UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

ROGER SILVA

Classificação textual por assuntos em aplicativo provedor de notícias coletadas de rede social direcionado à plataforma móvel

Trabalho de Graduação.

Orientador: Prof. Dr. Leandro Krug Wives

Porto Alegre
2014

RESUMO

Classificação de textos é uma área responsável por agrupar textos em categorias. Ela aborda soluções sobre como impor algum tipo de organização em um conjunto de textos através de rótulos, os quais servem de categorias. Um aplicativo móvel que colete notícias e possibilite a seu usuário consumi-las pode conter um mecanismo de agrupamento delas por assunto de modo que melhore sua experiência de uso. O desafio é ainda maior quando a fonte dessas notícias é o Twitter, uma rede social que permite somente a publicação de textos curtos. O objetivo deste trabalho é criar um módulo de classificação textual que realize a categorização das notícias coletadas por um aplicativo móvel, acoplar o módulo a esse aplicativo e prover uma melhor experiência de uso aos usuários do aplicativo uma vez que as notícias estarão rotuladas por assunto. O enfoque para a construção desse módulo é escolher a melhor configuração que maximize a quantidade de acertos na categorização de notícias com textos curtos.

Palavras-chave: Classificação de textos, Aplicativo, Mobile, Notícias, Rede Social, Twitter.

Text classification applied for a mobile news feed application based on social networks

ABSTRACT

Text classification is responsible for grouping texts into categories. It covers solutions designed to impose some kind of organization in textual sets by the use of labels, which serve as categories. A mobile application that collects news and enables users to consume them must contain a classification mechanism able to automatically organize them by subjects in order to improve the user experience. The challenge is even greater when the news source is Twitter, a social network that only allows the publication of short texts. The objective of this work is to create a text classification module that performs the categorization of news collected by a mobile application, attach the module to this application and provide a better user experience, since the news will be labeled by subject. The approach to the construction of this module is to choose the best configuration that maximizes the amount of correct categorization of news with short texts.

Keywords: Text Classification, Application, Mobile, News, Social Network, Twitter.

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

ADT	Abstract Data Type
API	Application Programming Interface
BDD	Behavior Driven Development
HTTP	Hypertext Transfer Protocol
JSON	Javascript Object Notation
MVC	Model-View-Controller
REST	Representational State Transfer
URL	Uniform Resource Locator

SUMÁRIO

1 INTRODUÇÃO

Documentos precisam ser organizados de tal forma que a busca por algum deles seja facilitada. Dentre métodos de organização possíveis está a rotulação. A um grupo de documentos que compartilhe características em comum é atribuído um rótulo que o identifique e o distinga em relação a outros grupos. Um grupo rotulado de documentos é denominado de classe de documentos. O processo de associar um ou mais rótulos de classe a um documento é denominado de classificação textual (BAEZA-YATES, 2013).

Textos acadêmicos, de páginas Web, de documentos pertencentes a um projeto, de artigos em jornais, etc, todos podem ser classificados de modo a ser organizados através de rótulos. Quaisquer tipos de rótulos que sirvam como meio de organização a documentos são válidos como, por exemplo, rótulos que classifiquem um texto quanto a seu tamanho ("pequeno", "médio", "grande", ...); quanto a sua época ("década de 70", "década de 80", "década de 90", ...); quanto ao assunto a que se refere ("nutrição", "medicina", "química", ...); dentre outros.

1.1 Motivação

Em maio de 2014, o autor deste trabalho desenvolveu um aplicativo para o sistema operacional Android chamado *News* (GOOGLE PLAY, 2014). Esse aplicativo possibilita com que seus usuários consumam notícias dos principais veículos de comunicação do mundo através de um dispositivo móvel (smartphone, tablet, dentre outros).

As notícias disponibilizadas por esse aplicativo são coletadas de fontes de notícias através de seus perfis na rede social *Twitter*, a qual permite pessoas a lerem e escreverem mensagens curtas em até 140 caracteres (TWITTER WIKIPEDIA, 2014).

Porém, o aplicativo em questão coleta notícias de diversas fontes. E essas (em quase sua totalidade) não oferecem uma forma de classificação das notícias por assunto (como, por exemplo, esportes, política, música, etc.) de modo que facilite o seu consumo.

1.2 Objetivo

Por meio deste trabalho, é proposta a solução de, uma vez o aplicativo *News* tendo coletado notícias do perfil no *Twitter* de fontes de notícias dos mais diversos gêneros, prover uma forma de organização dessas notícias por assunto de modo que facilite o consumo dessas pelo leitor.

Para tornar isso viável, será implementado um módulo de classificação textual. Esse módulo fará uso da API de um software de aprendizagem de máquina chamado *Weka* que, dentre outras funcionalidades, realiza predições sobre um dado baseado em um modelo construído previamente. O módulo construído será acoplado ao aplicativo *News* para que, uma vez que esse tenha coletado um conjunto de notícias, repasse-as ao módulo e esse classifique-as por assunto. Por fim, uma vez as notícias classificadas, sejam mostradas por assunto ao usuário do aplicativo.

1.3 Organização do texto

O texto desse trabalho foi organizado em cinco capítulos. O capitulo dois, seguinte a este, trata dos conceitos relacionados que envolvem a construção do aplicativo e do módulo de classificação textual, além de soluções semelhantes a proposta por esse trabalho e uma comparação entre elas. O capitulo três menciona questões de design tanto arquitetural, quanto de interface da solução. O capitulo quatro descreve aspectos da implementação de funcionalidades do aplicativo, do módulo de classificação de notícias e, em especial, das adaptações do aplicativo original para comportar o módulo de classificação. O capitulo cinco trata sobre experimentos realizados para concepção ideal do classificador. Por fim, o capitulo seis é a conclusão deste trabalho que contém um breve resumo sobre todos os aspectos tratados e desenvolvidos sobre a solução e perspectiva de trabalhos futuros.

2 CONCEITOS RELACIONADOS

Neste capítulo, serão mostrados conceitos fundamentais relacionados a este trabalho. Primeiramente, será descrita a metodologia de desenvolvimento ágil adotada para o projeto. Também, será exposta uma visão geral da plataforma-alvo para a qual o aplicativo *News* foi desenvolvido. O conceito da *Twitter API* é explicado logo após, pois é através dela que o aplicativo acessa as notícias publicadas por fontes. Em seguida, serão apresentados os fundamentos teóricos que sustentam a solução, principalmente referente à área de classificação de textos e algoritmos supervisionados, e o tipo de classificador específico adotado. Por fim, serão relatadas soluções existentes similares ao aplicativo *News* e uma análise comparativa entre esses.

2.1 Metodologia

O método de trabalho escolhido para produção deste projeto foi o Kanban (tratado aqui como metodologia, e não simplesmente como um quadro de tarefas). Essa metodologia trata-se de um mecanismo de controle de fluxo de trabalho (ANDERSON, 2010). Ele é formado por um quadro e cartões. Cada cartão é associado a um item de trabalho. Caso haja um cartão liberado, ele poderá ser associado a um item de trabalho e ser colocado no quadro de tarefas. Conforme o item de trabalho for sendo desenvolvido, o cartão associado a ele será deslocado pelo quadro de tarefas até sua conclusão. Quando concluído, o cartão será removido do quadro de tarefas e o item de trabalho desassociado dele, liberando, assim, o cartão para ser associado a um novo item de trabalho.

Essa metodologia de desenvolvimento se baseia em um sistema puxado. Conforme a entidade atuando sobre o projeto ou trabalho esteja livre de tarefas, essa poderá puxar um novo item de trabalho para atuar sobre ele. Então, Kanban também auxilia a não haver sobrecarga de trabalho durante o processo, principalmente por ser um mecanismo visual de controle. Ou seja, caso haja acúmulo de cartões em uma determinada etapa do processo, isso será rapidamente detectável por esse acúmulo estar explícito no quadro de tarefas. No contexto deste trabalho, Kanban foi adotado por ser um mecanismo simples (e satisfatório) de controle de fluxo de trabalho e por ajudar visualmente a controlar os

tópicos sendo desenvolvidos durante todo o processo.

Para implantação dessa metodologia neste trabalho, foi adotada uma ferramenta virtual online chamada Trello (TRELLO, 2014), pois ela permite o gerenciamento colaborativo entre os membros envolvidos em um projeto. No caso deste trabalho, os membros trataram-se do professor orientador e o aluno orientado. Abaixo, é mostrada a Figura 2.1, a qual contém uma amostra do quadro de tarefas em um determinado estágio do projeto hospedada pela ferramenta Trello:

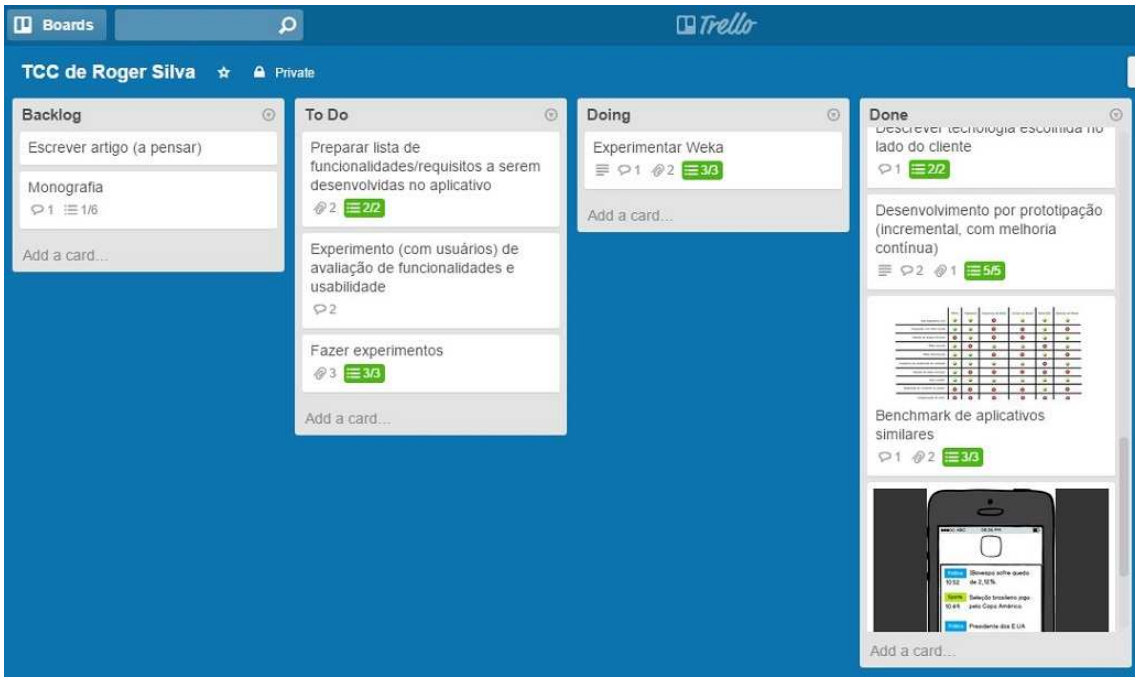

Figura 2.1: Quadro de tarefas usado no Kanban por este projeto

2.2 Plataforma-alvo

O aplicativo *News* foi desenvolvido para a plataforma móvel *Android*. Trata-se de um sistema operacional desenvolvido pelo Google baseado no *kernel* do sistema operacional Linux, o qual, segundo dados divulgados por essa companhia, alcançou o número de um bilhão de usuários ativos (ANDROID WIKIPEDIA, 2014).

A escolha da plataforma *Android* como hospedeira do aplicativo *News* foi determinada pelo fácil acesso ao ambiente de desenvolvimento necessário para sua produção (já que todas as ferramentas usadas são gratuitas) e, principalmente, pelo alto número de usuários que possuem dispositivos com esse sistema operacional - quase 85%, segundo a organização global de análise estratégica *Strategy Analytics* (STRATEGY ANALYTICS, 2014) e conforme a Figura 2.2 - visando, assim, maximizar a audiência do aplicativo.

O sistema operacional *Android* conta com diversas versões já lançadas. Conforme

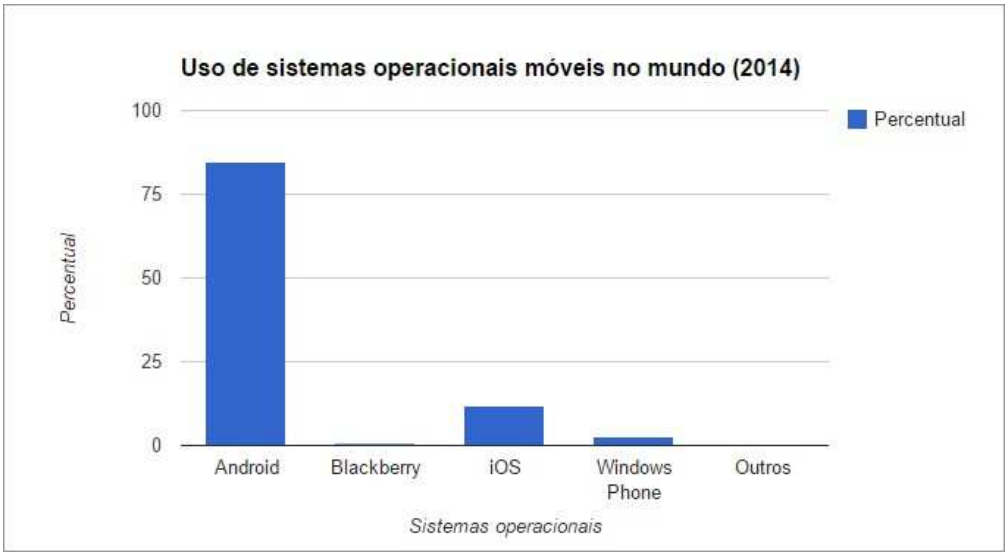

Figura 2.2: Dados referentes ao uso de sistemas operacionais móveis

novas versões são disponibilizadas, novas funcionalidades são adicionadas a elas. Para o desenvolvimento de um aplicativo para essa plataforma, caso ele faça uso de *features* somente existentes nas versões mais novas do sistema operacional, dispositivos que hospedam versões mais antigas do sistema não poderão executar esse aplicativo. Com base nisso, com o objetivo de oferecer suporte ao maior de número de versões de *Android*, o aplicativo *News* tem suporte mínimo à *API* 15 (*Ice Cream Sandwich*, versão 4.0.3). Dessa forma, 89,6% dos dispositivos no mundo são capazes de executar o aplicativo, segundo dados do Google (DEVELOPER ANDROID, 2014) e conforme a Figura 2.3 abaixo:

Version	Codename	API	Distribution
2.2	Froyo	8	0.6%
2.3.3 - 2.3.7	Gingerbread	10	9.8%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	8.5%
4.1.x	Jelly Bean	16	22.8%
4.2.x		17	20.8%
4.3		18	7.3%
4.4	KitKat	19	30.2%

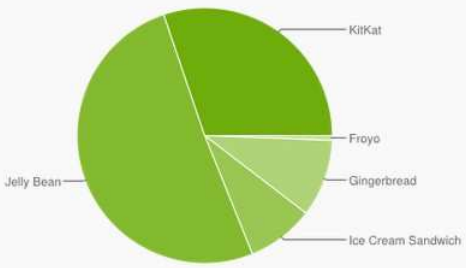

Figura 2.3: Dispositivos com versões de Android no mundo em 2014

2.3 Twitter API

A *Twitter API* permite aplicações externas serem integradas ao *Twitter*. O aplicativo *News* tratado nesse trabalho, é totalmente dependente dessa API, pois é através dela que as notícias do *Twitter* das fontes de notícias são coletadas. O meio pelo qual a *Twitter API* permite essas requisições é através de HTTP GET (pertencente ao conjunto de métodos

de REST, que trata-se de um estilo arquitetural que consiste de um conjunto coordenado de restrições arquiteturais aplicadas a componentes em um sistema de hipermídia distribuído).

2.4 Classificação Textual

A classificação textual de documentos é um ramo da área de Recuperação de Informação que é fundamentada sobre aprendizado de máquina. Aprendizado de máquina é uma área da Inteligência Artificial responsável pelo projeto e implementação de algoritmos que visam reconhecer padrões genéricos em dados providos como entrada, tal que aqueles padrões possam ser usados para realizarem predições sobre dados novos ainda não manipulados (BAEZA-YATES, 2013).

Algoritmos de aprendizado de máquina são compostos por uma etapa de aprendizado. Essa etapa é baseada na construção de um modelo (padrão) sobre dados de entrada para que, conforme novos dados tenham que ser avaliados, o algoritmo possa inferir um julgamento sobre esses dados com base no modelo gerado na fase de aprendizado. Esses algoritmos podem ser definidos de três tipos: supervisionados, não-supervisionados e semisupervisionados. Neste trabalho, o processo de aprendizado aplicado na solução usada pelo aplicativo *News* é uma abordagem supervisionada.

2.4.1 Definição

O processo de classificação textual pode ser definido formalmente da seguinte forma:

Dado um conjunto D de documentos e um conjunto C de classes, um classificador textual é uma função F: D x C $\rightarrow$ {0, 1}, tal que, dado um par [d_i, c_j], $d_i \in D$ e $c_j \in C$, a função produz como resultado valor 0, caso o documento d_i pertença à classe c_j. Caso a função produza o valor 1 como resultado, é dito que o documento d_i não pertence à classe c_j (BAEZA-YATES, 2013).

A definição acima permite que o algoritmo de classificação textual atribua mais de uma classe a um determinado documento. Um classificador que realize esse tipo de atribuição é chamado multirrótulo. Caso o classificador restrinja essa atribuição para somente uma classe por documento, ela passa a ser denominada uma atribuição de rótulo único.

No contexto do aplicativo *News*, pelo fato dele permitir que uma dada notícia seja classificada somente para um assunto, o algoritmo classificador presente na solução é considerada ser de rótulo único.

2.4.2 Algoritmos Supervisionados

Esses algoritmos são caracterizados por dependerem de informações fornecidas por seres humanos (por exemplo, classes e documentos pertencentes a elas) de modo que

essas informações possam ser usadas como sendo um conjunto de treinamento para um classificador, para que esse aprenda uma função de classificação e consiga classificar novos dados ainda não vistos com uma boa margem de acerto. Um conjunto de treinamento pode ser definido formalmente da seguinte forma:

Dado um subconjunto $D_i \in D$ e um conjunto de classes C, uma função de conjunto de treinamento é definida como T: D_i x C $\rightarrow$ {0, 1}, tal que, dado um par [d_i, c_j], $d_i \in D$ e $c_j \in C$, o valor 0 é atribuído ao par caso o documento d_i não esteja associado à classe c_j e 1 caso o documento d_i esteja associado à classe c_j, com base no julgamento realizado por especialistas humanos (BAEZA-YATES, 2013).

A função de treinamento é usada para ajustar a função do classificador, de modo que o classificador consiga predizer novos dados. O processo de treinamento e classificação é descrito na Figura 2.4 abaixo.

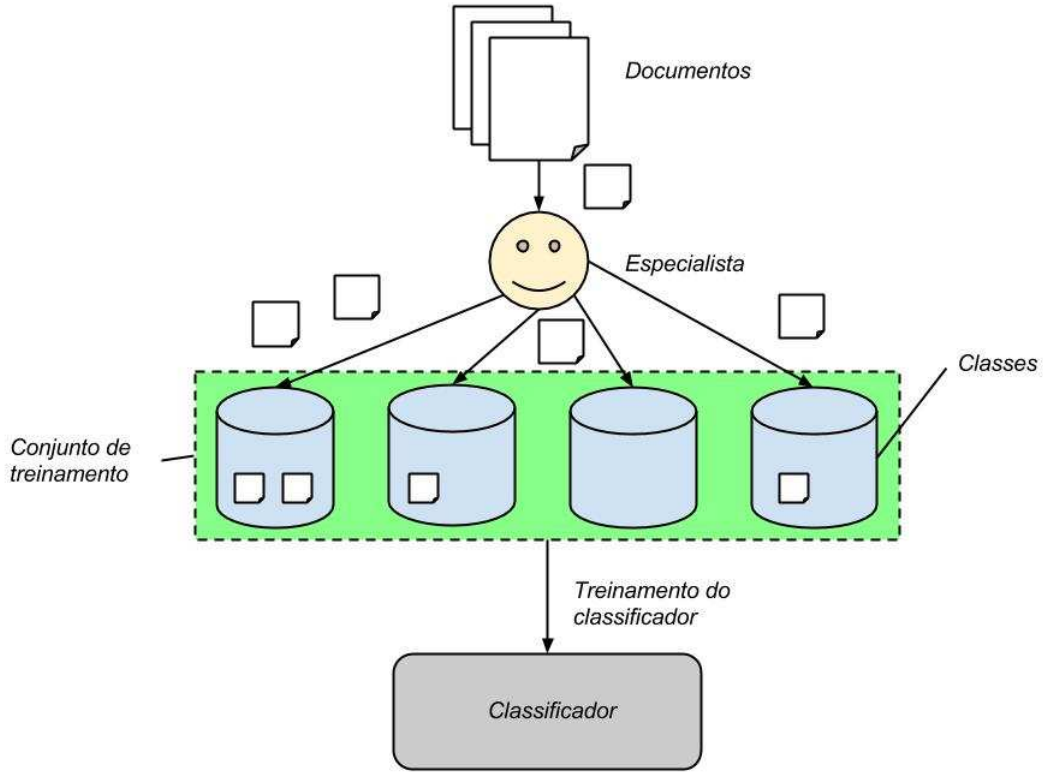

Figura 2.4: Treinamento de classificador

Um fator a ser levado em conta é o fato do conjunto de treinamento ser avaliado por humanos. Logo, existe subjetividade no julgamento das associações documento-classe do conjunto de treinamento. Assim, deve ser considerado que o classificador realizará predições sobre novos dados com base no julgamento realizado por especialistas humanos, de modo que o resultado produzido pelo classificador não deve ser considerado objetiva-

mente correto sob todos os pontos de vista.

2.4.3 Classificador Bayes Ingênuo Multinomial

O módulo de classificação de notícias usado no aplicativo deste trabalho é baseado em um algoritmo de classificação chamado Bayes Ingênuo Multinomial. Esse classificador é uma instância específica de um classificador Bayes Ingênuo (BAYES INGÊNUO WIKIPEDIA, 2014) que usa distribuição multinomial para cada termo encontrado em documentos. Ou seja, é um classificador que leva em conta, no contexto da solução proposta por este trabalho, a frequência com que termos aparecem em notícias disponibilizadas por fontes.

Seja $C = \{c_1, c_2, ..., c_n\}$ um conjunto de classes, $K = \{k_1, k_2, ..., k_m\}$ um conjunto de termos e $W = \{w_1, w_2, ..., w_m\}$ a frequência com que termos aparecem em um documento de teste. Seja **D** o número total de documentos em um conjunto de treinamento. D_c o número de documentos que estão associados a uma classe c. A probabilidade que uma classe c esteja associada a algum documento no conjunto treinamento, nomeada aqui como $P(C)$, pode ser definida como:

$$P(C) = D_c / D$$

A quantidade de vezes que um termo k ocorre em uma classe c associada a documentos no conjunto de treinamento, pode ser definida como $count(k, c)$. A quantidade de termos contidos em documentos (incluindo repetições nessa contagem) os quais estejam associados a uma classe c pode ser tratado como $count(c)$. E, considerando o número de elementos do conjunto **K** (podendo ser interpretado como o vocabulário disponível), a probabilidade da ocorrência de um termo **k**, dado uma classe **c**, pode ser definido como:

$$P(k|c) = (count(k, c) + 1) / (count(c) + |K|)$$

Dessa forma, para a predição (associação) de uma classe a um documento (**CD**), a seguinte expressão é usada pelo algoritmo:

$$P(CD) = max(P(c_i) \times P(k_1|c_i) \times P(k_2|c_i) \times ... \times P(k_m|c_i)), i = 1...n$$

Ou seja, a classe c com o maior $P(CD)$ será tida como o rótulo do documento de teste **D** sendo avaliado pelo classificador.

2.5 Soluções Existentes

Assim como o aplicativo *News*, existem outras soluções semelhantes para a plataforma *Android* que permitem aos usuários consumirem notícias de fontes das formas mais diversas. A seguir serão mostrados os principais aplicativos encontrados para *Android* com funções semelhantes ao *News* e suas características, e, por fim, uma tabela comparativa

exaltando as diferenças entre eles com os prós e contras de cada solução.

2.5.1 Flipboard

O Flipboard (`http://goo.gl/ehTB1g`) é uma revista personalizada que permite acompanhar notícias e assuntos diversos, como futebol, moda e tecnologia. Notícias do Brasil e do mundo. Permite visualizar fotos, vídeos e artigos compartilhados por amigos através de redes sociais. Abaixo, segue a Figura 2.5 da tela desse aplicativo:

Figura 2.5: Tela do Flipboard

2.5.2 Imprensa de Bolso

Imprensa de Bolso (`http://goo.gl/LQdjHK`) trata-se de uma solução de fornecimento de notícias mais simples. Ele permite folhear os jornais principais brasileiros e estrangeiros. Segue, na Figura 2.6 a tela do aplicativo.

2.5.3 Jornais do Brasil

O aplicativo Jornais do Brasil (`http://goo.gl/XcDbrs`) reúne os principais jornais e revistas no Brasil acessíveis sem que seja necessário navegar para os sites desses. Na Figura 2.7, é possível ver a variedade de fontes.

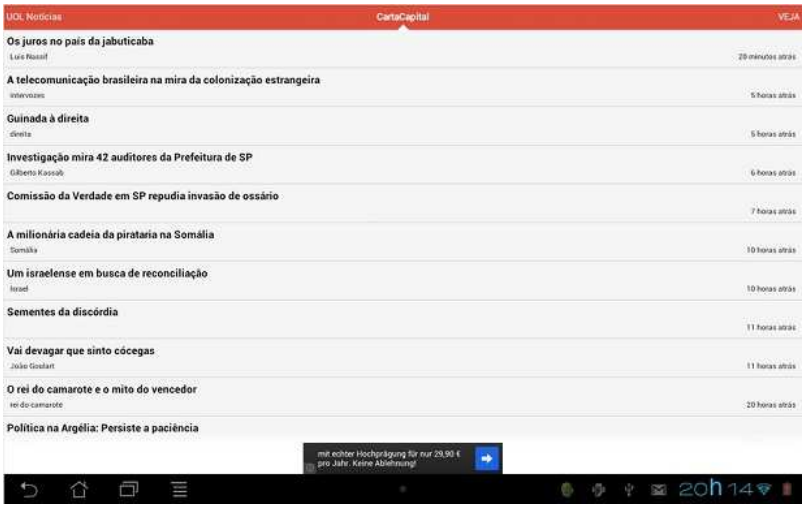

Figura 2.6: Tela do Imprensa de Bolso

Figura 2.7: Tela do Jornais do Brasil

2.5.4 News360

News360 (`http://goo.gl/rsfoOn`) é um leitor de notícias que aprende sobre as notícias prediletas do usuário. As notícias fornecidas são baseadas no histórico de uso do usuário. Conectando-se via redes sociais, o aplicativo pode aprender sobre características do usuário de modo a aumentar a chance de acerto sobre as notícias que o usuário teria preferência em ler. Na Figura 2.8 é possível ver sua tela:

2.5.5 Notícias do Brasil

Com uma proposta simples, o aplicativo Notícias do Brasil (`http://goo.gl/prABuH`) é um leitor das principais fontes de notícias do Brasil. Segue a tela desse aplicativo na Figura 2.9:

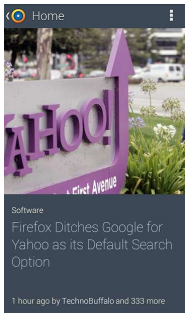

Figura 2.8: Tela do News360

Figura 2.9: Tela do Notícias do Brasil

2.6 Análise comparativa

Apesar dos aplicativos apresentados cumprirem, em sua essência, o mesmo objetivo de exibição de notícias de diversas fontes aos seus usuários, eles apresentam algumas peculiaridades que se destacam positiva e negativamente. Abaixo, na Figura 2.9, segue uma comparação de todos os aplicativos listados anteriormente, incluído o aplicativo *News* que, mais adiante, será detalhado mais a fundo:

	News	*Flipboard*	*Imprensa de Bolso*	*Jornais do Brasil*	*News360*	*Notícias do Brasil*
User Experience (UX)	✓	✓	✗	✓	✓	✓
Integração com redes sociais	✓	✓	✗	✗	✓	✗
Seleção de artigos favoritos	✗	✓	✗	✗	✓	✓
Mídia nacional	✓	✗	✓	✓	✗	✓
Mídia internacional	✓	✓	✗	✗	✓	✗
Frequência de atualização de conteúdo	✓	✓	✓	✓	✗	✓
Seleção de mídias favoritas	✓	✗	✗	✗	✗	✗
Auto-contido	✓	✓	✓	✓	✓	✓
Adaptação do conteúdo ao usuário	✗	✗	✗	✗	✓	✗
Categorização de texto	✗	✗	✗	✗	✗	✗

Figura 2.10: Comparação entre aplicativos de notícias Android

Conforme a figura acima, nenhum dos aplicativos em questão apresenta a caracterís-

tica de categorização textual. Logo, o módulo de classificação textual do aplicativo *News* é tido como o diferencial em relação a todos os outros aplicativos de gerenciamento de notícias.

3 DESIGN DA SOLUÇÃO

Neste capítulo serão tratadas questões de design da solução da aplicação proposta. Primeiramente, será abordado o problema que ela tenta resolver e o quanto a classificação de notícias por assunto beneficia os usuários. Após, será mostrada como foi projetada a arquitetura do aplicativo e quais componentes fazem parte dessa. Também serão mostrados os requisitos do aplicativo na forma de histórias de usuário. Por fim, a interface gráfica do aplicativo será discutida e a adaptação dessa para suportar a funcionalidade de categorização de notícias por assuntos.

3.1 Identificação do problema

O aplicativo *News* foi desenvolvido com a proposta de atender a necessidade de usuários que gostam de manter-se informados por meio das principais mídias, porém sem o tempo para conseguir se dedicar à leitura. Além disso, pelo fato de tratar-se de uma solução para plataforma móvel, ela deve oferecer simplicidade e uma boa experiência de usuário para que possa ser usada por pessoas em trânsito de modo que a leitura não seja prejudicada.

Uma vez que as notícias são coletadas pelo aplicativo, elas são mostradas ao usuário em um formato *timeline* sem uma identificação adequada sobre a qual tipo de assunto se refere uma notícia em específico. Para tratar esse problema, o módulo classificador de notícias visa rotular todas as notícias com o assunto a qual cada uma se refere de modo facilite a experiência do usuário do aplicativo.

3.2 Design da arquitetura

O aplicativo *News* foi estruturado sob a forma de uma arquitetura híbrida, pois, para sua organização dentro da plataforma *Android* foi utilizado o padrão MVC. Porém, o aplicativo necessita realizar requisições à API do Twitter de modo que as notícias sejam coletadas. Logo, também pode ser dito que a solução faz uso do padrão REST, resultando, assim, em um padrão híbrido (MVC + REST).

3.2.1 Padrão arquitetural MVC

Este padrão caracteriza-se por separar a camada de apresentação da camada de interação do usuário com o aplicativo (SOMMERVILLE, 2009). Ou seja, toda comunicação existente do usuário com telas do aplicativo (*views*), às quais necessitam manipular entidades (*models*) no contexto da aplicação fazem isso por meio de classes (*controllers*) que são responsáveis por gerenciar toda a lógica necessária. Os três componentes do padrão MVC podem ser vistos na Figura 2.9 de acordo com o contexto do aplicativo *News*:

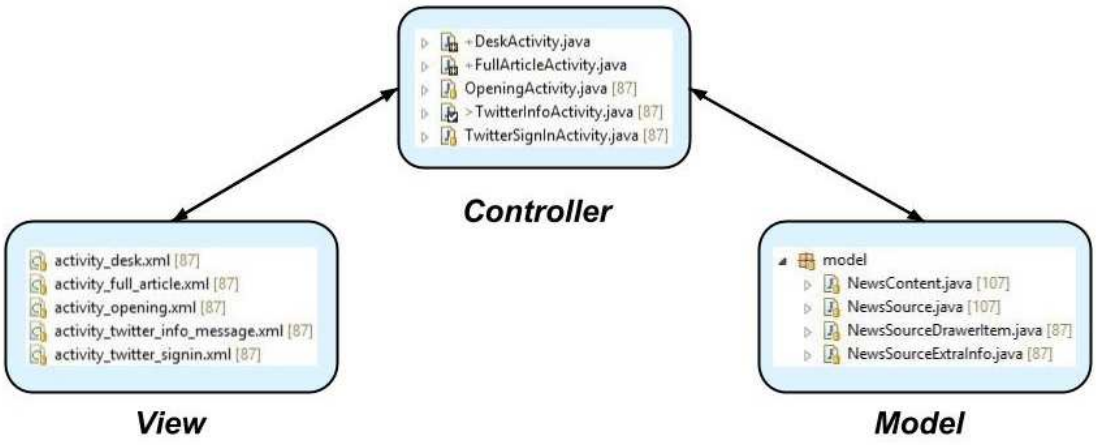

Figura 3.1: Padrão MVC no contexto do aplicativo News

3.2.2 Padrão arquitetural REST

Segundo Richardson e Amundsen, "*REST is not a protocol, a file format, or a development framework. It's a set of design constraints: statelessness, hypermedia as the engine of application state, and so on*"[1] (RICHARDSON; AMUNDSEN, 2013). Esse padrão possui operações, mais conhecidas como verbos, que permitem o acesso a recursos através do uso de URLs. Esses verbos são denominados GET, POST, PUT e DELETE. Em especial, merece destaque o verbo GET, pois ele é essencial para o aplicativo *News*.

A função do método GET é permitir o acesso à representação de algum recurso. Esse recurso no contexto da solução em questão neste trabalho são notícias. Essas, disponibilizadas por fontes de notícias pelo *Twitter*, são possíveis de acesso através da API `https://api.twitter.com/1.1/statuses/user_timeline.json`, o aplicativo acessa as últimas notícias publicadas por uma determinada fonte. O aplicativo pode receber como resultado à invocação ao serviço REST através da URL ou um JSON, o qual é um formato de troca de dados (que contém as informações requeridas pelo aplicativo), ou algum código de erro. Esse processo de requisição e resposta, é mostrado na Figura 3.2,

[1]REST não é um protocolo, um formato de arquivo, ou um framework de desenvolvimento. É um conjunto de restrições de projeto: sem estado, hipermídia como o motor de estado da aplicação, etc.

seguinte.

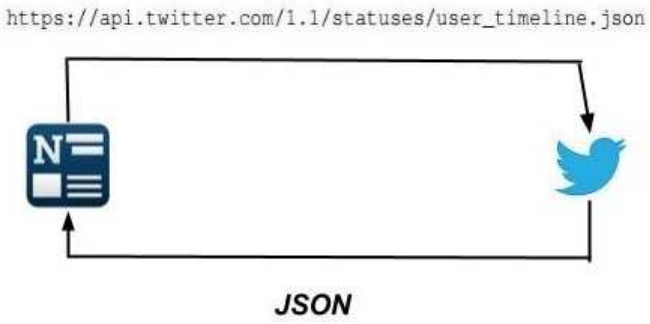

Figura 3.2: Padrão REST

3.3 Requisitos do aplicativo

Os requisitos da solução são expostos no formato de histórias de usuário. Essas são formas que descrevem funcionalidades que agregam valor ao usuário do software (COHN, 2004) de maneira enxuta e ágil.

Cada história relacionada ao aplicativo *News* terá testes de aceitação, tais que servirão para validar se a história de usuário, após implementada no aplicativo, validam ou não a funcionalidade implementada. Esses testes são descritos em um formato chamado BDD, sendo um padrão de fácil escrita e de fácil compreensão por quem necessita compreender a intenção dos testes.

3.3.1 Seleção de mídia qualquer

A história *Seleção de mídia qualquer* caracteriza-se por descrever um usuário selecionando um veículo de comunicação qualquer, sem especialização ou especializado em um determinado assunto, conforme a Figura 3.3

SENDO um usuário que deseja ler notícias
POSSO selecionar uma mídia qualquer
PARA QUE visualize sua timeline com notícias sobre assuntos específicos ou sobre os assuntos mais diversos

Figura 3.3: História “Seleção de mídia qualquer”

O primeiro teste de aceitação relacionado a essa história chama-se *Timeline vazia*. Ele representa o caso em que não existem notícias a serem mostradas, como mostrado na Figura 3.4.

Já o segundo, nomeado *Timeline preenchida*, descreve o caso em que existe ao menos uma notícia a ser mostrada, como a Figura 3.5 descreve.

DADO QUE seja selecionada uma fonte de notícias
QUANDO essa não contém nenhuma notícia publicada
ENTÃO sua timeline aparece vazia
E o usuário não pode interagir com essa timeline

Figura 3.4: Timeline vazia

DADO QUE seja selecionada uma fonte de notícias
QUANDO essa contém notícias publicadas
ENTÃO sua timeline aparece preenchida com notícias, seus horários de publicação e possíveis links para suas matérias completas
E o usuário pode interagir com ela deslizando a lista de notícias ou clicando sobre um link para ser redirecionado para suas matérias completas

Figura 3.5: Timeline preenchida

3.3.2 Seleção de mídia genérica

A história *Seleção de mídia genérica* pode ser interpretada como um usuário selecionando um veículo de comunicação sem especialização em um assunto, ou seja, que pode publicar notícias como, por exemplo, esporte, economia, política, arte, dentre outros, como é mostrado na Figura 3.6:

SENDO um usuário que deseja ler notícias
POSSO selecionar uma mídia genérica
PARA QUE visualize sua timeline com notícias sobre os mais diversos assuntos

Figura 3.6: História "Seleção de mídia genérica"

Um teste de aceitação para validar essa história pode ser rotulado como *Notícias de apenas um assunto*. Ele verifica se a fonte de notícias provém conteúdo apenas de assuntos especializados, tais como esporte, política, etc, como mostrado na Figura 3.7 abaixo:

DADO QUE seja selecionada uma fonte de notícias genérica
QUANDO as notícias publicadas pela fonte de notícias são categorizadas para um assunto em específico
ENTÃO será possível visualizar as notícias publicadas somente sobre esse assunto em uma única timeline
E o usuário poderá interagir com essa timeline

Figura 3.7: Notícias de apenas um assunto

Outro possível teste para essa história acontece quando notícias sobre mais de um assunto são publicadas. Teste esse que pode ser chamado *Notícias sobre mais de um assunto*. Esse teste de aceitação pode ser descrito como na Figura 3.8.

3.3.3 Seleção de fontes de notícias favoritas

A história *Seleção de fontes de notícias favoritas* pode ser entendida como a ação do usuário em gerenciar as fontes de notícias de modo que elas estejam acessíveis facilmente

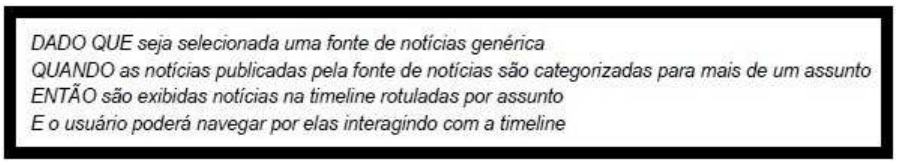

Figura 3.8: Notícias sobre mais de um assunto

na barra lateral do aplicativo, conforme fica claro na Figura 3.9 a seguir:

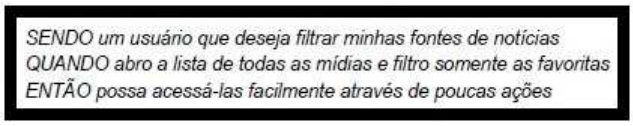

Figura 3.9: História "Seleção de fontes de notícias favoritas"

Para validar a história, um teste chamado *Seleção de nenhuma mídia favorita* pode ser realizado. Para tal, a Figura 3.10 mostra o contexto dele:

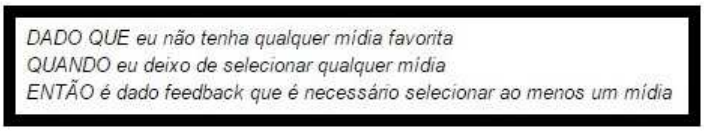

Figura 3.10: Seleção de nenhuma mídia favorita

Outro possível contexto ocorre quando o usuário resolve selecionar uma ou mais fontes favoritas. Tal teste de aceitação, nomeado *Seleção de uma ou mais mídias favoritas* pode ser realizado como descrito na Figura 3.11 abaixo:

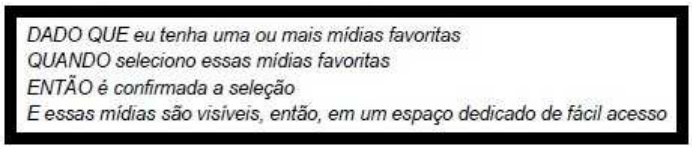

Figura 3.11: Seleção de uma ou mais mídias favoritas

3.3.4 Visualização de matéria completa

A história de usuário *Visualização de matéria completa* trata-se do desejo do usuário em ler a matéria completa, caso exista, no site da fonte de notícias de uma determinada notícia contida na *timeline* de uma fonte. Esse contexto, pode ser visto na Figura 3.12.

Um modo de validar essa história, é através de um teste que pode ser nomeado como *Exibição de matéria completa*, como pode ser visto na Figura 3.13.

SENDO um usuário que deseje mais informações sobre uma notícia
QUANDO realizo a ação de ser redirecionado para a matéria completa de uma notícia
ENTÃO a matéria do site é exibida no aplicativo
E possa aprofundar-me mais sobre o assunto em questão

Figura 3.12: História "Visualização de matéria completa"

DADO QUE deseje ler mais sobre uma determinada notícia publicada por uma mídia
E existe um link para o site da mídia sobre essa notícia
QUANDO aciono o link que leva para o site dessa mídia
ENTÃO a matéria completa sobre a notícia é exibida com sucesso

Figura 3.13: Exibição de matéria completa

3.4 Design de interface

Pelo fato do aplicativo alvo desse trabalho já existir muito antes de ser proposta a classificação das notícias por assunto, a interface de usuário teve de ser adaptada, de modo que a experiência de usuário não fosse prejudicada no uso do aplicativo.

Para a mudança no design da interface foram construídos uma série de *mockups*, tais que pudessem facilitar a visão de uma nova interface ideal que comportasse a adição dos rótulos de assunto sobre cada notícia disponibilizada. Na Figura 3.14 abaixo, é possível analisar como foi projetada a interface do aplicativo antes da funcionalidade de classificação de notícias e após a adesão dessa funcionalidade:

Figura 3.14: Mockups antes e após a adoção da categorização por assunto

4 IMPLEMENTAÇÃO DA SOLUÇÃO

Este capítulo trata das funcionalidades funcionais e não-funcionais implementadas no aplicativo e, em específico, da implementação e do acoplamento do módulo classificador de notícias ao aplicativo.

4.1 Descrição de funcionalidades

O aplicativo *News* foi desenvolvido com a intenção de ser uma solução enxuta e minimalista de modo que maximizasse a memorização do usuário sobre as funcionalidades contidas na solução e, assim, provesse uma qualidade acima da média de *user experience*.

4.1.1 Timeline de notícias

Esse é o recurso principal do aplicativo. Na *timeline*, são exibidas as notícias publicadas por um veículo de comunicação. As notícias, providas através do Twitter do veículo de comunicação, podem ser atualizadas através de um movimento vertical para baixo da *timeline* por parte do usuário (padrão de design conhecido como “Pull To Refresh”). Cada notícia publicada na *timeline* pode ou não conter um ícone em formato de jornal (como mostrado na Figura 4.1). Esse ícone, possibilita que, uma vez o usuário clicando sobre ele, seja redirecionado para a matéria completa daquela notícia no site do veículo de comunicação em questão.

Figura 4.1: Timeline de notícias

4.1.2 Matéria completa

Dentro do aplicativo, é possível acessar a matéria completa sobre uma notícia publicada por um determinado veículo de comunicação (como mostrado na Figura 4.2). A matéria é mostrada dentro de um componente chamado “WebView”, o qual caracteriza-se por possibilitar páginas web serem mostradas dentro dele. Logo, esse componente pode ser colocado dentro da janela do aplicativo, evitando com que o usuário tenha que abrir a matéria completa em um navegador, hipótese que daria uma má sensação em quesito de *user experience* (UX) a esse usuário.

Figura 4.2: Matéria completa

4.1.3 Escolha de fonte de notícias

As notícias acessíveis através do aplicativo são disponibilizadas por diversos veículos de comunicação. Esses veículos, são visíveis a partir do momento que o usuário abre uma barra lateral (conhecida na plataforma Android como “Navigation Drawer” e mostrada na Figura 4.3). Uma vez escolhendo uma mídia, a *timeline* desse é exibida na tela com as notícias publicadas através de sua conta no Twitter.

Figura 4.3: Escolha de fonte de notícias

4.1.4 Seleção de fontes de notícias favoritas

As fontes de notícias de um usuário do aplicativo são facilmente acessíveis através de uma barra lateral na tela principal do aplicativo. Essa barra lateral pode ter seu conteúdo personalizado (no caso, diferentes veículos de comunicação). Como mostrado na Figura 4.4, é possível escolher as fontes favoritas de modo que elas passem a ocupar a barra lateral da tela principal do aplicativo para que o usuário possa selecionar qualquer uma delas com facilidade.

Figura 4.4: Seleção de fontes de notícias favoritas

4.1.5 Notícias rotuladas por assunto

Fontes que provém conteúdo genérico tem cada uma de suas notícias mostradas na tela principal no aplicativo com um rótulo, o qual descreve o tipo de assunto o qual a notícia se refere, como mostrado na Figura 4.5:

Figura 4.5: Notícias rotuladas por assunto

4.2 Construção do módulo classificador

O processo de construção de um módulo de classificação de notícias requer, basicamente, a geração de um conjunto de treinamento de instâncias e o treinamento de um classificador, de modo que seja gerado um módulo de classificação que possa ser acoplado ao aplicativo tratado nesse trabalho para que ele possa realizar o processo de classificação de notícias.

4.2.1 Conjunto de treinamento

O processo de classificação foi realizado com o uso de um algoritmo supervisionado. Logo, esse requer a construção de um conjunto de treinamento com instâncias já classificadas por especialistas humanos.

O conjunto de treinamento foi estruturado no formato de um arquivo .arff (Attribute-Relation File Format). Esse é um tipo de arquivo compreendido por uma ferramenta de software chamada *Weka* (WEKA, 2014), a qual contém um conjunto de algoritmos de aprendizagem de máquina que agem sobre um conjunto de dados.

Foi determinado que o aplicativo *News* fosse capaz de classificar notícias em sete categorias de assuntos diferentes. Dentre elas esportes, política/economia, tecnologia, música, cidade, trânsito e clima. A escolha por um número menor de categorias implicaria em notícias sobre temas bem definidos serem classificadas erroneamente (por exemplo, uma notícia esportiva ser classificada como ou sobre política, ou sobre clima). Já a escolha por um número muito alto de categorias obrigaria notícias a conterem em seu texto palavras-chave que são facilmente definidas como pertencentes a uma categoria devido a características comuns que categorias comuns possam compartilhar entre si. O conjunto de treinamento construído contém 107 instâncias para cada categoria de assunto. Essa quantidade foi escolhida, pois foi suficiente de modo que produzisse resultados satisfatórios. O aumento desse número não melhorou a eficácia do classificador gerado.

No arquivo de conjunto de treinamento, a classificação de uma instância é descrita da seguinte forma:

<notícia publicada>, <assunto>

Onde <notícia publicada> trata-se de um exemplo de notícia coletada de uma fonte pelo aplicativo e <assunto> é a categoria de assunto a qual a notícia pertence. Lembrando que o assunto no conjunto de treinamento é determinado por um especialista humano. Logo, o julgamento do especialista humano contém um elemento de subjetividade. Assim, o algoritmo de classificação de notícias será afetado pelo fator de subjetividade.

4.2.2 Geração do classificador

A partir da ferramenta Weka se dará a geração do classificador de notícias. O processo de geração do classificador é visível através da Figura 5.1 abaixo:

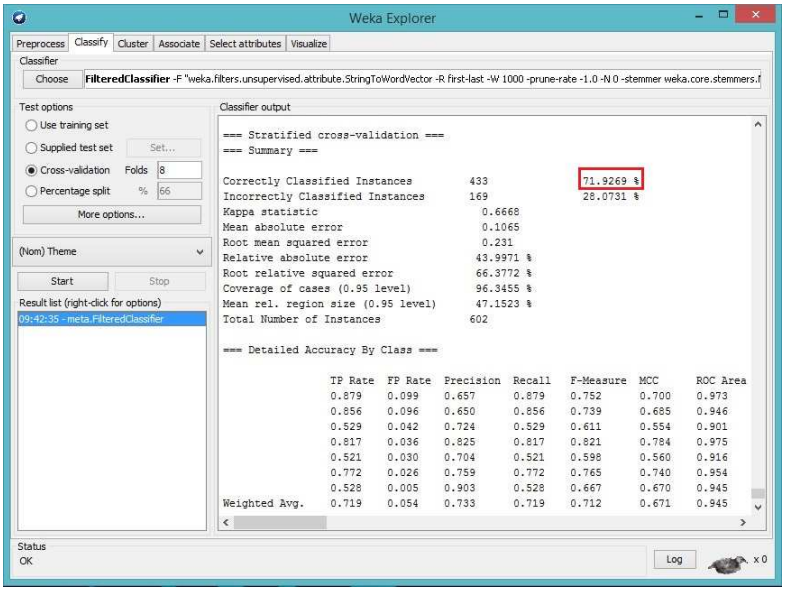

Figura 4.6: Geração do classificador de notícias

Algumas informações são relevantes nessa figura. A partir do treinamento do classificador, é desejável torná-lo o mais genérico possível na classificação de novos dados (no caso desse trabalho, novas notícias). Para avaliar a capacidade de generalização de um classificador, é usada uma técnica denominada validação cruzada (CROSS VALIDATION WIKIPEDIA, 2014). Essa técnica divide o conjunto de treinamento em um número de subconjuntos para que alguns sejam usados como conjuntos de dados de treinamento e outros de teste. O método de particionamento selecionado para a aplicação dessa técnica foi *k-fold*. Esse método caracteriza-se por dividir o conjunto de treinamento em k subconjuntos disjuntos, sendo um deles usado como conjunto de teste para avaliar a qualidade do classificador e os outros k-1 subconjuntos como conjunto de treinamento. De forma circular, o mesmo processo é repetido utilizando-se sempre um distinto subconjunto como conjunto de teste. No caso do classificador deste trabalho, o conjunto de treinamento foi particionado em oito subconjuntos, pois um maior particionamento não melhorou a acurácia do classificador. Outro elemento importante é a taxa de acertos obtidos para novas instâncias pelo classificador, no valor de 71,92%. O que significa que, em uma amostra de dez novas notícias, o classificador deduz corretamente, em média, o assunto de sete notícias.

Esse processo gera um modelo de classificador no formato de um arquivo .model, o qual será usado para a implementação do módulo (na forma de biblioteca de classes) de classificação de notícias.

4.2.3 Implementação

O arquivo .model gerado pelo *Weka* (que contém o modelo do classificador) é usado pela biblioteca que contém o módulo de classificação de notícias, conforme mostra a Figura 4.7:

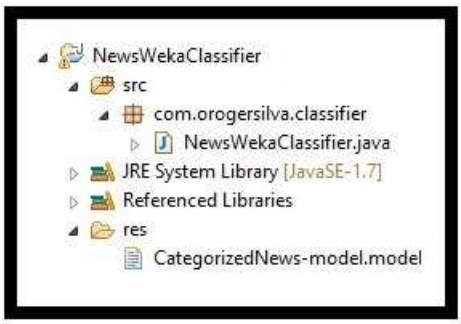

Figura 4.7: Estrutura de diretório do módulo de classificação de notícias

A classe *NewsWekaClassifier* contém, basicamente, dois métodos de interesse, de mesmo nome, sobrecarregados. O método *classify* recebe como entrada notícia(s) e, produz como saída a(s) categoria(s) de assunto, a qual (as quais) as notícias se referem. O diagrama de classe de *NewsWekaClassifier* é mostrado na Figura 4.8 abaixo:

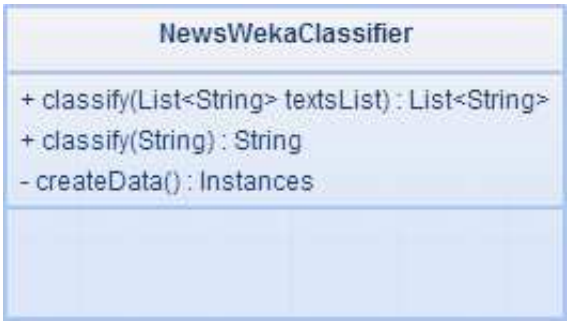

Figura 4.8: Diagrama de classe de NewsWekaClassifier

Uma vez gerado um arquivo .jar desse módulo de classificação de notícias, é realizada a integração desse módulo ao aplicativo *News*.

4.3 Adaptação do aplicativo ao módulo classificador

Devido a já existência do aplicativo *News* antes da adoção do módulo de classificação de notícias por assunto, algumas adaptações tiveram de ser realizadas de modo a comportar a funcionalidade de classificação. Dentre as adaptações estão alterações na interface gráfica, lógica de controle e base de dados.

Em primeiro lugar, é realizada a importação do módulo de classificação ao aplicativo. Para isso, o arquivo .jar teve de ser gerado a partir do projeto que contém a classe

NewsWekaClassifier mencionada anteriormente nesse trabalho. A importação do módulo é mostrada na Figura 4.9:

Figura 4.9: Importação do módulo classificador

Também a interface gráfica teve de ser adaptada com a adição de um campo de texto que conterá a descrição do tipo de assunto da notícia, conforme a Figura 4.10:

```
<TextView
    android:id="@+id/story_type_text_view"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:paddingLeft="6dp"
    android:paddingRight="6dp"
    android:layout_marginLeft="6dp"
    android:layout_marginRight="6dp"
    android:layout_alignParentLeft="true"
    android:layout_alignParentTop="true"
    android:layout_alignLeft="@+id/posted_news_clock"/>
```

Figura 4.10: Alteração no XML da interface gráfica

O aplicativo foi modelado de modo a comportar um ADT para o conteúdo de uma notícia publicada. Foi necessária a adaptação desse ADT de modo a passar a conter um atributo referente ao tipo de assunto de uma notícia. O atributo no ADT foi nomeado *storyType*, como mostra a Figura 4.11:

```
public class NewsContent implements Comparable<NewsContent> {

    private long tweetId;
    private long twitterId;
    private Date createdAt;
    private String content;
    private String fullArticleLink;
    private String mediaLink;
    private int storyType;
```

Figura 4.11: Adaptação do ADT

Adicionalmente, foi definida uma enumeração responsável por conter todos os tipos os quais uma notícia pode ser classificada. A Figura 4.12 mostra:

```
public enum StoryType {

    NONE(0), SPORTS(1), POLITICS_ECONOMY(2), TECHNOLOGY(3), MUSIC(4), CITY(5), TRANSIT(6), WEATHER(7);
```

Figura 4.12: Definição de enumeração de tipos de notícias

Uma vez uma notícia tendo sido classificada pelo classificador, esse produzirá uma cadeia de caracteres com o nome do tipo da notícia. De modo a melhorar a legibilidade

```
public static StoryType getEnumStoryType(String storyTypeStr) {

    if (storyTypeStr.equals("sports"))
        return SPORTS;
    else if (storyTypeStr.equals("politicseconomy"))
        return POLITICS_ECONOMY;
    else if (storyTypeStr.equals("technology"))
        return TECHNOLOGY;
    else if (storyTypeStr.equals("music"))
        return MUSIC;
    else if (storyTypeStr.equals("city"))
        return CITY;
    else if (storyTypeStr.equals("transit"))
        return TRANSIT;
    else if (storyTypeStr.equals("weather"))
        return WEATHER;
    else
        return NONE;
}
```

Figura 4.13: Conversão para enumeração de tipos de notícias

do código do aplicativo, foi implementado um método convertendo o nome do tipo da notícia para sua enumeração correspondente, como na Figura 4.13.

Evidentemente, uma invocação ao método de classificação é necessária a partir do objeto *NewsWekaClassifier*, como na Figura 4.14:

```
List<String> classificationResultsList = textClassifier.classify(tweetsText);
```

Figura 4.14: Invocação a método de classificação

Tendo a notícia sido coletada da fonte, é necessário atualizar o conteúdo coletado com o tipo de assunto o qual se refere, como mostrado na Figura 4.15:

```
StoryType storyType = StoryType.getEnumStoryType(classificationResult);
newsContent.setStoryType(storyType.getValue());
```

Figura 4.15: Definição do tipo de assunto

A tabela na base de dados referente aos registros de notícias coletadas necessita ter uma coluna adicional para comportar o tipo de assunto da notícia, conforme a Figura 4.16:

```
/**
 * "NewsContent" table contents definition
 * @author RogerSilva
 *
 */
public static abstract class NewsContentTable {

    public static final String TABLE_NAME = "NewsContent";

    public static final String COLUMN_NAME_TWEET_ID = "TweetId";
    public static final String COLUMN_NAME_CREATEDATTEXT = "CreatedAt";
    public static final String COLUMN_NAME_TWEETTEXT = "TweetText";
    public static final String COLUMN_NAME_TWITTER_ID = "TwitterId";
    public static final String COLUMN_NAME_FULL_ARTICLE_LINK = "FullArticleLink";
    public static final String COLUMN_NAME_MEDIA_LINK = "MediaLink";
    public static final String COLUMN_NAME_STORY_TYPE = "StoryType";

    public static final String SQL_CREATE_TABLE =
            "CREATE TABLE " + TABLE_NAME + " (" +
                    COLUMN_NAME_TWEET_ID + " " + INTEGER_TYPE + " " + "PRIMARY KEY" + COMMA_SEPARATOR +
                    COLUMN_NAME_CREATEDATTEXT + " " + TEXT_TYPE + COMMA_SEPARATOR +
                    COLUMN_NAME_TWEETTEXT + " " + TEXT_TYPE + COMMA_SEPARATOR +
                    COLUMN_NAME_TWITTER_ID + " " + INTEGER_TYPE + COMMA_SEPARATOR +
                    COLUMN_NAME_FULL_ARTICLE_LINK + " " + TEXT_TYPE + COMMA_SEPARATOR +
                    COLUMN_NAME_MEDIA_LINK + " " + TEXT_TYPE + COMMA_SEPARATOR +
                    COLUMN_NAME_STORY_TYPE + " " + INTEGER_TYPE + COMMA_SEPARATOR +
                    "FOREIGN KEY" + "(" + COLUMN_NAME_TWITTER_ID + ") REFERENCES " + NewsSourceTable.TABLE_NAME + "(" + NewsSourceTable.COLUMN_NAME_TWITTER_ID + ")" + ")";
```

Figura 4.16: Alteração na tabela de notícias

Por fim, no instante da inserção na base de dados da notícia coletada, é necessária a persistência do assunto da notícia. Uma vez que a tabela referente ao conteúdo de notícias já esteja alterada na base de dados, torna possível que a notícia seja inserida, recuperada

e mostrada na tela do dispositivo quando necessário, como mostra a Figura 4.17 logo a seguir:

```
public void addNewsContents(ArrayList<NewsContent> newsContents) {

    SQLiteDatabase db = this.getWritableDatabase();

    String sqlInsertQuery = "INSERT INTO " + NewsDbContract.NewsContentTable.TABLE_NAME + " (" +
            NewsDbContract.NewsContentTable.COLUMN_NAME_TWEET_ID + "," +
            NewsDbContract.NewsContentTable.COLUMN_NAME_CREATEDATTEXT + "," +
            NewsDbContract.NewsContentTable.COLUMN_NAME_TWEETTEXT + "," +
            NewsDbContract.NewsContentTable.COLUMN_NAME_TWITTER_ID + "," +
            NewsDbContract.NewsContentTable.COLUMN_NAME_FULL_ARTICLE_LINK + "," +
            NewsDbContract.NewsContentTable.COLUMN_NAME_MEDIA_LINK + "," +
            NewsDbContract.NewsContentTable.COLUMN_NAME_STORY_TYPE + ") " +
            "VALUES (?, ?, ?, ?, ?, ?, ?)";
```

Figura 4.17: Alteração em inserção na tabela de notícias

5 EXPERIMENTOS E RESULTADOS

Para a geração do classificador de notícias, foram realizados diversos experimentos de modo a encontrar a melhor configuração que maximizasse características de generalização do classificador para que esse realizasse o maior número de predições corretas sobre novas notícias coletadas. A adição de um stemmer foi considerada de modo a tentar melhorar a acurácia do classificador e os resultados produzidos com esse componente foram analisados.

5.1 Resultados gerados

Inúmeros experimentos foram realizados de modo a maximizar a acurácia do classificador. Após uma bateria de experimentos, a configuração ideal para o contexto do aplicativo *News* foi alcançada conforme mostra a tela do *Weka* na Figura 5.1 abaixo:

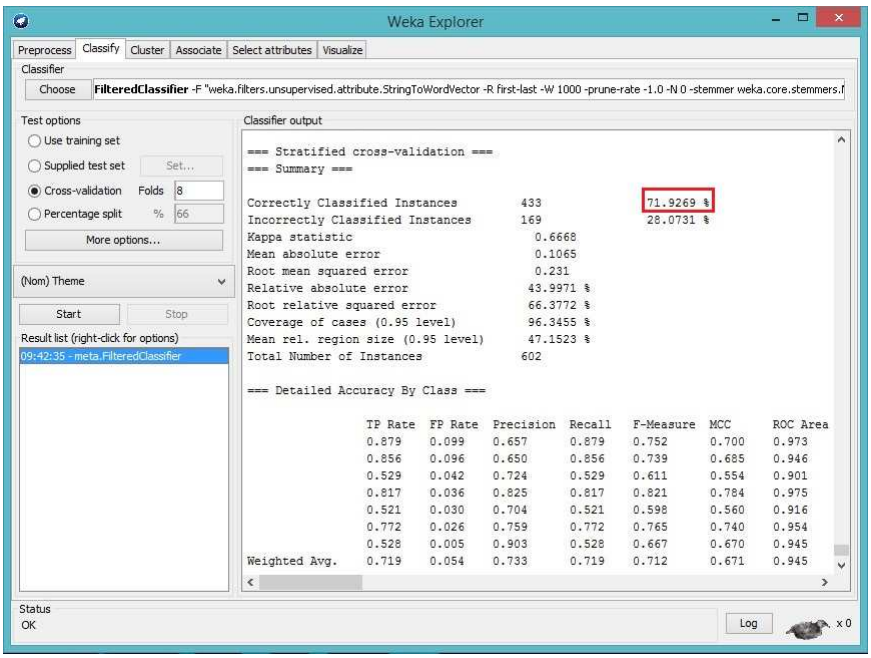

Figura 5.1: Melhor configuração gerada para o classificador

Tal configuração foi obtida com a utilização de um algoritmo de classificação Bayes

Ingênuo Multinomial (já mencionado na seção 2.4.3) e uma estratégia de testes de validação cruzada com particionamento em oito subconjuntos disjuntos (pelos motivos já mencionados na seção 4.2.2) para medir a capacidade de generalização do classificador.

O nível de acurácia atingido pelo classificador é dependente do padrão do conteúdo publicado por fontes. Pelo fato de todas notícias publicadas conterem, no máximo, 140 caracteres (já que são coletadas da rede social *Twitter*), a capacidade de generalização do classificador é prejudicada pela falta de termos contidos em uma notícia. Outro fator que influi na qualidade do classificador é a variação na forma de escrita de cada notícia disponibilizada, pois, como o arquivo de treinamento usado na construção do classificador contém notícias de variadas fontes e não existe uma padronização no estilo de escrita do conteúdo das notícias pelas diferentes fontes, há um prejuízo na qualidade da classificação. Por fim, outro elemento que prejudicou a qualidade do classificador foram as "notícias de saudações". Diversas fontes tendem a publicar como primeira notícia do dia uma saudação de modo a comunicar que as publicações daquele dia iniciaram, como mostra a Figura 5.2 abaixo:

Figura 5.2: Notícias de saudações

Esse tipo de publicação não agrega em termos de qualidade no treinamento do classificador, devido aos tipos de assuntos determinados os quais uma notícia pode ser classificada. O que resulta em uma classificação incorreta por parte do classificador, pois ele, obrigatoriamente, terá que classificá-la de alguma forma.

5.2 Experimentos com stemmer

Para melhorar a qualidade do classificador, foi experimentada a adição de um *stemmer* (ou "radicalizador") à configuração do classificador. *Stemmers* são algoritmos que tem por objetivo reduzir palavras a formas comuns de representação por meio de uma técnica chamada *conflação* que funde ou combina as morfologias variantes de um termo (FRAKES; BAEZA-YATES, 1992).

A adição desse *stemmer* ao módulo classificador trouxe uma melhora, em média, de 4,2% em relação à configuração sem *stemmer*. Os resultados dos experimentos com a adoção de *stemmer* são mostrados na Figura 5.3, a qual contém sinalizada em vermelho a melhora em relação aos resultados mostrados na Figura 5.1.

Porém, foi determinado que sua adoção não seria realizada para a solução final, pois, apesar da melhora nos resultados apresentados, eles não foram significativos o suficiente.

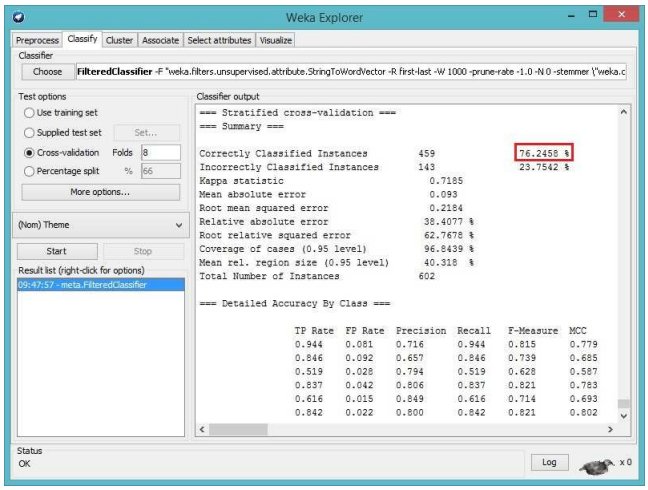

Figura 5.3: Resultado de experimentos com stemmer

E, principalmente, pelo processamento adicional requerido para classificar cada notícia e esse processamento consumir mais recursos do dispositivo móvel executando o aplicativo, o qual deve-se minimizar o uso de recursos de hardware para minimizar, também, seu consumo de bateria.

6 CONCLUSÃO

Neste trabalho foi construído um módulo de classificação de notícias, o qual age sobre notícias publicadas por uma fonte e mostra-as na tela do dispositivo com o nome do assunto o qual a notícia se refere.

Todos conceitos adjacentes para a construção do módulo foram abordados, desde caráter algorítmico até questões arquiteturais sob as quais a solução foi projetada. Também foram expostos o modo que foram realizados o design e a implementação do aplicativo, com enfoque especial sobre o módulo de classificação. Por fim, experimentos realizados durante a construção do classificador foram analisados e os porquês de não adotar estratégias mais elaboradas para a construção do classificador.

Uma limitação encontrada na fase de adaptação da funcionalidade de classificação textual foi a forma que seriam expostos os assuntos os quais cada notícia foi classificada pelo módulo de classificação no aplicativo. O design de interface preferencial de exposição dos assuntos, inicialmente, seria na forma de abas, onde notícias agrupadas por assunto estariam dispostas em suas devidas abas. Porém, devido à complexidade de manipulação dos componentes usados pela interface (já presentes no aplicativo antes da fase adaptação), foi preferível dispor uma interface mais simples com o objetivo principal de, simplesmente, fornecer ao usuário o assunto de cada notícia.

Visando trabalhos futuros, de forma a aprimorar a acurácia do módulo de classificação de notícias, é possível, uma vez coletadas um conjunto de notícias de fontes distintas, usá-las como um novo conjunto de treinamento para retreinar o classificador para que aumente as características de generalização desse e esteja mais capacitado a classificar corretamente novas instâncias.

REFERÊNCIAS

ANDERSON, D. **Kanban**: successful evolutionary change for your technology business. [S.l.]: Blue Hole Press, 2010.

ANDROID WIKIPEDIA. **Definição de Android na Wikipedia**. Disponível em: <http://en.wikipedia.org/wiki/Android_(operating_system)>. Acesso em: Novembro 2014.

BAEZA-YATES, R. **Recuperação de informação**: conceitos e tecnologia das máquinas de busca. Porto Alegre: [s.n.], 2013.

BAYES INGÊNUO WIKIPEDIA. **Definição de Bayes Ingênuo na Wikipedia**. Disponível em: <http://en.wikipedia.org/wiki/Naive_Bayes_classifier#Multinomial_naive_Bayes>. Acesso em: Novembro 2014.

COHN, M. **User Stories Applied**: for agile software development. [S.l.]: Addison-Wesley Professional; 1.ed., 2004.

CROSS VALIDATION WIKIPEDIA. **Definição da técnica de validação cruzada**. Disponível em: <http://en.wikipedia.org/wiki/Cross-validation_(statistics)>. Acesso em: Novembro 2014.

DEVELOPER ANDROID. **Dados gerais sobre uso de características em Android tais como versões Android e tamanhos de dispositivos**. Disponível em: <https://developer.android.com/about/dashboards/index.html>. Acesso em: Novembro 2014.

FRAKES, W.; BAEZA-YATES, R. **Information Retrieval**: data structures and algorithms. London, UK: Prentice Hall, 1992.

GOOGLE PLAY. **Aplicativo News no Google Play**. Disponível em: <https://play.google.com/store/apps/details?id=com.orogersilva.newsandroidapp>. Acesso em: Novembro 2014.

RICHARDSON, L.; AMUNDSEN, M. **RESTful Web APIs**. Gravenstein Highway North, Sebastopol: O'Reilly Media, 2013. 29p.

SOMMERVILLE, I. **Software Engineering**. [S.l.: s.n.], 2009.

STRATEGY ANALYTICS. **Dados sobre o uso da plataforma Android no mundo**. Disponível em: <http://blogs.strategyanalytics.com/WSS/post/2014/07/30/Android-Captured-Record-85-Percent-Share-of-Global-Smartphone-Shipments-in-Q2-2014.aspx>. Acesso em: Novembro 2014.

TRELLO. **Ferramenta Trello**. Disponível em: <https://trello.com/>. Acesso em: Novembro 2014.

TWITTER WIKIPEDIA. **Definição de Twitter na Wikipedia**. Disponível em: <http://en.wikipedia.org/wiki/Twitter>. Acesso em: Novembro 2014.

WEKA. **Homepage da ferramenta de aprendizado de máquina Weka**. Disponível em: <http://www.cs.waikato.ac.nz/ml/weka/>. Acesso em: Novembro 2014.

APÊNDICE A PESQUISA DE AVALIAÇÃO DE FUNCIONALIDADES E USABILIDADE

Foi disponibilizado a um grupo de cinco usuários o aplicativo *News* para que suas funcionalidades fossem testadas e avaliadas conforme seu nível de dificuldade de uso. Além disso, perguntas referentes à qualidade da usabilidade também foram incluídas. Ainda, questões mencionando a qualidade da classificação das notícias por assunto realizadas pelo aplicativo e se foi tida como satisfatória.

Figura A.1: Tempo de uso de Android

Figura A.2: Alterar fonte de notícias

Figura A.3: Alterar fonte de notícias

Figura A.4: Alterar fonte de notícias

Figura A.5: Alterar fonte de notícias

Figura A.6: Alterar fonte de notícias

Figura A.7: Alterar fonte de notícias

Figura A.8: Alterar fonte de notícias