para este teste foram:

   - 1;
   - 2;
   - 4;
   - 6;
   - 8.

   Como alguns áudios são pequenos o número máximo de DS que pode ser utilizado nos testes foi 8.

3.3.2 Parâmetros do Classificador

O K-NN é versátil, podendo ter diversas classes como possível resultado. Contudo, para este método (Espectrograma), foram definidas duas classes: **Normal** e **Perigo**. A classe normal é composta por arquivos de som ambiente enquanto a classe perigo é composta por sirenes e buzinas.

1. **Amostra:** Utilizou-se 15 áudios para teste, sendo eles apenas classe Normal ou Perigo;

2. **Treinamento:** Composto pelo áudio 1 como classe Normal e o áudio 12 da classe Perigo, ambos são distintos dos que foram utilizados como amostra.

3. **Classe:** O valor 0 (zero) foi associado a classe Normal e o valor 1 (um) para a classe Perigo.

4. ***K*:** Como o resultado da classificação está diretamente associado a quantidade de vizinhos próximos, realizou-se a variação do mesmo de 5 a 25. Afim de produzir um resultado consistente foi estipulado que para cada amostra o número de vizinhos máximos não poderia exceder o número máximo de pontos comparáveis de uma das classes da matriz de treinamento.

O resultado esperado é que para toda a extensão da amostra que contenha apenas som ambiente seja 0 (classe normal) e para a amostra que contenha um som de sirene seja 1 (classe perigo).

3.4 MÉTODO 2: POTÊNCIA, *KURTOSIS*, *SKEWNESS*, VARIÂNCIA E BANDA

Os resultados obtidos na Seção 5.1.1 não foram satisfatórios e por isto outra abordagem foi utilizada. Em vez de utilizar a resposta do Espectrograma, agora leva-se em consideração Potência, *Kurtosis*, *Skewness*, Variância e Banda. Estas características estatísticas são usadas como medidas para extrair as informações importantes sobre uma série de tempo (DIYKH; LI; WEN, 2016). Neste caso a serie de tempo é amostra que pretende-se classificar, definida por:

$$a[n], n = 0, 1, ..., N - 1. \tag{3.1}$$

Onde N é o tamanho da amostra.

- **Potência:** é a potência do sinal da amostra, definida pela Equação 3.2.

$$P_i = \frac{1}{N-1} \sum_{n=0}^{N-1} (a_i[n] - \mu_i)^2. \tag{3.2}$$

Onde μ_i é definido pela Equação 3.3 e a_i são os pedaços da amostra que correspondem a um tempo de 0.25s.

$$\mu_i = \frac{1}{N} \sum_{n=0}^{N-1} a_i[n]. \tag{3.3}$$

- ***Kurtosis* (Curtose):** é o grau de achatamento da distribuição em comparação ao que se considera normal, quanto maior a *Kurtosis* maior é a presença de valores que estão distantes da média

(RIBEIRO et al., 2012), cuja é definida pela Equação 3.4.

$$\kappa_i = \frac{\frac{1}{N}\sum_{n=0}^{N-1}(S_i[n]-\nu_i)^4}{\left(\frac{1}{N}\sum_{n=0}^{N-1}(S_i[n]-\nu_i)^2\right)^2}. \tag{3.4}$$

Sendo A_i é a transformada de *Fourier* do sinal a_i (OPPENHEIM, 1997), pode-se obter a densidade espectral de potência através da Equação 3.5. E com auxílio da Equação 3.6 calcula-se o ν.

$$S_i[k] = |A_i[k]|^2, k = 0, 1, ..., N-1. \tag{3.5}$$

$$\nu_i = \frac{1}{N}\sum_{n=0}^{N-1} S_i[n]. \tag{3.6}$$

- ***Skewness*:** é uma medida da simetria dos dados em relação a média de forma que um resultado positivo implica que os dados são espalhados para a direita da média e um resultado negativo implica para a esquerdada média (GEORGIOU; VOIGT, 2015), que está definida na Equação 3.7

$$\psi_i = \frac{\frac{1}{N}\sum_{n=0}^{N-1}(S_i[n]-\nu_i)^3}{\left(\sqrt{\frac{1}{N}\sum_{n=0}^{N-1}(S_i[n]-\nu_i)^2}\right)^3}. \tag{3.7}$$

- **Variância:** é uma medida de dispersão que indica o grau de variabilidade em determinadas situações (RIBEIRO et al., 2012), a Equação 3.8 representa a Variância.

$${\sigma_i}^2 = \frac{1}{N-1}\sum_{n=0}^{N-1}(S_i[n]-\nu_i)^2. \tag{3.8}$$

- **Banda:** o espectro é dividido na quantidade de banda utilizada. O valor escolhido foi de 100, o que acarreta em uma largura de banda com o valor de 441 Hz. A Equação 3.9 representa como é

obtido o valor para cada Banda.

$$B_i = \sum_{n=ib}^{(i+1)b-1} S_i[n], i = 0, 1, ..., 99. \tag{3.9}$$

Onde b é definido pela Equação 3.10.

$$b = \frac{N}{100}. \tag{3.10}$$

A utilização de um método que é composto de mais variáveis visa obter as nuances que diferem um áudio do outro, assim classificando ele de maneira adequada.

No Método 3.3 os valores para tempo foram de 0,5; 1,0 e 1,5 enquanto no Método 3.4 o valor de tempo foi fixado em 0,25 segundo.

3.4.1 Parâmetros do Classificador

A mesma abordagem vista na Seção 3.3.2 foi utilizada nos testes com esta metodologia. O áudio só pode ser rotulado como normal ou perigo, mesmo que o sinal de perigo tenha sua origem em sirenes ou buzinas.

1. **Amostra:** afim de obter resultados mais precisos foram utilizados os mesmos 15 áudios e acrescentados 10 áudios variados, de forma que a amostra contenha sons ambientes, sons com sirenes, sons com buzinas e sons que misturem ambiente com sirene e/ou buzina.

2. **Treinamento:** diversas variações no conjunto de treinamento foram realizadas. Analisando os resultados no formato apresentado na Seção 2.1 obteve-se a seguinte configuração:

   - Áudio 0 e 1 para classe Normal;
   - Áudio 13, que é uma sirene, áudio 26, que são buzinas em sequência e não sobrepostas, 57 buzinas com duração de 0,25s compõem a classe Perigo. As 57 buzinas são amostras gravadas com o aplicativo desenvolvido e visam atribuir maior variedade ao conjunto de treinamento. Elas estão descritas na Tabela 4 e 5 (Anexo A).

3. **Classe:** o valor 0 (zero) foi associado a classe normal, o valor 1 (um) para a classe perigo, que engloba sirenes e buzinas.

4. ***K*:** De forma similar ao utilizado na Seção 3.3 o valor do K foi variado de 1 a 25 e utilizou-se os resultado de *Sensitivity and Specificity* (Seção 2.3.2) para definir o melhor valor de K.

4 IMPLEMENTAÇÃO DO APLICATIVO

Os aparelhos para telefonia móvel evoluem em suas características de *software* e *hardware* constantemente. Anualmente grandes empresas do ramo da tecnologia apresentam suas mais novas inovações em grandes conferências que atraem milhares de espectadores e possíveis compradores.

Um telefone móvel que é capaz de realizar muitas das funções de um computador, normalmente com interface *touchscreen*, com acesso a internet e um sistema operacional que possibilita rodar e realizar *download* de aplicativos é considerado um *smartphone*.[1]

Segundo pesquisa da ERICSON, no Q1 (Janeiro à Março) de 2016 existiam 7,4 bilhões de inscrições de telefones móveis ao redor do mundo e aproximadamente 80% (63 milhões) de telefones móveis vendidos no mesmo período (Q1/2016) são *smartphones* (CERWALL et al., 2016). Inscrições de *smartphones* e usuários de *smartphones* eram estatísticas diferente, dado ao fato que muitas vezes trocamos de telefone e não cancelamos a inscrição do anterior. Segundo um levantamento da Nakono (2016) existem cerca 2 bilhões de dispositivos pessoais conectados a internet.

Dessa maneira uma estimativa válida é que 30% da população mundial possuí um *smartphone*. Assim, caso apenas 10% dessas pessoas utilizam de fones de ouvido diariamente em seu trajeto para escola, trabalho ou lazer chega-se ao valor de aproximadamente 6 milhões de brasileiros que podem cometer involuntariamente algum ato contra a sua própria segurança enquanto ficam verificando seus *smartphones*.

Segundo pesquisa realizada pelas empresas *Millward Brown* Brasil e *NetQuest* apresentada na revista Exame, o brasileiro passa em média cerca de 3h14min conectado ao celular por dia, enquanto os jovens nascidos do ano 2000 em diante gastam em média 4h por dia. A pesquisa aponta que realizar chamadas e navegar na internet correspondem à 89% e 87% respectivamente, das atividades mais realizadas pelos usuários (AMARAL, 2016).

Existe uma variedade quase inimaginável de *smartphones* e de empresas que os fabricam no mercado. Contudo, há poucos sistemas operacionais diferentes, praticamente 3 deles dominam o mercado, *Android* desenvolvido pela *Google*, *iOS* mantido pela *Apple* e em menor escala o *Windows* da *Microsoft*. De acordo com o levantamento da Forni

[1] Tradução do autor do verbete *smartphone* do *English Oxford Dictionarie*. Disponível em: <`https://en.oxforddictionaries.com/definition/smartphone`>.

e Meulen (2016), no segundo quarto de 2016, 86,2% de *smartphones* vendidos para o usuário final possuíam o *Android* como sistema operacional, 12,9% eram com *iOS* e 0,6% com *Windows*. A diferença ocorre porque os *smartphones* com *iOS* e *Windows* são apenas comercializados pela *Apple* e *Microsoft* respectivamente, enquanto o *Android* é utilizado por inúmeras empresas, uma vez que a *Google* não produz *smartphones* atualmente.

Com os dados apresentados e buscando uma maior abrangência do aplicativo de segurança, optou-se por implementar a solução encontrada no sistema operacional (plataforma) *Android*.

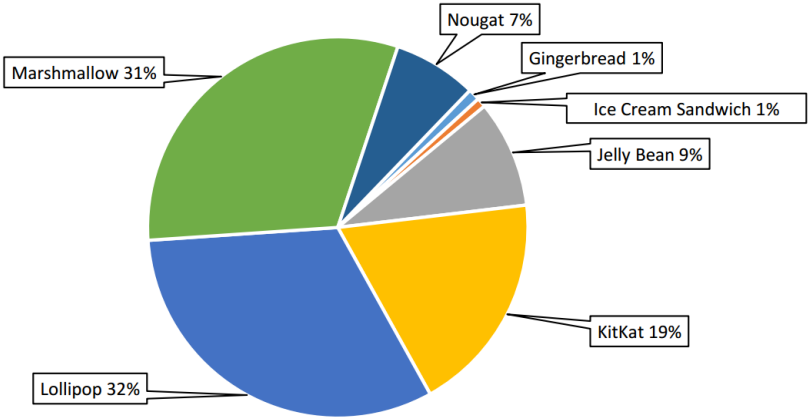
4.1 *ANDROID*

Segundo Lecheta (2013), o *Android* é uma plataforma de desenvolvimento para aplicativos móveis, baseada no sistema operacional *Linux* e com diversas aplicações, sendo um ambiente de desenvolvimento poderoso, ousado e flexível.

Devido ao fato de ser uma plataforma compatível com mais de 300 *hardwares* diferentes, presente em mais de 190 países, o *Android* é a maior base para qualquer dispositivo *mobile*, e cerca de 1 milhão de novos usuários a utilizam pela primeira vez em cada dia (GOOGLE, 2017a).

A primeira versão do *Android* surgiu em 2008 e praticamente a cada ano uma versão é lançada estando atualmente na versão 7.1 (API 25) conhecida como *Nougat*. Entretanto a versão com codinome *O* já está em fase de *preview*. Essa grande quantidade de versões, telas e *hardwares*, faz com que o *Android* seja bastante fragmentado. A Figura 4 destaca a diversidade dessas versões.

Figura 4: Distribuição das versões do *Android* nos *smartphones*.



Fonte: Adaptado de Google (2017c).

A Tabela 1 apresenta a divisão das versões do *Android* (API) que possuem mais de 0,1% de utilização. Esses dados foram computados pela *Google* verificando-se os acessos a *Google Play* (Loja de aplicativos para *Android*) durante o período de 7 dias finalizado em 7 de maio de 2017 (GOOGLE, 2017c). Observa-se que as versões mais atuais, (7.0 e 7.1) correspondem apenas a 7,1% dos dispositivos atuais. Isso ocorre porque cada empresa faz suas modificações sobre o sistema base disponibilizado pala *Google*, de forma que os aparelhos dos usuários acabam defasados em relação ao sistema base.

Ao desenvolver para *Android* não é recomendado focar apenas nas versões mais atuais, contudo escolher uma versão demasiadamente antiga, visando abranger um maior nicho, pode ser uma decisão arriscada, uma vez que novas tecnologias, sensores, e possibilidades são apresentadas a cada versão. Com o intuito de abranger uma faixa considerável e que possua os mínimos requisitos para o bom funcionamento da solução encontrada no Capítulo 3 optou-se pela API 19 (Versão 4.4) denominada *KitKat*, dessa maneira consegue-se abranger 89,1% (API 19 + API 20 + API 21 + API 22 + API 23 + API 24 + API 25) dos dispositivos atualmente em funcionamento.

Tabela 1: Versões do *Android*, Codinome, API e Distribuição.

Versão	**Codinome**	**API**	**Distribuição (%)**
2.3.3 - 2.3.7	*Gingerbread*	10	1.0
4.0.3 - 4.0.4	*Ice Cream Sandwich*	15	0.8
4.1.x	*Jelly Bean*	16	3.2
4.2.x		17	4.6
4.3		18	1.3
4.4	*KitKat*	19	18.8
5.0	*Lollipop*	21	8.7
5.1		22	23.3
6.0	*Marshmallow*	23	31.2
7.0	*Nougat*	24	6.6
7.1		25	0.5

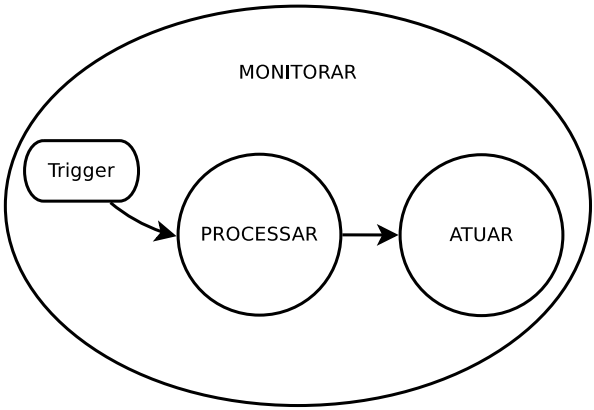
Fonte: Adaptado de Google (2017c).

4.2 APLICATIVO

Para o desenvolvimento do aplicativo utilizou-se o *Android Studio*, que é uma IDE (*Integrated Development Environment*) disponibilizada gratuitamente pela *Google* que está na versão 2.3.3.

A Figura 5 apresenta a visão global do funcionamento do aplicativo. Nota-se que o aplicativo foi divido em 3 grandes áreas, Monitorar, Processar e Atuar, as quais serão explicitadas na Seção 4.2.2. Observa-se na Figura 5 que monitorar engloba todo o algorítimo, esta é a premissa do aplicativo. Os dados só são processados após um *trigger*, o qual leva em consideração se o fones de ouvido está conectado, se há música sendo reproduzida no dispositivo e o usuário está em movimento. Caso alguma dessas condições não seja satisfeita o processamento é parado visando economia de recursos do *smartphone*.

Figura 5: Visão global do Aplicativo.



Fonte: O autor.

À medida que o processamento é realizado, amostras de 0,25s são capturadas pelo microfone enviadas para classificação com K-NN e caso a classe de saída seja Perigo (representada por 1) o aplicativo requer a disponibilidade do áudio para reprodução. Como ele não reproduz nenhum padrão sonoro, é o mesmo que dizer que o som foi pausado, depois de alguns segundos o áudio anterior retorna a execução. Contudo, caso a classe de saída seja a normal (representada por 0) nada acontece com a reprodução do conteúdo. Esse *loop* é contínuo até que uma das condições elencadas anteriormente não seja satisfeita.

4.2.1 Recursos

O *Android* disponibiliza muitos recursos para o desenvolvedor e a cada nova versão alguns recursos são adicionados, outros são substituídos ou melhorados. Contudo, não é o escopo apresentar todos os inúmeros recursos presentes no *Android* ou na versão utilizada (*KitKat*) e sim aqueles que são indispensáveis para o funcionamento do aplicativo, com o intuito de ajudar na compreensão. Esse recursos são apresentados brevemente nas subseções a seguir.

4.2.1.1 *Broadcast Receiver*

A classe *BroadcastReceiver* é uma importante classe presente no *Android* sendo utilizada para que as aplicações possam reagir a eventos gerados por uma *Intent*. De forma sucinta é a classe que reage as mensagens enviadas ao sistema operacional (GOOGLE, 2017b).

Esta classe é executada apenas em segundo plano, de forma a reagir com as mensagens do sistema sem ter que apresentar uma tela para o usuário. Seu objetivo é a troca de informação (mensagens) sem que o usuário perceba (LECHETA, 2013).

4.2.1.2 *Intent* e *Intent Filter*

Uma *Intent* pode ser definida como uma mensagem da aplicação para o sistema operacional, solicitando que alguma ação seja realizada por outra aplicação, e representa um importante papel na arquitetura do *Android* e como aplicações interagem entre si (LECHETA, 2013).

Segundo Google (2017e) há três casos fundamentais para a utilização do *Intent*, elas são:

- **Para iniciar uma *Activity***: *Activity* (Atividade) representa as telas para o *Android*. Pode-se utilizar o *Intent* para abrir e carregar os dados de outra *Activity* da aplicação.
- **Para iniciar um *Service***: A *Intent* descreve o *service* que será iniciado e carrega todos os dados necessários.
- **Para fornecer uma *Broadcast***: *Broadcast* (transmissão) é uma mensagem que qualquer aplicativo pode receber.

Os *Intents* podem ser divididos em explícitos e implícitos. Os explícitos especificam qual componente irá iniciar utilizando o nome exato da classe e é utilizado principalmente para iniciar componentes do próprio aplicativo. Os implícitos declaram uma ação que deseja-se realizar e o sistema operacional executa com o aplicativo que consegue realizar a tarefa ou apresenta os possíveis aplicativos que possam ser utilizados, caso haja mais de um compatível (GOOGLE, 2017e).

A função do *Intent Filter* é filtrar as mensagens enviadas para o sistema operacional quando deseja-se utilizar o *Intent* é necessário configurar a classe *Intent Filter* para interceptar a requisição e executar o que foi pedido (LECHETA, 2013).

4.2.1.3 Sensores

Existem diferentes sensores nos *smartphones Android*. Cada fabricante adiciona os *hardwares* que acredita ser um diferencial ou necessários para sua aplicações específicas. Segundo (GOOGLE, 2017g). os sensores suportados são separados em três categorias:

- *Motion Sensors*: são os sensores responsáveis por mensurar as forças de aceleração e rotação nos três eixos (x, y, z) e incluí os seguintes sensores: acelerômetro, sensor de gravidade, giroscópio, etc.

- *Environmental Sensors*: são os sensores que mensuram os parâmetros de ambiente, tais como intensidade da luz, temperatura, etc.

- *Position Sensors*: são sensores que medem a posição física do dispositivo e enquadram-se nesta categoria os sensores de orientação e *magnetometers*.

É importante ressaltar que nem todos os sensores estão disponíveis, depende do *hardware* ou da versão do *Android*, e.g., é o sensor de *Step Detector Sensor* (Sensor de detecção de passos) que apesar de ser introduzido junto com o *KitKat*, a maioria dos modelos de *smartphones* que utilizam essa versão não possuem. Esta foi uma limitação encontrada na construção do aplicativo, optando-se por utilizar apenas o acelerômetro visto que este está presente em praticamente todos os dispositivos.

4.2.1.4 *Service*

Service ou serviço é o recurso mais importante para o aplicativo desenvolvido, visto que ele é o componente que pode realizar longas operações e não precisa de interface com o usuário (GOOGLE, 2017h). Pode-se dizer que um *service* possui duas formas, iniciado e vinculado.

O *service* iniciado é quando o aplicativo utiliza o método *startService* para iniciar o *service*. Dessa forma o *service* pode ficar executando indefinidamente até que o método *stopService* seja chamado, mesmo que a aplicação não esteja mais executando (GOOGLE, 2017h).

O *service* vinculado é quando o aplicativo utiliza o método *bindService* para iniciar o *service*. Neste caso ele fica vinculado ao aplicativo só mantendo-se em execução enquanto o vinculo é mantido. Vários

componentes podem ser vinculados ao serviço de uma vez só, de forma que quanto todos desfizerem o vinculo o serviço será destruído, (GOOGLE, 2017h).

Mesmo que o *service* seja encerrado pelo sistema operacional devido a falta de memória para executar outros aplicativos com maior prioridade, o sistema operacional tentará reiniciá-lo para que o processamento continue, desde que as condições dos recursos necessários estejam normalizados. Neste caso é importante que o *service* seja implementado de forma a poder recuperar o estado caso ele seja encerrado (LECHETA, 2013).

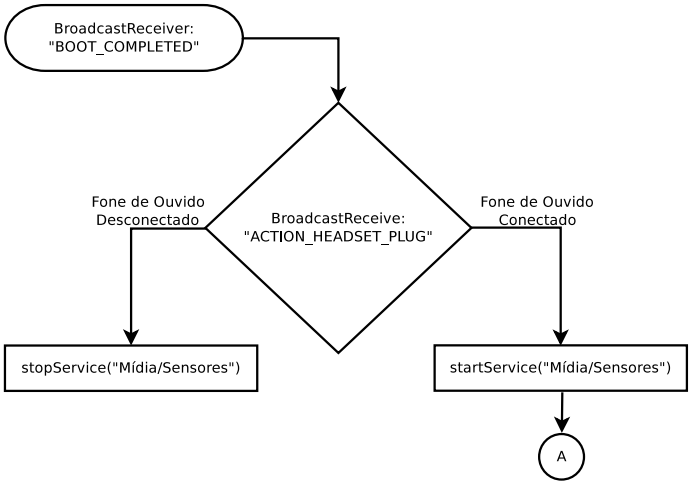
4.2.1.5 *Audio Focus*

De acordo com o apresentado na documentação oficial do *Android*, qualquer aplicativo que deseja reproduzir algum áudio para o usuário deve solicitar o foco de áudio para o aplicativo imediatamente antes da reprodução começar, (GOOGLE, 2017d). Isso é importante para que a mixagem de áudio não diminua a experiência do usuário. O fato de conseguir a permissão para executar o áudio e depois abandonar a reprodução é o que será utilizado para pausar qualquer áudio que esteja sendo executado, visto que a documentação também instrui os desenvolvedores a implementarem ações para quando os aplicativos perdem o foco do áudio.

4.2.2 Fluxograma de Funcionamento

O aplicativo teve seu desenvolvimento guiado pelos fluxogramas apresentados nesta seção, os quais também descrevem como o aplicativo funciona e como ele se comporta em situações específicas fundamentais. A Figura 6 apresenta o *trigger* para o sistema entrar em funcionamento. Inicialmente o aplicativo espera uma mensagem de *Broadcast* do sistema operacional informando que o sistema já foi carregado ("*BOOT_COMPLETED*") de forma que dispara o *service* responsável apenas pela identificação de que o fones de ouvido foi conectado ou desconectado.

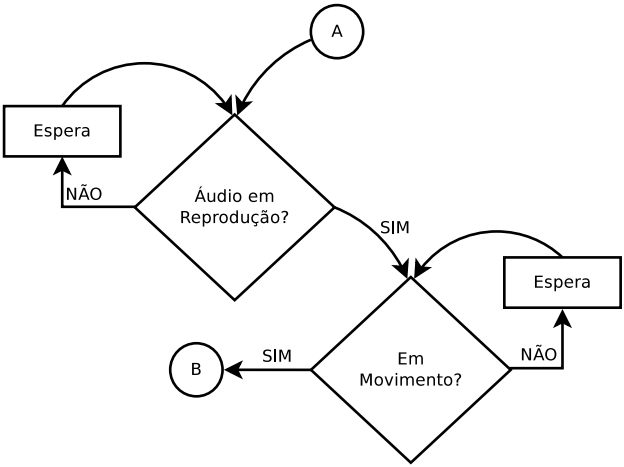
Figura 6: Monitora, parte 1/2.



Fonte: O autor.

A Figura 7 apresenta o comportamento do aplicativo após a identificação de que o fones de ouvido foi conectado. Esse é o *trigger* principal da aplicação e quando ativado ele inicia o *service* responsável por verificar em intervalos de tempos pequenos (segundos) se existem áudio em reprodução. Para esta aplicação, quando mencionado áudio, ele pode corresponder a musica, vídeo ou ligação. Caso o usuário esteja utilizando reprodução de áudio, é verificado através de leituras do acelerômetro, que leva 5 segundos, se o usuário (dispositivo) está em movimento. Essa verificação é realizada com intervalos de tempo maiores (minutos).

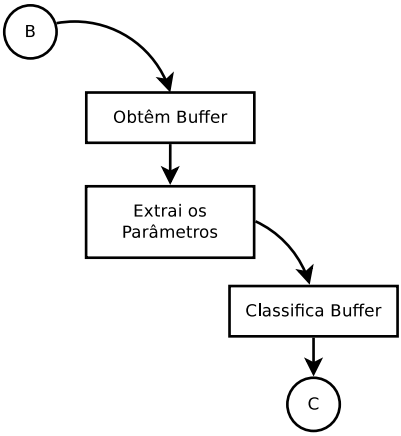
Figura 7: Monitora (Mídia/Sensores), parte 2/2.



Fonte: O autor.

Uma vez que todos os três *triggers* se mantenham ativados inicia-se o *service* de processar o áudio, que é responsável por obter um *buffer* deslizante de tamanho correspondente à 0,25s. Para cada *buffer* os dados estatísticos, apresentados na Seção 3.4, são extraídos e enviados para a classe responsável por classificá-lo utilizando o K-NN. A classe de saída é a responsável por ativar o atuador ou não. O fluxograma de funcionamento é apresentado na Figura 8.

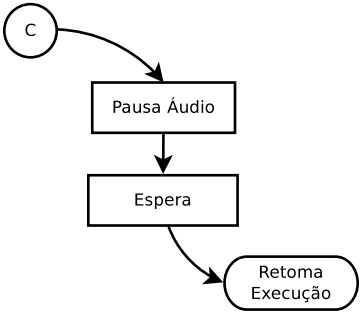
Figura 8: Processa.



Fonte: O autor.

Após ser identificada uma classe perigo, como apresentado na Figura 9, o sistema pausa a execução do áudio por alguns segundos de forma que não há nenhum som sendo reproduzido nos fones de ouvido. Assim, assegurará que o usuário do dispositivo escute o que está acontecendo ao seu redor e, caso seja algum perigo eminente para ele, execute alguma ação. Após esse período de silêncio o reprodução anterior é retomada.

Figura 9: Atuador.



Fonte: O autor.

As partes apresentadas nas Figuras 8 e 9 são executadas em *loop* infinitamente até que alguma condição apresentada na Figura 7 deixe de ser verdade. Contudo, os *services* atribuídos só serão realmente

destruídos quando o fone de ouvido for desconectado, como apresentado na Figura 6. Vale ressaltar que se o sistema não contemplar e/ou deixar de contemplar alguma das condições da Figura 7, os *services* associados à obtenção do *buffer*, extração dos parâmetros e classificação não serão executados de forma a preservar recursos do *smartphone*.

5 AVALIAÇÃO DO RESULTADOS

Devido ao fato do trabalho compreender o desenvolvimento e a implementação da solução encontrada no Capítulo 3, os resultados são apresentados em seções distintas, afim de prover uma melhor forma de analisá-los.

Os resultados da extração dos parâmetros são apresentados na seção a seguir, enquanto os da implementação em *smartphone* (Capítulo 4) estão descritos na Seção 5.2.

5.1 MÉTODOS DE EXTRAÇÃO DE PARÂMETROS

Com auxílio do *software MATLab* foi possível realizar variações dos parâmetros e analisar o comportamento do classificador em relação a estes parâmetros, de modo a escolher os melhores valores que se adequam a solução proposta. Utilizou-se dois métodos distintos, de forma que os resultados são apresentados em subseções distintas.

5.1.1 Resultados Método 1

Apesar de que todos as variações descritas na Seção 3.3 utilizarem os 15 áudios disponíveis até então, foram selecionados 4 áudios que caracterizam os resultados obtidos em todo o conjunto. Optou-se por utilizar o método de Percentual de Acerto, descrito na Seção 2.3.1, para expressar os resultados.

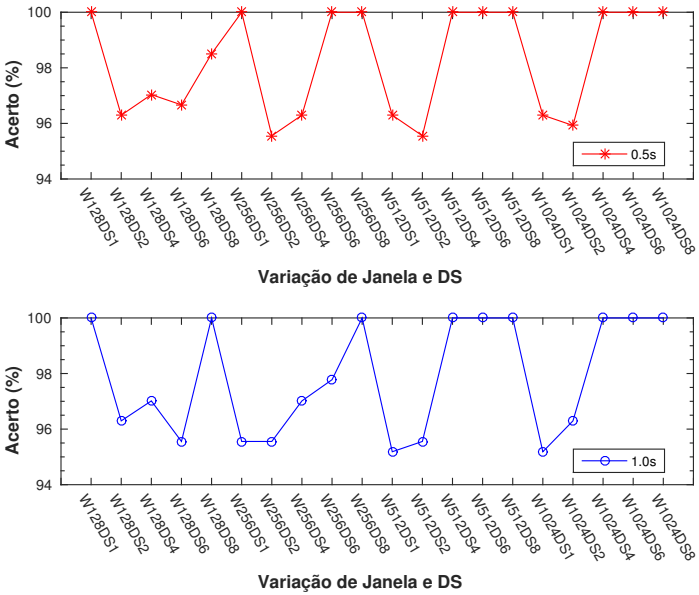
Ocorreu variação de vizinhos (K), Janela (W), *Downsampling* (DS) e tempo (0,5; 1,0 e 1,5 segundos). Contudo, os gráficos aqui apresentados contém o percentual de acerto para $K = 15$, que ao avaliar os resultados provou ser a melhor opção.

Utilizou-se o eixo x para abrigar a variação de Janela e DS, enquanto o eixo y corresponde ao percentual de acerto. O gráfico superior apresenta o $tempo = 0,5$ enquanto o inferior possui $tempo = 1,0$, ambos em segundos. Os resultados para o $tempo = 1,5$ segundos não são apresentados, visto que não acrescentam informações úteis para a definição dos demais parâmetros.

Os resultados apresentados nas Figuras 10 e 11 correspondem em sua totalidade a sons considerados classe normal. Nota-se na Figura 10 que a variação dos parâmetros fazem que o percentual de acerto

oscile. Contudo, para qualquer valor o resultado é satisfatório pois está acima de 94%. A variação de tempo também não compromete o resultado, sendo que o tempo de 0,5 segundo apresenta, no geral, o melhores resultados, considerando-se que a aplicação terá que possuir a menor latência possível, i.e., o tempo entre o início do evento (sinal de perigo) e a percepção do sistema desse evento. Para isto o tempo de 0,5 segundo é o ideal.

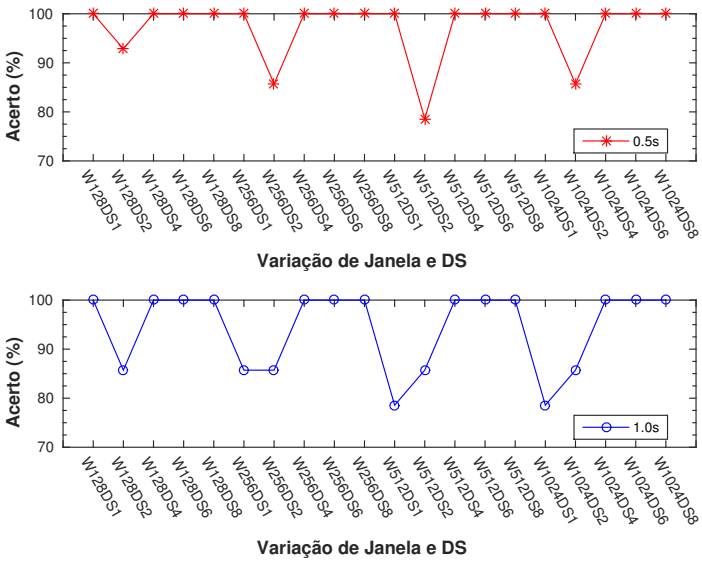
Figura 10: Gráfico para Áudio 3 com tempos de 0,5 e 1,0 segundo.



Fonte: O autor.

No outro áudio normal selecionado pode-se observar que apesar de possuir percentuais de acerto elevados (acima de 75%), eles são no geral, inferiores aos obtidos para o Áudio 3. Da mesma forma que o anterior, para este tipo de áudio o valor de $DS = 2$ apresenta os piores resultados, não importando o valor da Janela. O valor do tempo também não é um fator que influência na escolha do parâmetro.

Figura 11: Gráfico para Áudio 8 com tempos de 0,5 e 1,0 segundo.

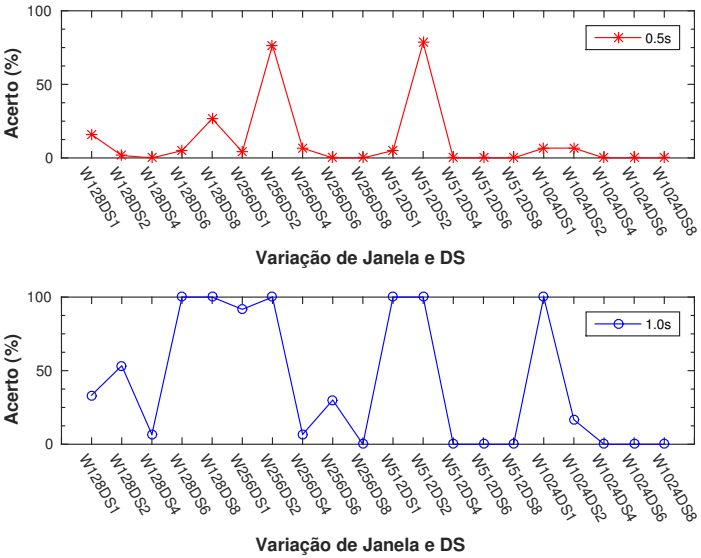


Fonte: O autor.

Avaliando-se os sons que pertencem a classe perigo (Figuras 12 e 13), nota-se uma queda drástica no percentual de acerto geral do classificador.

Na Figura 12, independente dos valores atribuídos para Janela e DSm os resultados não são satisfatórios. O ponto com valor de $Janela = 256$, $DS = 2$ e $Janela = 512$ e $DS = 2$ apresenta os melhores resultados, cujos valores encontram-se entorno de 80%. Qualquer outras configurações de janela e DS apresentam resultados que não podem ser utilizados numa aplicação de segurança. Diferentemente do comportamento para áudios normais, o valor $tempo = 1,0$ segundo apresenta melhores resultados. Partindo-se deste pressuposto, o tempo de latência seria maior visto que só de dados a serem analisados há 0,5 segundo de tempo acrescido e quanto mais dados é necessário processar maior será o tempo até que a resposta seja obtida.

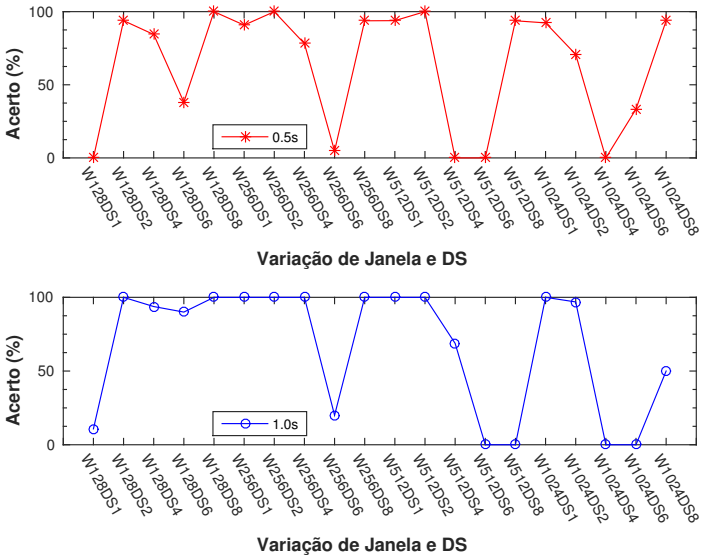
Figura 12: Gráfico para Áudio 13 com tempos de 0,5 e 1,0 segundo.



Fonte: O autor.

A Figura 13 mostra melhores resultados, obtendo-se acertos acima de 80%, porém ao contrário dos gráficos para sons de classe normal, no qual o valor de DS baixo ($DS = 2$) produz resultados abaixo dos demais, para este áudio temos que valores de DS elevados (4, 6 e 8) geram resultados com baixo percentual de acerto. Acredita-se que o valor de Janela não influencia de maneira significativa o percentual de acerto. De forma similar, a Figura 12, utilizando-se o *tempo* $= 1,0$ segundo, aumenta a eficácia do classificador. Entretanto, nota-se a existência de uma maior inconsistência no percentual de acerto no sons de perigo, o que pode representar a não adequação deste método para a aplicação proposta.

Figura 13: Gráfico para Áudio 14 com tempos de 0,5 e 1,0 segundo.



Fonte: O autor.

Após analisar os resultados para 15 áudios diferentes, optou-se por utilizar os valores de $K = 15$, $Janela = 512$, $DS = 2$, que apresentaram melhor resultado geral e que são apresentados na Figura 14. Neste ponto foi acrescentado mais dois áudios de perigo com características diferentes dos já utilizados.

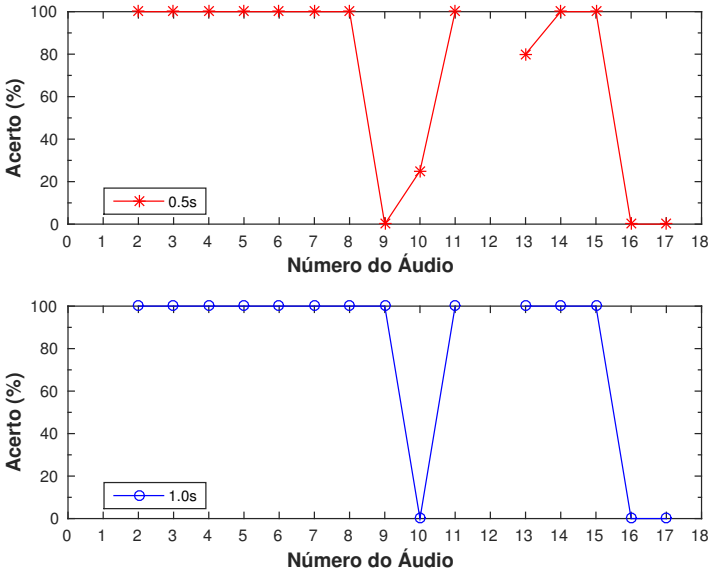
Os áudios 1 a 8 são áudios normais, enquanto os áudios 11, 12, 13, 14 e 15 são de diferentes tipos de sirenes e os áudios 8, 9, 16 e 17 são de buzinas distintas.

Verificou-se que o classificador, independente do tempo, classifica com mais de 90% de acerto os áudios normais e acima de 80% as sirenes. Contudo, quando trata-se das buzinas, o classificador não consegue atingir marcas satisfatórias. O áudio 9 passa de 0% a 100% com a modificação do tempo para 1,0 segundo e, como mencionado, isso aumenta o tempo de latência, o que não é desejado. Porém, mesmo com esta melhora os áudios 16 e 17 não conseguem ser classificado como perigosos. É valido mencionar que realizou-se a classificação dos áudios com uma composição de normal, perigo e perigo-buzina no conjunto de

treinamento. No entanto, isto não gerou modificações pertinentes nos resultados encontrados.

Nota-se que a Figura 14 inicia no áudio 2 e possuí uma descontinuidade no áudio 12, o que deve-se ao fato desses comporem o conjunto de treinamento.

Figura 14: Gráfico para os 17 áudios com tempos de 0,5 e 1,0 segundo.



Fonte: O autor.

Como os resultados apresentados não proporcionam valores adequados para um aplicativo de segurança, optou-se por utilizar a abordagem do Método 2, descrito na Seção 3.4.

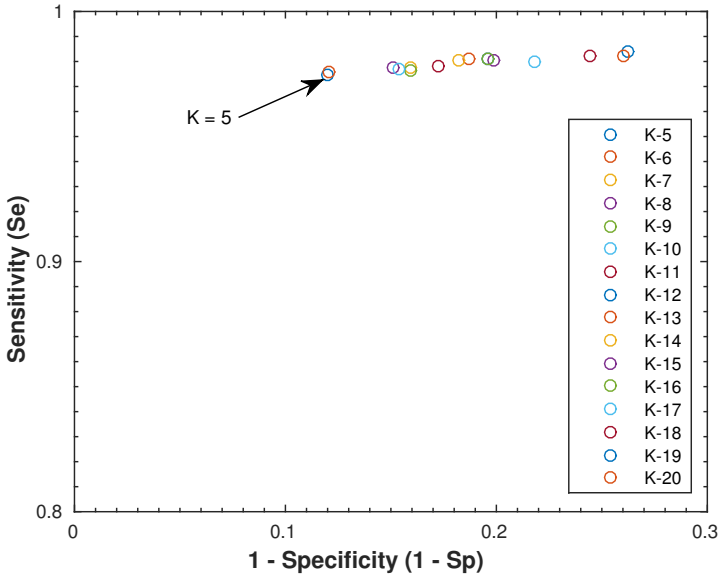
5.1.2 Resultados Método 2

Devido a formulação do Método 2 é possível gerar uma classificação pessoal com a classe que cada um das partes do áudio pertence. A partir dessa classificação pode-se utilizar a avaliação de resultados baseado no Método *Sensitivity* e *Specificity* apresentado na Seção 2.3.2,

o que facilita a escolha do valor de K.

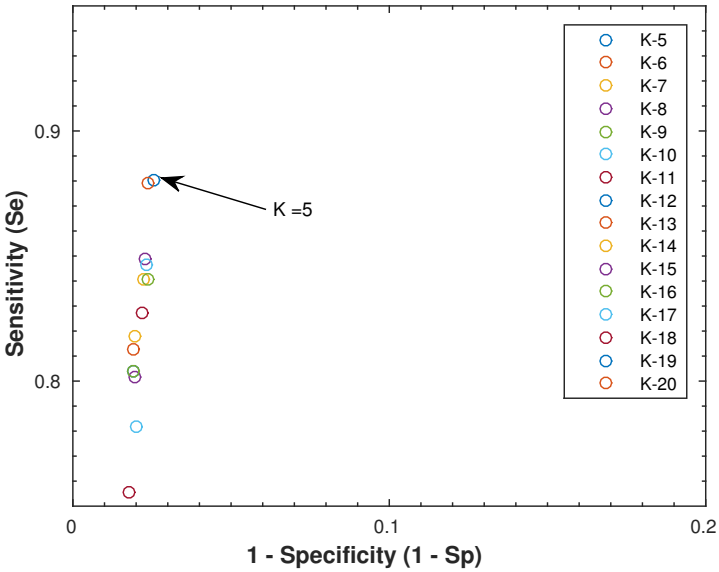
As Figuras 15 e 16 apresentam os valores de *Sensitivity* e 1 − *Specificity* para a classe normal e para classe perigo, respectivamente. Os melhores valores são aqueles que estão no canto superior esquerdo, os quais corresponde à uma alta *Sensitivity* e *Specificity*. Contudo, diferente do realizado no Método 1, todos os áudios utilizados nesta seção, excluindo-se os que compõem o treinamento, foram unidos formando um único áudio. Da mesma forma a classificação desejada também foi unida e este mecanismo foi utilizado para que os cálculos fossem justos e os valores encontrados pudessem ser utilizados, caso contrário seria necessário dois gráficos para cada áudio testado. Com esta técnica temos o mapeamento geral do comportamento do classificador.

Figura 15: Curva ROC para classe 0 (Normal).



Fonte: O autor.

Figura 16: Curva ROC para classe 1 (Perigo).



Fonte: O autor.

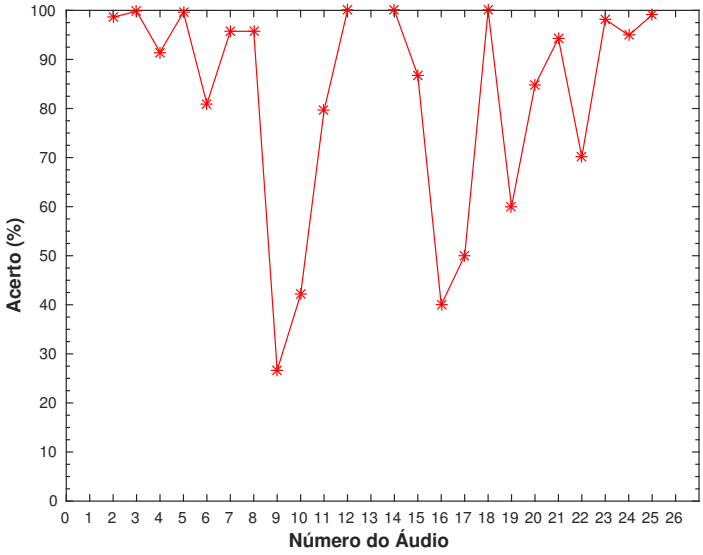
Como apresentado na Figura 15 e reforçado na Figura 16, existem dois valores de K que poderiam ser utilizados: $K = 5$ e $K = 6$. Apesar do valor $K = 6$ apresentar resultados muito semelhantes ao $K = 5$, optou-se por utilizar o $K = 5$ devido ao fato que não há peso atribuído as distâncias e por este ser um valor de K ímpar evita-se o empate, (ROGERS; GIROLAMI, 2011).

Nota-se que ambas Curvas ROC possuem valores superiores a 0,85 para S_e e inferiores a 0,15 para $1 - S_p$, mostrando resultados excelentes no contexto geral. Contudo, o classificador pode apresentar diferentes percentuais de acerto para os áudios testados e a Figura 17 apresenta o resultado individual destes. Os pontos de descontinuidade são dos áudios que compõem o conjunto de treinamento.

Observa-se que apesar do gráfico apresentar uma inconsistência maior para os áudios de classe normal (áudios 2 a 8) em relação a Figura 14, os valores de percentual de acerto para sons de perigo, os de Sirenes e, principalmente, aqueles de Buzinas (áudios 9, 10, 16, 17) apresentam uma melhora no percentual global de acerto. Claramente

o classificador pode evoluir afim de apresentar resultados melhores. Os áudio mistos (áudios 18 a 25) apresentam valores satisfatórios.

Figura 17: Gráfico para os áudios de teste.



Fonte: O autor.

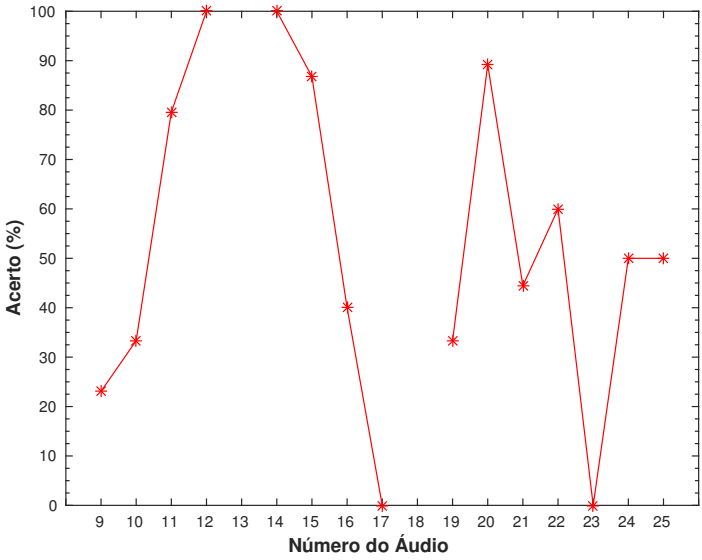
O Método 1, descrito na Seção 3.3, não obteve bons resultados com $tempo = 0,5s$, enquanto o Método 2 conseguiu um desempenho superior utilizando-se de metade deste tempo.

A Figura 18 apresenta os valores do percentual de acerto olhando-se apenas para a parte de perigo, i.e., quanto do que era para ele classificar como perigo que ele realmente classificou como tal. Apesar de apresentar valores não satisfatórios e ao compará-los com a Figura 17 aparentar ser inferior, alguns detalhes devem ser levados em consideração.

Os áudios 9, 10, 17 e 19 são amostras com características distintas daqueles que constituem o treinamento. Especificamente o áudio 19 é uma buzina de ar que em seu tom mais elevado equipara-se à uma buzina do treinamento. O áudio 23, que em termos da classificação dos sinais de perigo obteve um acerto nulo, é composto de buzinas com

baixa amplitude, o que pode ser considerado que elas estejam distante do usuário. Os áudios 24 e 25 apresentavam cerca de 0,5 segundo (2 amostras com classe de perigo, em sequência) de buzinas em áudios de 15 e 27 segundos, respectivamente. A distinção do percentual de acerto entre eles está no fato que o áudio 24 teve classes de perigo atribuídas a amostras de classe normal, enquanto o 25 teve apenas uma classe definida de forma errônea. Contudo, em uma situação real o aplicativo seria capaz de cumprir sua premissa, uma vez que as classes perigo estavam em sequência.

Figura 18: Gráfico para os áudios de teste, analisando-se apenas a classe perigo.



Fonte: O autor.

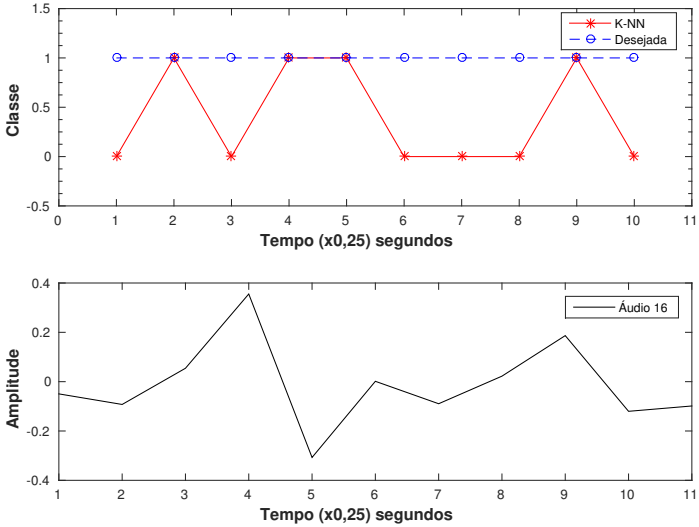
Utilizando-se das gravações obtidas com o aplicativo desenvolvido, onde gerou-se amostras de buzinas provenientes de 12 carros distintos e distâncias variando entre 5 e 30 metros, obteve-se 207 amostras, dentre elas 57 foram utilizadas para treinamento. As 150 restantes, algumas com a presença de falas ou vento, foram submetidas a classificação pela solução encontrada obtendo-se um resultado de percentual

de acerto equivalente à 66%.

Para melhor exemplificar as classes que o K-NN definiu em relação a classificação pessoal utilizada para calcular o percentual de acerto, as Figuras 19 e 20 apresentam o som utilizado, a classificação do K-NN e a classificação desejável.

A Figura 19 apresenta a classificação do Áudio 16, um áudio problemático no Método 1, cujo percentual de acerto foi 0%. Extraindo mais características foi possível aumentar o percentual de acerto para mais de 40%. Nota-se que apenas algumas frações de segundo não coincidem com a classificação desejada. Entretanto, é possível notar que o classificador optou pela classe errônea em no máximo 3 amostras em sequência. Isso representa que ele levaria no mínimo 1 segundo para classificar o sinal de perigo.

Figura 19: Gráfico contendo o áudio 16, a classificação do K-NN e a classificação desejada.

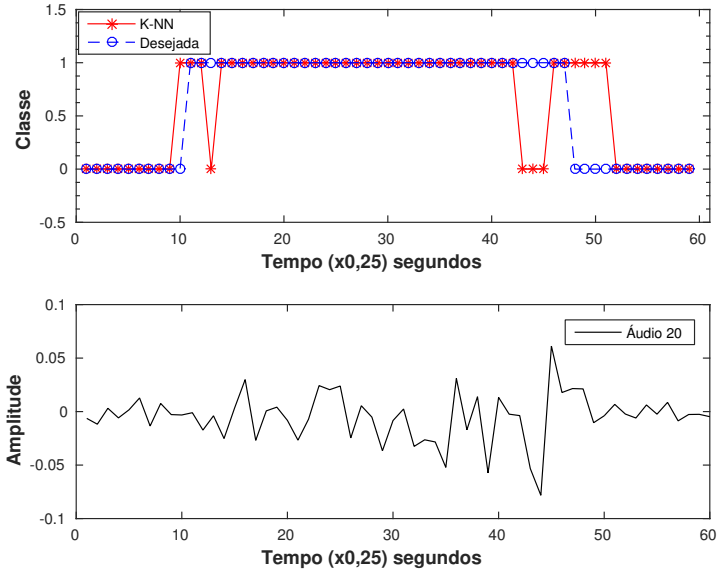


Fonte: O autor.

Com um resultado semelhante, a Figura 20 possuí algumas distorções do que seria desejável, principalmente no final do áudio. Porém o áudio é composto por um carro de polícia com sirene se aproximando

e parando. Mesmo que a classe desejada no final seja normal, porque é uma sirene sendo desligada. Não se pode considerar como um erro absoluto do classificador e caso a classificação desejada não fosse específica neste tipo de situação os resultados gerais seriam melhores.

Figura 20: Gráfico contendo o áudio 20, a classificação do K-NN e a classificação desejada.



Fonte: O autor.

Mesmo que o classificador não contenha um percentual de acerto perto de 100% não quer dizer que essas configurações não possam ser utilizadas em uma aplicação real. Apesar das diferentes buzinas, com distintas distâncias, este método obteve um percentual de acerto maior que 66%, tornando plausível a sua utilização. Contudo, melhorias na escolha dos parâmetros e/ou no treinamento pode agregar maior diversidade de buzinas com o intuito de otimizar o classificador.

Nota-se porém que cerca de 28% das buzinas classificadas de forma errada são as que possuem um nível maior de interferência do vento em sua gravação ou que possuem áudio de buzina menor que 250ms.

5.2 APLICATIVO DESENVOLVIDO

Após desenvolver uma solução com percentual de acerto viável, optou-se por implementá-la para *Android*, de forma que fosse possível visualizar o funcionamento do classificador em condições normais de uso.

Ao desenvolver um aplicativo para *Android* utiliza-se basicamente o *Software Development Kit* (SDK), o que proporciona uma programação orientada o objetos, neste caso Java. Contudo, existe a alternativa de utilizar o *Native Development Kit* (NDK), o qual permite que os aplicativos utilizem arquivos fontes em C ou C++ integrados aos escritos em Java, ou simplesmente eles. A utilização do NDK tem pouco valor para muitos aplicativos, visto que a complexidade ao utilizar é elevada e muitas vezes não justifica o uso, (GOOGLE, 2017f). Apesar de ser recomendada para obter-se mais desempenho, este aplicativo foi desenvolvido em Java na sua totalidade.

Os fluxogramas apresentados na Seção 4.2.2 determinam o fluxo lógico do aplicativo. Entretanto, não representam apenas uma classe, visto que o aplicativo foi modularizado a fim de proporcionar testes individuais de funcionamento e após o teste do aplicativo completo.

O primeiro módulo desenvolvido foi o responsável por capturar o áudio no formato WAV e sem compressão. Inicialmente desenvolveu-se um classe que adquiria o áudio por um período de 250ms e armazenava na memória do dispositivo. Nesta etapa, pode-se avaliar as diferenças entre obter o áudio *Stereo* e depois "convertê-lo" em MONO ou obter em MONO. Optou-se por obter diretamente em MONO visto que não houve variação significativa. Contudo, o *Android* não grava WAV nativamente, sendo necessário obter um arquivo temporário PCM de 16bits com extensão .raw e adicionar o cabeçalho de WAV após a conclusão da gravação. A partir dessa limitação, escolheu-se utilizar o arquivo sem a adição do cabeçalho na versão final do aplicativo, da mesma forma que não é salvo o arquivo na memória do telefone.

O segundo módulo é responsável por calcular o definido na Seção 3.4 e, devido a necessidade da transformada de *Fourier*, optou-se por utilizar uma disponível na internet, sendo a única classe não implementada. Essa decisão foi tomada baseada no fato que uma desenvolvida pelo autor não conseguiria um desempenho semelhante ao da Universidade de Columbia (fonte da FFT). Esta possui licença para utilização desde de que mantido o cabeçalho, o que foi feito.

Os resultados foram comparados com os obtidos pelo *MATLab* e neste ponto foi necessário ajustar a escala dos resultados obtidos com o

smartphone, o que não comprometeu o tempo de processamento total do aplicativo.

A classe que processa áudio também é responsável por pausar e retomar o áudio quando um áudio é classificado como perigoso.

O terceiro módulo corresponde ao desenvolvimento da classe que implementa o K-NN. Como o conjunto de treinamento é extenso, uma matriz de 1752 linhas por 104 colunas, utilizou-se um arquivo com extensão csv para carregá-lo na memória RAM do *smartphone*. Este processo leva cerca de 2 segundos e é carregado assim que as 3 condições (fone conectado, áudio em reprodução e *smartphone* em movimento) para começar a gravar são verdadeiras.

No quarto módulo, estão contidos todas as verificações das condições necessárias para o aplicativo entrar em funcionamento. Existe uma classe responsável apenas por verificar se o *boot* do celular for completado. Esta classe inicia o *service* que verifica se o fone foi conectado ou desconectado. Caso sim, ele inicia o *service* que verifica as condições ou, do contrário, destrói o *service*.

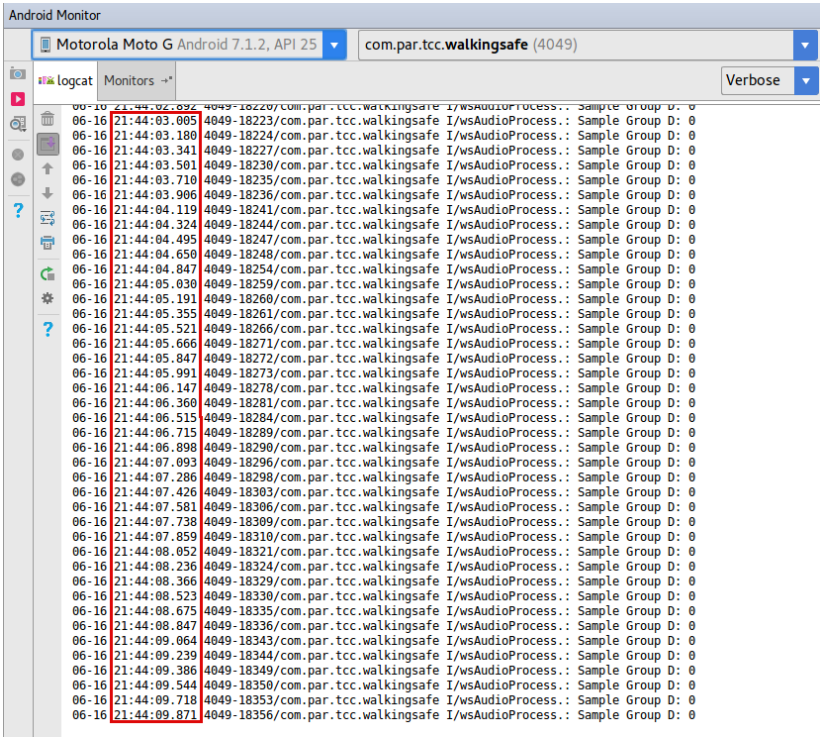
O *service* que verifica as condições busca a informação se há música em execução ou se existe um ligação em andamento a cada 10s. Com intervalo de 2min é executado a verificação se o dispositivo está em movimento, para ativar o monitoramento basta que ele esteja em movimento uma única vez. Contudo, para cancelar é necessário que 3 leituras indiquem que o dispositivo esteja parado. Para verificar se o usuário está ou não em movimento, utilizou-se o acelerômetro, que inclui o valor da gravidade. Em testes com *smartphones* verificou-se que se o valor absoluto da diferença de 25 leituras (cerca de 5s) seja superior à 2 para os eixos x e y e maior que 2,5 para o eixo z, o dispositivo está em movimento, caso essas três condições não sejam atingidas, diz-se que o *smartphone* está parado.

Pode-se expressar os resultados de duas maneiras: tempo de processamento e consumo de recursos.

5.2.1 Tempo de Processamento

Apesar da dificuldade de expressar esses valores com a precisão que são executadas no *smartphone*, o *Android Studio* provê uma *interface* capaz de apresentar os dados de *log* do dispositivo tornando possível determinar o tempo aproximado de cada módulo. A Figura 21 apresenta o *log* da classe de cada amostra testada, onde é possível visualizar o tempo médio entre cada processamento.

Figura 21: *Android Studio log* para cada amostra analisada.



Fonte: O autor.

Na versão final do aplicativo existe um *buffer* deslizante que manda 250ms de áudio para análise a cada 50ms. Contudo, o aplicativo leva em média 150ms para processar e obter o valor da classe para cada *buffer* analisado. O *smartphone* testado foi um Motorola Moto G de 1ª geração (2013) com um processador de 1.2 GHz *Quad Core*. Em dispositivos mais modernos a tendência é que todo processo seja executado com maior velocidade, de forma análoga em dispositivos mais antigos o tempo será maior. Contudo, pode-se afirmar que para a maioria dos *smartphones* o aplicativo executa em tempo real.

A Figura 22 apresenta *logs* e através deles podemos estimar os tempos para cada uma das principais etapas. Quando o *log* "Processando" é apresentado corresponde que uma amostra com 250ms de áudio começou a ser processada. Da mesma forma quando o *log* "Skip-

ping" é apresentado quer dizer que uma nova amostra deslocada 50ms foi recusada. O *log* "Sample Group: " representa o fim do processamento da determinada amostra, apresentando a classe 0 ou 1. Observa-se que o valor total de processamento para o destacado foi de 134ms. A transformada rápida de *Fourier* levou 51ms. Os cálculos dos coeficientes para o K-NN demorou 61ms e a classificação levou 13ms. É claro que esses valores são apenas uma amostra do que pode ser obtido. Existe uma variação esperada para esses tempos por causa do compartilhamento de recursos com outras aplicações executando no *smartphone*. Alguns testes apresentaram resultados de 73ms para todo o processamento, enquanto em outros os valores foram próximos à 200ms. Um fator que contribuí para tempos mais altos é a necessidade de enviar o *log*, os quais em uma versão de distribuição não haverá necessidade.

Figura 22: *Android Studio log* para cada amostra analisada.

```
23:01:22.412 25885-506/com.par.tcc.walkingsafe D/wsAudioProcess.: Processando
23:01:22.413 25885-1703/com.par.tcc.walkingsafe D/wsAudioProcess.: Padronização do Áudio
23:01:22.420 25885-1703/com.par.tcc.walkingsafe D/wsAudioProcess.: FFT - Início
23:01:22.450 25885-506/com.par.tcc.walkingsafe D/wsAudioProcess.: Skipping
23:01:22.471 25885-1703/com.par.tcc.walkingsafe D/wsAudioProcess.: FFT - Fim
23:01:22.471 25885-1703/com.par.tcc.walkingsafe D/wsAudioProcess.: Padronização do Amostra
23:01:22.472 25885-1703/com.par.tcc.walkingsafe D/wsAudioProcess.: Inicia Cálculos
23:01:22.511 25885-506/com.par.tcc.walkingsafe D/wsAudioProcess.: Skipping
23:01:22.533 25885-1703/com.par.tcc.walkingsafe D/wsAudioProcess.: Inicia Classificação
23:01:22.546 25885-1703/com.par.tcc.walkingsafe I/wsAudioProcess.: Sample Group: 0
```

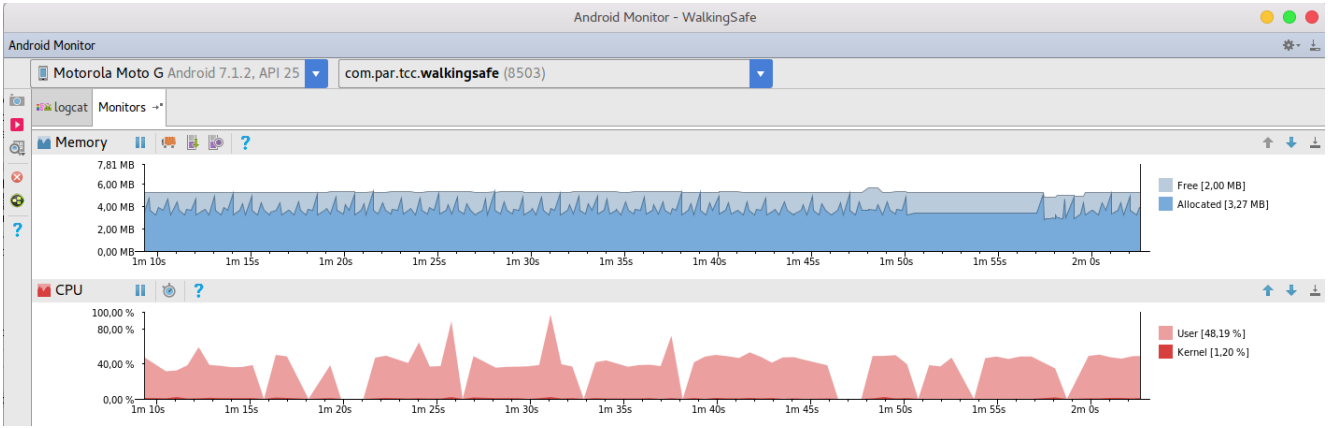
Fonte: O autor.

5.2.2 Consumo de Recursos

Para avaliar o consumo dos recursos do *smartphone* enquanto o aplicativo estava em plena execução foi utilizado o monitoramento disponibilizado no *Android Studio*. Deste modo foi possível estimar o uso de memória, processador e bateria.

A Figura 23 apresenta dois gráficos: o primeiro do consumo de memória e o segundo de processador. Observa-se que o consumo médio de memória é inferior à 6MB somando-se o alocado e liberado.

Figura 23: *Android Studio* Monitor, enquanto o aplicativo é executado.



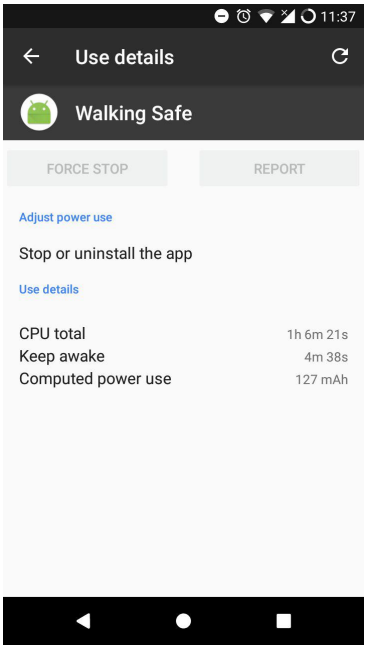
Fonte: O autor.

Em relação a utilização da CPU, o valor médio é de 50%, com picos de 80%. Espera-se valores bem elevados, dado a quantidade de *threads* executando. Isso pode afetar a utilização do *smartphone* pelo usuário. Contudo, assumindo que o aplicativo só ira executar quando o usuário está escutando música e se movimentado, presume-se que o dispositivo estará no bolso de forma que o usuário não sentirá a performance geral do aparelho diminuir.

Em relação ao consumo de bateria, testou-se o aplicativo diversas vezes e os valores de consumo ficaram em entorno de 10% da bateria drenada em cerca de 1h. Normalmente o aparelho consome cerca de 2 à 3% apenas com a reprodução de áudio.

Pelas informações do *Android*, Figuras 24 e 25, pode-se concluir que o aplicativo gasta da mesma forma que a tela do aparelho, ou seja, 1 hora com o aplicativo executando gasta de mesma forma que 12 minutos de tela ativa (acesa). Esses valores são elevados, porém assim como a tela o tempo de execução contínua é baixo, não impactando de forma agressiva no modo como o usuário utiliza seu *smartphone*.

Figura 24: *Screenshot* dos dados fornecidos pelo aparelho



Fonte: O autor.

Figura 25: *Screenshot* dos dados fornecidos pelo aparelho

Fonte: O autor.

5.2.3 Classificador

Apesar dos resultados encontrados na Seção 5.1.2 apresentarem valores razoáveis, algumas informações foram descobertas com o aplicativo executando. Inicialmente não havia o Áudio 0 da Tabela 2. Este áudio de silêncio só foi adicionado depois que os testes com o aplicativo em tempo real apresentaram inconsistência no resultado.

Esperava-se que o classificador apresentasse algumas falhas devido à diversidade de áudios a que estamos expostos diariamente.

Mesmo o aplicativo ter sido desenvolvido visando a conversação do usuário, esta parte está comprometida visto que alguns tons de voz acabam confundindo o aplicativo. Entretanto, como no caso anterior, o fato de adicionar um áudio deste tipo ao conjunto de treinamento pode melhorar o classificador significativamente.

6 CONSIDERAÇÕES FINAIS

A utilização cada vez mais frequente de *smartphones* tem aumentado exponencialmente as possibilidade de comunicação e de acesso a cultura, especialmente aquela disponível em meios audiovisuais. Por outro lado, seu uso em diferentes situações cotidianas, pode expor seus usuários a riscos de acidentes, especialmente quanto utilizado durante deslocamentos em vias públicas e com utilização de fones de ouvido.

O aplicativo desenvolvido durante este trabalho pode ser utilizado em situações reais e sua resposta frente ao sinais sonoros de alerta ao pedestres ocorre em tempo real, possibilitando que o usuário de *smartphone* tenha um tempo para uma ação defensiva. Assim, sua instalação e execução em *smartphones* pode contribuir para um trânsito mais seguro e com menores índices de lesão entre os pedestres.

Apesar de que o classificador K-NN apresentar resultados satisfatórios, a extrapolação para áudios similares aos utilizados no conjunto de treinamento não atingiu os valores esperados. Contudo, notou-se que a adição de uma maior diversidade de áudios contribui para a melhora do classificador de forma que os resultados encontrados são melhores quando a diversidade é maior, mesmo que estes possuam tempo de duração mais curto.

A implementação da solução encontrada na plataforma *Android* provou ser desafiadora e essencial para o aprimoramento do trabalho. A execução em tempo real contribuiu para que fosse possível evidenciar falhas no classificador que não haviam sido previamente previstas.

O conjunto de áudios utilizados poderia conter uma maior diversidade. Entretanto, optou-se por utilizar gravações padronizadas afim de proporcionar um melhor conjunto de treinamento. Notou-se em testes com gravadores para *smartphone* que ajustes realizados por *software* comprometem o desempenho do classificador.

6.1 TRABALHOS FUTUROS

Está seção apresenta alguns tópicos que podem ser explorados em trabalhos futuros:

1. Em relação a solução encontrada:

    - Melhorar o conjunto de treinamento, adicionando mais áudios com características específicas do problemas mencionados,

tais como falas;

- Verificar quais são os parâmetros mais relevantes para a classificação;
- Testar outros classificadores e comparar sua viabilidade (tempo e eficiência) em relação ao K-NN;
- Reduzir o tempo necessário de áudio para que o classificador o classifique corretamente;
- Estudar qual o tempo ideal dos áudios para o treinamento, visando um treinamento mais diverso e balanceado.

2. Em relação ao aplicativo:

   - Adicionar a verificação do sensor *Step Detector* e implementar a sua utilização, quando presente, afim de melhorar a detecção de movimento do dispositivo;
   - Analisar a viabilidade de implementar a classe que processa o áudio utilizando-se do NDK;
   - Analisar com maior precisão o consumo de recursos e verificar possíveis pontos de melhoria.

REFERÊNCIAS

AMARAL, B. do. **Brasileiro usa celular por mais de três horas por dia**. 2016. Disponível em: <http://exame.abril.com.br-/tecnologia/brasileiro-usa-celular-por-mais-de-tres-horas-por-dia/>.

ANATEL. **Relatório de acompanhamento do setor de telecomunicações: Serviço Móvel Pessoal (SMP)**. [S.l.], 2016. 46 p. Disponível em: <http://www.anatel.gov.br/dados/relatorios-de-acompanhamento/2016>.

BRESOLIN, A. D. A. **Estudo do Reconhecimento de Voz para o Acionamento de Equipamentos Elétricos via Comandos em Português**. 119 p. Tese (Mestrado) — Universidade do Estado de Santa Cataria, 2003.

BUNGE, M. **Treatise on basic philosophy**. Boston: [s.n.], 1985. V. 7 p.

CERWALL, P. et al. **Ericsson Mobility Report 2016**. [S.l.], 2016. 32 p. Disponível em: <https://www.ericsson.com/res/docs/2016-/ericsson-mobility-report-2016.pdf>.

CONTRAN. **Resolução Nº 35**. may 1998. 4 p. Disponível em: <http://www.denatran.gov.br/download/Resolucoes-/resolucao035_98.>

DENATRAN. **Portaria Nº 12**. feb 2002. 3 p. Disponível em: <http://www.denatran.gov.br/download/Portarias/2002-/PORTARIA12-02.rtf>.

DENATRAN. **Frota por UF e Tipo de Veículo**. [S.l.], jun 2016. 1 p. Disponível em: <http://www.denatran.gov.br/images-/Estatistica/RENAVAM/2016/Julho/Frota_por_UF_e_Tipo>.

DENG, Z. et al. Efficient kNN classification algorithm for big data. **Neurocomputing**, v. 195, p. 143–148, 2016. ISSN 18728286. Disponível em: <http://www.sciencedirect.com/science/article/pii-/S0925231216001132>.

DIYKH, M.; LI, Y.; WEN, P. EEG Sleep Stages Classification Based on Time Domain Features and Structural Graph Similarity. **IEEE Transactions on Neural Systems and Rehabilitation**

Engineering, v. 24, n. 11, p. 1159–1168, 2016. Disponível em: <http://ieeexplore.ieee.org/document/7452628/>.

ERICSSON. North America Mobility Report Appendix. n. June, p. 8, jun 2015. Disponível em: <http://www.ericsson.com/res/docs/2015-/ericsson-mobility-report-june-2015-rnam-appendices.pdf>.

FERRIS, M. H. et al. Using ROC curves and AUC to evaluate performance of no-reference image fusion metrics. **Aerospace and Electronics Conference (NAECON)**, 2015. ISSN 23792027. Disponível em: <http://ieeexplore.ieee.org/document/7443034/>.

FISCH, D.; KALKOWSKI, E.; SICK, B. Knowledge Fusion for Probabilistic Generative Classifiers with Data Mining Applications. v. 26, n. 3, p. 652–666, 2014. Disponível em: <http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6420835>.

FORNI, A. A.; MEULEN, R. van der. **Gartner Says Five of Top 10 Worldwide Mobile Phone Vendors Increased Sales in Second Quarter of 2016**. 2016. Disponível em: <https://www.gartner.com/newsroom/id/3415117>.

Freitas Junior, V. et al. A pesquisa científica e tecnológica. **Espacios**, v. 35, n. 9, p. 12, 2014. Disponível em: <http://www.revistaespacios.com/a14v35n09/14350913.html>.

GALAR, M. et al. An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes. **Pattern Recognition**, v. 44, n. 8, p. 1761–1776, aug 2011. ISSN 00313203. Disponível em: <http://www-.sciencedirect.com/science/article/pii/S0031320311000458>.

GEORGIOU, G. M.; VOIGT, K. Stochastic computation of moments, mean, variance, skewness and kurtosis. **Electronics Letters**, v. 51, n. 9, p. 673–674, 2015. Disponível em: <http://ieeexplore.ieee.org/document/7095681/>.

GLOWACZ, A.; GLOWACZ, Z. Recognition of images of finger skin with application of histogram, image filtration and K-NN classifier. **Biocybernetics and Biomedical Engineering**, v. 36, n. 1, p. 95–101, 2016. Disponível em: <http://www.sciencedirect.com-/science/article/pii/S0208521615300073>.

GOOGLE. **Android, the world's most popular mobile platform**. 2017. Disponível em: <https://developer.android.com/about/android.html>.

GOOGLE. **BroadcastReceiver**. 2017. Disponível em: <https://developer.android.com/reference/android/content-/BroadcastReceiver.html>.

GOOGLE. **Dashboards**. 2017. Disponível em: <https://developer.android.com/about/dashboards/index.html>.

GOOGLE. **Handling Changes in Audio Output**. 2017. Disponível em: <https://developer.android.com/guide/topics/media-apps-/volume-and-earphones.html>.

GOOGLE. **Intents e filtros de intents**. 2017. Disponível em: <https://developer.android.com/guide/components/intents-filters-.html?hl=pt-br>.

GOOGLE. **Primeiros passos com o NDK**. 2017. Disponível em: <https://developer.android.com/ndk/guides/index.html>.

GOOGLE. **Sensors Overview**. 2017. Disponível em: <https:/-/developer.android.com/guide/topics/sensors/sensors_overview.h>.

GOOGLE. **Serviços**. 2017. Disponível em: <https://developer-.android.com/guide/components/services.html?hl=pt-br>.

GUTIÉRREZ, P. D. et al. GPU-SME-kNN: Scalable and memory efficient kNN and lazy learning using GPUs. **Information Sciences**, Elsevier Inc., v. 373, p. 165–182, 2016. ISSN 00200255. Disponível em: <http://linkinghub.elsevier.com/retrieve/pii/S0020025516306739>.

HAYKIN, S. **Neural Networks and Learning Machines**. [S.l.: s.n.], 1999. 906 p. ISSN 14337851. ISBN 9780131471399.

HAYKIN, S. **Redes Neurais**. 2. ed. Porto Alegre: Bookman, 2001. ISBN 8573077182.

IBGE. **Pesquisa Suplementar**. [S.l.], 2014. Disponível em: <goo.gl/YXq3hL http://goo.gl/YXq3hL>.

IBGE. **Estimativas da População Residente no Brasil e Unidades da Federação**. [S.l.], sep 2016. Disponível em: <https://goo.gl/YP3aZj>.

JIAN, H. E.; CHEN, H. A Portable Fall Detection and Alerting System Based on k-NN Algorithm and Remote Medicine. **China Communications**, v. 12, n. 4, p. 23–31, 2015. ISSN 16735447. Disponível em: <http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7114066>.

KRAMER, O. **Dimensionality Reduction with Unsupervised Nearest Neighbors**. [S.l.: s.n.], 2013. ISBN 9783642386510.

LECHETA, R. R. **Google Android: Aprendar a criar aplicações para dispositivos móveis com Android SDK**. 3. ed. São Paulo: Novatec, 2013. 824 p. ISBN 9788575223444.

LICHENSTEIN, R. et al. Headphone use and pedestrian injury and death in the United States. **Injury Prevention**, v. 18, n. 5, p. 287–290, jan 2012. Disponível em: <http://injuryprevention.bmj.com-/content/early/2012/01/03/injuryprev-2011-040161.short>.

LIU, H.; WU, X.; ZHANG, S. Neighbor selection for multilabel classification. **Neurocomputing**, v. 182, p. 187–196, 2016. ISSN 18728286. Disponível em: <http://www.sciencedirect.com/science-/article/pii/S0925231215019724>.

LOURIDAS, P.; EBERT, C. Machine Learning. p. 110–115, 2016. ISSN 87567016. Disponível em: <http://ieeexplore.ieee.org/document/7548905/>.

MEIRELLES, F. S. 27ª Pesquisa Anual do Uso de TI. may 2016. Disponível em: <http://eaesp.fgvsp.br/sites/eaesp.fgvsp.br/files-/pesti2016gvciappt.pdf>.

NAKONO. **Number Of Connected Things**. 2016. Disponível em: <https://fusion.nakono.com/data/number-of-connected-things-personal-devices-smartphones-worldwide-annual>.

NGUYEN, T. T. T.; ARMITAGE, G. A Survey of Techniques for Internet Traffic Classification using Machine Learning. **IEEE Communications Surveys and Tutorials**, v. 10, n. 4, p. 56–76, 2008. Disponível em: <http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=4738466>.

OPPENHEIM, A. V. **Signals \& Systems**. [S.l.: s.n.], 1997. 957 p. ISSN 1098-6596. ISBN 9788578110796.

OZAY, M. et al. Machine Learning Methods for Attack Detection in the Smart Grid. **IEEE Transactions on Neural Networks and Learning Systems**, v. 27, n. 8, p. 1773–1786, 2016. Disponível em: <http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7063894>.

RIBEIRO, L. et al. Analysis of the relationship between EEG signal and aging through Linear Discriminant Analysis (LDA). **Revista Brasileira de Engenharia Biomédica**, v. 28, n. 2, p. 155–168, 2012. ISSN 15173151. Disponível em: <http://www.scielo.br/pdf/rbeb/v28n2/a06v28n2.pdf>.

ROGERS, S.; GIROLAMI, M. **A First Course in Machine Learning**. [S.l.]: Chapman and Hall/CRC, 2011. 306 p.

SCOPATZ, R. A. et al. Effect of Electronic Device Use on Pedestrian Safety: A Literature Review. p. 67p, 2016. ISSN 01960644. Disponível em: <https://trid.trb.org/view/1406500>.

SERRANO, A. J. S. et al. Feature selection using ROC curves on classification problems. **Neural Networks (IJCNN)**, p. 0–5, 2010. Disponível em: <http://ieeexplore.ieee.org/document/5596692/>.

SHIRAISHI, Y.; FUKUMIZU, K. Statistical approaches to combining binary classifiers for multi-class classification. **Neurocomputing**, v. 74, n. 5, p. 680–688, 2011. ISSN 09252312. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0925231210003735>.

SILVEIRA, D. T.; CÓRDOVA, F. P. A pesquisa científica. In: GERHARDT, T. E.; SILVEIRA, D. T. (Ed.). **Métodos de pesquisa**. 1. ed. Porto Alegre: [s.n.], 2009. cap. 2, p. 31–42. ISBN 9788538600718.

SURESH, S.; SUNDARARAJAN, N.; SAVITHA, R. **Supervised Learning with Complex-valued Neural Networks**. 1. ed. [S.l.]: Springer-Verlag Berlin Heidelberg, 2013. (Studies in Computational Intelligence 421). ISBN 978-3-642-29490-7,978-3-642-29491-4.

SWENSSON, J. R. P.; SWENSSON, R. P.; SWENSSON, R. C. IPOD®, MP3 Players e a audição. **Rev. Fac. Ciênc. Méd. Sorocaba**, v. 11, n. 2, p. 4–5, 2009. Disponível em: <http://revistas.pucsp.br/index.php/RFCMS/article/view/1952/1208>.

WANG, C. et al. Equation-Based InSAR Data Quadtree Downsampling for Earthquake Slip Distribution Inversion. **IEEE Geoscience and Remote Sensing Letters**, v. 11, n. 12, p.

2060–2064, 2014. Disponível em:
<http://ieeexplore.ieee.org/document/6815638/>.

YU, Z. et al. Hybrid k-Nearest Neighbor Classifier. **IEEE Transactions on Cybernetics**, v. 46, n. 6, p. 1263–1275, 2016. Disponível em:
<http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7137658>.

ANEXO A – Áudio e sua Descrição

Tabela 2: Tabela de com o número do áudio e sua breve descrição. Parte 1/2.

Áudio	**Descrição**
0	Som de ambiente externo. Silêncio
1	Som de ambiente externo. Estrada em uma cidade pequena, com tráfego e pedestres.
2	Som de ambiente externo. Estrada principal de um país.
3	Som de ambiente externo. Tráfego de cidade.
4	Som de ambiente (externo). Estação de trem com aviso de chegada do trem.
5	Som de ambiente (externo). Atmosfera externa do aeroporto, na área de observação.
6	Som de ambiente (transporte). Caminhão 10t, aproximação, parada e desligado.
7	Som de ambiente (transporte). Moto, aproximação e parada.
8	Som de ambiente (transporte). Carro, aproximação e parada.
9	Som de perigo (buzina). Interior do carro, 4 buzinas consecutivas com pequeno intervalo entre elas.
10	Som de perigo (buzina). Interior do carro, 6 buzinas consecutivas com pequeno intervalo entre elas.
11	Som de perigo (sirene). Sirene de polícia tipo 1, observada de fora.
12	Som de perigo (sirene). Sirene de polícia tipo 2, observada de fora.
13	Som de perigo (sirene). Sirene de polícia tipo 1, observada do interior do carro.
14	Som de perigo (sirene). Sirene de polícia tipo 2, observada do interior do carro.
15	Som de perigo (sirene). Sirene de ambulância tipo 1, observada de fora.
16	Som de perigo (buzina). Carro desligado, buzinas em sequências, sem intervalos entre elas.
17	Som misto (ambiente e buzina). Moto ligada com 3 buzinas com intervalos entre elas.

Tabela 3: Tabela de com o número do áudio e sua breve descrição. Parte 2/2.

Áudio	**Descrição**
18	Som ambiente (externo). Motor de carro ligado e uma fala.
19	Som misto (ambiente e buzina). 2 buzinas de ar de caminhão em sequência e depois silêncio.
20	Som misto (ambiente e sirene). Carro de polícia com sirene ligada, aproximação e passagem.
21	Som misto (ambiente e buzina). Tráfego de cidades com algumas buzinas no seu decorrer.
22	Som misto (ambiente e sirene). Sirene de ambulância tipo 2, aproximação e passagem.
23	Som misto (ambiente e buzina). Ambiente de cidades com algumas buzinas.
24	Som misto (ambiente e buzina). Carro om motor em funcionamento e uma buzina.
25	Som misto (ambiente e buzina). Moto com motor em funcionamento e uma buzina.
26	Som de perigo (buzina). Diversas buzinas em sequência, mas sem sobreposição.

Tabela 4: Tabela de com o número do áudio da buzina, o carro que proveu a amostra e a distância entre o *smartphone* e o carro. Parte 1/2.

Buzina	**Carro**	**Distância (m)**
2	Corsa	5
5	Clio	5
6	Clio	5
7	Clio	5
10	Fiesta	5
32	Corsa	10
36	Clio	10
39	Fiesta	10
42	Parati	10
43	Parati	10
50	Siena	10
51	Siena	10
52	Siena	10
62	Corsa	15
66	Clio	15
68	Fiesta	15
70	Fiesta	15
73	Parati	15
79	Fox	15
80	Fox	15
81	Siena	15
82	Siena	15
83	Siena	15
84	Ká B	15
85	Ká B	15
95	Clio	20
96	Clio	20
97	Fiesta	20
99	Fiesta	20
102	Parati	20
103	Parati	20
105	Ká	20
106	Ká	20
107	Fox	20
108	Fox	20
109	Fox	20

Tabela 5: Tabela de com o número do áudio da buzina, o carro que proveu a amostra e a distância entre o *smartphone* e o carro. Parte 2/2.

Buzina	**Carro**	**Distância (m)**
114	Corsa	25
117	Clio	25
120	Fiesta	25
121	Fiesta	25
126	Ká	25
127	Ká	25
129	Fox	25
130	Fox	25
137	Corsa	30
138	Corsa	30
140	Clio	30
141	Clio	30
142	Fiesta	30
143	Fiesta	30
144	Fiesta	30
146	Parati	30
152	Fox	30
153	Fox	30
155	Siena	30
156	Siena	30
159	Ká B	30