TrendsBot: Verificando a veracidade das mensagens do Telegram utilizando Data Stream

Wellison R. M. Santos[1], **Marcus R. Xavier**[1], **David C. P. da Cunha**[1], **José C. F. M. Júnior**[1], **Daniel A. R. Adauto**[1], **Carlos A. G. Ferraz**[1]

[1]Centro de Informática – Universidade Federal de Pernambuco (UFPE)
Caixa Postal 50740-560 — Recife-PE — Brasil

{wrms,mrxl,dcpc,jcfmj,dara,cagf}@cin.ufpe.br

Abstract. *Fake news is created and shared in large scale through applications such as WhatsApp and Telegram. The massive sharing of such news on these applications is caused by the impact of the headline, along with the ease of sharing and also by the lack of a verification method. Therefore, it is evident the need for a tool that facilitates the judgment of the news integrity. This work proposes TrendsBot: a Telegram bot capable of collecting evidence through Twitter and news portals, supporting users in a quick check of that news. The bot queries keywords through transcribed audios or texts sent to a group, collects evidence related to those words, and relates them to the terms of the news.*

Resumo. *As notícias falsas, conhecidas como Fake News, são criadas e compartilhadas em larga escala por meio de aplicativos como WhatsApp e Telegram. O compartilhamento em massa dessas notícias nesses aplicativos é causado pelo impacto do título, juntamente com a facilidade de compartilhamento e também pela falta de um método de verificação. Portanto, fica evidente a necessidade de uma ferramenta que facilite o julgamento da integridade das notícias. Este trabalho propõe o TrendsBot: um bot para Telegram capaz de coletar evidências através do Twitter e portais de notícias, dando suporte aos usuários em uma verificação rápida dessas notícias. O bot consulta as palavras-chave através de áudios transcritos ou textos enviados a um grupo, recolhe evidências relacionadas com essas palavras e as relaciona com os termos da notícia.*

1. Introdução

Recentemente, com o aumento de atividades em redes sociais a facilidade da propagação de mensagens também se intensifica. Com vários canais de comunicação disponíveis, constantemente, o leitor é estimulado a propagar noticias com procedência duvidosa ou falsa. As notícias falsas ou *Fake News*, apesar de ser um assunto antigo, vêm ganhando destaque de forma recorrente e teve seu estopim durante as corridas eleitorais norte-americanas de 2016 [Zhang and Ghorbani 2019]. Informações falsas podem ser imprecisas, onde a informação não é totalmente correta ou incorreta, que geralmente tem a finalidade de enganar o usuário com uma mensagem totalmente falsa. Do mesmo modo, alguns estudos sugerem que notícias falsas são direcionadas para um público que provavelmente não lê o conteúdo da manchete, sendo desta maneira propagadas como notícias verdadeiras [Trends 2017].

Apesar da mobilização dos governantes e das empresas em tentar conter a propagação de notícias falsas, essas ainda se difundem nas redes sociais. Pesquisas

mostram que em 2016 mais de 80% dos estudantes de nível médio tiveram problemas em distinguir notícias reais de anúncios patrocinados [Sadikoglu and Oktay 2018, Dey et al. 2018]. Em consonância com as redes sociais, os aplicativos de troca de mensagens instantâneas (IM) têm sido um dos principais disseminadores de *Fake News*.

O *Telegram* é uma aplicação de mensagens instantâneas que possui importantes funcionalidades que foram exploradas para a elaboração deste trabalho. O *Telegram* é uma aplicação em nuvem (do inglês *cloud-based*) com arquitetura distribuída e que utiliza seu próprio protocolo de comunicação *MTProto* para acessar a API (*Application Programming Interface*) do servidor a partir de um dispositivo móvel [Tel 2019]. Atualmente, o *Telegram* conta com duas APIs que são disponibilizadas gratuitamente para desenvolvedores, que são a *API Bot* e a *API TDLib*.

O termo *bot* vem da palavra inglesa *robot* que significa robô. São *scripts* automatizados que podem realizar tarefas de forma automática, se conectar com a *web*, ou até mesmo trocar informações com outros *bots* [Eslahi et al. 2013]. A *API bot* possibilita que os *bots* sejam implementados e/ou ligados em rede, e realizem funcionalidades customizadas. A *API TDLib* é uma biblioteca da base de dados multi-plataforma do *Telegram*, onde é possível implementar um cliente (da arquitetura cliente-servidor) totalmente customizado.

Neste trabalho, é proposto o *TrendsBot*, um *bot* para *Telegram* de interação semi-automática, utilizando a *API bot*. Sendo assim, é possível que um *bot* seja capaz de coletar indicativos em portais de notícias confiáveis utilizando a API do *Twitter* que coleta mais dados para auxiliar os usuários em uma rápida análise de informações tendenciosas. O *bot* atualmente consulta palavras-chave através de áudios ou textos, e refina os resultados com base na análise do conteúdo pesquisado, determinando a relevância, coincidindo e relacionando os termos consultados, e retornando informações que estão de fato ligadas às notícias falsas.

Este documento esta organizado da seguinte forma: a seção 2 apresenta os trabalhos relacionados sobre os temas abordados neste artigo. A seção 3 detalha as tecnologias utilizadas e as funcionalidades da ferramenta. A seção 4 descreve os resultados e trabalhos futuros. E por fim, a seção 5 disponibiliza os links para o TrendsBot.

2. Trabalhos Relacionados

A literatura apresenta trabalhos recentemente propostos para auxílio e detecção de notícias falsas (*fake news*). Alguns artigos também abordam, de forma mais geral, o uso de *chatbots* como ferramentas de auxílio ao usuário (assistentes). Outras abordagens genéricas apontam um caminho de pesquisa relacionado ao uso das informações disponíveis nas redes sociais e *sites* de notícias como um caminho viável para detecção de *fake news*.

Em [Hassan et al. 2017] é apresentada uma arquitetura para checagem de fatos, através de uma ferramenta baseada nas tecnologias de: Aprendizagem de Máquina, Processamento de Linguagem Natural e Consultas em Banco de Dados. A arquitetura do *software* propõe um fluxo que inicia com o monitoramento de sites e redes sociais, passa por um subsistema que classifica e seleciona os fatos para verificação, realiza comparações em fontes confiáveis (sites, serviços e mídias sociais), e retorna um relatório que pode ser

apresentado via API das redes sociais (por exemplo, uma conta do Twitter[1] para postagem de um comentário sobre a notícia).

Já [Long et al. 2017] propõe uma extensão de um modelo para detecção tradicional baseado em redes neurais, e lhe adicionam as informações de perfil do orador (informações pessoais, com intuito de lhe dar credibilidade). Em [Sample et al. 2019], é utilizada uma abordagem baseada em um modelo linguístico que identifica características das notícias verdadeiras e falsas, de maneira que o conteúdo das verdadeiras possa ser caracterizado quantitativamente. Além disso, é feita uma análise do padrão de propagação de notícias, no qual as notícias verdadeiras diferem das notícias falsas, sendo estas últimas muitas vezes originadas em *bots* com o objetivo de saturar o espaço de notícias.

O trabalho de [Granik and Mesyura 2017], observa a similaridade entre os sistemas de filtros de *Spam* e de detecção de *fake News*, além de propor outra abordagem que utiliza Inteligência Artificial (IA) baseada em redes Bayesianas para detecção de notícias falsas.

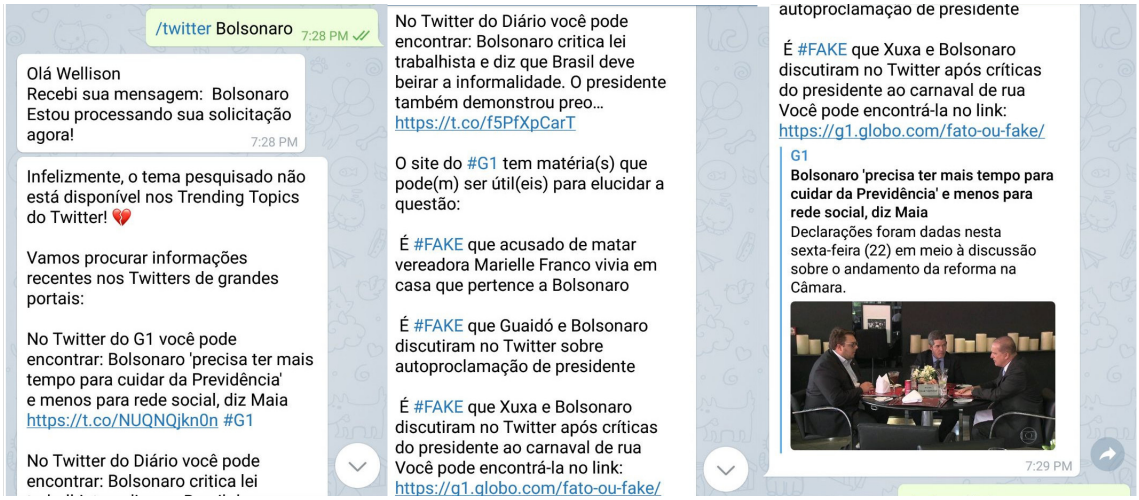
Recentemente, [Della Vedova et al. 2018] trouxe uma abordagem interessante, com o uso de IA, utilizando dois métodos de forma combinada, que consideram o conteúdo da notícia e os sinais sociais. Esse método de detecção foi validado no mundo real via *chatbot* do Messenger do Facebook e obteve taxa de acerto de 81,7%. Na análise dos sinais sociais, o *chatbot* recebe uma *string*, valida se é uma *string* de uma postagem do Facebook, depois lista o conjunto de códigos Identificadores (IDs) das pessoas que curtiram o *post* publicamente, e em seguida classifica o *post*.

Por outro lado, [Wang et al. 2018] faz uma revisão das principais técnicas de criação e disseminação das notícias falsas (com uso de *chatbots*) e conclui dizendo que a checagem das informações é um ponto chave para o combate das *fake news*, desde que seja realizada de forma automática.

Analisando as propostas, foi possível observar várias alternativas possíveis de serem executadas em termos de linhas de pesquisa, seja individualmente ou em conjunto. Como ponto de partida e de exploração, utilizamos as visões gerais de [Rubin et al. 2015], [Conroy et al. 2015] [Wang et al. 2018] e [Sample et al. 2019] para estudar as principais características e comportamentos dos processos de criação e propagação das notícias falsas. O trabalho bastante completo de [Buntain and Golbeck 2017] contribuiu essencialmente nas fases de entendimento do problema e de proposição da solução, fornecendo um mapeamento abrangente sobre as tecnologias empregadas nas áreas correlatas das chamadas *fake news* em mídias sociais, incluindo as etapas de caracterização e identificação (e diferentes técnicas, ferramentas e modelos utilizados). A abordagem ainda especifica outras áreas correlatas relevantes, e em geral revelam uma importante perspectiva de Mineração de Dados como solução para os problemas apresentados.

Através dos estudos de [Hassan et al. 2017], [Long et al. 2017], [Granik and Mesyura 2017], [Della Vedova et al. 2018] observamos que várias abordagens utilizam mecanismos inteligentes para classificação e detecção das notícias falsas, e têm se mostrado eficientes. Especificamente em [Della Vedova et al. 2018], foram utilizados, de forma aplicada, os *chatbots* para auxiliar na detecção de *fake news* (nesse caso, restrito às postagens do Facebook). Os autores de [Hassan et al. 2017] afirmam que

[1]http://twitter.com.br



(a) (b) (c)

Figura 1. Capturas de tela mostrando a interação com o TrendsBot

realizam o monitoramento de sites e redes sociais, e que utilizam a API do Twitter apenas para exibição do *feedback*.

3. Tecnologias e Funcionalidades

O TrendsBot consiste de um *chat bot* que auxilia os usuários a identificarem notícias de procedência duvidosa e/ou falsa em grupos do *Telegram*. O *bot* é capaz de interagir de forma semi-automática com os membros do grupo. Uma vez mencionada a palavra-chave de acionamento correspondente à notícia em foco através de mensagens de voz ou texto, o *bot* coleta os termos presentes na mensagem e realiza uma consulta em portais online do *Twitter* de alta confiabilidade como o G1[2], o BBC[3] e o Diário de Pernambuco[4]. Adicionalmente, a página web "Fato ou *Fake*"é consultada para complementar as informações já colhidas nos portais.

A Figura 1 mostra uma interação sendo realizada com o *bot* `(a)` através da palavra-chave `\twitter` e passando um texto para ser buscado, nesse caso a palavra `Bolsonaro`. Como pode ser visto ainda em `(a)`, a palavra não está entre as tendências do *Twitter* e também que existe uma notícia relacionada no site G1 e em `(b)` outra do Diário de Pernambuco. Em `(b)` em `(c)` é mostrado o resultado da busca por notícias feitas no site "Fato ou *Fake*". Assim, os membros do grupo podem olhar esses indícios e concluir sobre a veracidade de alguma notícia relacionada com o termo buscado, podendo assim compartilhar esses indícios e ajudar a outro contatos a refletirem se aquela notícia é verdadeira ou falsa.

Na Figura 2 é apresentada a arquitetura do *TrendsBot*. O *bot* do *Telegram* é responsável por intermediar a comunicação entre o usuário e o servidor, enviando requisições para o servidor através de métodos HTTP `GET` e `POST`. Para consultar os *Tweets* mais recentes dos Portais, é necessário que o *bot* envie um texto normalizado ao servidor utilizando um método HTTP `POST`. A partir deste momento, o texto normalizado, os *Tweets*

[2]https://g1.globo.com/fato-ou-fake
[3]https://www.bbc.com/portuguese
[4]http://www.diariodepernambuco.com.br

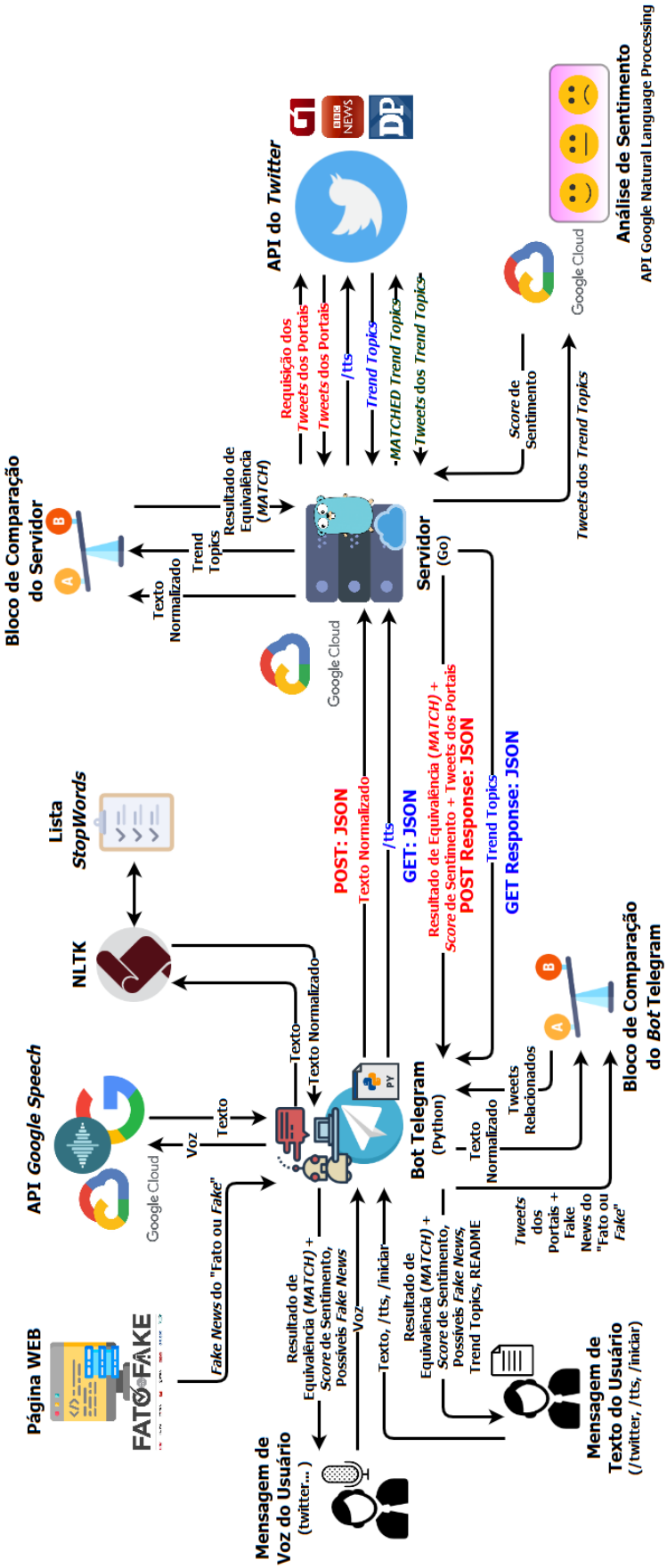


Figura 2. Arquitetura do Sistema

dos Portais e as *Fake News* do "Fato ou *Fake* são enviados a um bloco de comparação do *bot* para verificar que a mensagem enviada está entre as notícias falsas.

Para o desenvolvimento do *TrendsBot* foi necessário um conjunto de tecnologias que seriam imprescindíveis para a integração. Sendo um *bot* para o aplicativo *Telegram*, foi necessário utilizar uma API que pudesse implementar os comportamentos necessários quando o *bot* fosse solicitado. Logo, a biblioteca *Python Telegram Bot*[5] foi utilizada, permitindo a adição de comandos ao *bot*.

Com essa implementação, foi possível adicionar dois comportamentos fundamentais para o *TrendsBot*. O primeiro comportamento foi referente ao recebimento de um texto pelo *bot*, enquanto que o segundo foi relacionado ao recebimento de áudio. Desta forma, foi programado que quando uma mensagem for enviada contendo uma palavra específica predefinida, o *bot* é engatilhado e se comunica com a aplicação servidor passando esse texto na requisição.

Se durante o recebimento do áudio for encontrada uma palavra específica predefinida, o fluxo engatilhado possui mais uma etapa. Essa etapa é referente ao envio do áudio para a transcrição, que é realizada pela API chamada *Google Speech To Text*[6]. Logo, para cada requisição do *bot* que precisasse realizar transcrição, é necessário antes que o áudio seja enviado para o serviço de armazenamento da *Google Cloud*, o *Google Cloud Storage*, para que de lá a API de transcrição pudesse inferir o conteúdo do áudio que está sendo processado.

Finalizada a etapa de adquirir o texto, seja por texto ou por transcrição de áudio, o *bot* realiza uma etapa de normalização do texto. Nessa fase de normalização são removidas palavras do texto que não são representativas para a compreensão da mensagem, isso durante a fase de análise de sentimento. Para essa etapa de normalização foi utilizada a biblioteca NLTK (do inglês *Natural Language Toolkit*), e para remover as palavras menos representativas da língua portuguesa, usou-se uma lista de palavras definidas no repositório de *stop-words*[7].

Com o texto normalizado, a próxima etapa é realizada pelo código servidor. Para isso, o *bot* realiza uma requisição HTTP/POST para o servidor com o conteúdo do texto normalizado. Com o texto normalizado, é feita uma consulta no *Twitter* em busca das mensagens mais relevantes sobre aquele assunto. Uma vez que o servidor recebe essas mensagens, ele inicia o processo de análise de sentimento através da API *Google Natural Language*, que ao receber o texto realiza um processo de *score*.

Para a análise são usados valores de referência para definir o sentimento em quatro opções possíveis, que são os sentimentos: claramente positivo, quando o score é dado entre 0.8 e 1; neutro, entre 0.1 e menor que 0.8; misto, entre 0 e menor que 0.1; e claramente negativo, entre -0.6 e menor que 0.

Uma vez que o *bot* recebe o sentimento e as notícias coletadas que estão relacionadas ao tema, ele envia três informações relevantes que ajudarão o usuário a entender se aquela mensagem enviada por ele está relacionada a alguma notícia falsa, como já

[5]https://python-telegram-bot.readthedocs.io/en/stable
[6]https://cloud.google.com/speech-to-text
[7]https://github.com/Alir3z4/stop-words/blob/master/portuguese.txt

mencionado na explicação da Figura 1.

4. Resultados e Trabalhos Futuros

Um problema atual na sociedade são as notícias falsas (do inglês *Fake News*). Esse tipo de notícia pode ser completamente ou parcialmente falsas, e pela falta de uma mínima verificação dos usuários, seja pela natureza atrativa da notícia, e aliada com a facilidade do compartilhamento pelas aplicações, essas notícias são massivamente compartilhadas.

Esse tipo de notícia motiva diversos pesquisadores a buscarem mecanismos e ferramentas que apontem características para auxiliarem os usuários a julgarem essas notícias como verdadeiras ou falsas. Páginas de internet como a "Fato ou *Fake*"são mecanismos básicos que informam quando uma notícia é verídica. Entretanto, é necessário que um administrador da página decida e informe ao leitor os fatos que justificam uma notícia como verdadeira ou não.

Com isso, a necessidade de ferramentas que automatizem a coleta de indícios e dispensem a necessidade de jornalistas que classifiquem essas notícias está mais alinhada com a facilidade e o tempo necessário para que uma notícia se torne viral. Isso justifica aplicações como a do *bot* TrendsBot, que é uma aplicação que utiliza a rede social *Twitter* para coleta de evidências que ajudem os membros de grupos no aplicativo do *Telegram* a decidirem a veracidade da notícia.

Quando o *bot* é invocado, ele torna-se capaz de mostrar em um grupo no *Telegram* se uma notícia está entre os assuntos mais comentados do momento. Também mostra s e existe nos perfis do *Twitter* de três portais de notícias alguma menção sobre aquele assunto. E por fim consulta a página "Fato ou *Fake*"em busca de alguma notícia verdadeira ou falsa referente ao assunto.

Entre os trabalhos futuros, está a implementação de uma funcionalidade onde o usuário submete sua localização para que o *bot* identifique as notícias próximas ao usuário que tenham mais indícios de serem falsas. Com isso, esse usuário poderá compartilhar os indícios encontrados pelo *bot* de que as notícias que estão nos seus grupos são verdadeiras ou falsas. Outra adição será para que o *bot* inicie uma busca por indícios sempre que um *link* for compartilhado no grupo, assim todos saberão dos fatos e indícios antes de comentarem e compartilharem. Também como trabalhos futuros, estão a adição de mais integrações com portais de notícias de jornais locais (região, estado ou cidade) que podem trazer mais indícios do que um jornal de abrangência nacional no julgamento da notícia, e ainda a integração com mais portais do tipo "Fato ou *Fake*".

5. Demonstração

Nesta seção estão todos os *links* para os artefatos do TrendsBot.

- *Links* para os códigos do *bot*: *Bot*[8] e Servidor[9];
- *Links* para o manual de instalação e configuração da aplicação[10] e vídeos demonstrando a instalação[11] e uso da aplicação[12].

[8]https://github.com/wellisonraul/TrendsBotPython
[9]https://github.com/wellisonraul/TrendsBotGoLang
[10]https://git.io/fjUsZ
[11]https://youtu.be/HYEYfvtTbzQ
[12]https://youtu.be/jEgaV5YqABk

Referências

(2019). Mtproto mobile protocol. https://core.telegram.org/mtproto. Acessado em 23 de março de 2019.

Buntain, C. and Golbeck, J. (2017). Automatically Identifying Fake News in Popular Twitter Threads. *Proceedings - 2nd IEEE International Conference on Smart Cloud, SmartCloud 2017*, pages 208–215.

Conroy, N. J., Rubin, V. L., and Chen, Y. (2015). Automatic deception detection: Methods for finding fake news. ASIST '15, pages 82:1–82:4, Silver Springs, MD, USA. American Society for Information Science.

Della Vedova, M. L., Tacchini, E., Moret, S., Ballarin, G., DiPierro, M., and de Alfaro, L. (2018). Automatic online fake news detection combining content and social signals. In *2018 22nd Conference of Open Innovations Association (FRUCT)*, pages 272–279.

Dey, A., Rafi, R. Z., Parash, S. H., Arko, S. K., and Chakrabarty, A. (2018). Fake News Pattern Recognition using Linguistic Analysis. pages 305–309.

Eslahi, M., Salleh, R., and Anuar, N. B. (2013). Bots and botnets: An overview of characteristics, detection and challenges. *Proceedings - 2012 IEEE International Conference on Control System, Computing and Engineering, ICCSCE 2012*, pages 349–354.

Granik, M. and Mesyura, V. (2017). Fake news detection using naive bayes classifier. In *2017 IEEE First Ukraine Conference on Electrical and Computer Engineering (UKRCON)*, pages 900–903. IEEE.

Hassan, N., Zhang, G., Arslan, F., Caraballo, J., Jimenez, D., Gawsane, S., Hasan, S., Joseph, M., Kulkarni, A., Nayak, A. K., Sable, V., Li, C., and Tremayne, M. (2017). Claimbuster: The first-ever end-to-end fact-checking system. *Proc. VLDB Endow.*, (12):1945–1948.

Long, Y., Lu, Q., Xiang, R., Li, M., and Huang, C.-R. (2017). Fake news detection through multi-perspective speaker profiles. pages 252–256.

Rubin, V. L., Chen, Y., and Conroy, N. J. (2015). Deception detection for news: Three types of fakes. ASIST '15, pages 83:1–83:4, Silver Springs, MD, USA. American Society for Information Science.

Sadikoglu, S. and Oktay, S. (2018). Identifying Fake News and Fake Users on Twitter. *Procedia Computer Science*, 120:204–212.

Sample, C., Columbia, L. L. C., and Indianapolis, I. (2019). A Model for Evaluating Fake News. (February).

Trends, I. T. (2017). The Economics of "Fake News". (December).

Wang, P., Angarita, R., and Renna, I. (2018). Is this the Era of Misinformation yet? Combining Social Bots and Fake News to Deceive the Masses. In *The 2018 Web Conference Companion*, Lyon, France.

Zhang, X. and Ghorbani, A. A. (2019). An overview of online fake news: Characterization, detection, and discussion. *Information Processing & Management*, 56(March):1–26.