Arquitetura de *machine learning* para análise de reportagens textuais em redes sociais para a detecção de *fake news*

Laura Betti Monteiro Radicchi[1], Michele Cristiani Barion[2], Adriano Ferreira[3]

Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas – Instituto Federal de Educação, Ciência e Tecnologia de São Paulo (IFSP) – Câmpus Hortolândia – São Paulo – SP – Brasil

[1]lauramradicchi@gmail.com, [2]michele_barion@hotmail.com, [3]adriano.unicamp@gmail.com

Abstract. *The automatic detection of fake news represents a big challenge nowadays and has been object of study in several areas of knowledge, such as Sociology, Computer Engineering and Psychology. Some of these researches tried to minimize the fake news impact on the real world. Therefore, this article presents the creation of an architectural design using machine learning associated with neural network and deep learning to detect fake news on social media. On this project, we used as a parameter of analysis the portuguese language.*

Resumo. *A detecção automática de fake news representa um grande desafio na atualidade e tem sido objeto de estudo de diversas áreas de conhecimento, tais como sociologia, engenharia da computação e psicologia, objetivando minimizar os impactos das notícias falsas no mundo real com rapidez e eficiência. Diante deste contexto, o objetivo do artigo é documentação arquitetural de Machine Learning associada a redes neurais para a detecção de Fake News em redes sociais, utilizando como parâmetro para análise a língua portuguesa.*

1. Introdução

O crescimento da Internet e das redes sociais modificaram a maneira de como os indivíduos consomem informação e isso pode ser justificado através de um estudo realizado por [Matsa e Shearer 2018], constatando que 67% dos adultos americanos se informam através de mídias sociais. Já no Brasil, conforme [Secom 2016], este número é um pouco menor chegando a 49%.

Como apresenta [Castells 2013], esta mudança de paradigma quanto a interação das pessoas com a informação ocorre devido a inovação dos meios de comunicação e a ampliação desse alcance para todos os âmbitos da sociedade, ocasionando de maneira muito rápida a mudança no comportamento das pessoas com a leitura das mensagens recebidas. Ocorre então, o que o autor [Castells 2013] chama de autocomunicação: a Internet e as redes sem fio como plataformas da comunicação digital. A Internet, segundo ele, é de massa e com a potencialidade de alcançar uma gama variada de receptores, além de se comunicar com um número grande de

redes que transmitem as informações digitalizadas pelo mundo, criando o que o autor chama de horizontalização. Esta horizontalização levou ao aumento do volume de informação aos quais os indivíduos têm acesso, dificultando a checagem de fatos e facilitando, assim, a circulação de notícias falsas, ou seja, as chamadas *fake news*.

Segundo [Gahirwal *et al*. 2018], *fake news* é "uma reportagem normalmente escrita para ganho pessoal, econômico ou político". Exemplificando, em períodos de eleição, este tipo de notícia pode influenciar os resultados devido a nebulosidade que pode ocasionar na veracidade das informações. O Brasil demonstra isso nas eleições de 2018 que, conforme a [OEA 2018], "a propagação *on-line* de desinformação e notícias falsas foi uma constante durante a fase pré-eleitoral e estendeu-se, inclusive, ao dia das votações".

A divulgação de notícias falsas não é um fenômeno recente, como pode ser citada a notícia do boimate publicada pela revista Veja em 1983, afirmando que cientistas alemães haviam criado um híbrido que era feito através do boi e do tomate. Esta reportagem foi baseada em um artigo da revista britânica *New Scientist*, tendo sido usada como uma piada para o dia 1º de abril, sendo a mesma desmentida pela própria revista em julho de 1983. Atualmente, com as redes sociais há uma maior facilidade de se manipular as informações que são circuladas e ainda em um curto tempo, como cita [Shao *et al.* 2018] quando diz que as informações "podem ser facilmente exploradas para manipular a opinião pública devido ao baixo custo para se produzir *sites* fraudulentos e grande volume de perfis controlados por *softwares* ou páginas conhecidas como *social bots* [...]".

Conforme [Shao *et al.* 2018] a propagação de notícias falsas ocorre, devido a "uma complexa mistura social, cognitiva e de algoritmos enviesados que contribuem para a nossa vulnerabilidade e para a manipulação por notícias falsas *on-line*."

Diante deste cenário informativo e a sociedade cada vez mais explorando as redes sociais, há a preocupação em saber se as notícias publicadas, independente do segmento da reportagem, é verídico ou não.

Neste contexto, a proposta deste trabalho é uma arquitetura de aplicação que faça a análise de textos publicados em redes sociais na língua portuguesa e detecte o grau de *fake news*. A motivação deste projeto surgiu a partir da necessidade de criar mecanismos que compreendam a propagação e a estruturação linguística deste tipo de notícia.

As demais seções do artigo se organizam da seguinte maneira:

- Seção 2: apresentação de trabalhos correlatos que abordam o uso de *machine learning* para detectar *fake news* na Internet;
- Seção 3: fundamentação teórica dos conceitos abordados neste trabalho;
- Seção 4: abordagem da metodologia, das técnicas e dos materiais utilizados para o desenvolvimento da arquitetura da aplicação;
- Seção 5: desenvolvimento da proposta utilizando os materiais, métodos e metodologias

descritas na seção anterior;

- Seção 6: apresentação das considerações finais acerca do projeto e a possibilidade de desdobramento em trabalhos futuros.

2. Trabalhos Correlatos

Esta seção aborda três trabalhos correlatos que associam a aplicação da *machine learning* para detectar *fake news* na Internet.

- O *BS Detector* é um *plug-in*, disponível para os navegadores *Mozilla Firefox* e *Google Chrome*, que alerta os usuários sobre *Fake News*, discurso de ódio e teorias da conspiração. Para isto a aplicação pesquisa nas páginas da *web*, *URLs* e referências que foram marcadas como não confiáveis em seu banco de dados [Sieradski 2016]. O *BS Detector* mostra uma mensagem de alerta se o artigo encontrado é falso. Entretanto, não especifica a porcentagem de erro e nem classifica as notícias em níveis de "veracidade" ou "falsidade". Em 2017 o *Facebook* disponibilizou-o para analisar o aumento de *Fake News* na plataforma, entretanto a rede social suspendeu o uso do *plug-in* alegando o desenvolvimento de uma ferramenta própria [Gahirwal *et al.* 2018].

- *Flock Fake News Detector FND* é um recurso da plataforma colaborativa de mensagens *Flock.* Este recurso foi implementado em 2017 e funciona da seguinte maneira: quando duas ou mais pessoas estão conversando no *Flock* e uma delas compartilha um *link*, o *FND* é ativado e seu algoritmo checa se a *URL* é confiável ou não. Caso seja, uma barra verde aparece ao lado da prévia da notícia. Caso não, uma barra vermelha aparece ao lado do prévia e o usuário é notificado [Flock 2017]. Esta verificação é feita a partir da referência cruzada com os bancos de dados da aplicação. Até o início do ano de 2017 o *Flock Fake News Detector* possuía mais 600 fontes de notícias falsas confirmadas [Marketwired 2017].

- *FakeChatBot* é uma aplicação em português que faz o uso de *chatbots* para auxiliar usuários a detectar rumores e notícias falsas na Internet. Os usuários inserem um trecho textual e a partir da interação com *chatbot* uma *api* recebe este texto e realiza uma busca em um *dataset*. Este *dataset* contém rumores e notícias falsas além de *crawlers* que percorrem *sites* da Internet buscando mais conteúdo e armazenando no *dataset*. É retornado ao usuário um resultado se a notícia é falsa ou não [De Lima Araujo *et al.* 2018].

O *dataset LIAR* foi criado por [Wang 2017] para utilizar técnicas de *Machine Learning* e Processamento de Linguagem Natural (PNL) para detectar *Fake News*. Este *dataset* contém aproximadamente 12.800 declarações retiradas da *api* do *site politcfact.com*, sendo uma página *web* ganhadora do prêmio *Pulitzer* que produz a checagem de fatos da política americana [Wang 2017]. Cada uma das declarações foram avaliadas por editores do *site* de acordo com o seu nível de veracidade. Os criadores consideraram seis etiquetas para definir o nível de veracidade sendo *pants-fire* definida como *Fake News*, seguindo em ordem crescente até chegarmos na etiqueta verdadeira, utilizada para notícias de fontes confiáveis. O *LIAR* pode ser utilizado para

treinamento de *Machine Learning* para a detecção de *Fake News*.

Na Tabela 1, apresenta-se uma comparação entre as principais funcionalidades das aplicações descritas acima com a arquitetura proposta neste trabalho. Considera-se N/A quando não se aplica a funcionalidade na ferramenta.

Tabela 1. Tabela comparativa das funcionalidades das diferentes aplicações

Requisitos	*Bs Detector*	*Flock-FND*	*FakeChatBot*	Artefato
Informar procedência da *URL*	Sim	Sim	N/A	Sim
Utiliza *PNL*	N/A	N/A	Sim	Sim
Machine Learning	N/A	N/A	Sim	Sim
Dataset em português	N/A	N/A	Sim	Sim
Informar porcentagem de veracidade da notícia	N/A	N/A	N/A	Sim
Opensource	Sim	N/A	N/A	Sim
Chatbot	N/A	N/A	Sim	Sim

3. Fundamentação teórica

Nesta seção são abordados os principais conceitos que fundamentam os aspectos técnicos e de negócio quanto ao desenvolvimento deste trabalho.

3.1. Arquitetura Monolítica e Microsserviços

Uma arquitetura monolítica descreve uma aplicação ou sistema composto por um único serviço. Este serviço normalmente utiliza o padrão de projeto *Model View Controller* (*MVC*) e se divide em três camadas sendo: *User Interface (UI*), lógica de negócio do sistema e, por fim, a camada que faz a comunicação com a base de dados [Kalske *et al.* 2018].

Já a arquitetura de microsserviços são serviços pequenos e autônomos que funcionam juntos. São entidades separadas, de maneira que devem ser implementados como serviços isolados em plataformas isoladas ou com o seu próprio sistema operacional. A comunicação entre os serviços ocorre entre chamados de *request* e *response* [Newmann 2015].

3.2. *Machine Learning* e Redes Neurais

A Inteligência Artificial (IA) é uma área da ciência da computação que surge logo após a Segunda Guerra Mundial. Pode ser definida em quatro categorias diferentes, sendo elas: pensando como um humano; pensando racionalmente; agindo como seres humanos e agindo racionalmente. Em cada uma destas categorias existem modelos representativos práticos e teóricos sendo respectivamente: a abordagem do teste de Turing, a estratégia de modelagem cognitiva, a abordagem das leis do pensamento e a abordagem de agente racional [Russell e

Norvig 2013].

Atualmente é uma área interdisciplinar e abrange diversos subcampos principalmente *machine learning e* redes neurais. Tem como objetivo a criação e desenvolvimento de técnicas computacionais e a construção de *softwares* que sejam capazes de aprender de maneira independente. Redes Neurais é um desdobramento da área da inteligência artificial e podem ser definidas como uma máquina desenvolvida para replicar a maneira como o cérebro humano realiza determinada tarefa. Esta rede pode ser implementada através de componentes eletrônicos ou através de um algoritmo de programação [Haykin 2001].

3.3. Processamento Natural de Linguagem e *Chatbots*

Alan Turing, o pai da computação moderna, quando criou o teste de Turing (1950), afirmou que para que o computador fosse capaz de passar no teste ele deveria possuir as seguintes capacidades: processamento de linguagem natural, aprendizado de máquina, representação do conhecimento e raciocínio automatizado [Russell e Norvig 2013]. O Processamento Natural de Linguagem (PNL) é um conjunto de técnicas variadas da computação para o processamento de textos, sejam eles orais ou escritos, com a finalidade de obter um processamento de linguagem para uma variedade de tarefas e aplicativos [Liddy 2001].

Já *chatbots* podem ser conceituados como *softwares* que interagem com usuários utilizando linguagem natural. Seu objetivo é simular uma conversação humana e sua arquitetura normalmente integra um modelo de linguagem associado a um algoritmo computacional para emular a comunicação entre usuário e máquina [Shawar e Atwell 2007].

3.4. Clusterização, Vetorização, Tokenização e Classifiers

O principal objetivo da PNL é transformar a linguagem humana em um modelo que seja manipulável por computadores e algoritmos. Devido a complexidade da tarefa, estudiosos e cientistas optaram por dividirem em sub-tarefas tais como: clusterização, vetorização, tokenização e *classifiers* [Collbert e Weston 2008]. Os próximos parágrafos descrevem cada uma destas sub-tarefas.

- Clusterização é o agrupamento de objetos similares. Objetos que diferem em detalhes insignificantes recebem o mesmo nome e podem ser tratados da mesma forma. As principais funções da clusterização são: nomear, exibir, resumir e prever. Desta maneira, todos os objetos que estão dentro de um mesmo *cluster* recebem o mesmo nome [Hartigan 1975].

- Vetorização é a distribuição representativa de dados em vetores a fim de auxiliar algoritmos de aprendizado a obter uma melhor performance agrupando itens similares [Mikolov *et al.* 2013].

- Tokenização é o processo de dividir blocos de textos em frases, palavras, símbolos ou elementos significativos chamados de *tokens* [Verma *et al.* 2014]. Seu principal objetivo é explorar palavras dentro de contextos frasais para identificação de palavras-chaves.

- Classifiers são algoritmos de *machine learning* que mapeiam dados e os classificam de

acordo com categorias específicas. A classificação preditiva é a tarefa de aproximação de uma função de mapeamento (f) de variáveis de *input* (x) para variáveis de output (y) [Sebastiani 2002].

3.5. *Design Science Research Methodology (DSRM*)

DSRM (Design Science Research Methodology) é um processo metodológico para projetar artefatos a fim de solucionar problemas, avaliar o que foi projetado e comunicar os resultados. Esta metodologia é composta por seis etapas, descritas a seguir [Peffers *et al.* 2007]:

1. Identificação do problema e motivação: nesta etapa define-se o problema de pesquisa e a justificativa;

2. Definição de objetivos para uma solução: com os conhecimentos obtidos na etapa anterior infere-se uma solução. As soluções propostas podem ser quantitativas ou qualitativas;

3. *Design* e desenvolvimento: criação do artefato. Tais artefatos podem ser construtores, modelos, métodos, entre outros. Nesta etapa deve-se determinar as funcionalidades e a arquitetura;

4. Demonstração: demonstrar o uso do artefato para resolver uma ou mais instâncias do problema;

5. Avaliação: nesta etapa é observado e avaliado o desempenho do artefato para a solução do problema;

6. Comunicação: divulgação do problema e da relevância da solução.

As etapas da *DSRM* são representadas na Figura 1.

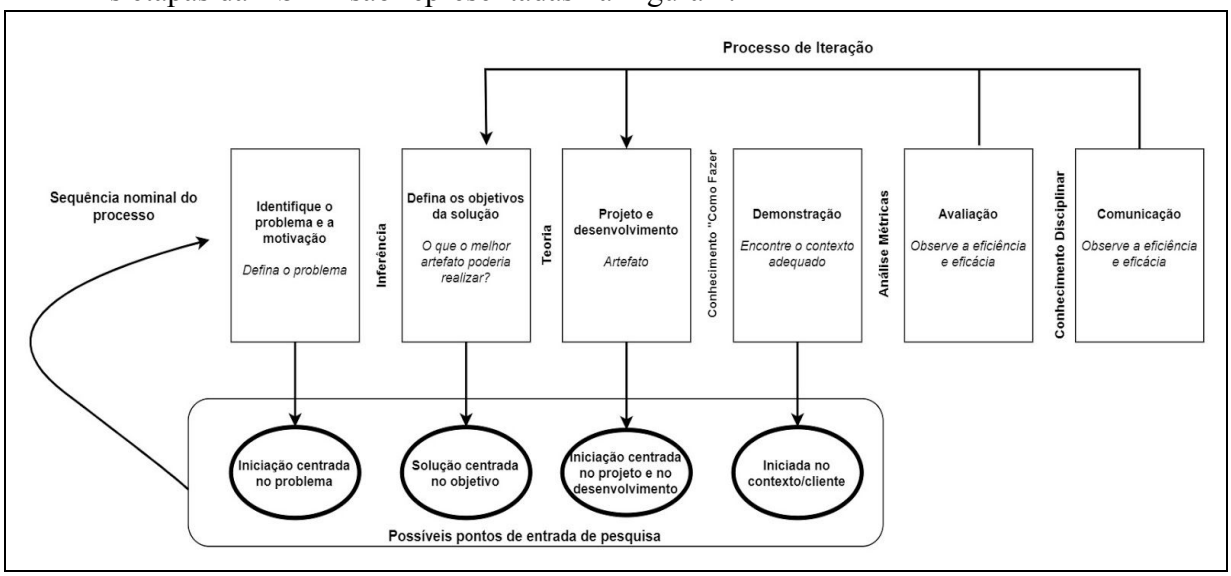

Figura 1. Etapas do *DSRM*

4. Metodologia

Esta seção aborda os métodos utilizados para apresentação da arquitetura proposta, sendo:

- Um estudo bibliográfico sobre propagação de *fake news* na Internet, arquitetura monolítica, microsserviços, *Machine Learning,* redes neurais, processamento natural de linguagem, *chatbots*, clusterização, vetorização, classificação e *DSRM.*
- Aplicação da *DSRM* (*Design Science Research Methodology*) para construção da arquitetura como um artefato.
- Documentação dos requisitos funcionais através do diagrama de caso de uso.
- Documentação dos requisitos não funcionais.
- Criação de um *dataset* contendo *fake news* e notícias de fontes confiáveis.
- Abordagem do desenho arquitetural da aplicação utilizando arquitetura monolítica e microsserviços.

5. Desenvolvimento

Esta seção aborda a criação da arquitetura do artefato conforme a *DSRM*, sendo dividida nas seguintes subseções: o uso da *DSRM* no desenvolvimento, documentação de requisitos funcionais e não funcionais, detalhamento do desenho arquitetural e definição de estórias de usuário.

5.1. O uso da *DSRM* para o desenvolvimento da arquitetura

A revisão sistemática de literatura ajudou a fomentar a solução do problema levantado no início do projeto. Isto incluiu a identificação de formas, técnicas, desafios e ferramentas utilizadas para a detecção de *fake news* em redes sociais.

Desta maneira, alguns objetivos foram delimitados a fim de desenhar o artefato arquitetural através dos resultados anteriores, sendo eles: o uso de PNL para interpretação do texto das notícias, o desafio de implementar uma arquitetura voltada a microsserviços para inteligência artificial. Esta sessão é o terceiro passo da *DSRM*, *design* e desenvolvimento, ilustrando como o referencial teórico e os objetivos para a solução foram traduzidos em um artefato.

5.2. Requisitos Funcionais e Não funcionais

Nesta seção são elencados os requisitos funcionais e não funcionais do artefato. Os principais requisitos funcionais são:

- Inserir trecho textual: o usuário insere parte ou uma notícia integralmente;
- Inserir *URL*: o usuário insere uma *URL* de uma notícia;
- Retornar procedência da *URL*: o *chatbot* retorna para o usuário se aquele *site* produz notícias confiável ou não;

- Retornar porcentagem de veracidade da notícia: o *chatbot* retorna para o usuário a porcentagem de veracidade da notícia;
- Interpretar texto inserido: o *chatbot* usando PNL é capaz de interpretar os textos inseridos pelo usuário;
- Acessar *datasets*: a *Machine Learning* acessa o *dataset* para realizar os processos de aprendizagem;
- Uso de PNL: a *Machine Learning* utiliza técnicas de PNL para aprendizagem e classificação de *fake news.*

A partir destes requisitos funcionais foi criado um caso de uso do artefato, conforme a Figura 2:

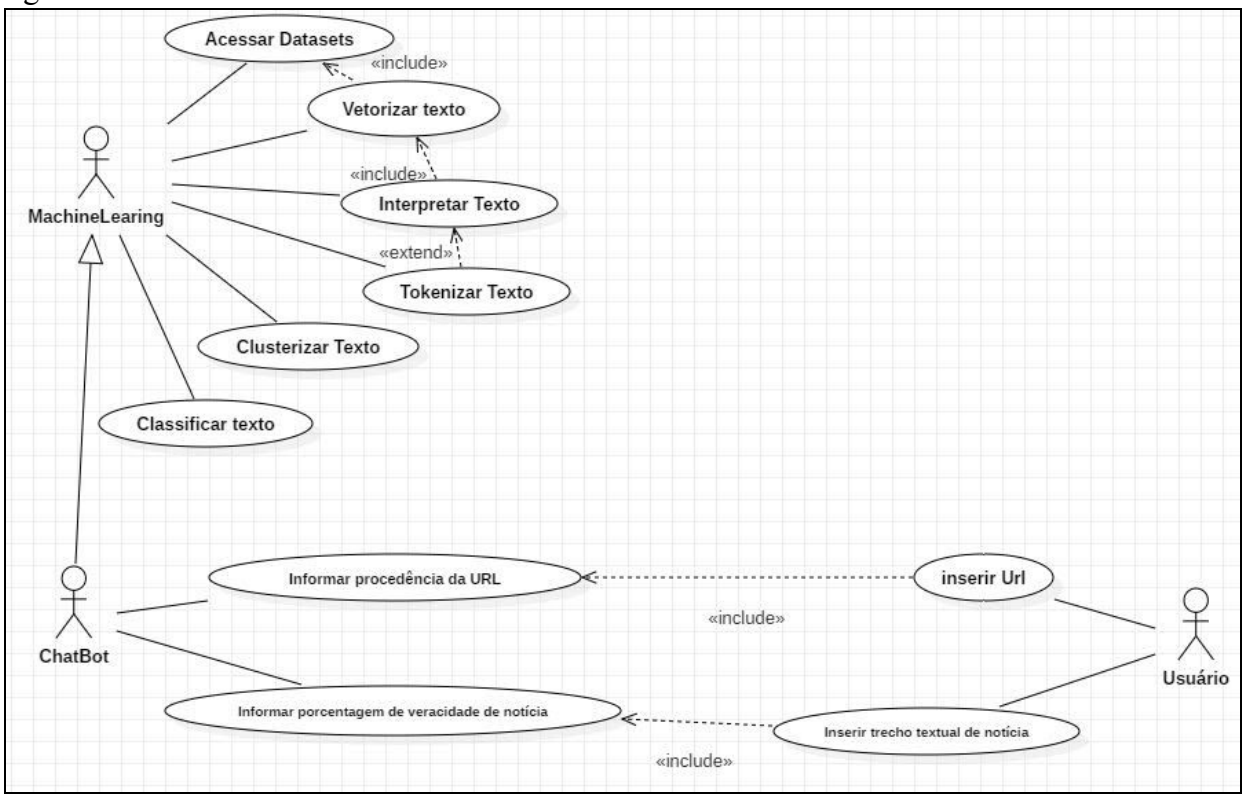

Figura 2. Diagrama de Caso de Uso do Artefato

Quanto aos principais requisitos não-funcionais do artefato são definidos a seguir:

- O artefato deve funcionar nas plataformas *web* e *mobile;*
- O artefato deve ser seguro, a proteção de dados de usuário deve estar em conformidade com a Lei Geral de Proteção de Dados (LGPD)[1];

[1]A LGPD é uma lei que demanda a proteção e confidencialidade de dados sensíveis. Disponível em: http://www.planalto.gov.br/ccivil_03/_ato2015-2018/2018/lei/L13709.htm.

- O *dataset* de notícias deve estar em formato *JSON* (*JavaScript Object Notation*) e deve ser armazenado em um banco de dados NoSQL;
- O processo de aprendizagem da *Machine Learning* deve ser protegido com camadas de segurança;
- A conexão entre *back-end* e *front-end* deve ser feita via serviços *REST*;
- A base de dados com *URLs* deve ser um banco de dados, podendo ser relacional, NoSQL ou uma tabela do Excel.

5.3. *Datasets* de notícias

Foi criado um *dataset* contendo *fake news* e notícias de fontes confiáveis, sendo construído no formato *JSON*.

Para criação deste *dataset* as notícias falsas foram retiradas de redes sociais como *Twitter*, *Facebook* e *Whatsapp*, sendo utilizados como fontes de *fact-checking*[2]: Agência Lupa, eFarsas e Aos fatos. Para as notícias de fontes confiáveis foram utilizados os jornais Folha de São Paulo, Estado de São Paulo, O Globo e El País.

A estrutura do *JSON* possui os seguintes atributos: fonte, título, texto e *label* significando respectivamente, de onde a notícia foi retirada, o título da notícia se houver, o texto integral na notícia e a classificação da notícia, ou seja, se é uma *fake news* ou se é uma notícia de fonte confiável. A Figura 3 traz uma demonstração.

Figura 3. *Dataset* contendo *fake news* e notícias de fontes confiáveis

O *dataset* foi composto por 120 notícias sendo 50 delas notícias de fontes confiáveis e 70 notícias falsas. A coleta dessas notícias foi feita da seguinte maneira: foi disparado um formulário para um pequeno grupo de pessoas, na qual elas tinham a possibilidade de enviar notícias encontradas em redes sociais diversas. Era necessário informar o texto da notícia, de onde foi retirada e a *URL* (se houvesse). Além deste formulário foi feita a busca por notícias em grupos do *WhatsApp,* perfis do *Twitter* e *Facebook*, além de *sites* de jornais e nos perfis destes nas redes sociais.

Após agrupar todas estas notícias foi dado início ao processo de *fact-checking*, utilizando as fontes mencionadas anteriormente. Quando confirmado que uma notícia era falsa, no campo

[2] *Fact-Checking* é uma checagem de fatos, ou seja um confrontamento de histórias com dados pesquisas e registros. É uma maneira de qualificar o debate público por meio da apuração jornalística. Disponível em: https://apublica.org/2017/06/truco-o-que-e-fact-checking/.

label do *JSON* era associado o valor “notícia falsa”. Caso a notícia fosse verdadeira era colocado o valor de “notícia verdadeira” no campo label do *JSON*. Este fluxo é representado no diagrama da Figura 4.

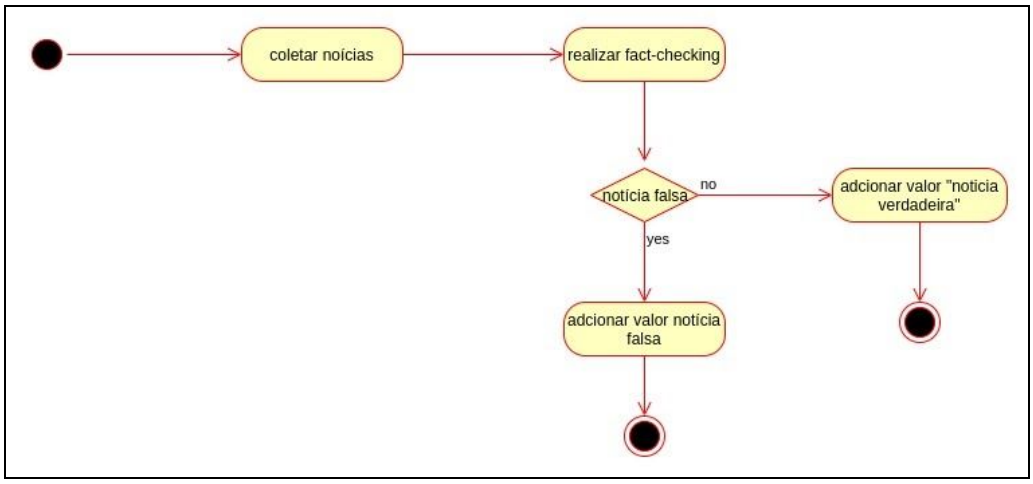

Figura 4. Diagrama de fluxo para criação do *dataset*

5.4. Arquitetura Monolítica X Microsserviços

Para a criação da arquitetura do artefato, foram utilizados dois paradigmas, sendo eles o monolítico e o de microsserviços. Esta abordagem tem o objetivo de delinear quais as vantagens e desvantagens de ambas na criação do artefato.

Para a arquitetura monolítica, a aplicação foi dividida utilizando o modelo de projetos *MVC*, na qual a camada *view* é representada pela interface do *chatbot*. Já na camada *model* estão as regras à respeito do processamento e da classificação de dados e, por fim, a camada *controller* faz a conexão com o banco de dados, estando contidos os *datasets* e as *URLs* que publicam notícias não confiáveis. A Figura 5 demonstra a exploração do conceito apresentado.

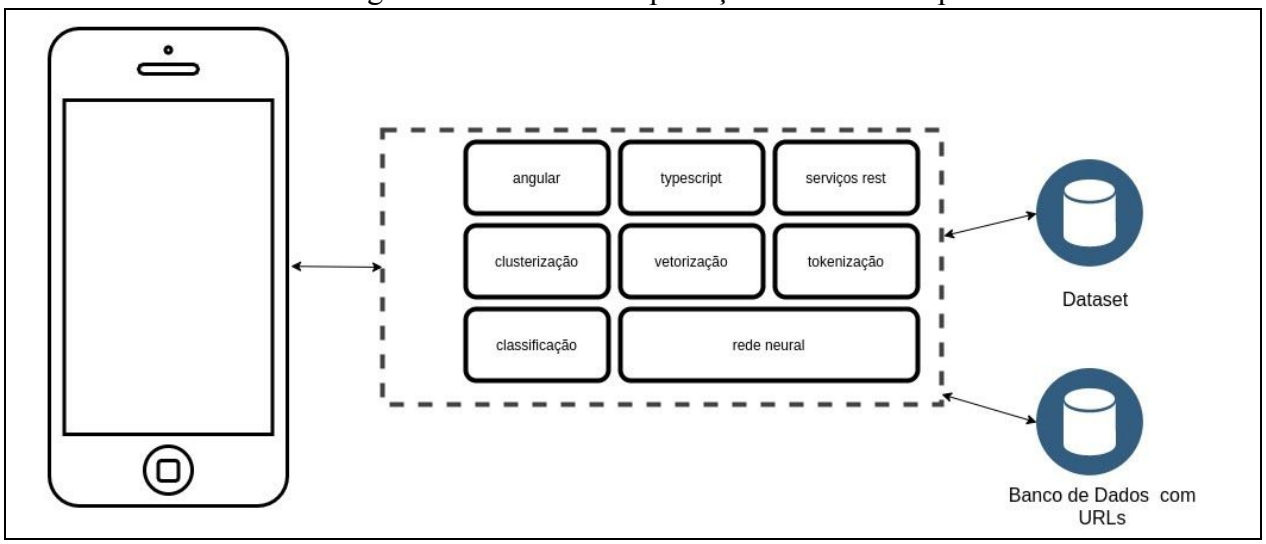

Figura 5. Arquitetura Monolítica

A arquitetura da Figura 5 é executada da seguinte maneira:

- O *front-end* troca informações com as camadas de controle e modelos via serviço *REST*.
- O usuário insere as informações na interface gráfica do *chatbot* que, realiza uma requisição *REST*.
    - Caso o dado inserido seja uma *URL*, o *chatbot* acessa diretamente o banco de dados contendo *URLs* e realiza uma *query* a fim de verificar se a *URL* inserida se encontra na base de dados. Caso o resultado da *query* seja positivo, o *chatbot* devolve para o usuário uma resposta afirmando que aquela *URL* não produz notícias confiáveis.
    - Caso o dado inserido seja um texto serão executados os processos da PNL: clusterização, vetorização e tokenização. Após o dado é enviado para a rede neural que irá classificar a porcentagem de veracidade da notícia.

Para a arquitetura orientada a microsserviços foram considerados os requisitos básicos para sua construção, criando-se serviços pequenos e autônomos. A estrutura desta arquitetura é representada na Figura 6.

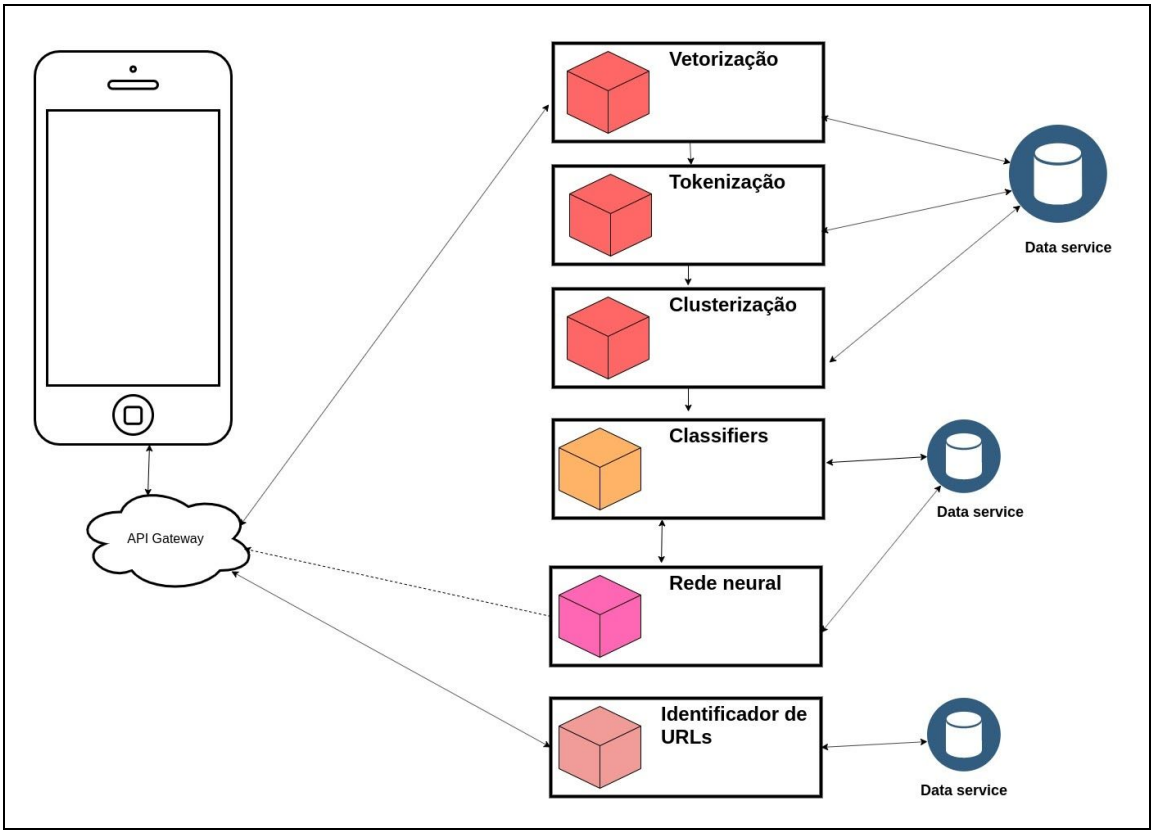

Figura 6. Arquitetura orientada a microsserviços

A arquitetura orientada a microsserviços, conforme apresentada na Figura 6, realiza as seguintes operações: o *front-end* faz requisições para a *API gateway*, dependendo de qual informação for enviada - *URL* ou trecho textual - diferentes microsserviços serão acionados.

- Caso seja inserido uma *URL*, a *API gateway* irá requisitar via serviço *REST*, o serviço de identificador de *URLs* que irá acessar seu respectivo banco de dados. Será executada uma *query,* caso a *URL* esteja no banco é devolvido para o usuário a informação de que este *site* não é uma fonte de notícias confiáveis.

- Uma vez que o usuário insere um trecho textual, o primeiro serviço a ser chamado é o de vetorização. Para ser classificado pela *machine learning* os dados precisam ser transformados em vetor.

    - Logo em seguida a informação é tratada de acordo com os métodos PNL. Assim, se houver necessidade serão utilizados os métodos de tokenização e clusterização. Isto se dá ao fato de muitas vezes existirem palavras no trecho inserido que já foram tokenizadas anteriormente.

    - Por fim os dados serão classificados e enviados para a rede neural que irá fazer a classificação de veracidade da notícia. Esta classificação será feita em porcentagem e o valor é retornado para o usuário.

Considerando as duas arquiteturas apresentadas, destacam-se as vantagens quanto ao uso:

- A arquitetura monolítica permite que o desenvolvedor não tenha maiores preocupações em relação ao monitoramento de performance, uma vez que está trabalhando com um código único. Existe uma maior facilidade para testes e *debug* da aplicação, uma vez que uma aplicação monolítica é uma unidade indivisível e, consequentemente, havendo maior facilidade em executar testes *end-to-end.* Possui um único *deploy*.

- O uso da arquitetura orientada a microsserviços permite uma melhor definição dos domínios do artefato. Esta arquitetura permite que sejam utilizadas, se necessário, diferentes linguagens de programação para escrever cada um dos serviços. Permite a escalabilidade de forma individual, ou seja, é possível escalar determinados serviços quando necessário. Possui alto nível de agilidade uma vez que qualquer defeito na aplicação afeta apenas um serviço, desta maneira as mudanças e correções acontecem de maneira que não afetam a aplicação inteira.

Quanto às desvantagens do uso associadas as duas arquiteturas apresentadas, destacam-se:

- Na arquitetura monolítica não é possível escalar apenas um trecho do código. Fazer alterações no código torna-se uma tarefa complexa, já que todas as modificações afetam o sistema como um todo. Dificulta a aplicação de novas tecnologias, afinal implicaria na refatoração de todo o código.

- Na arquitetura orientada a microsserviços existe uma complexidade extra, uma vez que microsserviços é um sistema distribuído e cabe ao desenvolvedor escolher e criar as

conexões entre os módulos e os bancos de dados. A arquitetura é um sistema complexo de múltiplos sistemas e banco de dados, desta maneira todas as conexões devem ser tratadas com cuidado. Quando é criada uma aplicação com microsserviços podem ocorrer problemas com configuração, métricas, *health checks* entre outros componentes. Como é uma arquitetura composta por componentes diferentes, criar testes *end-to-end* pode ser algo complexo.

5.5. Estórias de Usuário e materiais para implementação

Foram delineadas as estórias de usuário para a implementação do artefato. As estórias foram divididas em duas categorias, sendo elas: construção do *front-end* e construção da *machine learning*. Estas tarefas abrangem aspectos gerais da construção do artefato, podendo ser executadas de maneira concorrente.

- *Front-end*:

  Estória: Como usuário eu preciso de uma interface gráfica para conversar com o *chatbot*.

  Sub-Tarefa 1: *setup* do ambiente de programação para o *front-end*;

  Sub-Tarefa 2: criação do componente da tela do *chatbot;*

  Sub-Tarefa 3: criação dos serviços *REST* e dos *endpoints*;

  Sub-Tarefa 4: Caso seja escolhida a arquitetura orientada a microsserviços deve ser feita a criação do *back-end* do *front-end*, serviço que irá fazer a conexão entre ambos.

- *Machine Learning*:

  Estória 1: Como administrador do sistema eu preciso vetorizar os *datasets*.

  Estória 2: Como usuário eu quero inserir uma *URL* de notícia e saber se ela é uma fonte confiável ou não.

  Sub-Tarefa 1: criar o banco de dados contendo as *URLs* de fontes não confiáveis;

  Sub-Tarefa 2: criar *endpoint* de chamada de método;

  Sub-Tarefa 3: criar *query* para devolver resposta para o usuário.

  Estória 3: Como administrador preciso implementar um algoritmo para classificar *cluster* gerado pelo meu tratamento do *dataset*.

  Estória 4: Como administrador do sistema preciso implementar um algoritmo para clusterizar o *dataset*.

  Sub-Tarefa 1: retirar *stopwords* do *dataset*;

  Sub-Tarefa 2: *stemming* (reduzir palavras para as suas respectivas raízes etimológicas);

  Sub-Tarefa 3: tokenizar *dataset*;

  Sub-Tarefa 4: criar *term frequency–inverse document frequency (tf-df*), um número estatístico que informa a importância de uma palavra dentro de um

documento;

Sub-Tarefa 5: implementar algoritmo para clusterização;

Sub-Tarefa 6: criar multidimensional *scaling* (*MDS*) do *dataset*.

Estória 5: Como administrador preciso implementar uma rede neural para treinar o *chatbot*.

Sub-Tarefa 1: criar matriz de incorporação de vetores;

Sub-tarefa 2: criar camada convolucional *pooling,* preparação de mapa de características condensadas.

Para a execução destas tarefas, o presente trabalho sugere o uso de alguns algoritmos e *frameworks* específicos, conforme abordagem da Tabela 2.

Tabela 2. Materiais para desenvolvimento

Tarefa	Algoritmo/ Framework	Motivo	Materiais para Referência
Desenvolvimento do *Front-End*	Angular	Framework que funciona em múltiplas plataformas, escalável e permite ao desenvolvedor um maior controle sob o código	https://angular.io/
Clusterizar Dataset	K-means	O método *k-means* é utilizado para a quantificação de vetores. O algoritmo tem como objetivo particionar n observações em k *clusters*. De tal maneira que, cada observação pertence ao *cluster* com o significado mais próximo	Hartigan, J. A. (1975). Clustering Algorithms
Vetorizar dataset	*word2vec*	Este modelo é uma rede neural de duas camadas treinada para reconstruir o contexto linguístico das palavras	https://pathmind.com/wiki/word2vec Mikolov, T. Sutskever, I., Chen, K., Corrado, G., Dean, J. (2013) Distributed Representations of words and Phrases and their Compositionality.
Remover Stopwords do dataset	NLTK	A biblioteca é implementada como uma coletânea independente de módulos, onde cada um define os tipos básicos de dados a serem processados.	http://www.nltk.org/howto/portuguese_en.html Loper, E., Bird S.(2002) NLTK: The Natural Language Toolkit.
Implementar Rede neural	TensorFlow	É um sistema de *Machine Learning* que opera escalado e em ambientes heterogêneos.	https://www.tensorflow.org/ Abadi M. et al.(2016) TensorFlow: A System for Large-Scale Machine Learning. In: Proceedings of the 12th USENIX Symposium on Operating System Design and Implementation (OSDI'16).

			Savannah, GA, USA. pp.265-283.
Implementar algoritmo para classificar cluster	XGBoost	É utilizada para classificação e problemas de regressão, produzindo um modelo de predição em formato de árvores de decisão. Cria seus modelos e os generaliza por meio de uma otimização arbitrária de uma função de diferenciação de custos	https://xgboost.readthedocs.io/en/latest/ Chen,T., Guestrin,C. (2016). XGBoost: A scalable Tree Boosting System

6. Considerações finais

Considerando a maneira como a população tem utilizado as redes sociais para manter-se informado, existe uma necessidade latente de fazer com que este processo seja mais rápido a fim de minimizar os impactos que as notícias falsas causam no cotidiano. O uso de técnicas de *Machine Learning* associadas a um *chatbot* pode acelerar o processo da checagem de fatos. Uma vez que o processo estaria mais acessível a todos os usuários de redes sociais.

Através do uso da *DSRM* foi possível desenhar uma arquitetura monolítica e uma arquitetura orientada a microsserviços para a implementação do artefato. Entretanto, é importante destacar que a escolha do melhor paradigma arquitetural está associada à equipe de desenvolvedores.

A apresentação de duas possibilidades arquiteturais, uma monolítica e a outra orientada a microsserviços, propõe para trabalhos futuros a implementação do artefato utilizando uma das duas arquiteturas, considerando as ferramentas apresentadas.

Como segunda proposta para futuros estudos é o desenvolvimento de uma *API* para conectar o *chatbot* com as principais redes sociais, além da automatização da coleta de notícias para o *dataset* utilizando *crawlers* e o aumento de labels para a classificação das notícias e estruturação dos dados.

O desenvolvimento deste trabalho proporciona a prática dos conhecimentos adquiridos nas disciplinas de Engenharia de *Software*, Análise Orientada a Objetos, Arquitetura de *Software* e Inteligência Artificial.

Referências

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y. e Zheng, X. (2016) "TensorFlow: A System for Large-Scale Machine Learning", https://www.usenix.org/system/files/conference/osdi16/osdi16-abadi.pdf, Novembro.

Castells, M. (2013), Redes de Indignação e Esperança - Movimentos sociais na era da internet, Zahar.

Collbert, R. e Weston, J. A. (2008). Unified Architecture for Natural Language Processing: Deep

Neural Networks with Multitask Learning. In *Proceeding ICML '08 Proceedings of the 25th international conference on Machine Learning*, páginas 160-167. Helsinki, Finlândia.

Chen, T. e Guestrin, C. (2016). XGBoost: A scalable Tree Boosting System. In *arXiv:1603.02754v3 [cs.LG]*, páginas 1-13.

De Lima Araujo, Y., Chares, A. C. e Oliveira, J. S. (2018). Identificação de fake news: uma abordagem utilizando métodos de busca e chatbots. In *Anais do VII Brazilian Workshop on Social Network Analysis and Mining*, Porto Alegre, RS, Brasil. SBC.

Flock. (2017) "Fake news detector", https://www.youtube.com/watch?v=6IBRjS4InYE, Janeiro.

Gahirwal, M., Moghe, S., Kulkarni, T., Khakhar, D. e Bhatia, J. (2018). Fake news detection. International Journal of Advance Research, páginas 817–819. Ideas and Innovations in *Technology*.

Hartigan, J. A. (1975), Clustering Algorithms, John Wiley I & Sons, Inc.

Haykin, S. (2001), Redes Neurais: Princípios e Prática, Bookman.

Kalske M., Mäkitalo N., Mikkonen T. (2018). Challenges When Moving from Monolith to Microservice Architecture. In *Garrigós I.*, Wimmer M. (eds) Current Trends in Web Engineering. ICWE 2017. Lecture Notes in Computer Science, vol 10544. Springer, Cham.

Liddy, E. D. (2001). Natural Language Processing. In *Encyclopedia of Library and Information Science*, 2nd Ed. NY. Marcel Decker, Inc.

Loper, E., Bird, S. (2002). NLTK: The Natural Language Toolkit. In *arXiv:cs/0205028v1*[cs.CL, páginas 1-8.

Marketwired. (2017). Flock fake news detector protects messaging and collaboration platform users against fake news epidemic. Yahoo! Finance.

Matsa, K. E. e Shearer, E. (2018). News use across social media platforms 2018. Technical report, Pew Research Center.

Mikolov, T. Sutskever, I., Chen, K., Corrado, G. e Dean, J. (2013). Distributed Representations of words and Phrases and their Compositionality.

Newmann, S. (2015). Building Microservices. O'REILLY Califórnia.

OEA. (2018). Relatório preliminar MOE. Technical report, OEA.

Peffers, K., Tuunanen, T., Rothenberger, M. e Chatterjee, S. (2007). A Design Science Research Methodology for Information Systems Research. In *Journal of Management Information Systems*, 24:3, páginas 45-77.

Russell, S. e Norvig, P. (2013), Inteligência Artificial, Elsevier.

Sebastiani, F. (2012). Machine Learning in Automated Text Categorization. In *ACM Computer Surveys*, vol. 34, páginas 1-47.

Secom. (2016). Pesquisa brasileira de mídia 2016: hábitos de consumo de mídia pela população brasileira. Technical report, Presidência da República. Secretaria Especial de Comunicação Social.

Shao, C., Ciampaglia, G. L., Varol, O., Yang, K., Flammini, A. e Menczer, F. (2018). The spread of low-credibility content by social bots. ar-Xiv:1707.07592v1, páginas 1–41.

Shawar, B. A. e Atwell, E. (2007). Chatbots: Are they Really Useful?. LDV- Forum 2007 - Band 22(1) - páginas 29-49.

Sieradski, D. (2016) “Bs detector”, https://gitlab.com/bs-detector/ bs-detector/tree/master.

Stroppa, T. (2018) “Primeiro de abril, a descoberta do boimate e o combate às fake news”, https://www.cartacapital.com.br/blogs/intervozes/primeiro-de-abril-a-descoberta-do-201cboimate201d-e-o-combate-as-fake-news/, Março.

Verma T. e Renu, G.D. (2014). Tokenization and Filtering Process in RapidMiner. International Journal of Applied Information Systems (IJAIS). Volume 7 - No2, páginas 16-18.

Wang, W. Y. (2017). Liar, liar pants on fire: a new benchmark dataset for fake news detection. arXiv:1705.00648v1 , páginas 1–5.

Documento Digitalizado Restrito

Versão Final TCC

Assunto: Versão Final TCC
Assinado por: Michele Barion
Tipo do Documento: Projeto
Situação: Finalizado
Nível de Acesso: Restrito
Tipo do Conferência: Documento Original e Cópia

Documento assinado eletronicamente por:

- **Michele Cristiani Barion**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 31/03/2020 16:54:06.

Este documento foi armazenado no SUAP em 31/03/2020. Para comprovar sua integridade, faça a leitura do QRCode ao lado ou acesse https://suap.ifsp.edu.br/verificar-documento-externo/ e forneça os dados abaixo:

Código Verificador: 383245
Código de Autenticação: 55804ad5d5