ESCOLA POLITÉCNICA DA UNIVERSIDADE DE SÃO PAULO

THIAGO KNOPF SILVA

DESENVOLVIMENTO DE UM ASSISTENTE DE COMPRAS MÓVEL UTILIZANDO TECNOLOGIA DE RECONHECIMENTO DE VOZ E DE PADRÕES DE COMPRAS

SÃO PAULO

2018

THIAGO KNOPF SILVA

DESENVOLVIMENTO DE UM ASSISTENTE DE COMPRAS MÓVEL UTILIZANDO TECNOLOGIA DE RECONHECIMENTO DE VOZ E DE PADRÕES DE COMPRAS

Projeto de conclusão de curso apresentado ao Departamento de Engenharia Mecatrônica da Escola Politécnica da Universidade de São Paulo para a obtenção do título de Engenheiro

Orientador: Professor Dr. Marcos Ribeiro Pereira Barreto

SÃO PAULO

2018

Dedico esse trabalho à minha família e amigos por toda a ajuda e paciência na incrível jornada que foi o desenvolvimento dessa ideia.

RESUMO

O processo de se realizar compras tem se revolucionado nas últimas décadas, sofrendo alterações significativas desde a popularização da internet, nos anos 90 e, mais recentemente com os avanços na computação móvel. Apesar disso, certos serviços ainda estão se adaptando à essa realidade e processos repetitivos e desgastantes, como realizar as compras no mercado ainda não possuem uma solução considerada satisfatória pelo mercado. A fim de atingir esse problema, e melhorar a experiência de ir ao mercado, principalmente para aqueles que possuem alguma restrição de mobilidade, foi desenvolvido um aplicativo auxilia essa tarefa. É proposto um *software* que seja capaz de entender quais são os produtos a serem adquiridos e disponibilizar o orçamento, através de uma análise gramatical e de um reconhecimento de entidades nomeadas, para a extração de informações relevantes à aplicação. Como avaliação propõe-se uma análise qualitativa de aderência do reconhecimento dos produtos de uma lista padrão.

Palavras-chave: aplicativo, voz, processamento de linguagem natural, banco de dados, android, rede neural.

ABSTRACT

The process of buying shares has revolutionized in the last decades, suffering the change of graphics since the popularization of the Internet, in the 90s and, more recently with the advances in the mobile communication. However, services are still adapting to the actual repetitive and exhausting practices and processes, such as making purchases in the marketplace and still not having a satisfactory solution produced by the market. In order to achieve this problem, and improve the market experience, especially those who exhibit some mobility constraint, was developed an auxiliary application this task. It is software that may be able to understand what data to acquire and make available for budget, through a grammar analysis and recognition of entities for an extraction of information relevant to the application. How the evaluation presents a qualitative analysis of product book adherence from a standard list.

Keywords: application, voice, natural language processing, database, android.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1. Introdução

1.1 Justificativa

Desde 2008, devido aos avanços tecnológicos ocorridos no campo da telefonia, observou-se uma revolução na forma como produtos e serviços são oferecidos ao público. O surgimento do *smartphone*, com a capacidade de rodar diversos tipos de programas, assim como um computador, só que portátil, fez com que diversas aplicações fossem desenvolvidas, a fim de suprir as necessidades das pessoas por determinados produtos antes inexistentes.

O contínuo avanço nessa área levou à criação de novas ferramentas e tecnologias embarcadas, tais como a integração do GPS ao celular, a comunicação com a web via rede de telefonia de alta velocidade (3G e 4G) e a capacidade de transformar texto para voz e voz para texto.

Aliando essa capacidade de prover serviços diversos de forma simples com a alta tecnologia embarcada, pode-se perceber a grande capacidade dos *smartphones* de ajudar grupos diversos da sociedade. Ao levar isso em consideração e somar o fato de que a computação móvel é a que tem mais aceitação pelo usuário pouco familiarizado com tecnologias [20], devido à sua gradativa inserção mercadológica, torna-se natural a escolha dessa plataforma como base de desenvolvimento para um produto que se espera fornecer para o público em geral.

O problema que se busca abordar é o de prover uma alternativa simples à realização de compras nos supermercados. A rotina de ir aos mercados locais várias vezes por mês para se realizar a compra de itens recorrentes e pesados, tais como frutas, verduras, bebidas e congelados pode não ser tão trivial para uma parcela da população que possua dificuldades de locomoção, tais como idosos, deficientes físicos ou pessoas que não possuem um veículo próprio. Logo, pretende-se desenvolver uma aplicação para celular que seja capaz de reconhecer uma lista de compras, captada pelo microfone do celular, e que ao comparar com um banco de dados, seja capaz de informar o cliente dos produtos disponíveis e do custo total da compra. Ao confirmar a compra, o mercado (ou o provedor do serviço) recebe o pedido, realiza a compra e o entrega no endereço do cliente.

Optou-se por essa estrutura devido ao público alvo, que pode não se sentir à vontade para realizar várias escolhas (produtos, marcas, tamanho e quantidades) para os diversos produtos de uma lista de compras de uma família. Em um sistema de compras tradicional, em que o usuário seleciona o produto a partir de uma busca por digitação, define todos os parâmetros e confirma a sua adição ao carrinho de compras, o número de cliques se torna um problema, pois pode levar o usuário à fadiga.

1.2 Objetivos

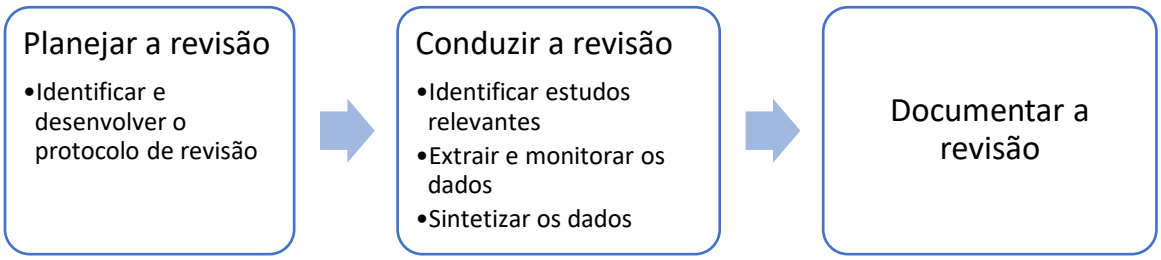
Ao fim desse estudo, pretende-se ter um produto validado pelo mercado, com uma boa capacidade de entendimento do padrão de compras do usuário, composto, ao menos, por três telas: a tela inicial deve ser capaz de mostrar os mercados cadastrados a partir de uma base de dados, a segunda tela terá o sistema de reconhecimento de voz, onde o usuário ira ditar os produtos da lista e a terceira tela será a compilação dessa lista. Essa última tela deverá mostrar cada um dos 5 atributos de um produto (tipo, marca, tamanho, modelo e quantidade), e deverá ser capaz de enviar essas informações para um servidor.

2. Revisão Bibliográfica

2.1 Metodologia

Para se revisar o estado da arte, é necessário identificar, avaliar e interpretar pesquisas importantes na área de desenvolvimento de aplicativos móveis, principalmente naqueles focados em compras do varejo e nas tecnologias envolvidas. No caso dos aplicativos disponíveis somente foram considerados aqueles que possuem versão para Android que, por se tratar de um *software* livre, facilita o seu acesso. Seguindo a metodologia proposta por Barbara Kitchenham em 2004 [1], para se realizar uma pesquisa de forma eficiente, deve-se proceder como descrito abaixo:

Figura 1- Metodologia de pesquisa proposta por Kitchenham



Segue abaixo a revisão considerada relevante para o desenvolvimento desse trabalho.

2.2 Sistemas de Compras Móveis

Um aplicativo móvel, também conhecido como app, é um *software* desenvolvido para ser instalado em um dispositivo eletrônico móvel tal como um celular do tipo *smartphone* ou um *tablet.* Em forte crescimento, o número de *downloads* de aplicativos chegou a 197 bilhões em 2017; um crescimento de 31,9% em relação ao ano anterior [2]. Criado em 2008, os aplicativos encontram-se disponíveis para *download* em plataformas de distribuição, tais como a App Store, da Apple e a Google Play, para celulares com sistemas operacionais do tipo iOS e Android, respectivamente.

Em 2017, o um dos aplicativos mais utilizado nos EUA foi o da Amazon, sendo o primeiro entre a geração dos *millenials,* com 35% de presença nos dispositivos móveis registrados [3], o que mostra uma tendência de aumento de demanda nesse setor. Com uma estimativa de crescimento de 9% ao ano [4], 2016 registrou mais de 2 bilhões de usuários de smartphones no mundo e, mesmo com 91% dos aplicativos instalados sendo gratuitos, estima-se que esse mercado deve superar a marca dos USD 80 bilhões de receita nos próximos 3 anos [5].

Figura 2- Distribuição de uso de aplicativos entre pessoas nascidas após janeiro de 1994. Fonte: [3].

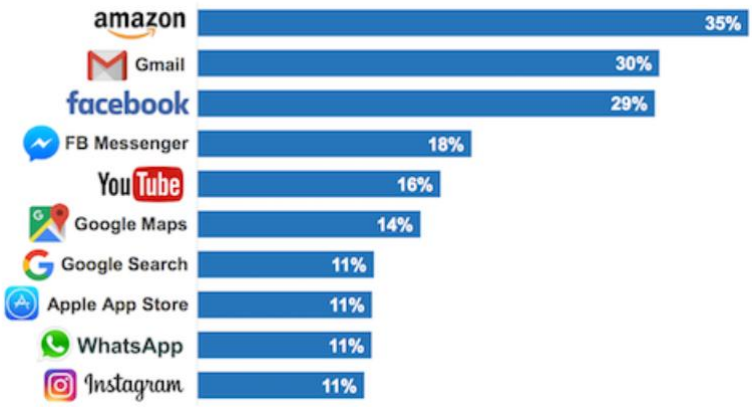
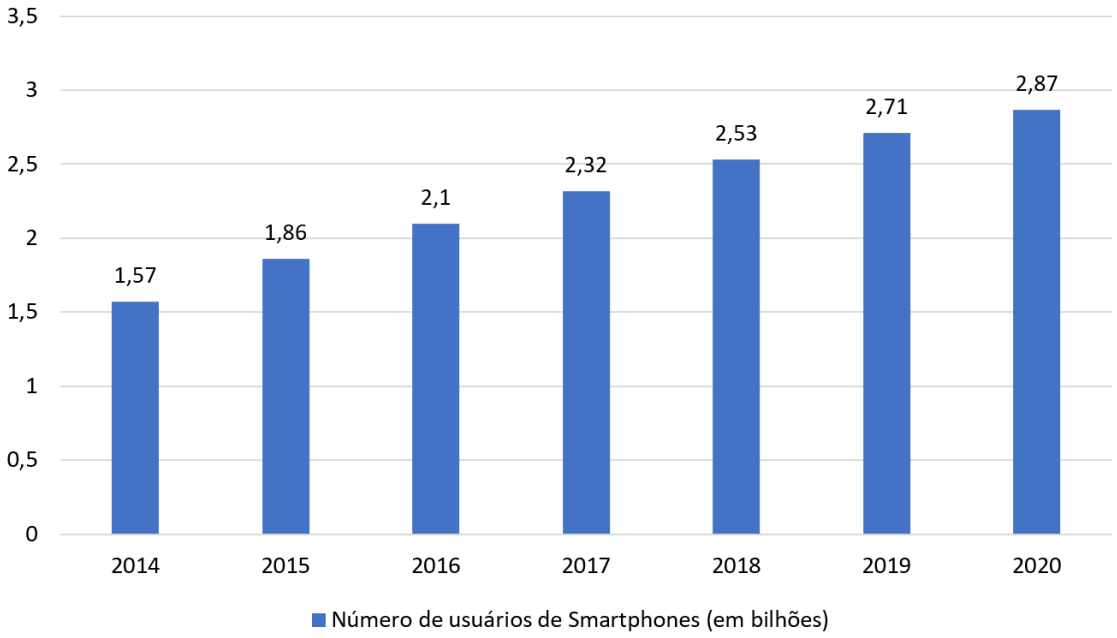


Figura 3 - Estimativa de crescimento no número de usuários de smartphones no mundo. Dados a partir de 2017 são estimados. Fonte: [4].



2.2.1 Amazon

Fundada em 1994, a Amazon é a maior rede de varejo virtual do mundo em termos de receita e a segunda maior em número total de vendas, atrás apenas do Alibaba.com. Em outubro de 2016, a Amazon tornou pública a sua intenção de construir lojas de conveniência que usam sensores para realizar a cobrança dos produtos adquiridos pelo cliente, eliminando a necessidade de ter que passar pelo caixa [7]. Chamadas de Amazon GO, a primeira loja foi

inaugurada exclusivamente para funcionários da empresa em dezembro de 2016 [6].

Figura 4 - Aplicativo de compras da Amazon. Fonte: amazon.com.

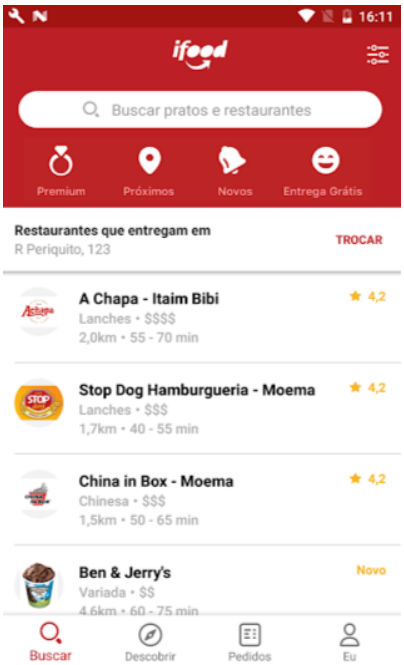
Com uma estrutura simples, a versão móvel do sistema da Amazon não difere muito dos seus pares, e pode ser usado como referência para um aplicativo de compras padrão. Com uma tela rápida de cadastro, exigindo somente nome e e-mail, o usuário já é direcionado à página principal de compras, que possui uma barra de pesquisa e um campo para inserir o endereço de entrega. Na página de configurações, algumas poucas opções são divididas em 3 grupos: tipos de lojas e promoções, acompanhamento de pedidos e informações sobre a conta do usuário, e configurações.

Buscando facilitar a experiência do usuário com o aplicativo, a Amazon utiliza uma tecnologia de reconhecimento de produtos pela câmera do celular, capaz de identificar o produto pelo código de barras. Essa função direciona o usuário para a tela de compras do produto identificado, evitando, assim que o usuário tenha que digitar o tipo do produto ou que tenha que buscá-lo a partir de uma lista com várias possibilidades, da maneira convencional.

2.2.2 iFood

O iFood é uma plataforma virtual que reúne diversos restaurantes que realizam entrega à domicílio de alimentos preparados para consumo. Essa empresa brasileira, criada em 2011, atingiu o valor de R$ 1 bilhão em 2014, após a sua fusão com a plataforma RestauranteWeb. Com mais de 10 milhões de *downloads* [8], esse aplicativo procura eliminar a necessidade de o usuário ter de guardar vários cardápios de diversos restaurantes [9].

Figura 5 - Aplicativo de compras de alimentos do iFood. Fonte: [9].



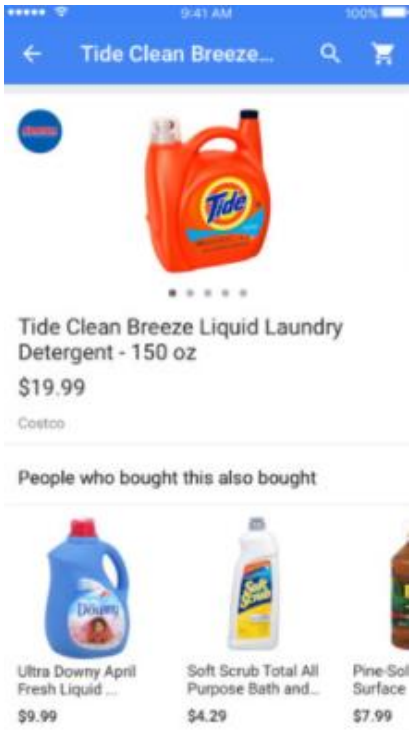
Diferentemente da Amazon, o iFood não realiza a venda do produto diretamente, mas age como um intermediário, o que implica em não trabalhar com estoque. Assim, cabe ao aplicativo centralizar os mais diversos tipos de restaurantes, categorizando-os, e permitindo que o usuário escolha os restaurantes mais próximos da sua localização. Trabalhando com duas *toolbars*, esse aplicativo exige somente o cadastro do endereço, e disponibiliza todos os produtos de cada um dos restaurantes cadastrados. Além da vantagem de mostrar vários restaurantes próximos ao endereço de entrega, uma outra facilidade de que o aplicativo disponibiliza é o pagamento online, com cartão de crédito.

2.2.3 Google Express

Criado em 2013, esse serviço de compras do Google, disponível em algumas partes dos EUA, possui mais de 1 milhão de *downloads* e permite aos seus usuários realizarem compras em instituições parceiras cadastradas, tais como os hipermercados Wal-Mart e Costco, redes varejistas de roupas e diversos como a Target, de materiais de escritório como Staples e de outros.

Ainda em fase de testes, as condições do serviço podem variar bastante, mas a empresa divulga que, de forma geral, a taxa de entrega cobrada é de aproximadamente USD 4,00 e prazo médio de entrega é de 48 horas. Apesar dessas condições, o *feedback* dos seus usuários pode ser considerado mediano (nota 3,8/5), sendo que as principais reclamações incluem diferenças entre os preços dos produtos no aplicativo e na loja, e o não cumprimento do prazo de entrega, que em alguns casos podem chegar a até duas semanas [10].

Figura 6 - Aplicativo de compras do Google Express. Fonte: [10].



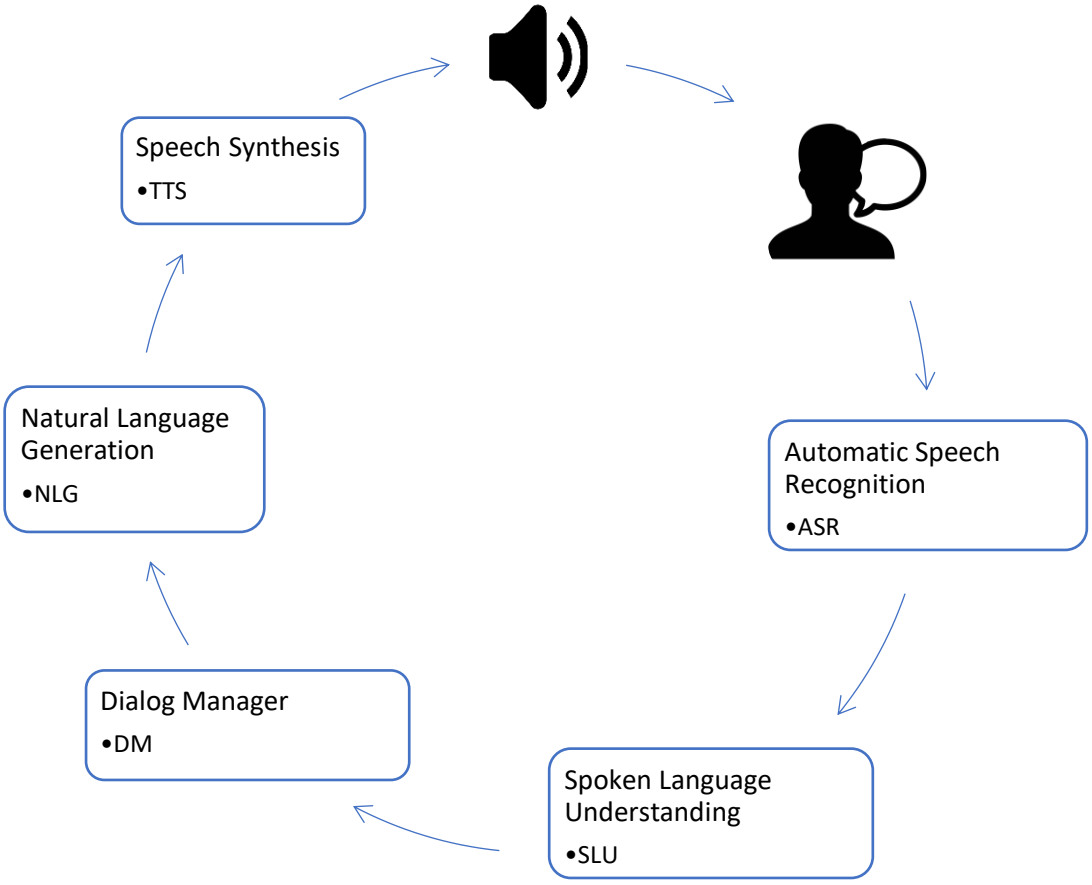
2.3 Sistemas de Diálogo

Um sistema de conversação tem por objetivo prover ao usuário, uma interface simples, que o permita acessar e gerenciar qualquer tipo de informação. Esse tipo de sistema recebe sinais sonoros do usuário e responde com uma ação ou com a informação requerida.

Inicialmente, os sistemas de diálogo eram restritos à alguns poucos setores, como redirecionadores de chamadas telefônicas e pesquisas meteorológicas, no início da década de 1990. Atualmente, suas aplicações se estendem de sistemas de navegação embarcados em veículos à robôs domésticos inteligentes [13].

De maneira geral, um sistema de diálogo segue uma arquitetura unidirecional composta por 5 módulos que irão realizar o tratamento do som, identificá-lo e responder de acordo com o que foi dito [12].

Figura 7 - Arquitetura de sistemas de diálogos tradicionais. Autoria própria.



- ASR: o módulo de reconhecimento de fala (speech-to-text) é responsável por transformar o sinal de audio recebido pelo microfone em texto
  .
- SLU: o semantizador recebe o texto do módulo anterior e o analisa, através de um processador de linguagem natural (NLP). O semantizador mapeia a fala pré-processada, na qual ele procura identificar o objetivo da frase e as entidades nomeadas, através de um pareamento semântico ou uma análise estatistica.

- DM: considerado o centro de todo o sistema de diálogo, esse gerenciador coordena as atividades de todos os componentes, controla o fluxo do diálogo e a comunicação com aplicações externas. Utilizando o significado da frase, detectado no módulo anterior, ele determina como o computador irá responder.

- NLG: o sistema cria respostas com base na linguagem natural, utilizando uma base de dados que poderá identificar uma resposta adequada para cada pedido do usuário.

- TTS: o sintetisador disponibiliza a resposta para o usuário, seja em forma de texto ou em áudio.

Além desses componentes do framewotk, outro ponto relevante para a estruturação de um sistema de diálogo é o grau de iniciativa estabelecido para o programa. Como um processo de diálogo pode ser visto como uma troca de informações na qual a iniciativa varia entre o usuário e o sistema, existem três possibilidades para quem direcionará o diálogo: o sistema, o usuário, ou os dois.

Num sistema de diálogo com a iniciativa fixada no *software*, o programa irá realizar algumas perguntas ao usuário para identificar as informações que preenchem cada um dos pontos pré-programados. Após serem devidamente preenchidos, o DM poderá analisar a questão e compará-la com o banco de

dados. Dessa maneira, o diálogo deve ser construido de tal forma que o usuário responda às questões com palavras ou frases simples, permitindo que as respostas do usuário possam ser previstas. Por outro lado, se o sistema precisar de várias respostas, a tarefa se torna muito complexa, exigindo mais tempo para ser executada e aumentando a chance de erro.

Já num sistema em que o direcionamento do diálogo é definido pelo usuário, o sitema passa a agir passivamente, realizando perguntas de confirmação caso alguns pontos não estejam claros. Em outras palavras, o usuário passa a realizar as perguntas e o sistema deve ser capaz de respondê-las. Apesar de fazer com que a conversação flua de uma forma mais natural, essa estrutura é mais complexa, pois exige que o banco de dados possua um vocabulário maior, levando em consideração diferentes combinações de palavras para um mesmo significado. Mesmo com esse cenário o DM ainda deve ser capaz de processar todas essas combinações de diálogo.

Por fim, num sistema com iniciativa mista, o programa é, pelo menos inicialmente, o agente responsável por controlar o dialogo. Porém o usuário possui uma flexibilidade de interação com *software*, podendo fornecer mais informações do que o necessário, e até modificar o objetivo da interação no meio da conversa. Essa arqitetura é a mais complexa das três pois exige que o sistema seja capaz de responder mesmo quando o usuário toma a iniciativa. Os sistemas comerciais mais modernos, como Siri, da Apple, e a Alexa, da Amazon, procuram utilizar esse sistema, pois é o que mais se assemelha à uma conversa natural [12].

Tabela 1 - Tipos de iniciativas de diálogos de sistemas de diálogos. Autoria própria.

Tipos de Iniciativas	Exemplos
Iniciativa do sistema	Sistema: Qual o nome da cidade que você irá visitar? Usuário: São Paulo. Sistema: Cidade São Paulo. Que tipo de restaurante você irá visitar? Usuário: Churrascaria
Iniciativa do usuário	Usário: Eu gostaria de visitar uma churrascaria em São Paulo. Sistema: Existem 25 churrascarias em São Paulo. Usuário: Onde fica o Fogo de Chão?
Iniciativa mista	Sistema: Aonde você está indo? Usuário: Numa churrascaria em São Paulo. Sistema: Existem 25 churrascarias em São Paulo. Qual delas você gostaria de visitar? Usuário: Quais são os mais baratos?

Cabe ao Dialog Manager encontrar a melhor reposta para uma dada situação, a fim de manter o discurso fluindo naturalmente. Existem três possíveis abordagens para o DM analisar o texto: baseada no conhecimento (*knowledge-based*), baseada nos dados (*data-driven*), e mista (*hybrid*).

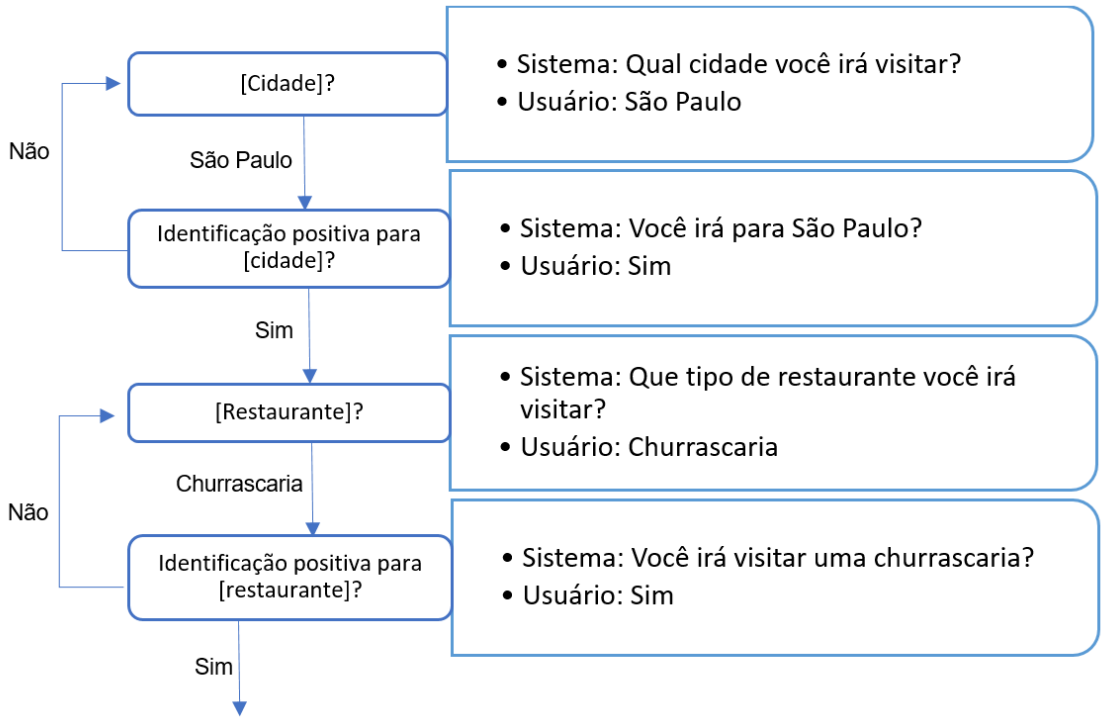
2.3.1 Knowledge-based

Esse tipo de sistema, geralmente restrito a tarefas altamente estruturadas e desenhadas para utilizar uma iniciativa do sistema, possui uma abordagem de estados finitos, onde uma linguagem definida como padrão é inserida manualmente no sistema. Esse tipo de estrutura é frequentemente utilizada em processos de prototipação rápida de sistema de diálogo com objetivos bem definidos.

Apesar de ser uma abordagem simples, programar cada uma das possíveis interações do usuário com o sistema é extremamente trabalhoso e torna o programa muito rígido. Por exemplo, caso o usuário forneça mais informações do que o necessário o sistema será incapaz de manter o fluxo da conversa, pois ele não foi desenhado para isso.

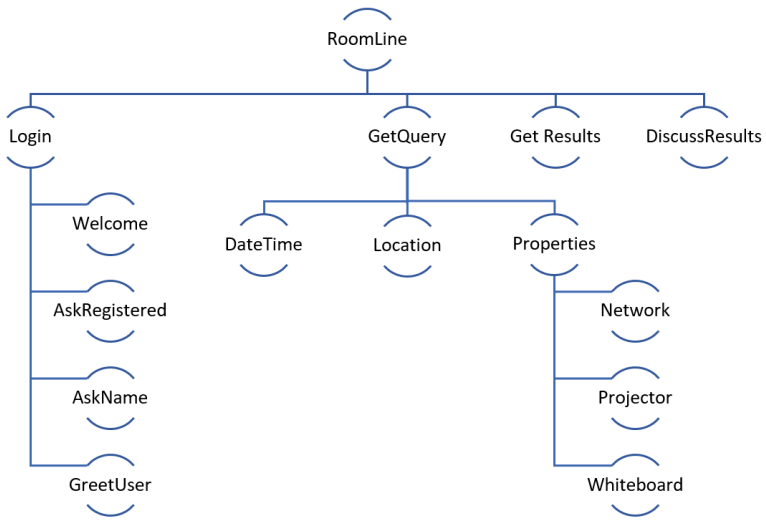
Para superar essas limitações, vários grupos de pesquisas exploraram formas de modelar o diálogo, segmentando tarefas grandes e complexas em tarefas menores. Um dos melhores exemplos de *framework's* de sistemas de diálogo que utiliza essa abordagem é o RavenClaw. Esse gerenciador de diálogo busca realizar uma clara separação entre os domínios da lógica de controle do diálogo [12].

Figura 8 - Exemplo de diálogo de um sistema *knowledge-based*. Autoria própria.



Uma parte do diálogo é identificada pela lista de ações específicas do *software*, (basicamente uma estrutura em forma de árvore de hierarquia de interações) [14]. Enquanto isso, a outra parte é controlada por um sistema de processamento que procura identificar as ações faltantes.

Figura 9 - Exemplo da lista de ações específicas de sistemas *RoomLine*, semelhantes ao RavenClaw. Autoria própria.



2.3.2 Data-driven

Uma forma mais recente de se estruturar o DM do que o *knowledge-based*, a capacidade do computador analisar um texto com base nos dados (*data-driven*) exige que uma grande variedade de combinações de estruturas verbais seja identificada pelo programa. Tal estrutura pode ser vantajosa no longo prazo pois o treinamento da máquina passa a ser automático, exigindo menos supervisão humana.

Essas vantagens motivaram o desenvolvimento de métodos estocásticos de modelagem de diálogo, usando *machine learning* baseados nos Processos de Decisão de Markov. Frameworks com esse tipo de estrutura aplicam uma análise estatística ao diálogo, possibilitando que o sistema o modele dinamicamente e permitindo-o alterar a estratégia do diálogo. Sistemas com essa estrutura, como o Watson, da IBM, utilizam algoritmos com sistemas de recompensa, que indicam qual a resposta que melhor se encaixa naquele diálogo. Outra vantagem desse sistema é que uma vez reconhecido o contexto do diálogo, o DM pode prever quais serão as próximas ações a serem tomadas, pois o sistema irá comparar a conversa atual com diálogos similares prévios [12].

2.3.3 Hybrid Approach

Sistemas baseados em dados exigem uma quantidade muito grande de exemplos de conversas para serem capazes de identificar qual a resposta ótima para cada tipo de diálogo. Para ajudar a resolver esse problema, os sistemas de simulação de diálogo podem ser incorporados para ajudar a criar cenários-base, com a supervisão humana, estimando quais seriam as melhores ações a serem tomadas pelo sistema. Somando-se a isso um conhecimento prévio de um sistema *knowledge-based*, pode-se obter uma estrutura muito mais robusta, capaz de se recuperar de erros e prever ações a serem tomadas dentro de cada cenário [12].

2.4 Sistemas Multiagente

Sistemas multiagente, também conhecidos como MAS (*multi-agent systems*), são sistemas computacionais em que vários agentes inteligentes e autônomos, sejam eles robôs ou humanos, interagem dentro das suas capacidades para atingir um objetivo mais rapidamente, quando comparado um sistema individual. Um agente deve ser capaz de receber informações, analisá-las e decidir se será necessário realizar alguma ação. As interações entre os agentes devem ocorrer dentro das restrições impostas pela organização e pelos meios que foram projetados dentro ambiente, permitindo que cada indivíduo atinja o seu objetivo [15].

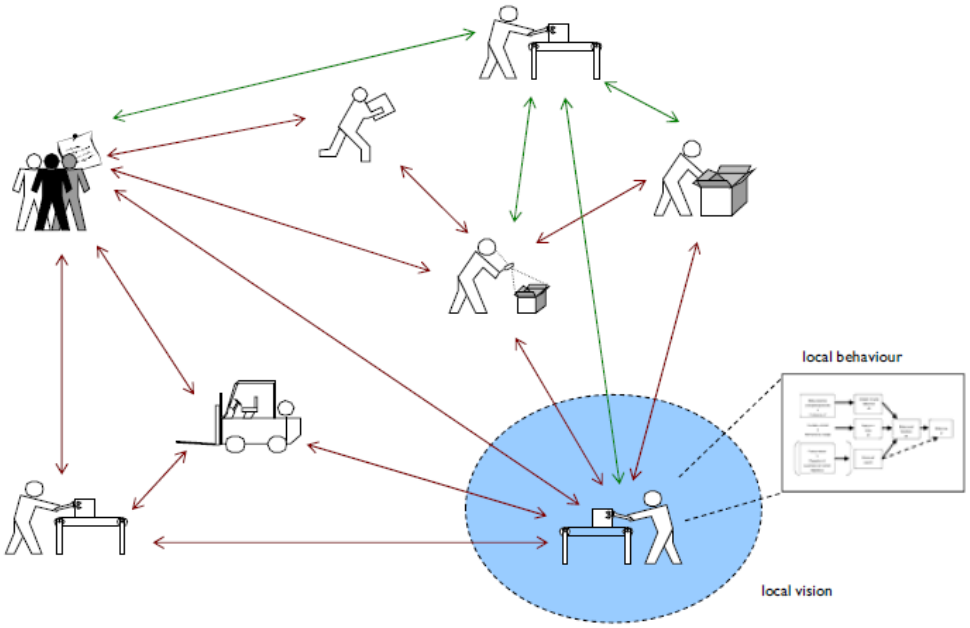
Atualmente, sistemas multiagentes possuem duas aplicações em que os seus benefícios se tornam mais visíveis: permitir que *softwares* sejam mais flexíveis e realizar modelagens. No caso da construção de sistemas, a flexibilidade implica na capacidade do programa de responder corretamente às situações dinâmicas, selecionando a melhor ação dentre todas as possibilidades, e ser capaz de receber novas funções de maneira prática, sem haver a necessidade de reestruturar o programa. Isso melhora a tolerância à erros e faz com que o sistema atinja o objetivo sem necessariamente ter de recorrer ao usuário [16].

Criar regras redundantes dentro dos sistemas é umas das principais formas de tornar o programa menos suscetível à erros, mas no caso do MAS, criar redundância significa criar apenas mais um agente. Além disso, caso um agente precise de um segundo agente para atingir seus objetivos e esse segundo agente falhar, o primeiro pode ativamente buscar novos caminhos para atingir seus objetivos [17].

Estruturas de sistemas de inteligência artificial, como o MAS estão se tornando cada vez mais importantes, tanto na tecnologia industrial como na solução de problemas complexos, como otimização de logística, reconhecimento de padrões de compras e reconhecimento de diálogo. Exemplos de aplicações do MAS variam desde de o controle do conforto térmico nos navios da marinha

americana [18] ao desenvolvimento de soluções para problemas específicos da rede de telefonia na Itália [19].

Figura 10 - Exemplo de um sistema multiagente. Fonte: [18].



2.5 Sistema Operacional Android

Todos os aplicativos para sistemas operacionais do tipo Android são desenvolvidos seguindo os mesmos padrões, disponibilizados pelo Google em seu site para suporte aos desenvolvedores [11], e utilizando o Android Studio, que é a plataforma de programação para aplicativos. Atualmente na versão 3.1, essa plataforma fornece ferramentas importantes que auxiliam no desenvolvimento do programa, tais como o *Instant Run*, que demonstra instantaneamente como alterações no código podem afetar o app, e o Emulador, que simula o funcionamento do aplicativo no celular.

Desenvolvidos na mesma plataforma e seguindo as mesmas diretrizes, todos os aplicativos citados possuem recursos importantes, como um menu lateral com possibilidade de personalização, filtro por geolocalização, sistema de busca de produtos, estrutura em telas laterais, banco de dados com imagens, descrição e preço dos produtos e sistemas de pagamentos online. A integração

à todas essas ferramentas está disponível nas versões mais recente do Android Studio [11].

Presente em diversos aparelhos móveis, o Android possui diversos atributos, tais como browser rápido, sincronização em nuvem, sistema multitarefa, facilidade para conexão e compartilhamento com outras plataformas, diversos aplicativos disponíveis *open-source*, fóruns de programação entre outros [21]. Esses fatores justificam a escolha dessa plataforma como base para o desenvolvimento de novas aplicações.

2.5.1 Banco de Dados

Um banco de dados nada mais é do que uma maneira estruturada de armazenar informações que se correlacionam, tais com dados pessoais, quantidades e preços de produtos. Existem diversos modelos de base de dados, tais como o modelo plano, o em rede, o hierárquico, o relacional, entre outros. O mais simples deles, e mais utilizado é o plano, que consiste numa matriz simples, bidimensional, muito utilizada em planilhas eletrônicas.

Uma tabela típica, possui colunas com diferentes tipos de dados, e em cada linha, uma gravação de dados. Numa programação orientada a objetos, em geral, cada tabela representa um objeto (designado por uma classe), e cada coluna representa um atributo dessa classe.

No caso do Android, ele possui um banco de dados relacional interno, com código aberto em sintaxe SQL. Esse tipo de banco de dados consiste num variação aprimorada do banco de dados plano, pois inclui a capacidade de realizar operações algébricas com os valores em cada célula e a capacidade de limitar o tipo de valor inserido nessa célula, podendo ser uma restrição de domínio (só aceita valores entre 1 e 100, por exemplo), de atributo (somente números inteiros), entre outras.

O Android não possui a capacidade de se comunicar o seu banco de dados interno com bases externas de maneira direta, por isso torna-se essencial entender a mecânica desse banco de dados. Devido à essa limitação, o SQLite

aceita somente variáveis do tipo *integer* (inteira), *real*, e texto, se fazendo necessária a conversão em um desses modelos para a sua inclusão no sistema.

Comunicações com outros bancos de dados podem ser realizadas mediante a configuração correta e a utilização das redes de internet. Além disso, para que as duas bases de dados se comuniquem propriamente é necessário converter quais quer informações para os tipos de variáveis aceitas pelo SQLite.

2.6 Estado da Arte

A fala, apesar de ser algo natural para os seres-humanos, é uma atividade muito complexa, sujeita a diversos fatores que podem alterar completamente o significado de uma frase, tais como o sarcasmo, sotaque, pronuncia, vernáculo, articulação, nasalidade, altura, volume, velocidade ou o estado emocional do interlocutor. Todos esses itens devem ser levados em consideração para a interpretação correta do significado de uma frase [20].

Devido a esses fatores, a interação entre as pessoas e os sistemas computacionais por meio do reconhecimento de voz, apesar de ser intuitivamente simples, se mostrou desafiadora na parte técnica. Sendo alvo de grandes pesquisas desde a década de 50, a ciência do reconhecimento de voz obteve seus maiores avanços a partir da década de 70, através da utilização de métodos diversos, tais como a correspondência de modelo, engenharia do conhecimento e modelagem estatística [21].

Esses avanços se intensificaram na última década, com a criação de uma plataforma de serviços cognitivos para negócios pela IBM em 2008, o Watson. Esse sistema se tornou tão avançado e famoso que em 2010 ele foi convidado para participar de uma competição de reconhecimento de palavras da televisão americana, *Jeopardy*, e logo se tornou campeão [23].

Em 2013, o Google lançou a sua plataforma para reconhecimento de voz, o Google Text-to-Speech, ou Google TTS, com extensão para várias linguagens e disponível em smartphones com sistema Android. Diferentemente de outras empresas, a inteligência artificial do sintetizador do TTS usa a abordagem *data-driven*, analisando as formas de onda inseridas no sistema e as comparando

com uma base de dados extensa à uma taxa de 24 mil amostras por segundo [22]. Essa estrutura torna o sistema mais robusto em comparação ao tradicional *knowledge-based*, que reconheceria as palavras individualmente, pois permite que o programa reconheça sotaques e expressões regionais, bem como dificulta a interpretação ambígua de uma frase. Essas vantagens tornam uma estrutura *data-driven* a escolha ideal para um sistema que irá relacionar os produtos de uma lista ditada com os itens de um banco de dados.

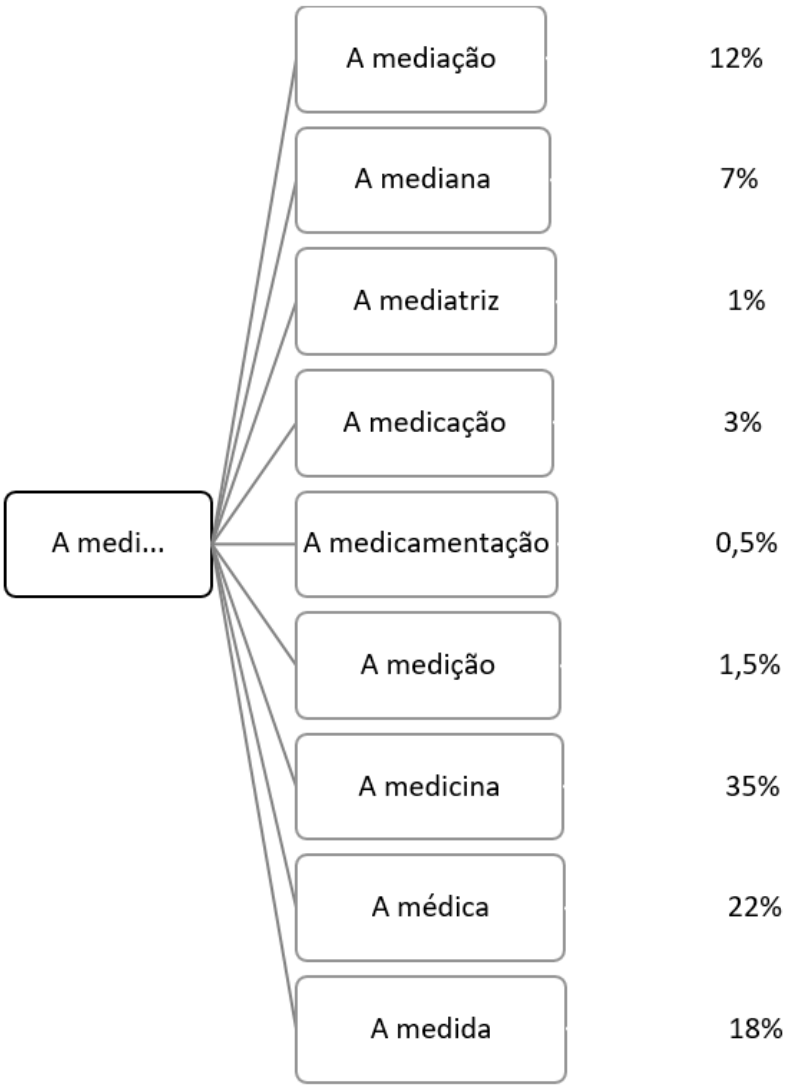
Como regra geral, pequenos conjuntos de palavras apresentam uma taxa de erros desprezível, porém conforme o tamanho do vocabulário aumenta, a chance de erro também aumenta. Estudos iniciais mostravam que alguns sistemas de reconhecimento de voz, podiam interpretar vocabulários de 100 mil palavras com taxas de erro próximas de 45% [23] [24] para sistemas com falas contínuas, que são mais difíceis de serem interpretados do que falas isoladas, já que o programa deve ser capaz de reconhecer o término de cada palavra.

Ainda assim, mesmo para um vocabulário pequeno e uma fala descontínua, o desempenho interpretativo do sistema também pode ser afetado pela sequência de palavras que são permitidas pelo reconhecimento. Uma lista de compras de supermercado gera uma restrição de tarefa (“manga” será sempre a fruta e não a parte de uma camisa), semântica (interpretando “comprar” ao invés de “com ar”) ou sintática (reconhecendo somente “3 caixas de suco” e desprezando “suco de 3 caixas”), por exemplo [25].

2.7 Fundamentos do Reconhecimento de Voz

De forma simplificada, pode-se dizer que o reconhecimento de voz se trata de um problema de identificação de padrões em diversos níveis, estruturados numa hierarquia que se inicia nas sílabas e pode se estender a sentenças, ou até a diálogos inteiros. Quanto mais alto o nível, mais restrições podem ser inseridas, diminuindo o tempo de processamento e a quantidade de erros, como demonstrado na a figura abaixo. Essa hierarquização permite que se combine uma tomada de decisão probabilística nos níveis inferiores com uma decisão discreta nos níveis mais elevados [27].

Figura 11 - Exemplo de uma árvore probabilística da tomada de decisão com restrições semânticas para a frase “A medi...”. Autoria própria.

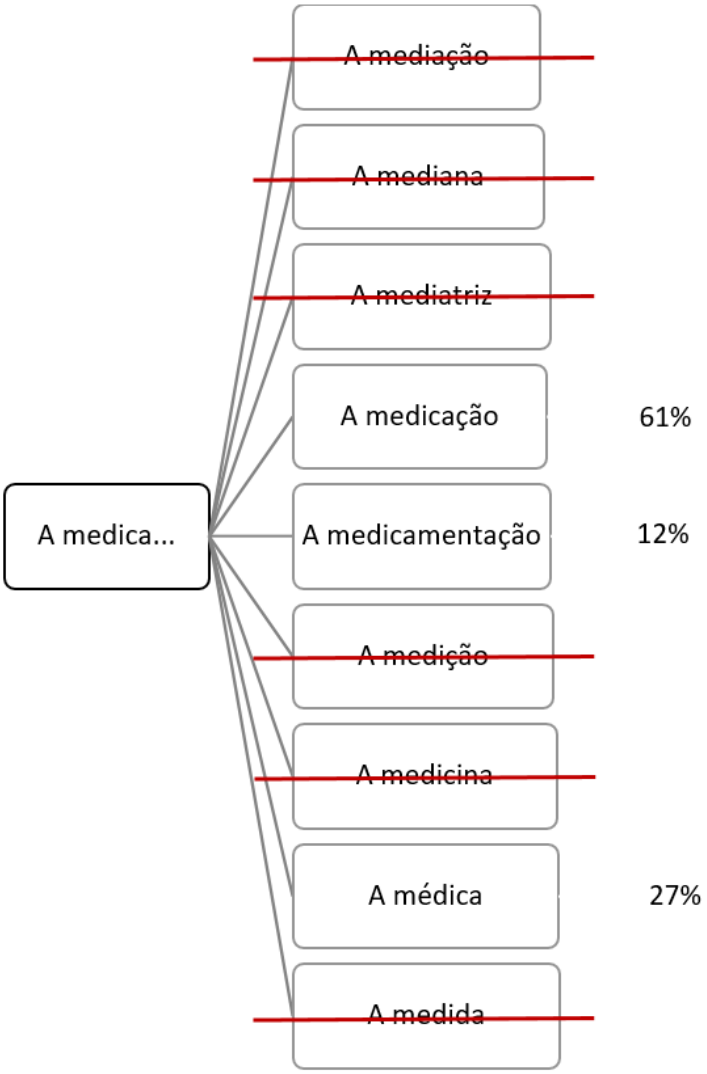


No exemplo acima, a formulação de uma sentença permite que o sistema estime quais são as palavras que poderão ser formuladas pelo interlocutor com base num banco de dados de sentenças, demonstrando quais são as terminações mais frequentes. Somando-se a existência do artigo “A” no início da frase, com a palavra “medi”, obtém-se um universo pequeno de possibilidades para os fonemas seguintes, com somente 5 alternativas: “a”, “ca”, “ção”, “ci”, “da”.

Essa limitação de possibilidades, que acaba por filtrar frases irracionais, faz com que o reconhecedor de voz avalie somente frases plausíveis, diminuindo a sua perplexidade, ou seja, diminuindo o fator médio gramatical de ramificação,

que é o número de palavras que podem se seguir a uma dada palavra. Isso torna a tarefa mais simples, o que acaba por reduzir o tempo de processamento [28].

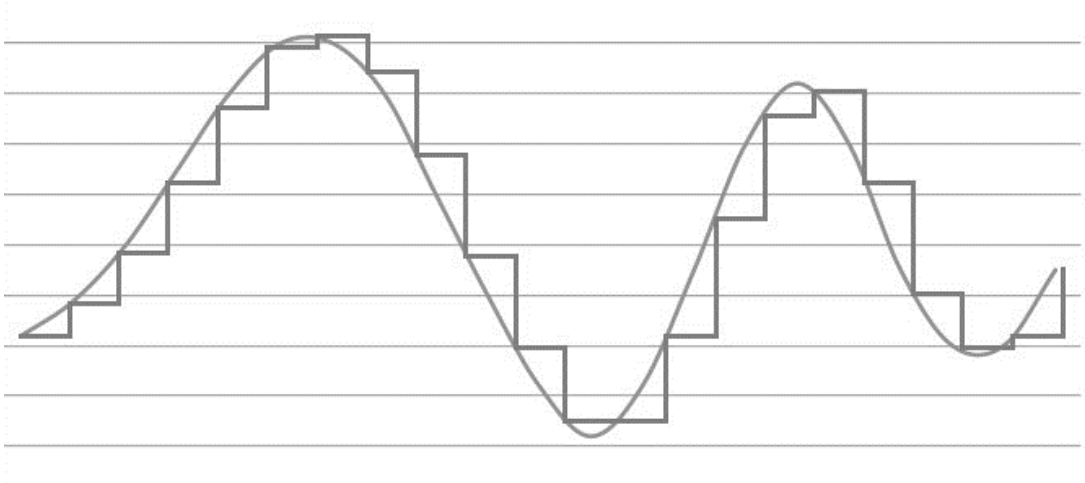
Figura 12 - Exemplo de uma árvore probabilística da tomada de decisão com novas restrições semânticas para a frase após uma nova entrada. Autoria própria.



A fala nativa é amostrada a uma alta frequência, de aproximadamente 16 kHz, fornecendo assim, uma sequência de valores de amplitude ao longo do tempo, que deve ser transformada e comprimida a fim de simplificar os processamentos posteriores [29], como exemplificado na figura abaixo. Dentre os processos de análise de sinal, o mais popular é a análise de Fourier (FFT), que produz frequências discretas ao longo do tempo, distribuídas na escala Mel, que é linear na faixa mais baixa e logarítmica na faixa mais alta, para se adequar

às características do ouvido humano. Outras técnicas também podem ser utilizadas, porém, elas apresentam pouca melhora em relação à FFT [23].

Figura 13 - Ilustração da amostragem de uma onda genérica por transformada de Fourier. Fonte: [23].

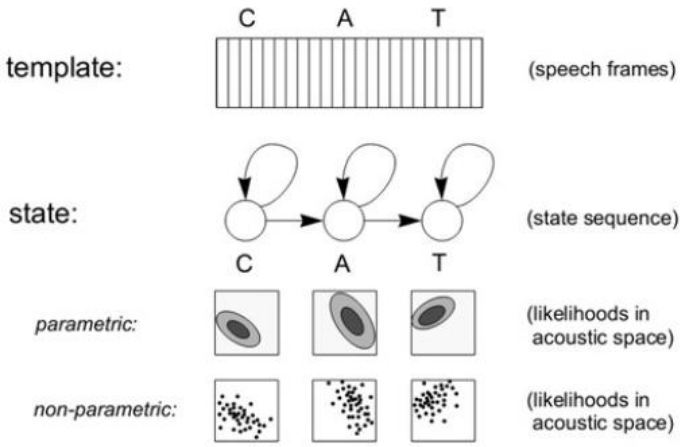


2.7.1 Modelos Acústicos

Após a divisão da onda sonora em valores discretos de amplitude de onda pela FFT, também chamado de *speech frames*, deve-se analisar o conteúdo acústico, ou seja, a mensagem em si, através de modelos acústicos. Dos modelos disponíveis, o mais simples é aquele que consiste apenas de uma amostra armazenada de uma palavra a ser modelada. Assim, se a entrada do sistema for suficientemente parecida com o modelo, a comparação terá um resultado positivo e o sistema reconhecerá a palavra. Essa estrutura apresenta duas limitações: não consegue levar em consideração as variabilidades acústicas e é limitada à quantidade de palavras existentes no banco de dados.

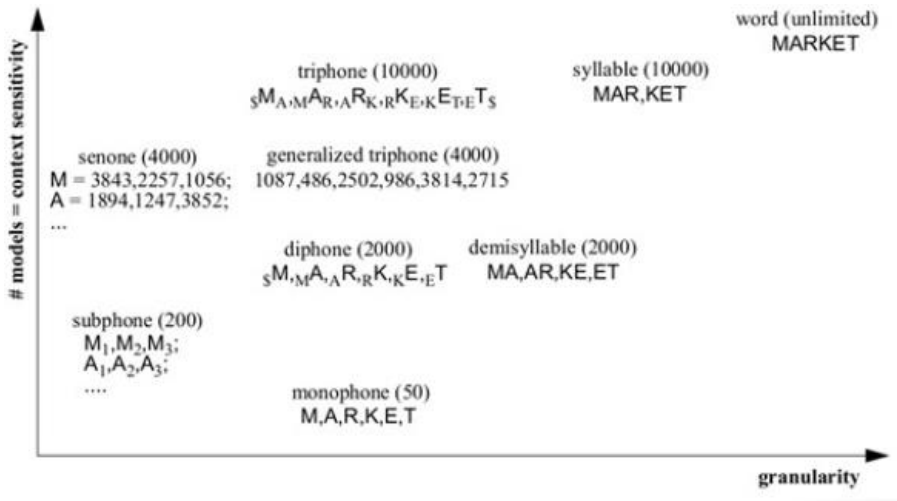
Uma maneira de se obter um sistema mais flexível é a baseada em modelos acústicos treinados, ou seja, em estados. Nesse caso, cada palavra é modelada por uma sequência de estados, onde cada estado indica os sons suscetíveis a serem ouvidos naquele momento, para uma dada palavra. Isso gera uma distribuição de probabilidade no espaço acústico, tal como exemplificado nas figuras 14 e 15 [24].

Figura 14 - Representação de modelos acústicos e seus estados para a palavra "CAT". Fonte: [31].



Os modelos acústicos adotados podem variar em granularidade, indo desde sílabas até a frases inteiras. Quanto maior a sua granularidade, tal como uma palavra, maior a sensibilidade do modelo de reconhecer o contexto e, portanto, maior a sua capacidade de identificar a entrada corretamente. No entanto, quanto mais específico o modelo, pior será o treinamento, pois ele terá necessariamente menos amostrar disponíveis. Assim, sistemas mais complexos, como o Watson da IBM e o TTS do Google, preferem modelos menos granulados, porém, melhor treinados [23] [24].

Figura 15 - Relação de granularidade x sensibilidade do modelo. Fonte: [31].



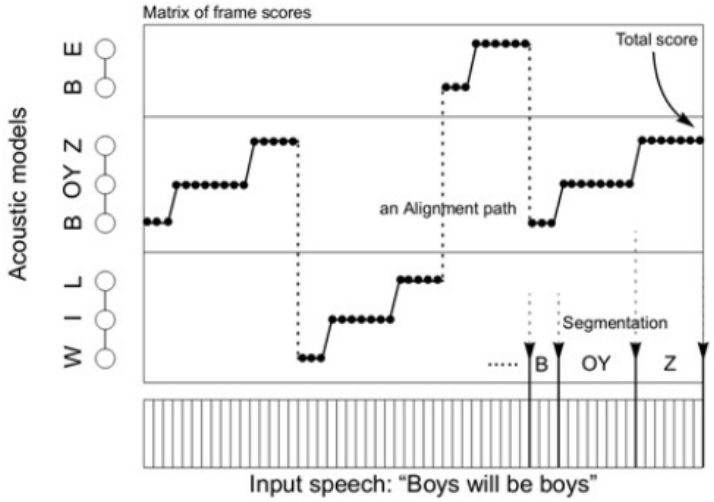
A análise acústica é realizada *frame* a *frame*, aplicando-se cada modelo acústico sobre cada quadro de fala, o que origina uma matriz de pontuação, tal

como na figura abaixo. Caso o modelo utilizado seja o mais simples, com base em padrões, a pontuação é a distância euclidiana entre o quadro de um padrão e o quadro de entrada. Caso o modelo utilizado seja o de estados, a pontuação representa a probabilidade de o estado gerar o quadro atual [24]. Essas pontuações identificam os modelos acústicos mais adequados e as transformam em palavras, assim, aquela combinação de modelos que possui a maior pontuação será a saída do sistema. Esse processo é chamado de alinhamento de tempo [30].

O alinhamento de tempo pode ser realizado por programação dinâmica, com um algoritmo gerando restrições no caminho local. Para um sistema baseado no estado, o caminho ideal induz uma segmentação na sequência da palavra, uma vez que indica quais quadros estão associados com cada estado. Esta segmentação gera marcadores, que irão treinar o modelo acústico recursivamente [30].

O resultado é uma gama de possibilidades de sequência de palavras, ou seja, várias frases hipóteses, com pontuações diversas, permitindo que sistemas complexos possam analisá-la mais de uma vez. Num primeiro momento, os modelos com poucas restrições geram as frases, e posteriormente, na segunda análise, modelos mais complexos aplicam mais restrições, recalculando a pontuação de cada frase-hipótese, retornando uma única saída.

Figura 16 - Alinhamento de tempo para a frase "Boyz will be boyz". Fonte: [31].

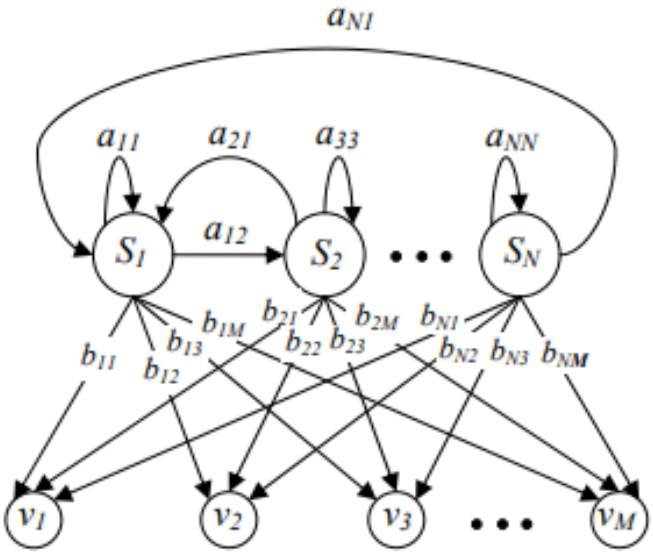


2.8 Modelos Ocultos de Markov

Um processo de Markov, que é um caso particular de um processo estocástico com estados discretos, possui a propriedade de que a distribuição de probabilidade do próximo estado depende apenas do estado atual, e não da sequência de estados que o precederam. Já o modelo de Markov, é um modelo estatístico, onde o sistema é modelado com sendo um processo de Markov com parâmetros a serem determinados, também chamados de parâmetros ocultos.

Em outras palavras, o modelo oculto de Markov é um conjunto de estados ligados por transições onde, a cada intervalo de tempo discreto, uma transição é levada a um novo estado e gera um marcador de saída. As escolhas das transições e dos marcadores são aleatórias, regidas por distribuições de probabilidade.

Figura 17 - Exemplo de Modelo de Markov Oculto com Si estados, vj marcadores observáveis, com probabilidades de transição aij e probabilidade de emissão de símbolos bij. Fonte: [32].



Da figura anterior, pode-se definir formalmente:

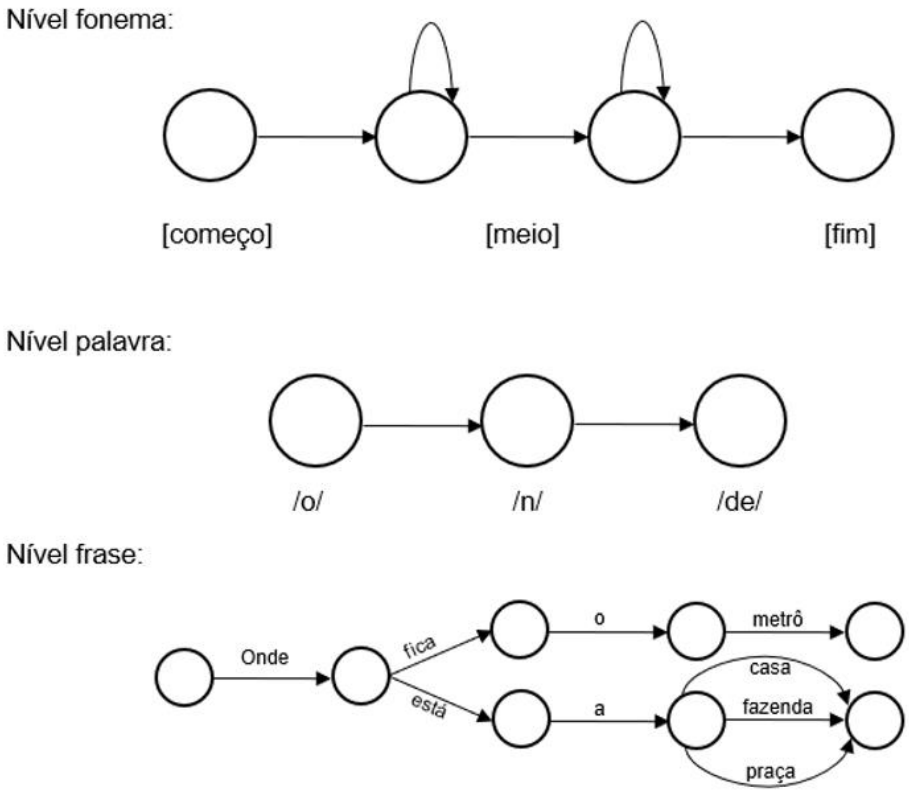
$\{S_i\}$ = conjunto de estados.

$\{a_{ij}\}$ = conjunto de probabilidades de transição, levando o estado de *i* para *j.*

$\{b_{mn}(v)\}$ = conjunto de probabilidades de emissão de marcadores, sendo *b* a distribuição de probabilidades de cada som possível *v* enquanto no estado *m*.

Os *Hidden Markov Models*, ou HMMs, possuem uma variedade de aplicações, como por exemplo, no reconhecimento da fala, onde os estados são os modelos acústicos, e as transições são as restrições temporais. Os estados indicam que os sons são susceptíveis a serem ouvidos durante os seus segmentos correspondentes da fala, e as restrições indicam como os estados podem seguir um ao outro em sequência. Como um discurso sempre avança no tempo, as transições sempre irão para frente (ou no máximo, terão um laço que permite um estado ter duração arbitrária) [32].

Figura 18 - Estrutura hierárquica de um HMM. Fonte [32].



Ao utilizar-se os HMM's de primeira ordem, obtém-se um número limitado de parâmetros treináveis, o que torna os algoritmos de reconhecimento de fala mais eficientes. Dos diferentes algoritmos básicos existentes que se pode associar com os HMM's, pode se destacar três: o progressivo, muito utilizado para a identificação de palavras isoladas; o de Viterbi, empregado no reconhecimento

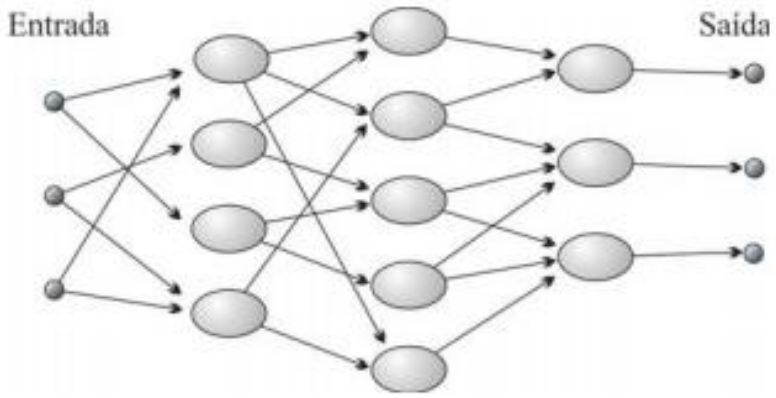
de fala contínua; e o progressivo-regressivo, utilizado no treinamento de HMM [24].

Apesar dessas vantagens, os HMM's possuem algumas limitações quando aplicados ao reconhecimento de voz. Inicialmente, a hipótese fundamental de que as probabilidades de transição dependem unicamente do estado atual, não é de fato verdade no reconhecimento de diálogos. Outro ponto é que as durações das transições são modeladas por uma distribuição exponencial decadente, que não é tão precisa quanto outras alternativas, como a de Poisson [24]. Outros estudos, ainda apontam limitações decorrentes da análise *frame*-à-*frame*, e na capacidade de discriminação dos modelos acústicos [33].

2.9 Redes Neurais

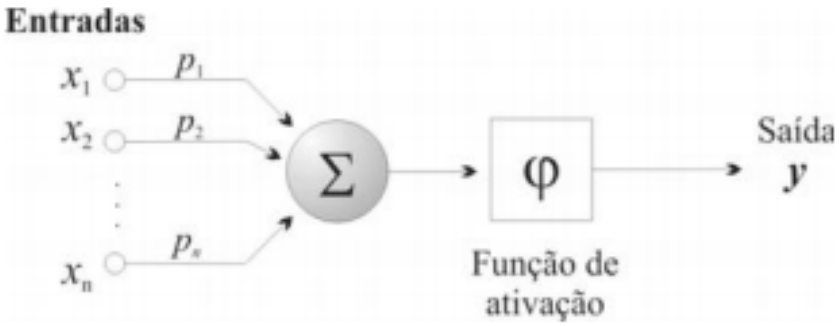
As redes neurais artificiais, ou RNA's, são modelos de processamento computacional inspirados pelo cérebro humano que possuem a capacidade de se aprimorarem sem a intervenção humana (*machine learning*). Ou seja, possuem algoritmos que permitem ao sistema reconhecer padrões e retornar os resultados mais relevantes. Comumente referidas como neurônios interconectados capazes de receberem vários valores de entrada, as RNA's podem ser representadas simplificadamente por um grafo, onde os nós representam os neurônios e as ligações são as sinapses, como na figura abaixo [34].

Figura 19 - Representação simplificada de uma rede neural. Fonte: [34]



O comportamento das conexões entre os neurônios é simulado por pesos diferentes, podendo até serem negativos no caso de conexões inibitórias. O efeito de um sinal enviado por um nó (neurônio) será determinado pela multiplicação da sua intensidade pelo peso da conexão utilizada. A somatória de todas as conexões é chamada de *função de ativação* e define a saída do sistema. Matematicamente tem-se [35]:

Figura 20 - Modelo matemático de um neurônio. Fonte: [34].



Essa área de estudo, também conhecida por conexionismo, foi inspirada na neurobiologia, mas, atualmente, envolve diversas outras áreas do conhecimento, tais como ciências da computação, engenharia elétrica, matemática, física, psicologia, linguística, entre outros. Vários estudos estão sendo realizados na área, principalmente quanto as propriedades da computação neural, utilizando modelos simplificados, tais como [26]:

- Treinabilidade – capacidade de reconhecer padrões de entrada e saída
- Generalização – utilização dos padrões reconhecidos em treinamento em novas situações semelhantes
- Não-linearidade – capacidade de realizar cálculos não-lineares e não-paramétricos nas suas entradas, permitindo-lhes realizar transformações complexas de dados
- Robustez – capacidade de tolerar danos físicos, ruídos de entrada e dados incorretos

- Uniformidade – habilidade de oferecer um paradigma computacional padrão que pode integrar as restrições de diferentes tipos de entrada
- Paralelismo – integração entre várias redes, permitindo o processamento de dados em paralelo, o que aumenta a velocidade de resposta do sistema

Existem diversos tipos de redes neurais, com diferentes arquiteturas, procedimentos de treinamento e aplicações, porém todas elas possuem os mesmos princípios. De forma direta, uma RNA é uma combinação de diversas unidades de processamento simples, chamados de nós ou neurônios, que influenciam umas às outras através de uma rede de pesos incentivadores ou inibidores. Cada unidade calcula a soma ponderada da sua entrada e transmite a sua soma para as outras unidades.

O treinamento de uma rede neural consiste em atribuir valores para a entrada e/ou para a saída, avaliar quais foram os resultados, e modificar os pesos das conexões de tal forma a priorizar os resultados ideais [36]. O reconhecimento de voz e de padrões de compras são exemplos de algumas aplicações que podem utilizar o conceito de redes neurais.

3. Desenvolvimento

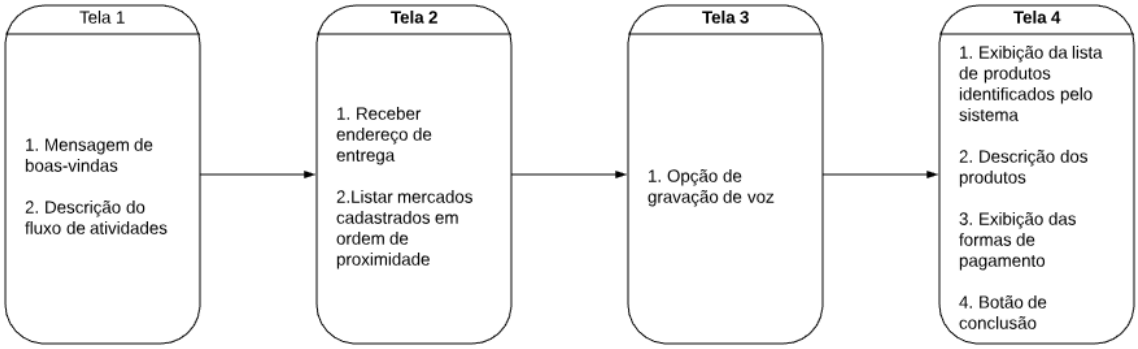
3.1 Modelo

Para o desenvolvimento da aplicação proposta, foram listadas as características dessa solução:

- Capacidade de armazenar o endereço de entrega e recuperar a informação dos mercados disponíveis para compra
- Armazenar o uma lista de ao menos 10 produtos e suas características em cada um dos 3 mercados que serão usados como referência
- Receber dados via voz
- Organizar os dados recebidos em uma tabela e exibi-los de forma simplificada
- Interface simples

O modelo inicial foi inspirado nos aplicativos já existentes, descritos no item 2.2, e está demonstrado no diagrama abaixo. Prevendo-se a utilização de 4 telas de interação com o usuário, sendo uma delas a tela de boas-vindas, a jornada do usuário passa por uma tela de seleção de mercado, onde deve-se inserir, também os dados do endereço de entrega, uma tela gravação da voz com os produtos listados, uma tela de revisão dos produtos e de confirmação da compra e pagamento.

Figura 21 - Diagrama de funcionalidades do aplicativo proposto. Autoria própria.



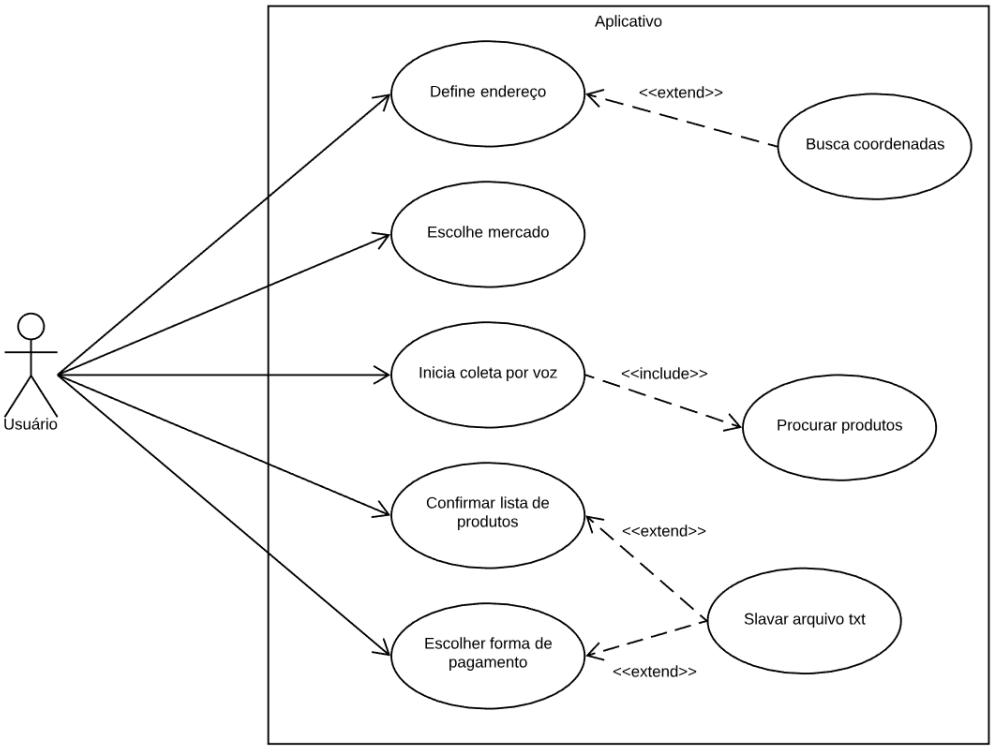
Nota-se que, como o objetivo desse aplicativo é a implementação de uma solução tecnológica para um problema comum, o fluxo de informações e a quantidade de telas foi reduzida, e funcionalidades diversas, como a visualização no mapa, ou mesmo uma tela de configurações não foram incluídas. Um esboço das telas está demonstrado abaixo.

Figura 22 - Esboço da sequência de telas do aplicativo. Autoria própria.

3.2 Diagramas UML

O primeiro passo no processo de análise dos produtos a serem adquiridos é a escolha do mercado e consequentemente, da base de dados a ser comparada. Ao consolidar essa etapa, o agente busca extrair da *string* da lista de produtos, palavras-chave que possam identificar o produto, sejam elas o nome do produto ou da marca, sempre buscando das combinações mais complexas para as mais simples. Portanto, o sistema verificará produtos como “água-de-coco” antes de “água”, saindo do nível frase e indo para o nível palavra, como descrito no item 2.8.

Figura 23 - Diagrama UML de casos de uso do sistema. Autoria própria.

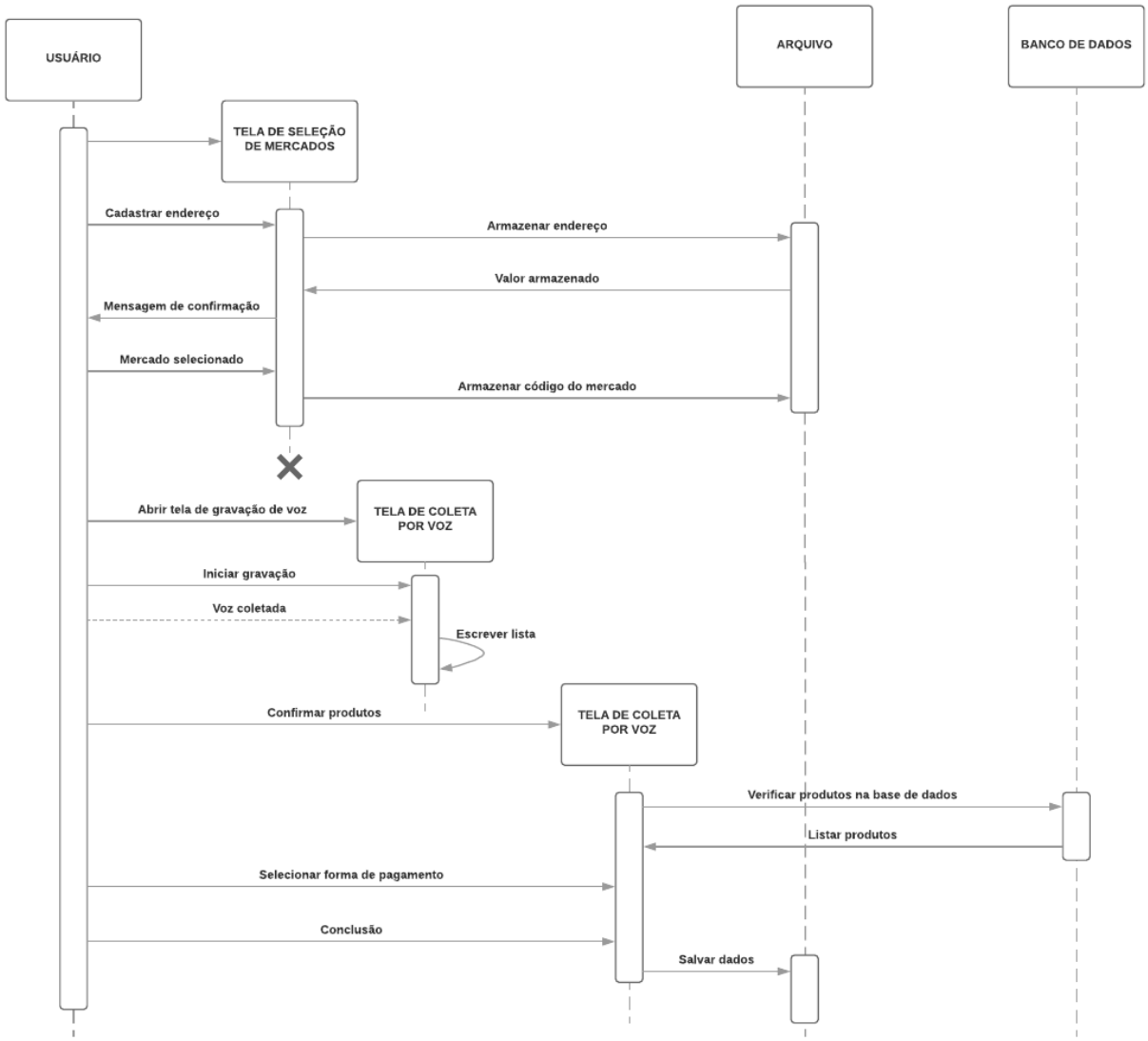


Assim, foi definido um sistema de diálogo em estado finito (item 2.3.1), onde a sequência da fala é determinada por um conjunto de estados e uma quantidade finita de transições pré-determinadas entre eles. Esse sistema receberá essa lista imutável, ou tupla, que é utilizada como estado de informação do sistema,

permitindo ainda, em estudos mais avançados, a inclusão de módulos de recomendação de produtos.

Na figura abaixo está ilustrado o diagrama de sequência do aplicativo que descreve as interações que ocorrem temporalmente entre o usuário e o aplicativo.

Figura 24 - Diagrama UML de sequência do sistema. Autoria própria.

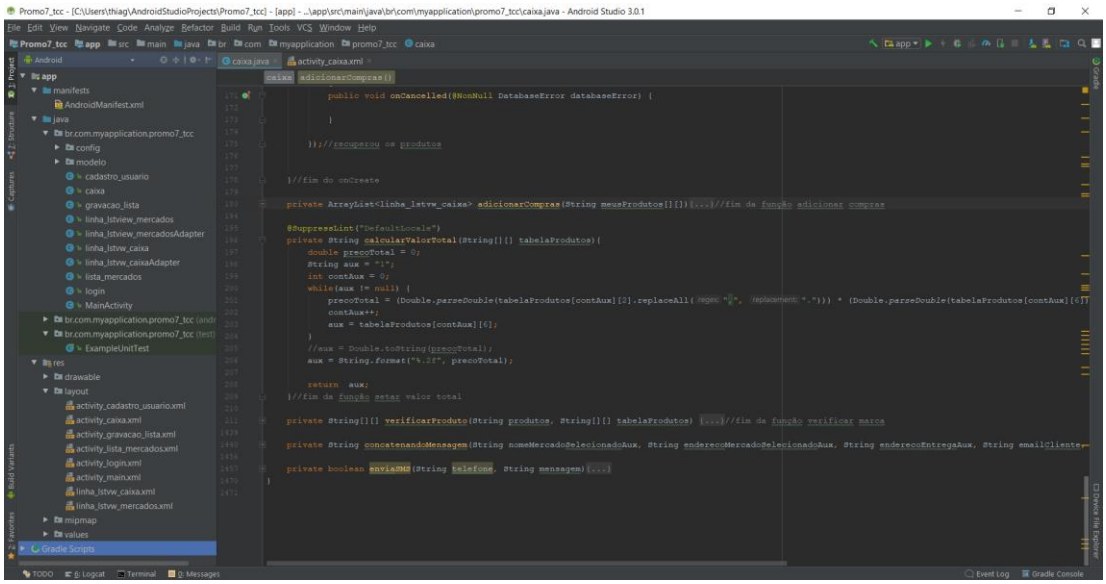


3.3 Ambiente de Desenvolvimento

Inicialmente, por ter a sua programação em Java, o ambiente de desenvolvimento de aplicações para Android mais utilizado era o Eclipse, porém,

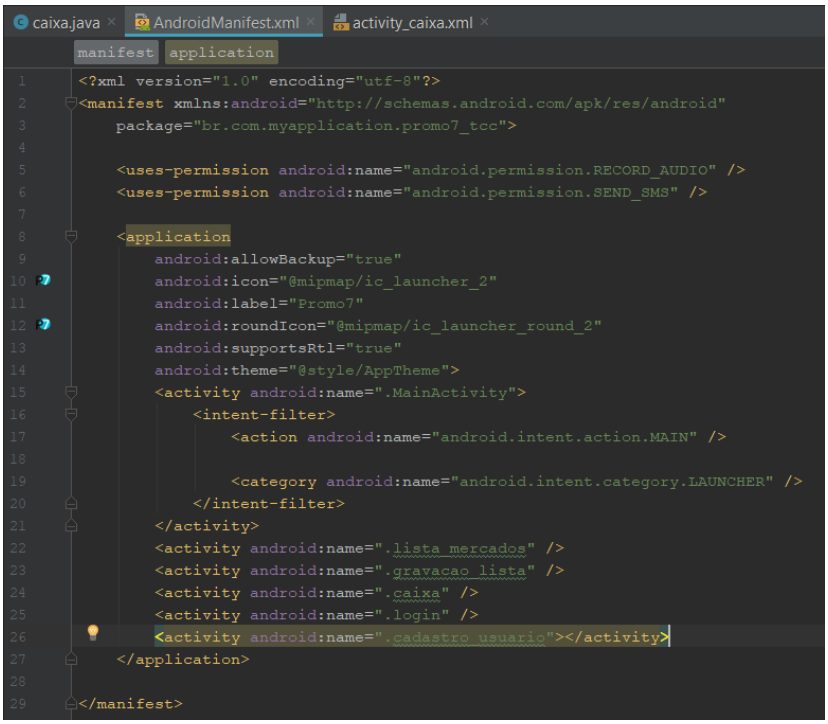
mais recentemente, o Google lançou a plataforma Android Studio (item 2.5) que possui algumas vantagens em relação ao seu predecessor, como um gerenciador de dependência Gradle, que oferece mais opções de compilação ao desenvolvedor. Além desse *software*, é necessário a instalação do *Java Development Kit*, (JDK).

Figura 25 - Ambiente de desenvolvimento Android Studio. Autoria própria.



O Android Studio quebra a programação do aplicativos em *Activities*, que podem ser consideradas as telas da aplicação. Além disso, para cada tela, é gerada duas áreas de código: uma com a terminação *.XML* e outra com a terminação *.JAVA*, responsáveis pelo *layout* e pelos métodos de cada *Activity* respectivamente. Outro ponto relevante desse ambiente de desenvolvimento é a criação do arquivo *Manifest.xml*, que é um arquivo mandatório para qualquer aplicação Android e que concentra informações sobre as *Activities*, as permissões do usuário e outras informações, como as destacadas na figura abaixo.

Figura 26 - Código do Android Manifest.xml com requisição de permissões do usuário nas linhas 5 e 6. Autoria própria.



Para a realização dos testes do aplicativo, pode-se utilizar emuladores no próprio computador ou um dispositivo físico, como o *smartphone*. O Android Studio já disponibiliza alguns emuladores embarcados, porém, eles consomem muita memória e são limitados. Para a realização dos testes da aplicação desenvolvida utilizou-se um dispositivo físico devido à necessidade de testar a função de captação de áudio, que não é possível de testar em emuladores.

3.4 Listas Utilizadas

Para a determinação dos itens a serem reconhecidos pelo semantizador, foi realizado um levantamento em supermercados com os itens mais relevantes para o cliente médio e chegou-se a uma lista de 469 itens diferentes, disponível para observação no apêndice A (levantamento de autoria própria). Considerando uma compra média de até 12 unidades de até 35 produtos diferentes, pode-se concluir que o cliente médio possui uma possibilidade praticamente infinita de compras, como demonstrado abaixo.

$$\frac{469!}{435! * 35!} * 12 = 2{,}251 * 10^{51}\ possibilidades$$

Para efeito de teste, foram selecionados os 10 primeiros itens diferentes da tabela do apêndice A, podendo ser adquiridos até 12 unidades de cada item. Essas condições de teste geram 43 milhões de possibilidades de compras, sem levar em consideração o tipo e a marca do produto.

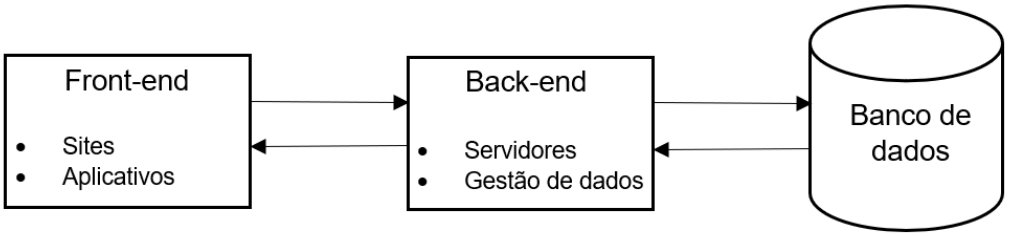
Tabela 2 - Lista de itens selecionados para teste. Autoria própria.

Produto	**Tipo**	**Tamanho**	**Marca**	**Preço**
Alface	Crespa Hidropônica	1 unidade		3,49
Repolho	Verde	1 unidade		1,99
Espinafre		1 unidade		4,99
Rúcula		1 unidade		3,99
Couve	Mateiga	1 unidade		4,99
Abacate		1 unidade		1,99
Abacaxi	Pérola	1 unidade		4,99
Ameixa		1 unidade		1,65
Banana	Prata	1 unidade		0,82
Caqui	Rama Forte	1 unidade		2,70

3.5 Sistema de *Back-End*

De forma genérica pode-se afirmar que serviços web são compostos de duas partes: a interface com o usuário e a gestão das informações do aplicativo ou do site. O Android Studio fornece todas as ferramentas necessárias para o desenvolvimento de uma interface completa, porém, é necessário a criação de uma método de comunicação que seja responsável por transmitir e armazenar todas as informações inseridas pelo usuário, para um sistema único, que terá a função de ser um banco de dados, permitindo o acesso a essas informações pelas pessoas responsáveis. Essa é a estrutura básica de um *back-end*.

Figura 27 - Esquemático do fluxo de informações em um serviço web. Autoria própria.



3.5.1 Firebase

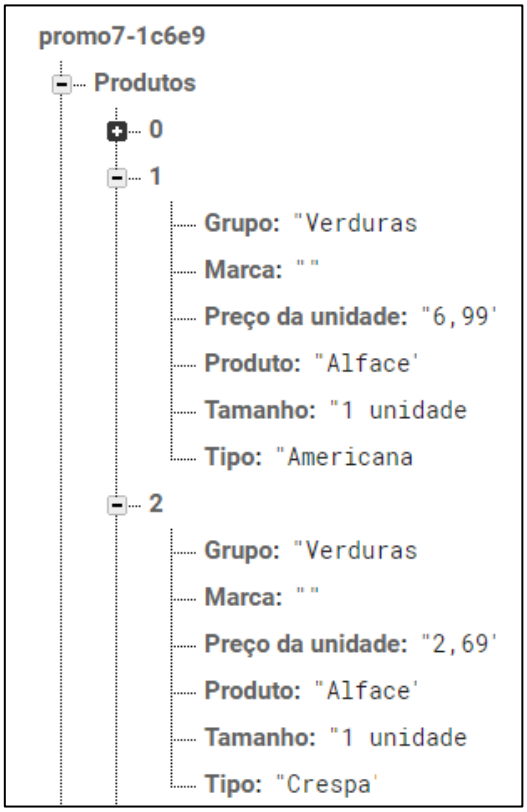
Para o desenvolvimento dessa aplicação, foi utilizado o sistema de *back-end* do Firebase. Essa plataforma possui algumas vantagens em relação à outras alternativas por ser capaz de sincronizar e armazenar grandes quantidades de dados em tempo real gratuitamente.

Adquirido pelo Google, o Firebase foi adaptado para ter uma comunicação simples com o Android Studio, concentrando todos os recursos necessários para um sistema de *back-end* em bibliotecas já prontas, sem a necessidade de refazer sistemas muito complexos. Fornecendo serviços de servidores, banco de dados em tempo real, códigos de servidor e códigos de rede, a implementação dessa plataforma agiliza a programação de aplicações. Todas as informações de métodos de implementação utilizados no desenvolvimento desse sistema estão disponíveis no anexo A, extraídas diretamente do *site* da empresa [37].

Dos recursos disponíveis para utilização via sistema Firebase, foram implementados 3 recursos: *Analytics, Realtime Database* e *Authentication.* Através do recurso de análise, o sistema é capaz de fornecer informações importantes sobre o desempenho do aplicativo e o comportamento do usuário, podendo até realizar análises segmentada por público. Já o sistema de autenticação por *e-mail* e senha, ou até por redes sociais, permite a identificação do usuário, essencial para a segurança de todos os usuários, flexibilizando o método de compras ao permitir que um mesmo usuário faça compras a partir de celulares diferentes. Por fim, o banco de dados em tempo real NoSQL com hospedagem na nuvem armazena os dados em formato JSON e consegue realizar a conexão com os dispositivos em milissegundos. Isso permite a

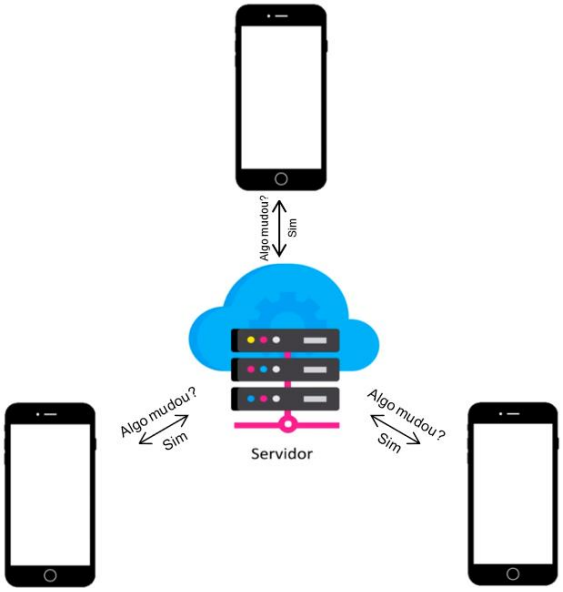
utilização do aplicativo mesmo com quando ele está *offline*, realizando a comunicação com o servidor quando é identificada uma nova conexão com a internet.

Figura 28 - Exemplo do sistema de armazenamento do banco de dados NoSQL em formato JSON. Autoria própria.



Numa estrutura convencional de *front*-end - back-*end*, o aplicativo fica constantemente realizando requisições de alteração de dados, o que pode gerar uma sobrecarga no sistema se o número de usuários crescer acima do previsto. Em termos práticos, o aplicativo “pergunta” ao servidor se houve alguma alteração na base de dados, diversas vezes por minuto, gerando um tráfego constante. Como o servidor, na maioria das vezes irá responder de forma negativa, pode-se perceber que essa abordagem é pouco eficiente.

Figura 29 - Esquema tradicional do sistema de requisições de um sistema back-end. Autoria própria.



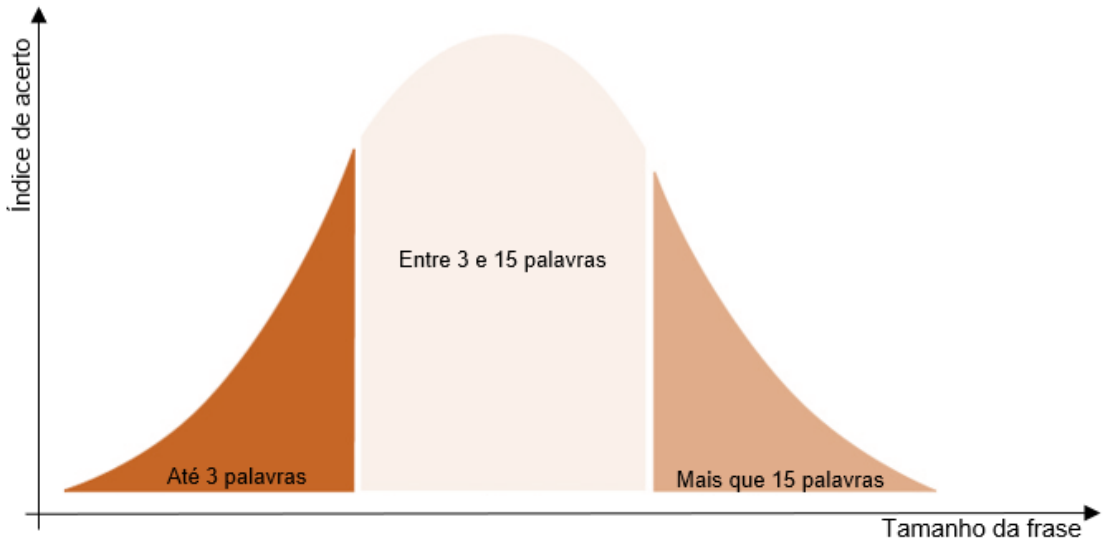
O Firebase permite a utilização de um outro formato de requisição, mais eficiente, em que o aplicativo se utiliza de uma classe *listener.* Dessa maneira, o aplicativo manda uma requisição do tipo *OnDataChange*, em que o servidor irá avisar cada um dos aplicativos em utilização, que houve uma alteração na base de dados. Isso permite que o tráfego de dados seja reduzido e que, mesmo em momento de pico, ou seja, durante a alteração de uma informação na base de dados, o servidor não fique sobrecarregado. Isso ocorre, pois, a parte ativa da comunicação é o servidor e ele não irá transmitir mais dados do que a sua capacidade, tornando o sistema mais robusto.

3.6 Semantizador

O semantizador embarcado do Google e utilizado nesse projeto, ou seja, o módulo responsável por receber e analisar a fala pré-processada pelo módulo de reconhecimento (figura 7), identificando qual a frase teria mais sentido dentro das diversas possibilidades, utiliza modelos ocultos de Markov para a definição estatística do resultado mais provável do discurso. Isso implica em uma probabilidade de acerto na conversão da fala para o texto que pode variar dependendo do tamanho do tamanho do discurso a ser analisado (item 2.7.1).

Para a determinação do formato que permite a maior taxa de acerto do discurso, foi realizado um estudo dos HMM's no nível frase, por já se saber que este é o que possui a maior taxa de acerto (itens 2.6, 2.7 e 2.8). Percebeu-se que frases muito curtas, com até 3 palavras registrou um alto grau de erro entre os usuários, e que frases muito longas, com mais de 15 palavras, geralmente identificavam ao menos uma palavra errada, o que impedia o sistema de reconhecer exatamente a intenção do usuário. Na faixa intermediária observou-se uma variância muito pequena no resultado do reconhecimento das palavras, o que indica que esse é o ponto ótimo do sistema e que a gravação deve ser finalizada em até 10 segundos. Caso seja necessária a continuação da gravação, o aplicativo deverá ser capaz de armazenar a primeira gravação e permitir uma segunda gravação, concatenando as duas listas de produtos.

Figura 30 – Esquemático da distribuição de eficiência de reconhecimento de fala eperada. Autoria própria.



Esse tópico foi analisado em maior profundidade através de experimentos e está disponível no item 4.1.

3.7 Extraindo palavras-chave: redes neurais e *machine learning*

Para o reconhecimento dos produtos a serem adquiridos pelo usuário, foi realizado o levantamento de formas diferentes de se referir a esses produtos:

“um alface”, “um pé de alface”, “uma alface”, “doze bananas”, “uma dúzia de bananas”, “uma penca de bananas”. Essas estruturas linguísticas levantadas a partir de entrevistas com diversas pessoas compõe o ponto chave da aplicação. Ao conseguir reconhecer a quantidade e os itens a serem adquiridos a partir de cada uma dessas frases, o sistema se torna robusto o suficiente para utilização prática, fora do ambiente de testes.

Aprimorando esse sistema, foi testado também a aplicação de *machine learning* por rede neurais, incorporadas ao banco de dados SQLite do *smartphone.* Para determinadas palavras-chave, a adequação da marca ou do tipo do produto irá ocorrer de forma automática, baseado no padrão de compras do usuário.

O sistema funciona da seguinte maneira:

Como a base de dados de produtos é única para todos os usuários, ela deve ser armazenada *online*, na plataforma do Firebase, para facilitar a atualização de produtos e preços. Porém apesar de também ser possível armazenar as informações do usuário dessa maneira, optou-se por praticidade, por armazenar as preferências do usuário no seu próprio dispositivo físico, individual. Essa alternativa, em compensação, apresenta algumas limitações, como o fato de, caso o usuário realize a compra dele em um outro celular, as suas preferências não estarão armazenadas ali.

Assim, pode-se dizer que ao realizar uma compra de um produto de um determinado tipo e marca, o aplicativo importa a base de dados do servidor (Apêndice A), reconhece a quantidade e o produto pelo tratamento da fala e extração de palavras-chave, e puxa do SQLite, o peso atribuído àquele produto, transformando a lista de produtos a serem comprados em uma matriz, como no formato abaixo.

Tabela 3 - Matriz de produtos e suas fontes. Autoria própria.

Produto	Tipo	Tamanho	Marca	Preço	Quantidade	Peso
Firebase	Firebase	Firebase	Firebase	Firebase	Semantizador	SQLite

O peso de um produto é ignorado quando a marca e/ou o tipo são especificados, porém, quando a palavra é genérica, esse número passa a ser considerado. O sistema busca todos os produtos iguais no SQLite, e realiza a somatória dos pesos já atribuídos com um limite de 10. Esse limite se justifica para atender os princípios de Treinabilidade e Generalização das redes neurais. Assim, a aquisição de um produto específico adicionará 1 para esse produto e removerá 1 do produto que possui o valor mais antigo, no esquema de fila. Não tenha realizado 10 compras daquele produto, não haverá remoção de peso.

Por exemplo, no caso de compra de um produto como alface, em que a variante não é a marca, mas o tipo, se o usuário disser “3 alfaces americana”, não haverá cálculo de probabilidade, porém, se a frase for genérica do tipo “3 alfaces”, esse cálculo será realizado. Por exemplo, suponha que esse usuário já realizou dez compras de alface na seguinte ordem:

Tabela 4 - Exemplo de distribuição de compra de alface por tipo no tempo. Autoria própria.

Compra número	**1**	**2**	**3**	**4**	**5**
Tipo de alface	Lisa	Lisa	Lisa	Americana	Americana

Compra número	**6**	**7**	**8**	**9**	**10**
Tipo de alface	Americana	Americana	Crespa	Crespa	Crespa

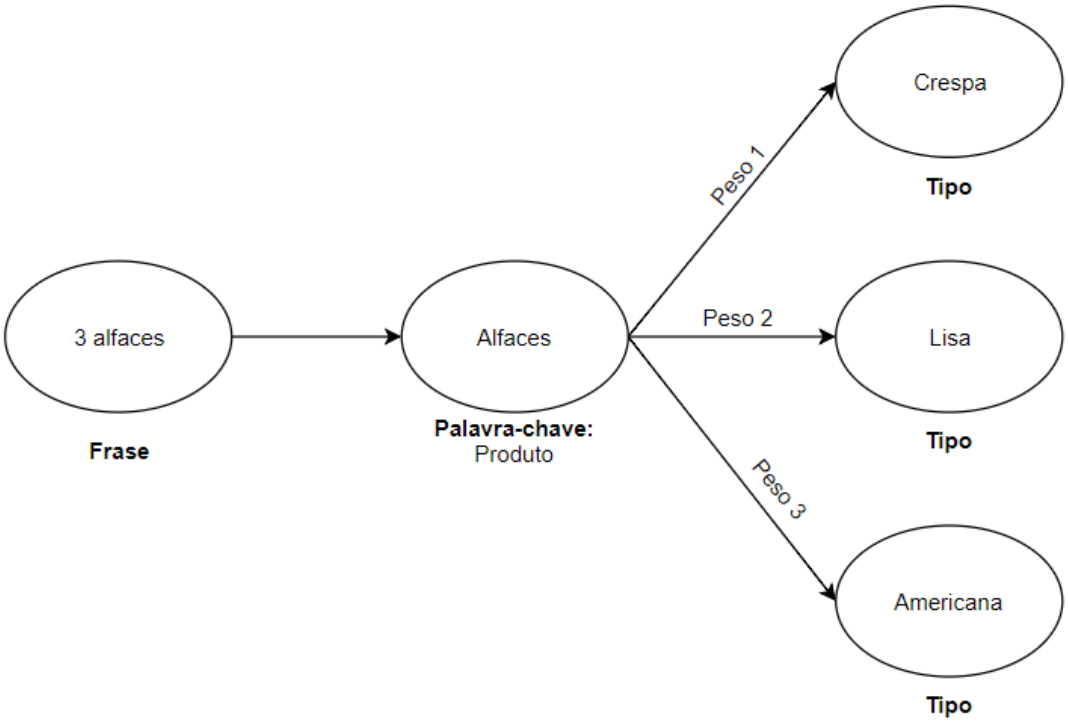
Assim, o aplicativo deve armazenar o valor 3 para alface do tipo lisa, 4 para o tipo americana e 3 para o crespa. Comparando os pesos, o sistema irá fornecer a alface americana para o cliente, que, ao finalizar a compra, adicionará mais 1 ao peso da alface adquirida e removerá 1 da alface mais antiga, no caso a lisa. A nova formação com os pesos atualizados ficará como exemplificada abaixo.

Tabela 5 - Exemplo de distribuição de compra de alface por tipo no tempo após a 11ª compra. Autoria própria.

Compra número	**1**	**2**	**3**	**4**	**5**
Tipo de alface	Lisa	Lisa	Americana	Americana	Americana

Compra número	**6**	**7**	**8**	**9**	**10**
Tipo de alface	Americana	Crespa	Crespa	Crespa	Americana

Figura 31 - Exemplo de rede neural para um produto específico. Autoria própria.



Segundo a imagem acima, os novos pesos após a realização da 11ª compra ficarão:

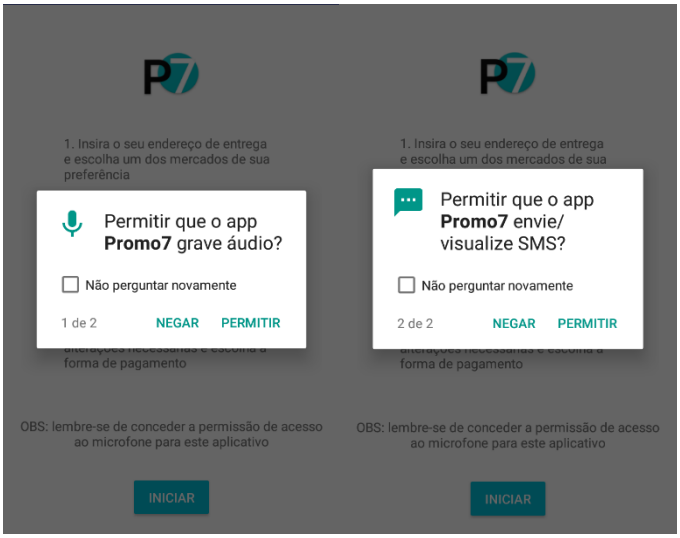
- Peso 1 = 3
- Peso 2 = 2
- Peso 3 = 5

3.8 Protótipo

O desenvolvimento dessa aplicação foi feito a partir do diagrama UML (figura 24) com a adição de um sistema de login de usuário e algumas pequenas modificações em relação ao design inicial (figura 22). Assim, o fluxo de telas ficou disposto da seguinte maneira:

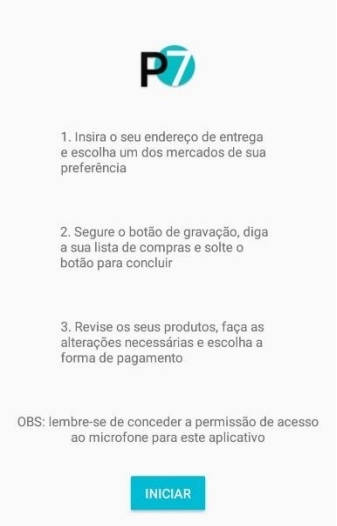
- Devido à necessidade de acesso ao microfone e ao SMS, foram inseridas requisições de funcionamento antes do acesso à primeira tela.

Figura 32 - Verificação de permissões de acesso do aplicativo. Autoria própria.



- Após conceder as permissões necessárias, o usuário tem acesso à primeira tela, com as instruções de funcionamento do aplicativo

Figura 33 - Tela inicial, com instruções de funcionamento do aplicativo. Autoria própria.



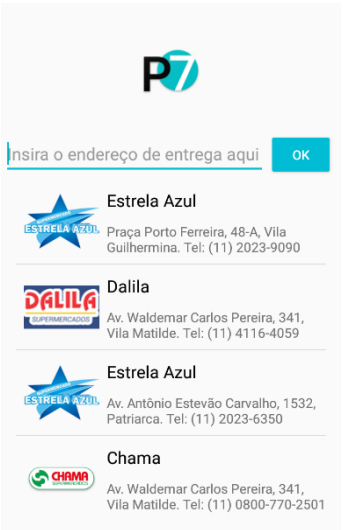
- Na tela seguinte, foi inserida uma área de login do usuário. Optou-se pelo método de *e-mail* e senha, mas outras formas de autenticação também podem ser utilizadas, como por redes sociais. Caso o usuário não tenha cadastro, pode-se criar uma conta facilmente pelo próprio aplicativo, pois foi criada uma conexão com o servidor que valida as

contas e verifica se todos os requisitos para a criação de um novo usuário foram atingidos (*e-mail* válido e senha com 6 dígitos).

Figura 34 - Segunda tela: tela de login e tela de cadastro. Autoria própria.

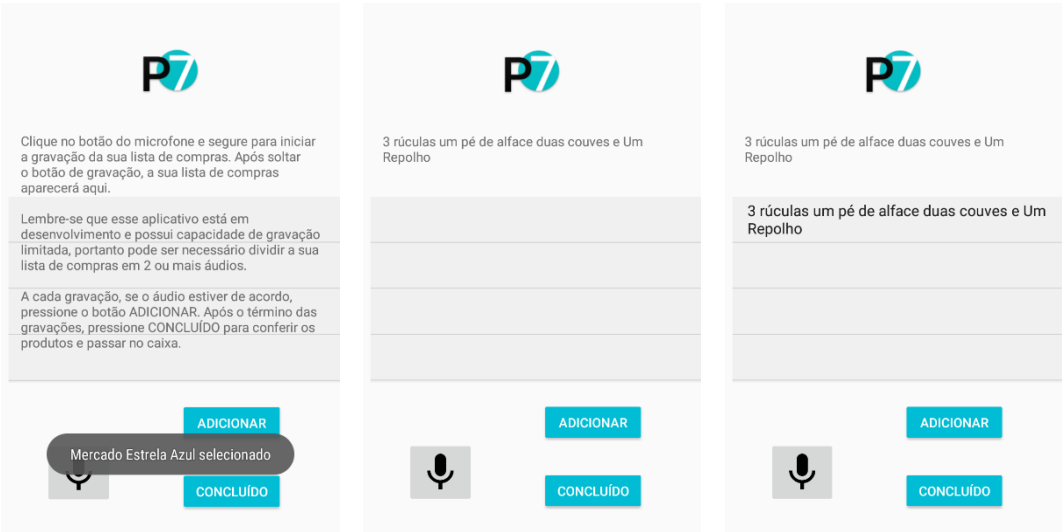
- Na terceira tela, tal como descrito na figura 22, insere-se o endereço de entrega e seleciona-se o mercado. Devido à complexidade de validação do endereço, esse campo não possui qualquer forma de validação se o endereço existe ou não, porém, possui algumas formas de tratamento de erro (item 4.3).

Figura 35 - Tela de seleção de mercados. Com o endereço registrado, o mercado selecionado é a transição para a tela de compras. Autoria própria.



- A quarta tela é a tela de compras. Nessa etapa o usuário irá utilizar o sistema de reconhecimento de voz para ditar os produtos que quer comprar. Inicialmente, ao ser carregada, a página mostra um breve texto de explicação do funcionamento do aplicativo (figura 36A), que desaparece ao se iniciar a gravação. Essa tela também pode ser dividida em 3 partes: a superior, a intermediária e a inferior. A superior é onde é impresso o que o usuário falou, porém ainda passível de confirmação (figura 36B). Na intermediária, ficam as listas dos produtos adicionados. Uma vez validado o reconhecimento do áudio, ao pressionar o botão “Adicionar”, o texto é armazenado (figura 36C). Por fim, na parte inferior ficam os botões que são usados como marcadores de validação.

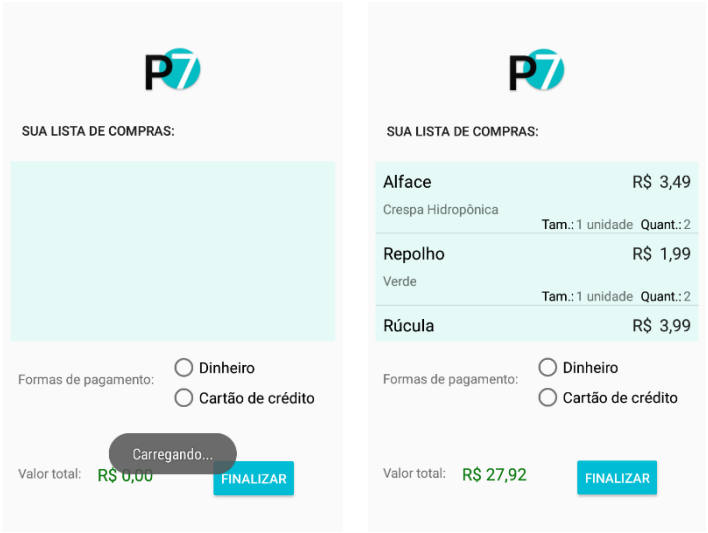
Figura 36 - Quarta tela: gravação da lista de compras. Instruções de funcionamento (A), reconhecimento do que foi dito pelo usuário após soltar o botão de gravação (B) e a confirmação da compra na lista ao pressionar o botão "Adicionar" (C). Autoria própria.



- Na última tela é realizado todo o processamento do que foi informado ao sistema. É realizado o *download* da base de dados do servidor, ela é ajustada para um formato adequado, para evitar conflitos com a base de dados do SQLite, e armazenada numa matriz de *strings*. A seguir, é realizada a análise de palavras-chave. Determinadas frases e palavras são atribuídas à certos produtos e quantidades, que irá gerar

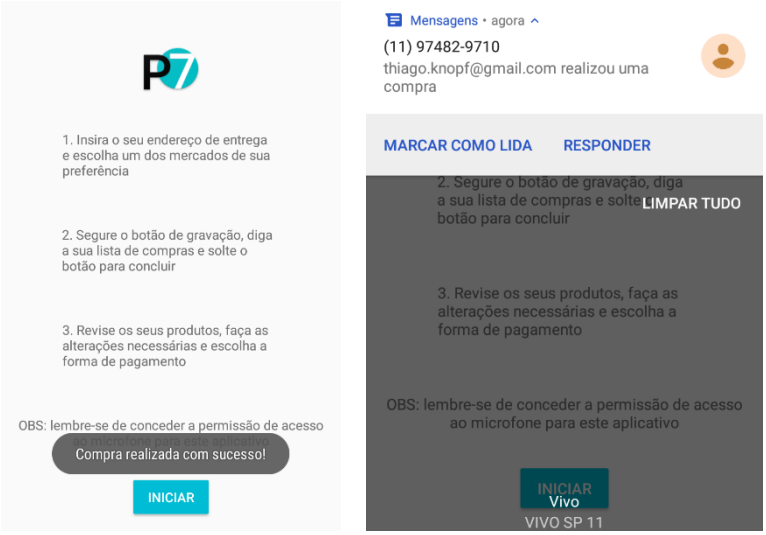
uma segunda matriz de *strings*. Por fim é feita uma comparação de ambas as matrizes, gerando os itens que foram reconhecidos e as suas quantidades. Multiplicando-se as quantidades pelos preços informados de todos os produtos tem-se o valor total.

Figura 37 - Última tela: reconhecimento dos produtos inseridos pelo usuário. Autoria própria.



- Ao selecionar a forma de pagamento e finalizar a compra, o usuário é redirecionado à tela inicial com um aviso de que a compra foi realizada com sucesso (figura 38A). Por fim, o telefone do usuário envia um SMS para o celular do responsável pelo *back-end* (figura 38A), confirmando o cliente e todas as informações da compra.

Figura 38 - Confirmação de que a compra foi realizada com sucesso (A) e recebimento de notificação de que um usuário realizou uma compra (B). Autoria própria.



- O responsável pelo *back-end*, com acesso ao servidor pode agora verificar a adição da compra na base de dados do aplicativo, sob o identificador “Pedidos”.

Figura 39 - Base de dados do aplicativo com 3 nós: Pedidos, produtos e usuários. Autoria própria.

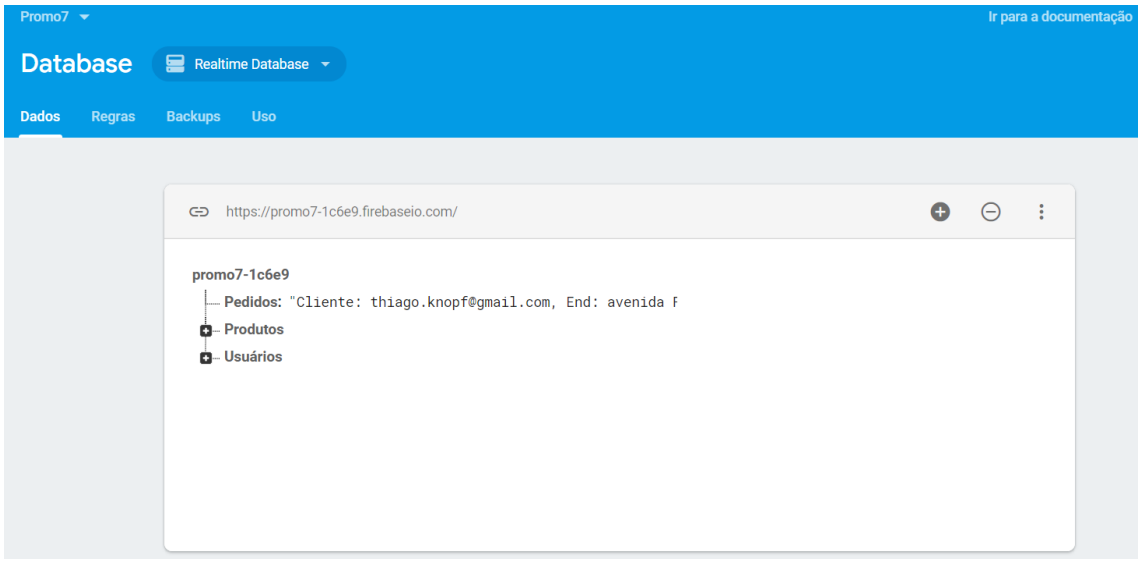
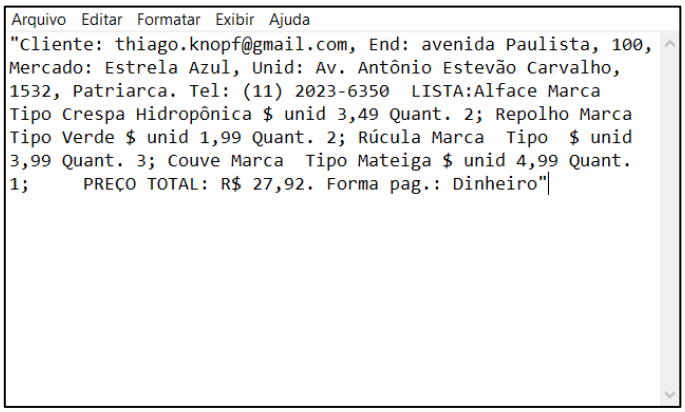


Figura 40 - Lista completa da compra realizada extraída da base de dados



4. Experimentos

4.1 Otimização do Semantizador

A capacidade de reconhecimento da fala do sistema embarcado no Android possui algumas limitações que foram levadas em consideração na hora de permitir a gravação. Pelo entendimento nos fundamentos do modo de funcionamento de sistemas de reconhecimento de voz (item 2.7.1) e no modo de análise de um modelo oculto de Markov (2.8), pode-se concluir que existe um ponto ótimo de acerto de reconhecimento de fala.

Caso a fala seja muito curta, como no nível fonema ou palavra (figura 18), o sistema tem dificuldade de calcular qual a palavra referida com precisão. Isso se deve ao grande número de palavras com sons parecidos. Para minimizar essa forma de erro, deve-se permitir que se gravem frases inteiras, para que o contexto das frases ajude na tomada de decisão do sistema. Isso é possível pelo método *hybrid-approach* de comparação de diálogos.

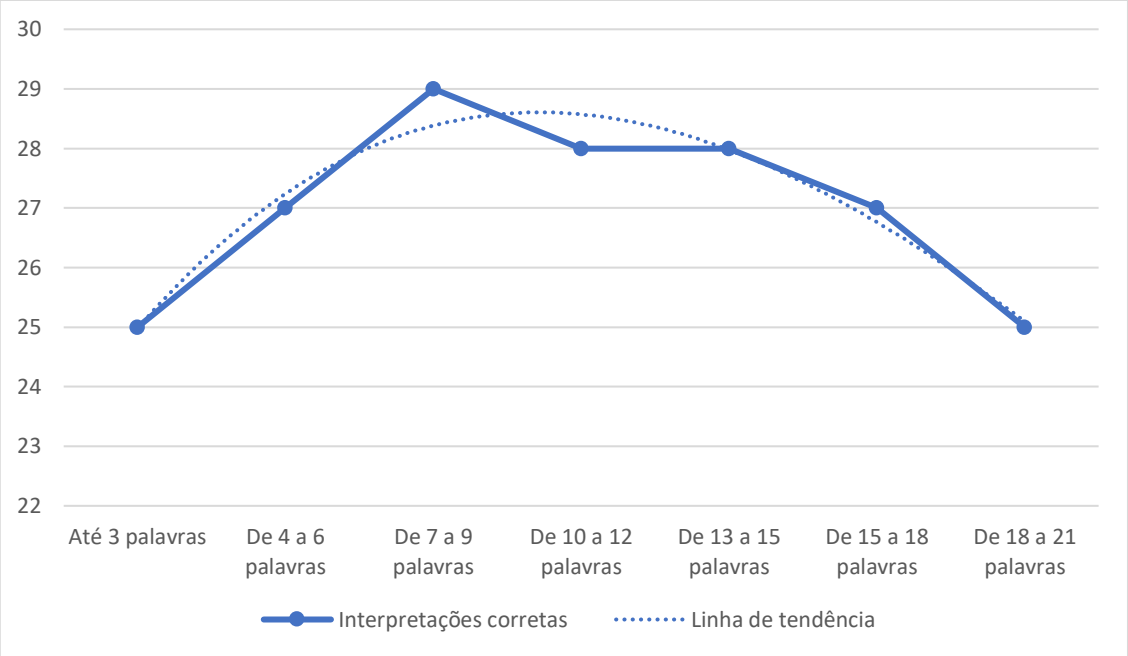
Por outro lado, quanto mais palavras forem captadas de uma vez, maior a chance de uma dessas palavras ser interpretada erroneamente. Numa sequência de palavras fornecidas por áudio, uma única palavra incorreta pode afetar por completo a interpretação da intenção do usuário, impossibilitando o seu uso pelo sistema. De forma simplificada pode-se entender o ponto de otimização no acerto da transcrição da fala como sendo o da figura 30.

Para realizar os testes de eficiência, foram utilizadas 3 frases-teste diferentes com os produtos já cadastrados, e foram adicionadas até 3 palavras por grupo de teste. Assim, para o grupo 1, de até 3 palavras, foram testadas 3 frases diferentes. Para o grupo 2, de 4 a 6 palavras, foram testadas mais 3 frases diferentes. Para o grupo 3, de 7 a 9 palavras, foram testadas mais 3 frases diferentes. E assim sucessivamente, até o teto de 21 palavras. Cada frase foi testada 10 vezes e os dados de testes detalhados estão no apêndice B.

A conclusão, como demonstrada no levantamento do apêndice B e replicada na figura abaixo, foi de que a partir de 4 palavras o sistema já possui um bom índice de acerto, que não varia muito para os grupos seguintes. A partir de 15

palavras, o sistema já começa a apresentar uma sensível queda na capacidade de reconhecer todas as palavras. Apesar de não ser tão significativa, essa queda estimulou a criação de uma trava no tempo de áudio, limitando-o a poucos segundos.

Figura 41 - variação de interpretações corretas pelo semantizador por tamanho da frase



Esse cuidado é recomendado no desenvolvimento de um produto para o varejo pois erros, mesmo que pouco frequentes e até erroneamente considerados desprezíveis, podem gerar uma aversão em potenciais clientes, que podem ou não recomendar o seu produto para outras pessoas. Durante os testes, percebeu-se que uma interpretação errada do áudio era, em geral, bem aceita pelo usuário, mas uma segunda interpretação errada já era suficiente para desestimular o uso do aplicativo, chegando ao ponto de usuários se recusarem a realizar uma terceira gravação para testes.

4.2 Rede Neural para Produtos

A identificação dos produtos ou marcas a partir de uma frase é uma atividade muito trabalhosa pois exige que uma gama de possibilidades de discursos seja

levada em consideração. Assim, para abordar o problema da identificação dos produtos e das quantidades ditadas pelo usuário, usou-se o princípio de análise do arranjo por redes neurais.

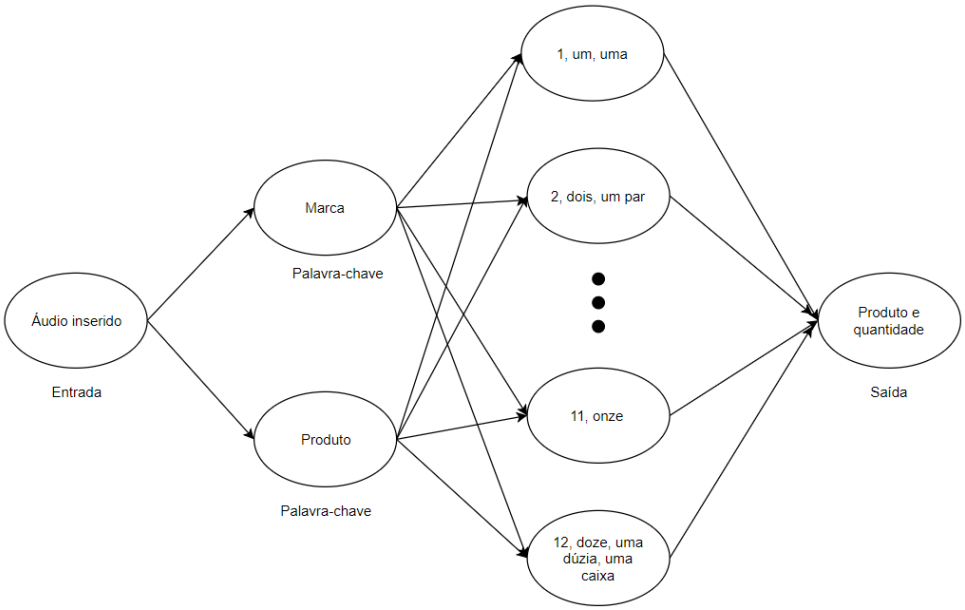
Usando-se a tabela de referência de produtos iniciais (tabela 2) e considerando que existem 10 possibilidades de compra, com até 12 unidades por produto, nota-se a necessidade da utilização de um modelo de reconhecimento de produtos que diminua o número de combinações de listas de compras.

O sistema realiza a concatenação de todos os áudios gravados pelo usuário em uma única *string* e passa a analisar as possibilidades a partir de palavras-chave. Ele percorre toda a frase procurando os produtos. Ao encontrar algum produto que existe na tabela do Firebase, ele verifica qual o número que foi atribuído àquele produto.

Por exemplo, caso o usuário insira o áudio “uma caixa de refrigerante”, o sistema irá buscar produto por produto, da lista de todos os produtos disponíveis (apêndice A), iniciando por “alface”, “repolho” “espinafre” e assim por diante. Todas essas comparações irão retornar *False*, porém, na comparação número 285, ela irá retornar *True* pois a palavra-chave (produto) da base de dados é “refrigerante”. No entanto, a segunda comparação irá retornar *False*, pois, o sistema irá buscar a quantidade “caixa” dentro das possibilidades desse produto. Como o produto é “palavra-chave: refrigerante, tamanho: 2l”, não existe a quantidade “caixa” para ele.

Continuando a comparação de palavras-chave, na verificação número 286, o sistema irá retornar o produto corretamente, pois o produto é “palavra-chave: refrigerante, tamanho: 200ml” e, para esse produto, a quantidade “caixa” está registrado e é equivalente a 12 unidades. Em um segundo momento, recomenda-se aprimorar o sistema para reconhecer marcas como sendo um nó na rede neural, inserido após os produtos ao invés de ser um nó complementar, como demonstrado abaixo.

Figura 42 - Modelo de pesquisa de produto por rede neural. Autoria própria.



Essa forma de pesquisar os itens a serem comprados se mostrou muito eficaz, reduzindo o número de possibilidades de pesquisa e sendo praticamente à prova de erros se a base de dados for povoada corretamente. Uma vez identificado o item e a quantidade, a comparação de texto é uma operação razoavelmente simples, que apesar de muito trabalhosa, se mostrou confiável.

Não foram observados erros decorrentes do sistema. Os únicos erros verificados foram humanos, e de homônimos. Os erros humanos são de simples solução e são decorrentes das diferentes maneiras que cada pessoa pode expressar a mesma informação. Assim, uma pessoa pode falar “uma dúzia de bananas” enquanto as outras falam “doze bananas”. Caso “uma dúzia” não esteja registrado como quantidade, o programa não irá reconhecer a o valor.

Já a questão do homônimo não é tão simples. Uma pessoa pode querer adquirir “uma batata palha”, mas, a não ser que exista um tratamento de exceção, o sistema irá reconhecer 2 produtos diferentes: uma batata e um pacote de batata palha. Nesse caso, deve-se atribuir ao produto de menor palavra-chave, no caso “batata”, uma exclusão se a palavra seguinte for “palha”. Mesmo assim, essa solução, apesar de ajudar a aumentar a eficiência do reconhecimento dos produtos, não é suficiente, pois o cliente pode querer comprar batata e batata-palha ao mesmo tempo, por exemplo.

4.3 Tratamento de Erros

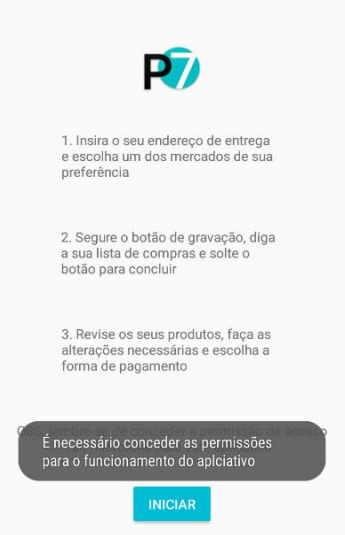
O tratamento de erros ou tratamento de exceções é a criação de métodos que corrigem alterações no fluxo normal da execução da aplicação. O tratamento dessas exceções parte do princípio de que o usuário é hostil e que, portanto, não utilizará o programa da forma como ele foi desenhado. Cabe ao desenvolvedor criar rotinas que reconhecem esses erros e decidem se o programa irá continuar ou não e, se decidir continuar, como fazê-lo (sistemas multiagente, item 2.4).

No desenvolvimento dessa aplicação foram encontrados diversos erros que devem ser tratados. Alguns bloqueiam o prosseguimento da aplicação, outros recuperam informação e outros não foram tratados para efeito demonstrativo.

A metodologia empregada na identificação desses erros se baseou em entrevistas com usuários diversos. 7 pessoas foram responsáveis por testar o aplicativo em diversas etapas do desenvolvimento, e geraram uma lista de erros que estão demonstrados abaixo.

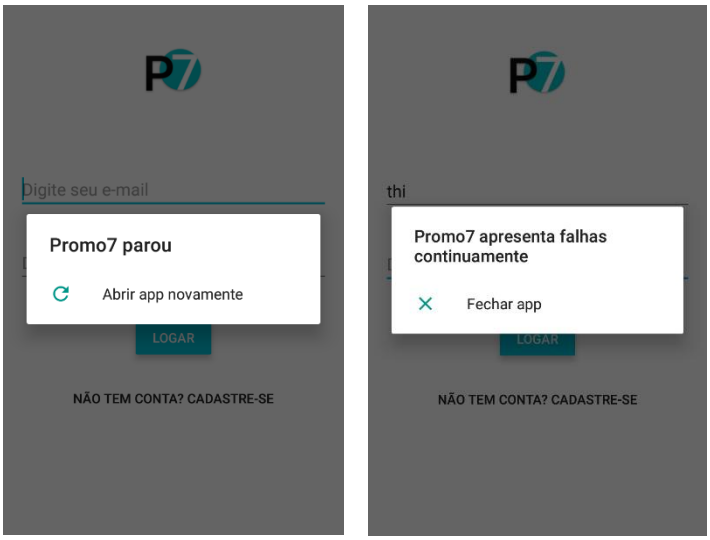
- Erro 1: na tela inicial, o usuário pode se recusar a conceder as permissões de gravação e envio de SMS ao aplicativo. Como essas funções são essenciais para o correto funcionamento do programa, optou-se por exibir uma notificação de alerta e travar o funcionamento do sistema. Assim, ao clicar no botão “Iniciar”, nada ocorrerá, obrigando o usuário a conceder as duas autorizações. Mesmo se uma única autorização for concedida e uma for recusada, o aplicativo não avançará para a segunda tela.

Figura 43 - Tratamento de erro na tela inicial. Autoria própria.



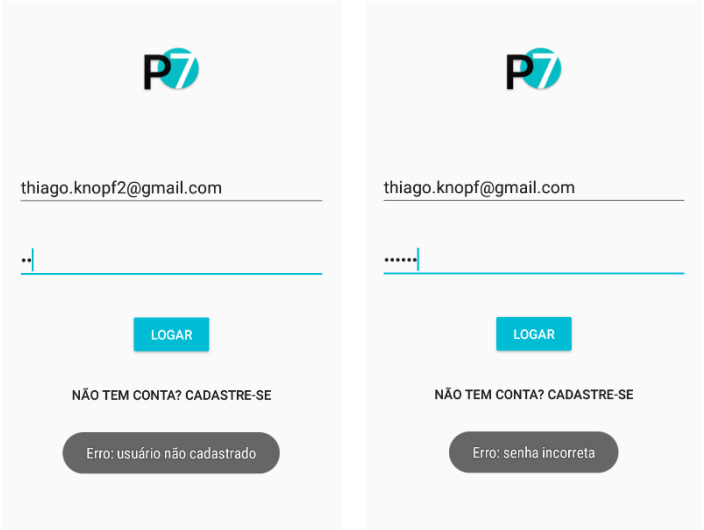
- Erro 2: na tela de login, pode ocorrer de o usuário não inserir o endereço de *e-mail* e clicar no botão logar ou inserir um *e-mail* inválido. Para efeitos demonstrativos, esse erro não foi tratado. Nota-se que apesar de as duas abordagens serem muito semelhantes, o sistema responde de maneira diferente. Ao não inserir nenhum endereço, a aplicação simplesmente para de funcionar (figura 42A), porém ao inserir um endereço que não possui a estrutura de um *e-mail* tradicional, uma mensagem diferente é exibida na caixa de alerta, o que indica dois erros diferentes. Para tratar esses erros, recomenda-se a inclusão de um comando *if*, que verifica se o *e-mail* possui mais de 5 caracteres e se possui @.

Figura 44 - Erro na validação do usuário. Nenhum endereço de e-mail inserido em A e endereço inválido em B. Autoria própria.



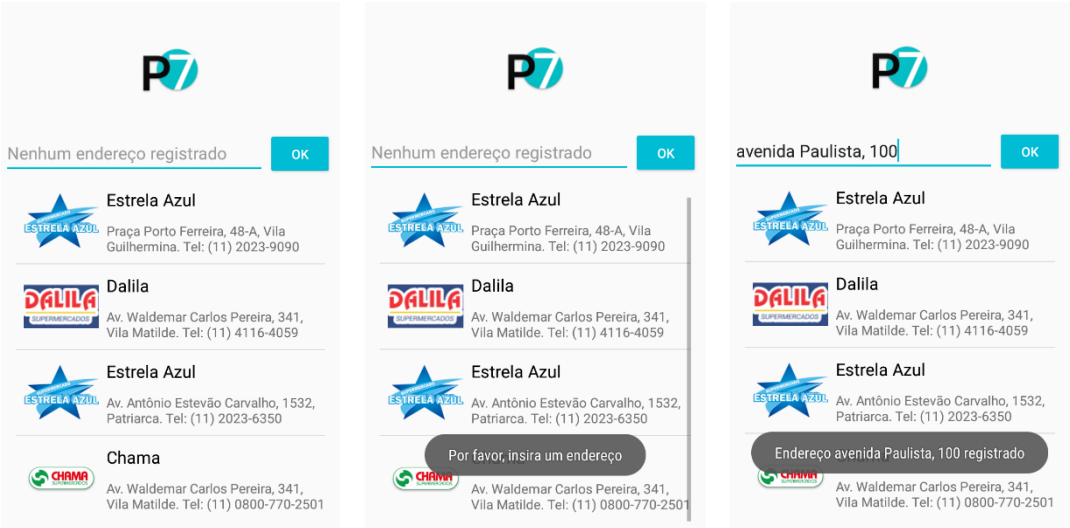
- Erro 3: caso o usuário insira um endereço de *e-mail* válido, porém incorreto ou caso ele insira uma senha incorreta, a verificação é feita pelo sistema, que compara as informações inseridas pelo usuário com a base de dados. Caso essas informações não sejam condizentes, uma mensagem de erro é exibida e o usuário não consegue avançar para a próxima tela.

Figura 45 – Erro de autenticação: usuário não cadastrado em A e senha inválida em B. Autoria própria.



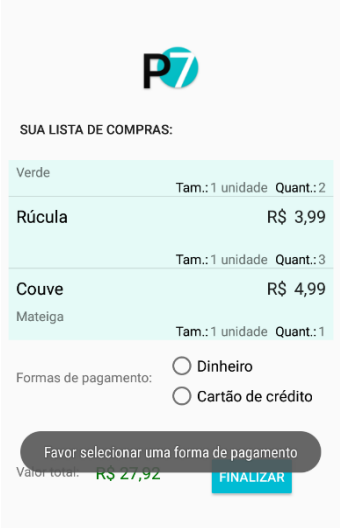
- Erro 4: existem 3 formas diferentes de interação nessa tela, sendo somente uma a correta. Para efeito de teste, as duas outras formas de interação foram tratas de maneiras diferentes. O usuário deve escrever o endereço de entrega, registrá-lo apertando o botão e selecionar o mercado de preferência. Ao simplesmente clicar no botão OK sem escrever nenhuma informação, o texto sombreado (*hint*) da caixa de texto muda para “Nenhum endereço registrado” (figura 44A). Essa forma de comunicação com o usuário se mostrou muito discreta nos testes, impedindo o entendimento pelo usuário e não é recomendada. A segunda abordagem é a exibição de uma caixa de alerta do tipo *Toast* que avisa de forma mais clara ao usuário que ele deve registrar o endereço. Nos testes, essa abordagem se mostrou muito mais eficaz (figura 44B). Somente após a confirmação do registro do endereço de entrega é que o usuário é permitido escolher o mercado de sua preferência (figura 44C).

Figura 46 – Tratamento de erros na tela de seleção de mercados: aviso discreto em A, aviso em destaque em B e confirmação de registro em C. Autoria própria.



- Erro 5: tal como nas telas anteriores, aqui é necessária a confirmação do método de pagamento, que é avisada caso o usuário tente finalizar a compra de forma prematura.

Figura 47 – Usuário não consegue finalizar a compra antes de selecionar a foram de pagamento. Autoria própria.



Além desses pontos, outros erros foram levantados pelos usuários, como ter um segundo campo de senha na tela de cadastro para validação, caso o usuário insira uma senha incorreta, e a necessidade de validar o endereço de entrega inserido como sendo um endereço real. No entanto, esses erros se mostraram de simples correção ou extremamente complexos, e foram considerados fora do escopo desse estudo.

5. Resultados

Como método de avaliação da solução, foram realizados testes com os voluntários. Se eles conseguissem realizar os seus pedidos com quaisquer itens da lista de 10 produtos cadastrados, de maneira bem-sucedida, em menos de 3 tentativas, o protótipo seria considerado adequado. A única orientação dada aos usuários foi sobre a de quantidade e os produtos disponíveis (tabela 2).

Não foi medida a etapa de cadastro, por ser considerada uma função extra que não faz parte do fluxo de interação do usuário comum. Essa etapa é muito tradicional em aplicativos e não apresenta nenhuma novidade ou desafio técnico que justifique a sua análise.

A avaliação foi feita de maneira binária para as seguintes perguntas:

- O usuário levou menos de 3 minutos para realizar as suas compras?

- O usuário precisou voltar de tela devido ao não entendimento do aplicativo?
- O usuário usou o aplicativo de maneira incorreta?
- O usuário conseguiu gravar seus produtos?
- O sistema reconheceu corretamente o que o usuário queria adquirir?

Foram realizados testes com 7 pessoas diferentes e os resultados obtidos são apresentados a seguir:

Figura 48 - Distribuição de respostas para os critérios 1 e 2 de avaliação do protótipo. Autoria própria.

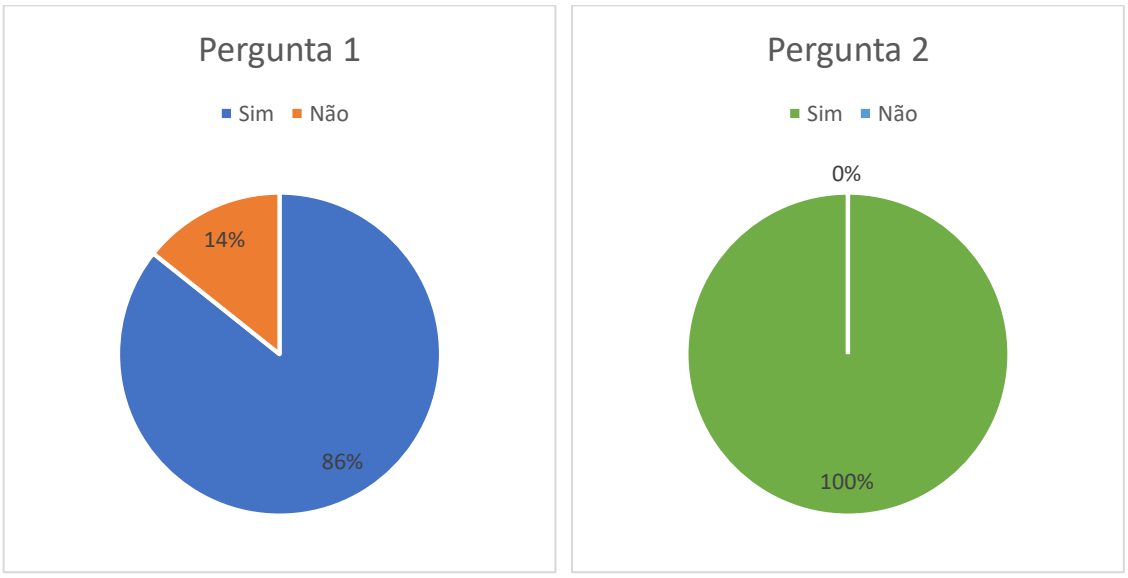


Figura 49 - Distribuição de respostas para os critérios 3 e 4. Autoria própria.

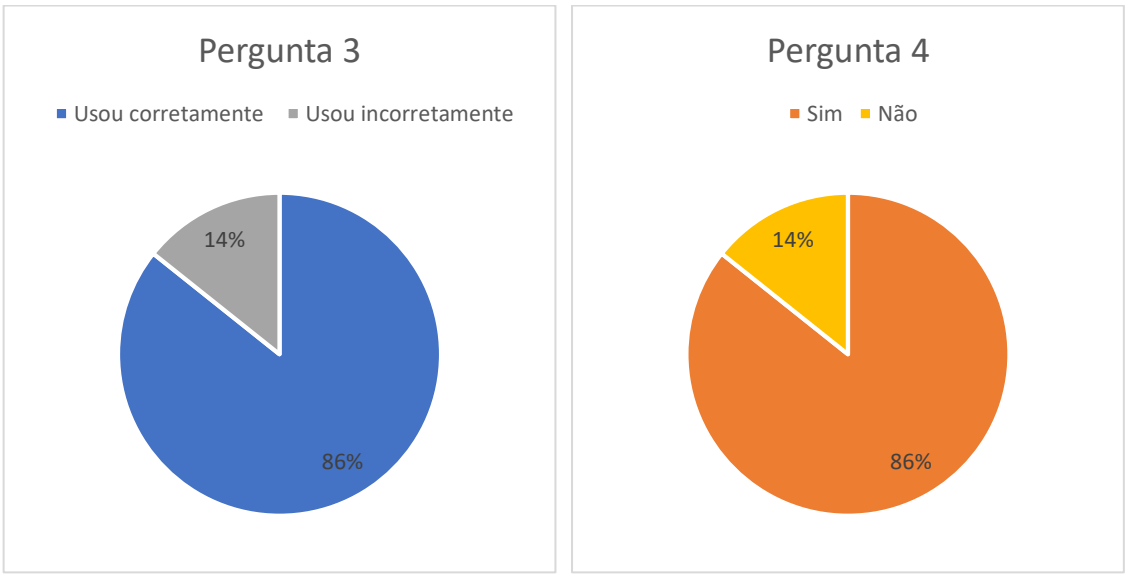
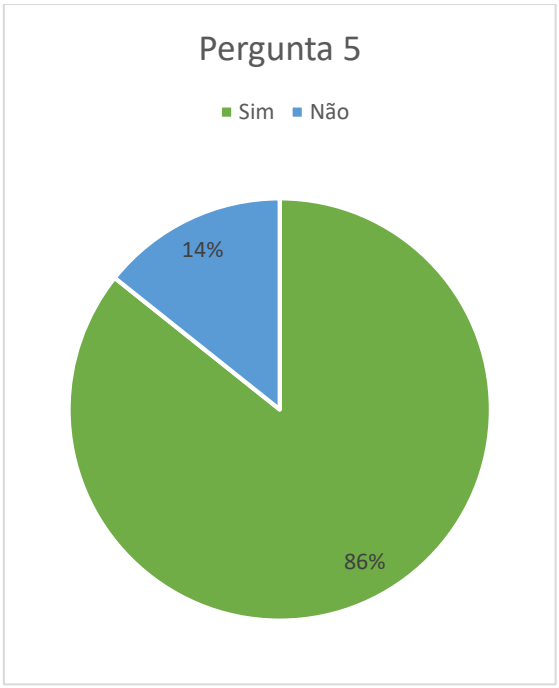


Figura 50 - Distribuição de respostas para o critério 5 de avaliação do protótipo. Autoria própria.



Todos os critérios de avaliação foram positivos e alcançados em mais de 70% dos casos. Além disso, a avaliação subjetiva de todos os usuários foi positiva de que usariam esse produto se ele estivesse disponível para o público. O único ponto de atenção é o fato de que todos os voluntários já tinham utilizado o aplicativo ao menos uma vez para testes durante a fase de desenvolvimento.

Na primeira pergunta, um usuário teve dificuldades com o reconhecimento de voz. O semantizador precisou de 5 tentativas para reconhecer corretamente as palavras “rúcula”, “alface” e “e 5”, traduzindo-as como “cúpula”, “mácula”, “a face” e “C5”. Isso fez com que o tempo de interação aumentasse significativamente, extrapolando o limite imposto.

Na terceira pergunta, um voluntário agiu de maneira instintiva, clicando no botão de *login* sem inserir nenhuma informação de *e-mail* e senha, levando o aplicativo a parar (item 4.3, figura 43). Ao ser questionado, ele explicou que como a maioria dos aplicativos já fazem a validação automática, ele agiu por reflexo. Esse erro é de simples correção e não implica na invalidação da solução.

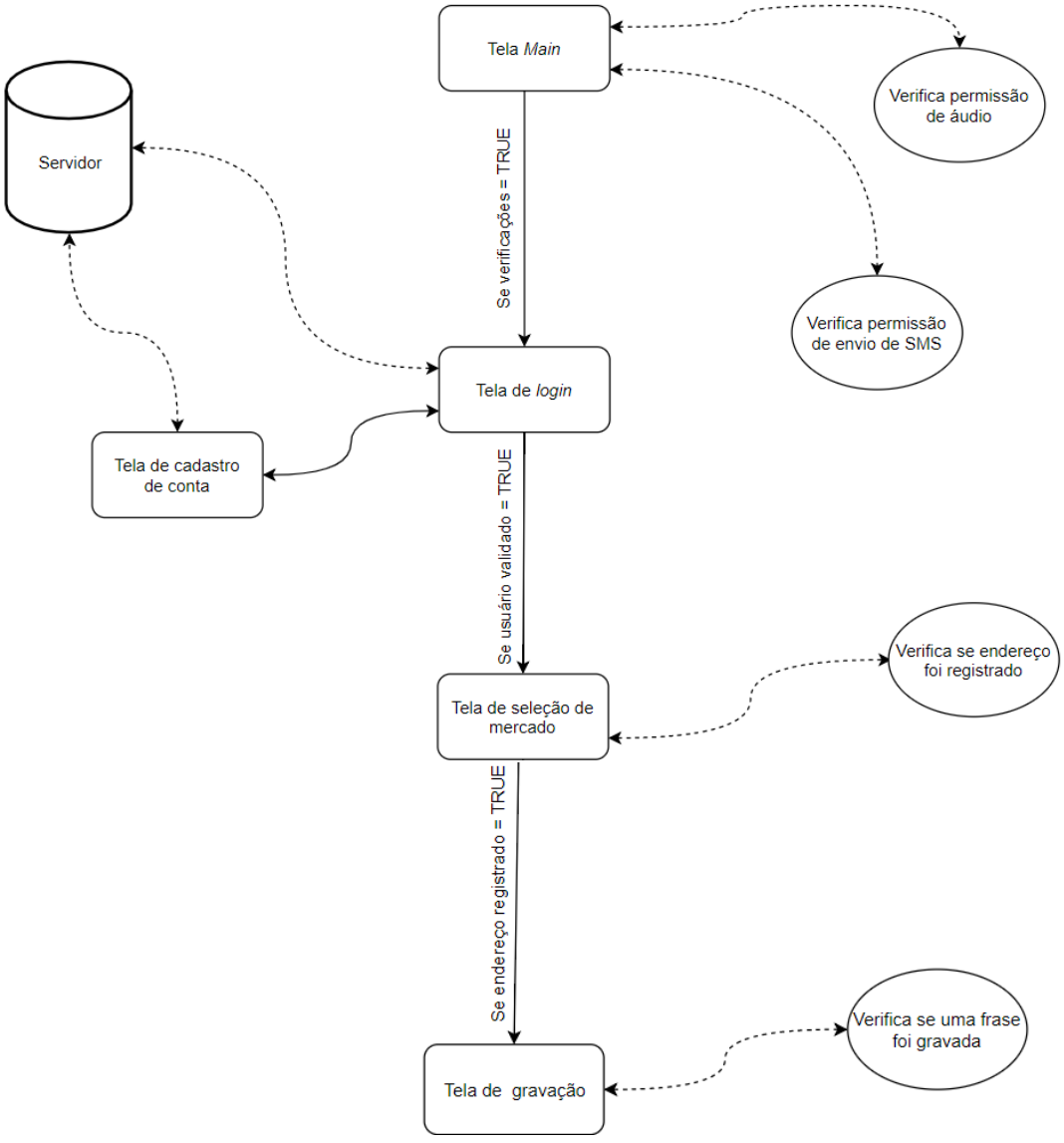
Na pergunta 4, o mesmo usuário teve dificuldade com o semantizador e, por iniciativa própria, abriu mão de adquirir um determinado produto. Esse dado é o mais importante e, portanto, tem mais peso que os demais. No entanto, foi

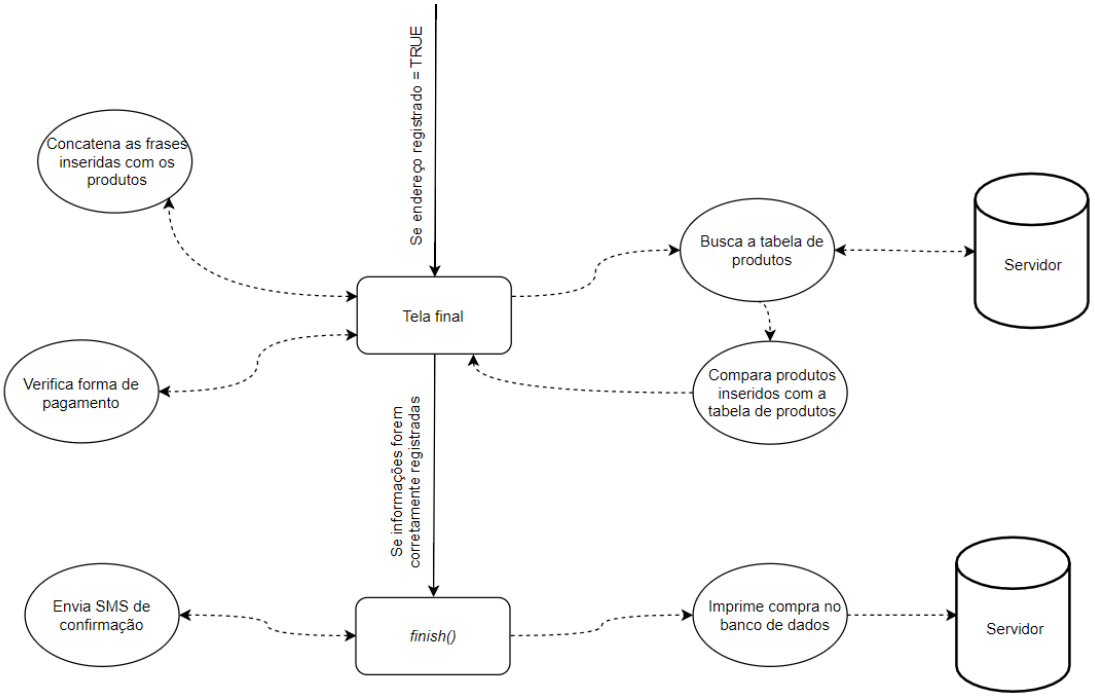
considerado que um único usuário é um número muito baixo para se avaliar a solução de maneira negativa. Isso impactou a pergunta número 5, onde foi considerado que o sistema não reconheceu a lista de compras corretamente porque não foi informado de todos os itens.

6. Estrutura

A estrutura final da aplicação, com as suas funções de verificação pode ser esquematizada da seguinte maneira:

Figura 51 - Fluxograma de comunicação do aplicativo





Observação: não se utilizou um diagrama UML devido ao tamanho da página, que inviabilizou a sua leitura. Considerou-se que para efeitos de comunicação, um fluxograma seria adequado.

7. Considerações Finais

Realizando os testes com os voluntários, pode-se perceber que o ponto de maior atenção deve ser o módulo TTS devido à sua grande quantidade de palavras e susceptibilidade à interpretação errada de palavras muito similares. Já a conversão de uma *string* em uma lista de produtos se mostrou trabalhosa, porém confiável, devido à metodologia de redes neurais.

Durante as avaliações com os voluntários, a percepção unânime foi a de que esse protótipo é sim uma alternativa melhor à necessidade de ir ao mercado, principalmente se o indivíduo possui algumas dificuldades, como não possuir um veículo ou não ter tempo suficiente para se deslocar até o local das compras. Além disso, todos os voluntários também afirmaram que possivelmente usariam esse aplicativo mesmo em comparação a outras soluções semelhantes já existentes no mercado que não possuem o módulo de voz.

A arquitetura utilizada permite uma rápida expansão da quantidade de produtos disponíveis para compra e é de fácil atualização. Ao inserir todos os produtos em um banco de dados *online*, o responsável pelo *back-end* consegue inserir e atualizar quaisquer produtos, com impacto imediato em todos os usuários, sem a necessidade de uma atualização. Por fim, o sistema de reconhecimento de palavras-chave permite a criação e um bloco de código único, semelhante para todos os produtos. Diferindo apenas nas palavras-chave, esse bloco de código, apesar de grande, é de entendimento simplificado, sem a necessidade de um programador especializado para atualizá-lo.

De maneira geral o resultado foi muito positivo e para trabalhos futuros, recomenda-se o aperfeiçoamento do modelo de *machine-learning* implementado (item 3.7) para o reconhecimento de padrões de compra e a expansão da complexidade do reconhecimento de produtos, marcas e quantidades (item 2.9 e 3.7).

8. Anexos

ANEXO A – DOCUMENTAÇÃO DE IMPLEMENTAÇÃO DO FIREBASE [37]

Adicionar o Firebase ao seu projeto do Android

Pré-requisitos

- um dispositivo com Android 4.0 (Ice Cream Sandwich) ou uma versão mais recente e o Google Play Services 15.0.0 ou posterior
- a versão mais recente do Android Studio

Se você ainda não tiver um projeto do Android Studio e quiser testar um recurso do Firebase, faça o download de um de nossos exemplos do guia de início rápido. Se você estiver usando um exemplo para início rápido, lembre-se de coletar o código do aplicativo do arquivo *build.gradle,* que geralmente fica na pasta *app/*, do módulo do seu projeto. Você precisará do nome desse pacote na próxima etapa. Observação: se você estiver fazendo upgrade de uma versão 2.X do SDK do Firebase, consulte nosso guia de upgrade para Android para começar.

Adicionar o Firebase ao app

Se você estiver usando o Android Studio versão 2.2 ou posterior, o Firebase Assistente é a maneira mais simples de conectar seu app ao Firebase. O Assistente pode conectar um projeto existente ou criar um novo para você com todas as dependências do Gradle necessárias. Se você usar uma versão mais antiga do Android Studio ou tiver uma configuração de projeto mais complexa, ainda poderá adicionar manualmente o Firebase ao seu app.

Usar o Firebase Assistente

Para abrir o Firebase Assistente no Android Studio:

Clique em “Ferramentas > Firebase” para abrir a janela “Assistente”. Clique para expandir um dos recursos listados (por exemplo, Analytics) e clique no link do tutorial fornecido (por exemplo, "Registrar um evento do Analytics"). Clique no botão “Conectar ao Firebase” para se conectar ao Firebase e adicionar

o código necessário ao seu app. Pronto! Você pode pular para as próximas etapas.

Para criar um projeto do Firebase:

Crie um projeto no Console do Firebase se ainda não tiver um. Clique em Adicionar projeto. Se você já tiver um projeto do Google associado ao seu aplicativo para dispositivos móveis, selecione-o no menu suspenso Nome do projeto. Caso contrário, insira um nome de projeto para criar um novo.

Opcional: edite o código do projeto. Ele receberá automaticamente um código exclusivo que será usado em recursos do Firebase visíveis ao público, como URLs de bancos de dados e seu subdomínio do Firebase Hosting. Você pode alterá-lo agora se quiser usar um subdomínio específico. Siga as demais etapas de configuração e clique em Criar projeto, ou Adicionar Firebasese você estiver usando um projeto existente, para começar a provisionar recursos para o projeto. Isso costuma levar alguns minutos. Quando o processo for concluído, você será levado à visão geral do projeto.

Agora que você tem um projeto, adicione seu app para Android a ele:

Clique em Adicionar o Firebase ao app para Android e siga as etapas de configuração. Se você estiver importando um projeto do Google, isso pode ocorrer automaticamente. Basta fazer o download do arquivo de configuração. Quando solicitado, digite o nome do pacote do seu app. É importante inserir o nome do pacote usado pelo seu app. Essa configuração só pode ser feita quando você adiciona um app ao seu projeto do Firebase.

Durante o processo, você fará o download de um arquivo *google-services.json*. Você pode fazer o download desse arquivo novamente a qualquer momento. Depois de adicionar o código de inicialização, execute seu aplicativo para enviar ao Console do Firebase a confirmação de que você instalou o Firebase com sucesso.

Observação: se você tiver muitas variantes de versão com diferentes nomes de pacote definidos, adicione cada app ao projeto no Console do Firebase.

Adicionar o SDK

Se você quiser integrar as bibliotecas do Firebase a um de seus projetos, precisará executar algumas tarefas básicas para preparar o projeto do Android Studio. Talvez você já tenha feito isso quando adicionou o Firebase ao seu app. Primeiro, adicione regras ao seu arquivo *build.gradle* no nível raiz para incluir o plug-in google-services e o repositório Maven do Google:

```
buildscript {
    // ...
    dependencies {
        // ...
        classpath 'com.google.gms:google-services:4.1.0' // google-services
plugin
    }
}

allprojects {
    // ...
    repositories {
        // ...
        google() // Google's Maven repository
    }
}
```

Em seguida, no arquivo Gradle do módulo (geralmente *app/build.gradle*), adicione a linha *apply plugin* na parte inferior do arquivo para ativar o plug-in do Gradle:

```
apply plugin: 'com.android.application'

android {
  // ...
}

dependencies {
  // ...
  implementation 'com.google.firebase:firebase-core:16.0.4'

  // Getting a "Could not find" error? Make sure you have
  // added the Google maven respository to your root build.gradle
}

// ADD THIS AT THE BOTTOM
```

```
apply plugin: 'com.google.gms.google-services'
```

Você também precisa adicionar as dependências dos SDKs do Firebase que quer usar. Recomendamos começar com *com.google.firebase:firebase-core*, que fornece a funcionalidade do Google Analytics para Firebase. Consulte a lista de bibliotecas disponíveis.

Bibliotecas disponíveis

As seguintes bibliotecas estão disponíveis para os diversos recursos do Firebase.

Linha de dependência do Gradle	Serviço
com.google.firebase:firebase-core:16.0.4	Analytics
com.google.firebase:firebase-database:16.0.3	Realtime Database
com.google.firebase:firebase-firestore:17.1.1	Cloud Firestore
com.google.firebase:firebase-storage:16.0.3	Storage
com.crashlytics.sdk.android:crashlytics:2.9.5	Crashlytics
com.google.firebase:firebase-auth:16.0.4	Authentication
com.google.firebase:firebase-messaging:17.3.3	Cloud Messaging
com.google.firebase:firebase-config:16.0.1	Configuração remota
com.google.firebase:firebase-invites:16.0.4	Invites e Dynamic Links
com.google.firebase:firebase-ads:16.0.1	AdMob
com.google.firebase:firebase-appindexing:16.0.2	Indexação de apps
com.google.firebase:firebase-perf:16.1.2	Monitoramento de desempenho
com.google.firebase:firebase-functions:16.1.1	SDK de cliente do Cloud Functions para Firebase
com.google.firebase:firebase-ml-vision:17.0.1	Kit de ML (Vision)
com.google.firebase:firebase-ml-model-interpreter:16.2.2	Kit de ML (modelo personalizado)

Observação: *com.google.firebase:firebase-core* é um alias recomendado para a biblioteca *com.google.firebase:firebase-analytics*.

9. Apêndices

APÊNDICE A – TABELA DE REFERÊNCIA DE PRODUTOS

Produto	Tipo	Tamanho	Marca	Preço
Alface	Crespa Hidropônica	1 unidade		3,49
Alface	Americana	1 unidade		6,99
Alface	Crespa	1 unidade		2,69
Alface	Lisa	1 unidade		1,99
Alface	Mimosa	1 unidade		1,89
Alface	Mimosa Hidropônica	1 unidade		3,49
Repolho	Verde	1 unidade		1,99
Repolho	Verde Hidropônico	1 unidade		4,99
Repolho	Roxo	1 unidade		4,99
Espinafre		1 unidade		4,99
Rúcula		1 unidade		3,99
Couve	Mateiga	1 unidade		4,99
Abacate		1 unidade		1,99
Abacaxi	Pérola	1 unidade		4,99
Ameixa		1 unidade		1,65
Banana	Prata	1 unidade		0,82
Banana	Nanica	1 unidade		0,68
Banana	Terra	1 unidade		1,61
Banana	Maçã	1 unidade		0,84
Banana	Ouro	1 unidade		1,40
Caqui	Rama Forte	1 unidade		2,70
Goiaba	Vermelha	1 unidade		1,40
Laranja	Pera	1 unidade		0,60
Laranja	Seleta	1 unidade		0,47
Laranja	Lima	1 unidade		0,66
Limão	Tahiti	1 unidade		0,38
Maçã	Argentina	1 unidade		1,93
Maçã	Fuji	1 unidade		1,84
Mamão	Formosa	1 unidade		7,49
Mamão	Papaya	1 unidade		1,85
Manga	Palmer	1 unidade		3,89
Manga	Haden	1 unidade		2,15
Maracujá		1 unidade		1,80
Melão	Amarelo	1 unidade		5,99
Mexerica	Ponkan	1 unidade		1,40
Mexerica	Carioca	1 unidade		0,90
Pêra	Willians	1 unidade		1,98
Pêra	Danjou	1 unidade		1,20
Uva	Niágara	1kg		13,00
Uva	Itália	1kg		7,00
Kiwi	Verde	1 unidade		1,56
Coco	Verde	1 unidade		2,49
Pêssego		1 unidade		1,40
Tomate	Italiano	1 unidade		0,48
Tomate	Carmen	1 unidade		0,72
Tomate	Holandês	1 unidade		1,00
Cebola		1 unidade		0,50
Cebola	Roxa	1 unidade		0,99
Cenoura		1 unidade		0,45
Batata	Lavada	1 unidade		0,81
Batata Doce	Rosada	1 unidade		1,04
Pepino	Caipira	1 unidade		0,56
Pepino	Japonês	1 unidade		0,92
Pepino	Comum	1 unidade		0,90
Abobrinha	Italiana	1 unidade		1,02
Abobrinha	Brasileira	1 unidade		1,68
Abóbora	Moranga	1 unidade		7,96
Abóbora	Paulista	1 unidade		3,24
Berinjela		1 unidade		1,48

Alho		1 unidade		1,29
Beterraba		1 unidade		0,60
Chuchu		1 unidade		0,60
Inhame		1 unidade		0,75
Mandioca		1 unidade		0,60
Pimentão	Amarelo	1 unidade		2,25
Pimentão	Vermelho	1 unidade		2,25
Pimentão	Verde	1 unidade		1,00
Jiló		1 unidade		0,50
Vagem	Holandesa	1kg		10,00
Ervilha	Torta	1kg		20,00
Rabanete		1 unidade		4,00
Ovo	Branco	30 unidades	Katayama	9,99
Ovo	Branco	10 unidades	Katayama	4,99
Ovo	Branco	12 unidades	Katayama	5,49
Ovo	Branco	6 unidades	Katayama	2,69
Ovo	Vermelho	10 unidades	Pufa Aja	7,99
Ovo de Codorna		30 unidades	Sol Nascente	4,49
Pão de Forma		500g	Panco	4,99
Pão de Forma		500g	Wickbold	4,79
Pão de Forma	Integral	400g	Visconti	4,49
Pão de Forma		400g	Visconti	2,99
Pão de Forma		500g	Pullman	6,99
Pão de Forma	Integral	500g	Nutrella	8,59
Bisnaga		300g	Panco	4,99
Bisnaga		300g	Pullman	4,79
Pão de Hot Dog	10 pães	500g	Juliana	4,99
Pão de Hot Dog	4 pães	200g	Wickbold	4,99
Pão de Hamburguer		200g	Wickbold	4,99
Pão de Hamburguer	Gergelim	200g	Wickbold	4,99
Pão de Hamburguer	Integral	200g	Wickbold	5,49
Pão de Alho		300g	Zinho	7,59
Pão de Alho		400g	Santa Massa	8,99
Torrada	Integral	160g	Bauduco	2,99
Torrada		160g	Bauduco	2,99
Torrada	Multigrãos	160g	Bauduco	2,99
Creme de Chocolate		140g	Nutella	7,99
Creme de Chocolate		380g	Ovomaltine	50,00
Creme de Chocolate		63g	Io-Io	2,99
Geleia de Frutas		320g	Queensberry	15,99
Geleia de Frutas	Diet	280g	Queensberry	17,99
Geleia de Morango		320g	Queensberry	15,99
Geleia de Morango	Diet	320g	Queensberry	17,99
Geleia de Damasco		320g	Queensberry	15,99
Café		500g	Três Corações	9,59
Café	Extraforte	500g	Três Corações	7,99
Café	Extraforte	500g	Melitta	10,99
Café		500g	Melitta	10,99
Achocolatado em Pó		400g	Nescau	4,80
Achocolatado em Pó		800g	Toddy	9,99
Achocolatado em Pó		200g	Nescau	4,59
Achocolatado em Pó		400g	Toddy	4,99
Suplemento	Chocolate	380g	Sustagem kids	17,99
Suplemento	Morago	380g	Sustagem kids	17,99
Suplemento	Baunilha	380g	Sustagem kids	17,99
Suplemento Zero	Morango	400g	Linea	25,99
Aveia		200g	Quaker	2,99
Aveia		450g	Quaker	6,99
Aveia	Farinha	170g	Yoki	2,79
Aveia	Farelo	200g	Quaker	2,59
Cereal	Multicereais	600g	Mucilon Nestlé	14,99
Cereal	Chocolate	270g	Nescau	6,99
Cereal		730g	Sucrilhos Kellogs	21,99
Cereal		300g	Snow Flakes	6,99
Bolo	Laranja	300g	Panco	5,79
Bolo	Chocolate	370g	Casa Suíça	13,59
Bolo	Mesclado	250g	Casa Suíça	6,29
Bolo	Laranja	250g	Pullman	5,99

Bolo	Laranja	370g	Casa Suíça	13,59
Bolo	Coco	250g	Pullman	5,99
Bolo	Chocolate	80g	Ana Maria	2,19
Bolo	Baunilha	80g	Ana Maria	2,19
Mistura para Bolo	Chocolate	400g	Italac	2,59
Mistura para Bolo	Chocolate	400g	Renata	3,99
Mistura para Bolo	Bolinha de chuva	260g	Dona Benta	3,99
Mistura para Bolo	Limão	400g	Italac	2,59
Mistura para Bolo	Coco	400g	Renata	3,99
Paçoca		320g	Santa Helena	12,99
Margarina		500g	Doriana	3,79
Iogurte	com Mel	170g	Batavo	1,59
Iogurte	Frutas Vermelhas	130g	Carolina	1,99
Leite Fermentado	6 unidades	80g	Yakult	6,99
Iogurte	Natural	170g	Nestlé	1,89
Presunto	Fatiado	100g	Sadia	2,60
Presunto	Fatiado	100g	Ceratti	1,70
Salsicha		400g	Sadia	4,80
Peito de Frango	Fatiado	100g	Ceratti	2,39
Peito de Peru	Fatiado	100g	Sadia	4,40
Mussarela	Fatiado	100g	Tirolez	2,69
Mussarela	Fatiado	100g	Três Marias	2,69
Prato		200g	Soltíssimo Sadia	8,49
Parmesão	Ralado	50g	Vigor	2,99
Requeijão		200g	Batavo	3,99
Requeijão		200g	Vigor	4,49
Cream Cheese	Light	150g	Danúbio	5,49
Cream Cheese		150g	Danúbio	5,49
Creme de Ricota	Light	250g	Tirolez	5,49
Creme de Ricota	Light	150g	Quatá	4,49
Arroz	Tipo 1	5kg	Camil	13,49
Arroz	Tipo 1	5kg	Namorado	9,99
Arroz	Tipo 1	5kg	Tio João	15,49
Arroz	Integral Parabolizado	1kg	Solito	3,99
Feijão	Carioca	1kg	Camil	3,99
Feijão	Carioca	1kg	Super Máximo	3,79
Feijão	Carioca	1kg	Broto Legal	4,49
Grão de Bico		500g	Yoki	10,99
Lentilha		500g	Yoki	7,59
Trigo para Kibe		500g	Yoki	3,69
Canjica	Milho	500g	Yoki	3,99
Açúcar	Refinado	1kg	União	2,29
Açúcar	Refinado	1kg	Da Barra	1,99
Acúcar	Orgânico	1kg	Denerara	4,99
Óleo	Soja	900ml	Liza	3,19
Óleo	Soja	900ml	Leve	2,99
Óleo	Canola	900ml	Purilev	8,49
Azeite		500ml	Olinda	4,99
Azeite		500ml	Carmelita	6,99
Sopa	Mandioquinha com Cebola	17g	Vono	1,59
Sopa	Batata com Carne	18g	Vono	1,59
Sopa	Milho com Frango	18g	Vono	1,59
Sopa	Caldo Verde	18g	Vono	1,59
Sopão	Feijão com Macarrão	194g	Vono	6,99
Sal	Refinado	1kg	Cisne	2,39
Sal	Refinado	1kg	Lebre	1,69
Sal	Marinho	1kg	Atlantis	4,99
Sal	Grosso	1kg	Cisne	2,79
Sal	com Saleiro	500g	Cisne	5,99
Farinha	Trigo	1kg	Dona Benta	3,29
Farinha	Trigo	1kg	Lili	2,19
Farofa	Temperada	500g	Yoki	3,99
Fubá	Mimoso	500g	Yoki	2,29
Papel toalha	Folha simples	2 unidades	Mascot	3,99
Papel toalha	Folha simples	2 unidades	Yuri	3,99
Papel alúmínio	30cm x 7,5 m	1 unidade	Kiko	3,49
Filtro de café	Papel	30 unidades	Jovita	2,59
Macarrão Instantâneo	Carne	85g	Adria	0,69

Macarrão Instantâneo	Carne	85g	Nissin Lámen	1,29
Macarrão	Espaguete	500g	Adria	2,19
Macarrão	Parafuso	500g	Adria	1,95
Macarrão	Espaguete	500g	Petybon	2,09
Macarrão	Parafuso	500g	Renata	2,99
Massa para Pastel		300g	Massa Leve	5,99
Capeletti	Frango	400g	Massa Leve	7,29
Capeletti	Carne	400g	Massa Leve	7,29
Molho de Tomate		340g	Salsaretti	1,69
Molho de Tomate		340g	Quero	1,69
Molho de Tomate		340g	Pomarola	1,69
Molho Branco		260g	Fugini	4,99
Molho Yakissoba		500ml	Sakura	13,99
Atum	Sólido	170g	Gomes da Costa	8,49
Atum	Natural Light Pedaços	170g	Coqueiro	6,99
Atum	Ralado	170g	Gomes da Costa	4,59
Atum	Ralado	170g	Coqueiro	4,99
Milho	Lata	200g	Quero	1,39
Milho	Lata	170g	Bonduelle	2,29
Ervilha	Congelada	300g	Pratigel	4,29
Ervilha	Lata	200g	Quero	1,59
Ervilha	Lata	200g	Predilecta	1,49
Ervilha	Lata	200g	Bonduelle	1,49
Seleta	Lata	200g	Quero	2,49
Seleta	Caixinha	200g	Quero	2,79
Seleta	Lata	300g	Bonduelle	3,29
Palmito	Inteiro	300g	Castelo	17,99
Palmito	Inteiro	300g	Pupunha	13,99
Grão de Bico	Lata	200g	Bonduelle	3,29
Pepino	Conserva	300g	Hemmer	11,99
Alcaparra	Conserva	60g	La Pastina	8,99
Azeitona	Verde Fatiada	330g	Rivoli	3,99
Azeitona	Verde	500g	Rivoli	9,99
Azeitona	Verde	500g	Couvert	6,99
Azeitona	Verde sem Caroço	160g	Violetera	4,79
Azeitona	Verde sem Caroço	155g	Cepêra	7,99
Cogumelo	Fatiado	100g	Hemmer	7,99
Azeite	Extra virgem	500ml	Andorinha	17,49
Azeite	Extra virgem	500ml	Gallo	18,69
Azeite	Tradicional	500ml	Gallo	16,99
Ketchup	Picante	397g	Heinz	8,49
Ketchup	Tradicional	400g	Quero	4,49
Ketchup	Tradicional	380g	Hellmanns	7,99
Mostarda	Tradicional	170g	Hellmanns	6,99
Mostarda	Tradicional	200g	Arrifana	3,29
Mostarda	Apimentada	496g	Heinz	19,99
Maionese	Tradicional	500g	Hellmanns	5,49
Tempero pronto	Sabores do nordeste	60g	Sazon	3,29
Tempero pronto	Caldo de galinha	57g	Knorr	1,69
Tempero pronto	Para arroz branco	60g	Sazon	3,29
Tempero pronto	Carnes vermelhas	60g	Sazon	3,29
Tempero pronto	Carnes vermelhas	57g	Knorr	1,69
Tempero pronto	Canela em pó	8g	Kitano	1,49
Tempero pronto	Orégano	10g	Kitano	2,49
Vinagre	Colorido	750ml	Castelo	2,49
Vinagre	Maça	750ml	Castelo	4,29
Vinagre	Balsâmico	500ml	Castelo	12,99
Pimenta	Do reino com moedor	42g	Kook	19,49
Pimenta	Do reino refil	45g	Kook	7,99
Cebolinha		1 unidade		2,99
Cheiro Verde		1 unidade		2,99
Coentro		1 unidade		2,99
Coentro	Orgânico	1 unidade		3,49
Gengibre		1 unidade		1,75
Chocolate	Com amendoim	32g	Chokito	1,79
Chocolate	Ao leite	16g	Baton	0,99
Chocolate zero	Meio amargo	30g	Linea	5,49
Bolacha	Recheada de chocolate	130g	Passatempo	1,39

Bolacha	Recheada de morango	136g	Trakinas	1,98
Bolacha	Maisena	400g	Zabet	3,69
Pipoca	Natural	50g	Yoki	1,29
Pipoca	Com manteiga	130g	Yoki	3,49
Pipoca de panela	Natural	500g	Yoki	2,69
Água	Natural	5l	Crystal	7,49
Água	Natural	500ml	Bonafont	1,79
Água	Natural	1,5l	Minalba	2,09
Água	Com Gás	1,27l	Bonafont	3,29
Água de coco	Natural	1l	Puro Coco	5,99
Água de coco	Natural	1l	Obrigado	7,49
Água de coco	Natural	200ml	Kero Coco	2,69
Chá	Erva Cidreira	15 unidades	Matte Leão	5,29
Chá	Canela	25 unidades	Matte Leão	5,49
Chá	Lichia	1l	Maguary	4,99
Chá Zero	Limão	300ml	Matte Leão	3,99
Suco	Pó de laranja	25g	Tang	0,99
Suco	Pó de uva	25g	Tang	0,99
Suco	Laranja 50%	1l	Del Valle	4,99
Refrigerante	Cola	2l	Coca-Cola	6,99
Refrigerante	Cola	200ml	Coca-Cola	1,55
Refrigerante Zero	Laranja	1,5l	Fanta	5,49
Leite	Integral	1l	Italac	2,99
Leite	Desnatado	1l	Paramalat	3,09
Leite	Desnatado	1l	Italac	2,49
Leite	Integral	1l	Ninho	4,99
Cerveja	Pilsen	269ml	Skol	2,29
Cerveja	Pilsen	350ml	Heineken	3,99
Cerveja	Pilsen	350ml	Eisenbhan	3,99
Cerveja	Pilsen	473ml	Skol	3,49
Cerveja	Pilsen	350ml	Amstel	3,28
Cerveja	Stout chocolate	600ml	Baden Baden	15,49
Cerveja	Red ale	600ml	Baden Baden	15,49
Cachaça	Maracujá Ice	275ml	51	3,69
Cachaça	Tradicional	965ml	51	6,49
Cachaça	Tradicional	910ml	Velho Barreiro	7,19
Vodka	Tradicional	1l	Orloff	27,90
Vodka	Red	1l	Smirnoff	35,99
Vodka	Ice	269ml	Smirnoff	5,29
Vodka	Tradicional	980ml	Skyy	39,90
Saquê	Tradicional	750ml	Daiki	17,90
Saquê	Tradicional	670ml	Jun Daiti	32,90
Conhaque	Tradicional	900ml	Dreher	13,99
Conhaque	Tradicional	1l	Domecq	32,90
Tequila	Tradicional	750ml	Sauza	79,90
Rum	Ouro	980ml	Bacardi	37,90
Rum	Prata	980ml	Bacardi	37,90
Whisky	Double black	1l	Johnnie Walker	199,90
Whisky	Gold Label	750ml	Johnnie Walker	259,00
Whisky	Red Label	1l	Johnnie Walker	119,90
Açaí Zero	Tradicional	2l	Frooty	43,90
Açaí	Com morango	2l	Frooty	39,90
Açaí	Tradicional	2l	Frooty	39,90
Sorvete	Baunilha com chocolate	1,5l	Nestlé	10,90
Sorvete	Napolitano	1,5l	Kibon	17,99
Sorvete	Flocos	1,5l	Kibon	17,99
Sorvete	Napolitano	1,5l	Nestlé	13,90
Hambúrguer	Bovino	672g	Texas Burguer	10,99
Hambúrguer	Bovino	672g	Sadia	14,90
Hambúrguer	Frango	672g	Sadia	9,90
Torta	Chocolate	470g	Miss Daisy	39,90
Torta	Mousse chocolate	500g	Miss Daisy	32,90
Torta	Holandesa	470g	Miss Daisy	39,90
Empanado	Frango com queijo	100g	Seara	1,29
Empanado	Nuggets de frango com queijo	300g	Sadia	6,99
Empanado	Frango com presunto	120g	Seara	3,29
Pão de queijo	Tradicional de forno	400g	Forno de Minas	9,89
Bovina	Filé mignon	1kg		45,99

Bovina	Coxão mole	1kg		24,90
Bovina	Contra filé	1kg		27,90
Bovina	Patinho	1kg		19,99
Bovina	Lagarto	1kg		23,90
Bovina	Musculo	1kg		19,00
Bovina	Coxão duro	1kg		18,99
Suína	Bisteca	1kg		9,90
Suína	Costela	1kg		10,90
Suína	Pernil	1kg		9,90
Suína	Linguiça toscana	1kg		9,90
Frango	Peito	1kg		9,90
Frango	Coxa	1kg		7,60
Frango	Asa	1kg		13,00
Frango	Coração	1kg		23,90
Frango	a Passarinho	1kg		7,90
Bacalhau	Desfiado	1kg		21,90
Bacalhau	Zarbo	1kg		36,90
Pasta de dente	Anti cárie	70g	Oral-B	1,69
Pasta de dente	Menta	90g	Colgate	2,49
Pasta de dente	Hortelã	90g	Colgate	2,49
Escova de dente	Infantil	1 unidade	Colgate	3,49
Escova de dente	Luminous White	2 unidades	Colgate	9,69
Escova de dente	Twister	3 unidades	Colgate	13,99
Fio dental	Menta	100m	Johnson & Johnson	8,49
Fio dental	Tradicional	100m	Dental Clean	6,49
Antiséptico bucal	Controle do tátaro	500ml	Listerine	21,90
Antiséptico bucal	Zero álcool sabor menta	500ml	Listerine	16,79
Antiséptico bucal	Infinity Plax	250ml	Colgate	9,69
Papel higiênico	Folha dupla	18 unidades	Neve	27,90
Papel higiênico	Folha dupla	12 unidades	Personal	13,79
Papel higiênico	Folha simples	16 unidades	Personal	9,90
Absorvente	Normal com abas	8 unidades	Intimus	3,99
Absorvente	Interno	10 unidades	OB	7,49
Absorvente	Seca com abas	8 unidades	Sempre livre	3,59
Haste flexível	Tradicional	150 unidades	Cotonete	6,99
Lenço	Folha dupla	50 unidades	Softy's	3,29
Desodorante	Aerosol	150ml	Nivea	14,69
Desodorante	Roll On Silver Protect	50ml	Nivea	6,99
Desodorante	Aerosol V8	90g	Rexona	11,99
Desodorante	Aerosol Antibacterial + Invisible	150ml	Rexona	11,99
Desodorante	Aerosol feminino	56g	Rexona	11,99
Desodorante	Aerosol Clean	89g	Dove	14,69
Hidratante	Tradicional Soft	200ml	Johnson & Johnson	11,99
Hidratante	com Camomila	400ml	Johnson & Johnson	18,49
Hidratante	Tradicional	100ml	Leite de Rosas	4,79
Hidratante	Romã e uva	400ml	Johnson & Johnson	11,49
Protetor Solar	50 fps	200ml	Sundown	43,90
Protetor Solar	30 fps	200ml	Sundown	37,90
Protetor Solar	30 fps	200ml	Nivea	54,90
Repelente	Elétrico 45 noites	33ml	Raid	10,99
Repelente	Spray	200ml	Off!	19,90
Repelente	Loção	100ml	Repelx	12,49
Sabonete	Hidratante	90g	Nivea	1,79
Sabonete	Leite e pétalas rosas	90g	Palmoçive	0,99
Sabonete	Magical Spell	85g	Lux	1,39
Sabonete	Amêndoas	90g	Johnson & Johnson	1,39
Talco	Menta	100g	Tenys Pé	8,49
Talco	Bebê	100g	Granado	11,49
Creme de barbear	Hidratação	65g	Bozano	4,99
Creme de barbear	Pele sensível	65g	Bozano	4,99
Lâmina de barbear	Depilação feminina	2 unidades	Bic	7,49
Lâmina de barbear	Masculino	2 unidades	Bic	6,49
Esponja	para Banho	1 unidade	Ponjita	2,49
Shampoo	Recuperação feminino	200ml	Dove	10,49
Shampoo	Seda hidratação anticaspa	325ml	Seda	9,49
Shampoo	Controle de queda anticaspa	400ml	Clear	14,79
Condicionador	Recuperaçãoextrema feminino	200ml	Dove	12,49
Condicionador	Neutro	500ml	Neutrox	8,99

Condicionador	Supreme control	400ml	Elseve	21,90
Gel	Fixação extra forte	250g	Hard Hair	6,49
Gel	Fixação média	250g	NYLooks	4,49
Tintura	Louro médio	1 unidade	Biocolor	8,98
Tintura	Louro natural	1 unidade	Loreal	32,90
Tintura	Preto onix	1 unidade	Loreal	32,90
Escova de cabelo	Redondo	1 unidade	Condor	13,99
Pente	Fino	3 unidades	Condor	10,49
Lenço demaquilante	Tradicional	25 unidades	Loreal	32,90
Desinfetante	Multiuso	500ml	Veja	4,49
Desinfetante	Perfumes Suave	500ml	Veja	2,99
Desinfetante	Fresh limão	1l	Ajax	9,49
Limpa vidro	Tradicional sem pulverizador	500ml	Vidrex	9,99
Limpa vidro	Tradicional com pulverizador	500ml	Vidrex	10,99
Limpa vidro	Tradicional	500ml	Mr Muscullo	6,49
Cera	Líquido incolor	750ml	Poliflor	15,49
Cera	Líquido incolor	750ml	Bravo	12,99
Cera	Líquido incolor	750ml	Nugget	10,99
Lustra móveis	Líquido	200ml	Destaque	5,29
Lustra móveis	Líquido aroma de lavanda	500ml	Poliflor	13,99
Lustra móveis	Líquido aroma de lavanda	500ml	Bravo	14,49
Álcool	Tradicional	1l	Coperalcool	6,99
Álcool	Tradicional	500ml	Coperalcool	4,29
Álcool	46% eucalipto	750ml	Coperalcool	7,49
Vassoura	Sem o cabo	1 unidade	Bettanim	9,98
Vassoura	Sem o cabo	1 unidade	Bettanim	11,99
Vassoura	Sem o cabo	1 unidade	Condor	12,99
Rodo	Com o cabo	1 unidade	Bettanim	12,99
Rodo	Sem o cabo	1 unidade	Bettanim	29,99
Rodo	Sem o cabo	1 unidade	Bettanim	37,90
Saco de lixo	30 litros	10 unidades	Dover Roll	3,49
Saco de lixo	Para banheiro	50 unidades	Dover Roll	12,49
Saco de lixo	50 litros	10 unidades	Dover Roll	4,79
Aromatizante	Refil	12 ml	Glade	5,49
Inseticida	Aero multiuso	300ml	SBP	9,99
Inseticida	Aero multiuso	395ml	Baygon	8,48
Removedor	Perfumado	500ml	Zulu	12,49
Removedor	Perfumado	1l	Zulu	21,90
Removedor	Perfumado	500ml	Bufalo	11,99
Lava roupas	Em pó	1kg	Omo	8,49
Lava roupas	Líquido concentrado	1,2l	Ariel	14,99
Lava roupas	Líquido concentrado	5l	Omo	43,99
Tira Mancha	Em pó	450g	Vanish	19,99
Tira Mancha	Roupas brancas	450ml	Omo	5,29
Tira Mancha	Refil em pó	420g	Utile	10,99
Alvejante	Líquido 3x1	1l	Super Candida	3,49
Alvejante	Refil	500ml	Vanish	6,99
Alvejante	Tradicional	500ml	Vanish	6,99
Amaciante	Concentrado	500ml	Ypê	5,99
Amaciante	Concentrado	1,5l	Downy	22,99
Amaciante	Concentrado	500ml	Ypê	5,49
Passa roupa	Refil	500ml	Passe Bem	9,49
Passa roupa	Com pulverizador	500ml	Passe Bem	13,99
Sabão em barra	Neutro	1 unidade	Ypê	1,69
Sabão em barra	Perfumado	5 unidades	Ypê	5,79
Sabão em barra	Coco	5 unidades	Ufe	14,99
Anti mofo	Tradicional	100g	Limpanno	6,99
Desengordurante	Cremoso	250ml	Cif	5,49
Desengordurante	Refil	500ml	Veja	9,49
Esponja	Tradicional	4 unidades	Bettanin	1,99
Esponja	Tradicional	4 unidades	3M	2,79
Palha de aço	Tradicional	60g	Bom bril	2,09
Palha de aço	Tradicional	60g	Assolan	1,34
Detergente	Neutro	500ml	Limpol	1,39
Detergente	Neutro	500ml	Minuano	1,25
Detergente	Neutro	500ml	Ypê	1,44
Água sanitária	Tradicional	1l	Super Cândida	2,99
Água sanitária	Tradicional	2l	Candura	3,89

APÊNDICE B – TABELA DE TESTES DE RECONHECIMENTO DE PALAVRAS

	Frases de teste	Testes 1	2	3
Até 3 palavras				
	2 alfaces	errado	certo	certo
	1 repolho	certo	errado	certo
	Espinafre	certo	certo	certo
De 4 a 6 palavras				
	2 alfaces e uma rúcula	errado	certo	certo
	1 repolho e um pé de alface	certo	certo	certo
	Eu quero espinafre e abacaxi	certo	certo	certo
De 7 a 9 palavras				
	2 alfaces, uma rúcula e uma dúzia de bananas	certo	certo	certo
	1 repolho, um pé de alface e umas 5 couves	certo	certo	certo
	Eu quero espinafre, abacaxi e duas ameixas	certo	certo	certo
De 10 a 12 palavras				
	2 alfaces, uma rúcula e uma dúzia de bananas prata	certo	certo	certo
	1 repolho, um pé de alface e 5 couves mateiga	certo	certo	certo
	Eu quero espinafre, abacaxi e umas duas ameixas bem vermelhas	certo	certo	certo
De 13 a 15 palavras				
	2 alfaces, uma rúcula e uma dúzia de bananas prata, E é isso.	errado	certo	certo
	1 repolho, um pé de alface e 5 couves. E um abacate	certo	certo	certo
	Eu quero espinafre, abacaxi, umas duas ameixas e 2 kilos de banana	certo	certo	certo
De 15 a 18 palavras				
	2 alfaces, uma rúcula e uma dúzia de bananas prata. E é isso. Ah e abacate.	certo	errado	certo
	1 repolho, um pé de alface e 5 couves. E dez abacates e dois pés de alface	certo	certo	certo
	Eu quero espinafre, abacaxi, umas duas ameixas, 2 kilos de banana e dois repolhos grandes	certo	certo	certo
De 18 a 21 palavras				
	2 alfaces, uma rúcula e uma dúzia de bananas prata. E é isso. Ah e abacate. E banana também.	errado	certo	certo
	1 repolho, um pé de alface e 5 couves. E dez abacates, dois pés de alface e dois de couve.	certo	certo	certo
	Eu quero espinafre, abacaxi, umas duas ameixas, 2 kilos de banana, dois repolhos grandes e 3 kilos de caqui	certo	certo	certo

4	5	6	7	8	9	10	Erros	Acertos
errado	certo	certo	certo	certo	errado	certo	3	7
certo	certo	errado	certo	certo	certo	certo	2	8
certo	certo	certo	certo	certo	certo	certo	0	10
						TOTAL	**5**	**25**
errado	certo	certo	certo	errado	certo	certo	3	7
certo	certo	errado	certo	certo	certo	certo	1	9
certo	certo	certo	certo	certo	certo	certo	0	10
						TOTAL	**4**	**26**
certo	certo	certo	certo	certo	certo	certo	0	10
certo	errado	certo	certo	certo	certo	certo	1	9
certo	certo	certo	certo	certo	certo	certo	0	10
						TOTAL	**1**	**29**
certo	certo	certo	certo	errado	certo	certo	1	9
certo	certo	certo	certo	certo	certo	certo	0	10
certo	certo	certo	certo	errado	certo	certo	1	9
						TOTAL	**2**	**28**
certo	certo	certo	certo	certo	certo	certo	1	9
certo	certo	certo	certo	certo	certo	certo	0	10
errado	certo	certo	certo	certo	certo	certo	1	9
						TOTAL	**2**	**28**
certo	certo	certo	certo	certo	certo	certo	1	9
errado	certo	certo	certo	certo	certo	certo	1	9
certo	certo	certo	certo	certo	errado	certo	1	9
						TOTAL	**3**	**27**
certo	errado	certo	certo	certo	certo	certo	2	8
errado	certo	certo	certo	certo	certo	certo	1	9
errado	certo	certo	certo	certo	certo	errado	2	8
						TOTAL	**5**	**25**

Bibliografia

[1] KITCHENHAM, B. Procedures for Performing Systematic Reviews. **Keele University Technical Report**, p. 3-22, jul, 2004.

[2] CHENEY, S. THOMPSON, E. **The 2017-2022 App Economy Forecast: 6 Billion Devices, $157 Billion in Spend & More**. 2018. Disponível em <https://www.statista.com/statistics/271644/worldwide-free-and-paid-mobile-app-store-downloads/>. Acessado em março de 2018.

[3] HERNANDEZ, R. **The 10 apps millennials say they can't live without**. 2017. Disponível em: <http://www.businessinsider.com/best-apps-according-to-millennials-2017-8/#1-amazon-10>. Acessado em março de 2018.

[4] LUNDEN, I. **6.1B Smartphone Users Globally By 2020, Overtaking Basic Fixed Phone Subscriptions**. 2015. Disponível em: <https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/>. Acessado em março de 2018.

[5] ALKIM, D. **Aplicativos irão gerar 79 bilhões de dólares em receita no mundo em 2020.** 2016. Disponível em: <http://www.mobiletime.com.br/20/04/2016/apps-vao-gerar-us-79-bi-em-receita-no-mundo-em-2020/438304/news>. Acessado em março de 2018.

[6] LESWING, KIF. This is Amazon's grocery store of the future: No cashiers, no registers and no lines. **Business Insider**. 5 de dezembro de 2016.

[7] MELVILLE, ANDREW. Amazon Go Is About Payments, Not Grocery. **Forbes**. 24 de janeiro de 2017.

[8] **iFood e RestauranteWeb se fundem em empresa de R$ 1 bilhão**. 2018. Disponível em: <http://g1.globo.com/economia/negocios/noticia/2014/09/ifood-e-restauranteweb-se-fundem-em-empresa-de-r-1-bilhao>. Acessado em março de 2018

[9] **iFood - Delivery de Comida.** 2018. Disponível em: <https://play.google.com/store/apps/details?id=br.com.brainweb.ifood>. Acessado em março de 2018.

[10] **Google Express.** 2018. Disponível em: <https://express.google.com>. Acessado em março de 2018.

[11] **Android Developers.** 2018. Disponível em: <developer.android.com/>. Acessado em março de 2018.

[12] LEE, C; JUNG, S; KIM, K; LEE, D; LEE, G. Recent Approaches to Dialog Management for Spoken Systems. **Journal of Computing Science and Engineering**, v. 4, n. 1, p. 1-22, mar, 2010.

[13] BOHUS, D; RUDNICKY, A. The RavenClaw dialog management framework: Architecture and systems. **Computer Speech & Language**, v. 23, n. 1, p. 332–361, 2009.

[14] BOHUS, D. RavenClaw: Dialog Management Using Hierarchical Task Decomposition and an Expectation Agenda. **Interspeech**., 2003. P. 597 - 600.

[15] REN, W; CAO, Y. Overview of Recent Research in Distributed Multi-agent Coordination. **Distributed Coordination of Multi-agent Networks**. Springer, London, 2011, p. 23 - 41.

[16] MCARTHUR, S. D. J., DAVIDSON, E. M., CATTERSON, V. M., DIMEAS, A. L., HATZIARGYRIOU, N. D., PONCI, F., FUNABASHI, T. Multi-Agent Systems for Power Engineering Applications—Part I: Concepts, Approaches, and Technical Challenges". **IEE Transactions on power systems**, v. 22, n. 4, 2007.

[17] MCARTHUR, S. D. J., DAVIDSON, E. M., CATTERSON, V. M., DIMEAS, A. L., HATZIARGYRIOU, N. D., PONCI, F., FUNABASHI, T. Multi-Agent Systems for Power Engineering Applications—Part II: Technologies, Standards, and Tools for Building Multi-agent Systems". **IEE Transactions on power systems**, v. 22, n. 4, 2007.

[18] LEITÃO, P. Multi-agent Systems in Industry: Current Trends & Future Challenges. **Beyond Artificial Intelligence**. Berlim: Ed. Springer, 2013. P. 197 – 201.

[19] Bergenti, F., Vargiu, E. Multi-Agent Systems in the Industry Three Notable Cases in Italy. **WOA.** 2010.

[20] AMADO, T.J.; GIOTTO, E. A sua lavoura na tela. **A Granja**, São Paulo, v.732, p. 38-42, dez, 2009.

[21] LECHETA, R. **Android Essencial**. 1. Ed. Novatech. 2016.

[22] GOODWIN, A. **Google, Hyundai show off new third-party Android Auto apps**. 2014. Disponível em: <https://www.cnet.com/roadshow/news/google-hyundai-demonstrate-android-autos-new-api/>. Acessado em abril de 2018.

[23] KOTELLY, B. **The art and business of speech recognition: creating the noble voice**. Indianapolis: Addison-Wesley, p. 208, 2003.

[24] CHOU, W.; JUANG, B.H. **Pattern recognition in speech and language processing**. Florence: CRC Press, p. 416, 2003.

[25] GOMÉZ, E. M. T. **Reconhecimento de fala para navegação em aplicativos móveis para português brasileiro**. 2011. 125 p. Dissertação (Mestrado em Ciência da Computação) - Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2011.

[26] DE PAULA, L. J. L. **Desenvolvimento de aplicativo para dispositivos móveis para coleta de dados georreferenciados através de reconhecimento de voz**. 2013. Dissertação (Mestrado em Engenharia de Sistemas Agrícolas) Escola Superior de Agricultura Luiz de Queiroz, Universidade de São Paulo, São Paulo, 2011.

[27] TEVAH, R. T. **Implementação de um sistema de reconhecimento de fala contínua com amplo vocabulário para o português brasileiro**. 2006. 115 p. Dissertação (Mestrado em Ciências da Computação em Engenharia Elétrica) - Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2006.

[28] SILVA, A. G. S. **Reconhecimento de voz para palavras isoladas**. 2009. 59 p. Trabalho de Conclusão de Curso (Graduação em Engenharia da Computação) - Universidade Federal de Pernambuco, Recife, 2009.

[29] FRANÇA FILHO, C. B. **Investigação do reconhecimento de fala baseado em um ambiente telefônico**. 2010. 53 p. Trabalho de Conclusão de Curso (Graduação em Ciências da Computação) - Universidade Federal de Pernambuco, Recife, 2010.

[30] DIVENYI, P. **Speech separation by humans and machines. Boston: Kluwer Academic**. 2. Ed. Springer US, p. 344 2005.

[31] TEBELSKIS, J. **Speech recognition using Neural Networks**. 1995. 10 - 26 p. Dissertação (Doutorado em Ciências da Computação) - Carnegie Mellon University, Pittsburgh, 1995.

[32] LEITE, P. B. C. **Identificação de Tipos de Culturas Agrícolas a partir de Seqüências de Imagens Multitemporais Utilizando Modelos de Markov Ocultos**. 2008. Dissertação (Mestrado em Engenharia Elétrica) - Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, 2008.

[33] YNOGUTI, C. A. **Reconhecimento de fala contínua usando modelos ocultos de Markov**. 1999. 151 p. Dissertação (Doutorado em Engenharia Elétrica) - Faculdade de Engenharia Elétrica, Universidade Estadual de Campinas, Campinas, 1999.

[34] FERNEDA, E. **Redes Neurais e sua aplicação em sistemas de recuperação de informação**. Brasília, v. 35, n. 1, p. 25-30, abril, 2006

[35] HAYKIN, S. **Redes Neurais: princípios e prática**. 1. Ed. Porto Alegre: Bookman, 2001.

[36] MORAIS, E. S. **Reconhecimento automático de fala contínua empregando modelo híbrido ANN + HMM**. 1997. 123 p. Dissertação (mestrado em Engenharia Elétrica) - Faculdade de Engenharia Elétrica, Universidade Estadual de Campinas, Campinas, 1997.

[37] **Adicionar o Firebase ao seu projeto do Android**. Disponível em: <https://firebase.google.com/docs/android/setup?authuser=0>. Acessado em novembro de 2018.