MINISTÉRIO DA DEFESA
EXÉRCITO BRASILEIRO
DEPARTAMENTO DE CIÊNCIA E TECNOLOGIA
INSTITUTO MILITAR DE ENGENHARIA
CURSO DE MESTRADO EM SISTEMAS E COMPUTAÇÃO

PAULO ALVES BRAZ

INVESTIGANDO A UTILIZAÇÃO DE INFORMAÇÕES TEXTUAIS NA DETECÇÃO DE BOTS SOCIAIS

Rio de Janeiro
2018

INSTITUTO MILITAR DE ENGENHARIA

PAULO ALVES BRAZ

INVESTIGANDO A UTILIZAÇÃO DE INFORMAÇÕES TEXTUAIS NA DETECÇÃO DE BOTS SOCIAIS

Dissertação de Mestrado apresentada ao Curso de Mestrado em Sistemas e Computação do Instituto Militar de Engenharia, como requisito parcial para a obtenção do título de Mestre em Ciências em Sistemas e Computação.

Orientador: Prof. Ronaldo Ribeiro Goldschmidt - D.Sc.

Rio de Janeiro
2018

INSTITUTO MILITAR DE ENGENHARIA
Praça General Tibúrcio, 80 - Praia Vermelha
Rio de Janeiro - RJ CEP 22290-270

Braz, Paulo Alves
INVESTIGANDO A UTILIZAÇÃO DE INFORMAÇÕES TEXTUAIS NA DETECÇÃO DE BOTS SOCIAIS / Paulo Alves Braz, orientado por Ronaldo Ribeiro Goldschmidt - Rio de Janeiro: Instituto Militar de Engenharia, 2018.

87p.: il.

Dissertação (mestrado) - Instituto Militar de Engenharia, Rio de Janeiro, 2018.

1. Curso de Sistemas e Computação - teses e dissertações. 1. Bots. 2. Processamento de Linguagem Natural. 3. Aprendizado Profundo. 4. Clusterização. 5. Redes Sociais. I. Goldschmidt, Ronaldo Ribeiro. II. Título. III. Instituto Militar de Engenharia.

INSTITUTO MILITAR DE ENGENHARIA

PAULO ALVES BRAZ

INVESTIGANDO A UTILIZAÇÃO DE INFORMAÇÕES TEXTUAIS NA DETECÇÃO DE BOTS SOCIAIS

Dissertação de Mestrado apresentada ao Curso de Mestrado em Sistemas e Computação do Instituto Militar de Engenharia, como requisito parcial para a obtenção do título de Mestre em Ciências em Sistemas e Computação.
Orientador: Prof. Ronaldo Ribeiro Goldschmidt - D.Sc.

Aprovada em 20 de agosto de 2018 pela seguinte Banca Examinadora:

Prof. Ronaldo Ribeiro Goldschmidt - D.Sc. do IME - Presidente

Prof. Julio Cesar Duarte - D.Sc. do IME

Prof. Eduardo Bezerra - D.Sc. do CEFET-RJ

Rio de Janeiro
2018

À minha família.

Ao Instituto Militar de Engenharia, alicerce da minha formação e aperfeiçoamento.

AGRADECIMENTOS

Agradeço a todas as pessoas que me incentivaram, apoiaram e possibilitaram esta oportunidade de ampliar meus horizontes.

José Braz (*In Memorian*), Jussara Braz, Liliane Oliveira, Thiago Braz e Luzia Oliveira, integrantes da minha família, saudo e agradeço o carinho e o cuidado diuturno. Aos meus amigos confrades: Flávio Vizeu, Dr. Fabrício Murai, Yanko Gitahy, Filipe Lima e Rafael Freitas, saudo e agradeço pela amizade e companheirismo de anos de amizade.

Agradeço aos meus outros queridos amigos e companheiros de academia da UFRJ ao IME, pelo apoio, suporte e incentivo durante o percurso: Rafael Pinto, Rodrigo Murta, Caio Sousa, Vinícius Dalto, Maj Leandro Ferreira, Marco Aurélio, Dr. Douglas Cardoso e Rodrigo Barboza.

Em especial ao meu mestre orientador, Dr. Ronaldo Ribeiro Goldschmidt, por suas disponibilidades, atenções, parceria, sobretudo por muitas vezes ultrapassar a esfera de orientador, figurando tal como um pai autêntico, orientando-me em outras dimensões da vida, alheias à academia, até mesmo me sustentando nos momentos mais sombrios e difíceis desta jornada. Jamais poderei esquecer e aqui manifesto minha sempiterna gratidão.

Finalmente, agradeço a Deus pela força e pela dádiva da vida, com Suas incontáveis bençãos derramadas em todo meu viver, sobretudo nos momentos de questionamentos acerca da minha fé em mim e Nele.

"When something is important enough, you do it even if the odds are not in your favor."

ELON MUSK

SUMÁRIO

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE SIGLAS

CBOW	Continuous Bag of Words
CNN	Convolutional Neural Network
URL	Uniform Resource Locator
CC-BY-SA	Creative Commons License
GPU	Graphic Process Unit
MLP	Multilayer Perceptron
OLAP	Online Analitycal Processing
DBSCAN	Density-Based Spatial Clustering with Noise
SOM	Self-Organizing Maps
SVM	Support Vector Machine

RESUMO

Atualmente, as redes sociais estão sujeitas a ações de *bots* sociais que executam atividades maliciosas como a disseminação de notícias falsas. Algumas pesquisas voltadas à detecção desse tipo de *malware* se baseiam em estatísticas extraídas a partir do conteúdo das mensagens postadas. Como a extração de estatísticas pode ocasionar perda de informação, este trabalho tem como objetivo apresentar evidências experimentais de que o uso de textos originais das mensagens pode melhorar a precisão de detecção. Para tanto, propõe-se um método que aplica uma rede neural convolucional para identificar mensagens suspeitas. Resultados utilizando dados do Twitter fornecem indícios de adequação do método proposto, de forma a aprimorar a acurácia dos classificadores de contas automatizadas.

ABSTRACT

Social bots are responsible for malicious activities in social networks. State-of-the-art on social bot detection combines account data with textual content statistics extracted from messages posted in those environments. Nevertheless, statistical consolidation may lead to information loss. In such scenario, this work searches for experimental evidence that supports the hypothesis that the usage of original textual content from messages may improve detection accuracy. To help our search, we developed an approach that applies convolutional neural networks to identify suspicious messages based on their original textual content. Those network models are trained on samples selected at random. Experiments with our method on Twitter data confirm our hypothesis.

1 INTRODUÇÃO

As redes sociais encontram-se expostas a ações dos bots sociais, contas automatizadas capazes de interagir com humanos e produzir conteúdo automaticamente (BADRI SATYA et al., 2016). Contas desta natureza podem ser utilizadas para manipulação de opiniões de usuários das redes sociais por meio de atividades maliciosas, tais como: disseminação de boatos e notícias falsas (do inglês, *fake news*), falsificação ou mesmo concepção de estatísticas de percepção pública, dentre outras (RATKIEWICZ et al., 2011b). Diante deste cenário, torna-se imperativo identificar e mitigar os efeitos negativos perpetrados por estas entidades, sobretudo pelo efeito extremamente negativo gerado pelas interações efetivadas.

Segundo Hwang et al. (2012) e Alvisi et al. (2013), a detecção de bots sociais é a tarefa de identificar contas automatizadas nos ambientes virtuais das redes sociais, tais como: Facebook, Linkedin, Twitter, etc. Portanto, para este fim, aplicam-se técnicas de aprendizado de máquina para, a partir dos dados sobre as diversas contas de uma rede social (do inglês, *Online Social Network*), seja possível construir modelos estatísticos de classificação binária[1] possibilitando a identificação eficaz, com a maior acurácia possível (LEE et al., 2011) e (DAVIS et al., 2016).

Neste expediente, é possível evidenciar que a grande maioria dos trabalhos apresenta técnicas baseadas na extração de conhecimento sobre os atributos ou características comportamentais das contas (do inglês, *feature-based approach*) (FERRARA et al., 2016). Desta forma, os atributos a que se refere esta abordagem são informações que descrevem o comportamento das contas nas redes sociais. Número de amigos ou seguidores de uma conta, quantidade de contas que cada conta segue, número de mensagens propagadas em uma janela de tempo, quantidade de caracteres na descrição do perfil da conta, quantidade de fotos postadas, dentre outros, são exemplos de atributos comportamentais utilizados na detecção de bots sociais (LEE et al., 2011), (DAVIS et al., 2016) e (WANG et al., 2016).

Todavia, além de informações comportamentais sobre as contas, alguns trabalhos também utilizam dados estatísticos sobre os textos das mensagens propagadas pelas contas, tais como: quantidade de links nas mensagens, número de menções a terceiros via

[1] Aprendizado Supervisionado com duas classes.

"@nomeusuário", etc (YANG et al., 2015). Nesses trabalhos, os textos brutos[2] das mensagens são submetidos a uma etapa de engenharia de atributos que é responsável por extrair dados estatísticos de interesse. No entanto, como em qualquer consolidação de dados, pode-se incorrer em perda de informação (BEAUDRY; RENNER, 2012).

O presente trabalho apresenta evidências relacionadas à perda de informação ou perda de indícios e similaridades importantes para identificar contas automatizadas, quando do uso estrito de estatísticas extraídas para representar os textos confeccionados e propagados pelas contas nos ambientes virtuais das redes sociais.

1.1 CONTEXTUALIZAÇÃO E MOTIVAÇÃO

Figurando como um espaço cada vez mais relevante no expediente de obtenção e disseminação de informações na Web [3](do inglês, *World Wide Web*), as redes sociais tem usufruído de evidente primazia e popularidade no que tange às mídias digitais atuais (MAHER et al., 2014). De acordo com Badri Satya et al. (2016), diariamente bilhões de usuários utilizam as mídias sociais, compartilhando informações sobre suas vidas, expressando opiniões através de mensagens, likes e avaliações. Como ilustração, existem hoje aproximadamente 3,7 bilhões de usuários conectados à Internet. Deste contingente, 1,5 bilhão utiliza o Facebook. Quanto ao Twitter, existem hoje 360 milhões de usuários cadastrados[4].

Desta forma, a proliferação de notícias falsas (do inglês, *fake news* também figura como um fenômeno bastante proeminente nos meios de comunicação (RATKIEWICZ et al., 2011b) (FERRARA et al., 2016). A partir da utilização dos *bots*, muitas notícias e boatos podem ser propagados de forma veloz, forjando assim a percepção pública no que tange aos mais diversos assuntos de interesse, provocando danos incomensuráveis nos espectros sociais, econômicos, políticos (tal como o ocorrido nas eleições presidenciais americanas), etc.

Dentro deste cenário, os *bots* sociais encontram ambientes ricos para perpetrar os mais diversos tipos de intervenções, comprometendo a experiência dos milhões de usuários destas redes sociais, tais como: propagação de boatos ou notícias falsas, manipulação dos famigerados *trending topics* [5], confecção de falsas avaliações relacionadas a bens de serviço

[2]Texto original confeccionado pela conta, desconsiderando anotações ou metadados sobre o texto

[3]Sistema de documentos em hipermídia que são interligados e executados pela Internet.**W3C**: https://www.w3.org/Help/webinternet

[4]**Statista** - www.statista.com/statistics/

[5]Indicador conhecido no Twitter que informa os assuntos mais comentados através da marcação *hashtag* "#"

em redes sociais voltadas para turismo e hotelaria, tais como: Trip Advisor e Booking, difamação de usuários ou pessoas públicas.

1.2 PROBLEMA

Como dito anteriormente, a detecção de *bots* sociais é a tarefa de identificar contas automatizadas, conhecidas como *bots* sociais (FERRARA et al., 2016). Logo, no contexto das diversas abordagens identificadas pelo presente trabalho, a utilização de conteúdo textual original é negligenciada, ou seja, explorada por uma pequena fração de trabalhos. Como alternativa, utilizam-se de estatísticas (metadados) relacionadas ao conteúdo textual gerado e disseminado pelas contas, tais como: quantidade de urls no corpo da mensagem, quantidade de menções, estatísticas relacionadas a processos de ***Pos Tagging***, etc.

Neste expediente, Beaudry e Renner (2012) preconiza que a utilização de estatísticas ou metadados acerca dos textos pode configurar perda de informação, pois traços ou padrões menos evidentes, relacionados à escrita, podem ser desconsiderados, impactando de forma negativa a acurácia dos rotuladores no processo de identificação de contas, pois estes traços são importantes, conforme a hipótese do presente trabalho.

1.3 HIPÓTESE E OBJETIVO

Diante do exposto, objetiva-se demonstrar que as mensagens podem de fato caracterizar uma conta numa rede social, de forma que traços e indícios menos evidentes (ou aparentemente ocultos) nos textos, podem ser utilizados para gerar rotuladores mais precisos.

Hipótese 1. *A utilização do conteúdo textual original das mensagens propagadas pelos usuários combinado com os seus respectivos atributos comportamentais pode propiciar a criação de classificadores mais precisos do que aqueles gerados com base em informações calculadas a partir dessas mensagens.*

Desta forma, o objetivo do presente trabalho é demonstrar a **Hipótese 1**. E, por conseguinte, a partir disso, gerar rotuladores mais precisos para identificar contas automatizadas.

1.4 CONTRIBUIÇÕES ESPERADAS

As contribuições esperadas para este trabalho são:

(i) Melhorias no desempenho de classificadores de bots sociais construídos a partir dos métodos propostos.

(ii) Novas técnicas de detecção de *bots* sob o paradigma das abordagens ***feature-based*** de detecção de *bots* sociais baseada em técnicas de Aprendizado Profundo, utilizando o conteúdo textual original das mensagens propagadas.

(iii) Nova taxonomia e modelo comparativo acerca das técnicas utilizadas para detecção de *bots* sociais utilizando Aprendizado de Máquina.

1.5 ORGANIZAÇÃO DO TRABALHO

O texto está organizado da seguinte forma: no capítulo 2 será apresentada uma revisão bibliográfica relevante e abrangente no tocante ao tema desta pesquisa, contendo também os principais termos e conceitos fundamentais utilizados neste trabalho; no capítulo 3 é então apresentado um levantamento sobre os trabalhos relacionados sobre o prisma da detecção de *bots*; no capítulo 4 são apresentados os métodos propostos para o tratamento do problema; no capítulo 5 são apresentados os resultados dos classificadores desenvolvidos sob o paradigma da técnica proposta; por fim, no capítulo 6 figuram as principais conclusões do trabalho como também contém um breve levantamento de algumas possibilidades de trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

No presente capítulo são apresentados detalhamentos sumarizados acerca de tópicos utilizados no decorrer do texto. Portanto, estas exposições são necessárias para alinhar o vocabulário e o conhecimento sobre tais assuntos, antes dos mesmos serem utilizados de forma livre no desenvolvimento do presente texto.

2.1 BOTS SOCIAIS

Os *bots* sociais são entidades controladas por *software*, que geram conteúdo e estabelecem interações com demais usuários baseados em algoritmos. Deste modo, importante ressaltar que algumas instâncias apresentam mecanismos sofisticados de evasão (WANG et al., 2016) e (JI et al., 2016), visando dirimir o efeito dos respectivos mecanismos de detecção utilizados pelos serviços de rede social, OSNs(*Online Social Networks*), acrônimo do inglês.

As consequências da disseminação indiscriminada dos *bots* sociais podem ser muito importantes e até mesmo irreversíveis, sobretudo no que tange à manipulação de informações de valor, impactando sobremaneira eventos de grande relevância para a sociedade (eleições), como também no furto de informações sensíveis para fins ilegais ou inconvenientes, como disposto no trabalho de Thomas et al. (2013).

O impacto destas entidades automatizadas sobre os diversos serviços de rede social pode ser bastante abrangente e diversificado. Existem evidências da adoção de *bots* sociais para efetuar manipulação política no contexto de eleições a partir de meados de 2010 (FERRARA, 2017). Tal abordagem é implementada na disseminação de milhares de mensagens (*tweets*) direcionando para páginas de internet que propagam informações falsas acerca de um grupo de candidatos que disputam o certame, enaltecendo um específico e prejudicando outros. Contudo, estudos até aqui têm demonstrado ser praticamente impossível determinar a autoria destas operações de acordo com Ferrara et al. (2016), Bessi e Ferrara (2016).

Além disso, existem muitas evidências acerca da utilização eminente de *bots* sociais para a disseminação de *spam* e *phishing* social, que objetiva a coleta de informações sensíveis dos diversos usuários que utilizam as redes sociais, isto pode ser verificado em Jagatic et al. (2007), Thomas et al. (2011), Chowdury et al. (2013). Estudos detalhados

sobre estes efeitos podem ser encontrados em Boshmaf et al. (2011) e Ferrara (2015).

Isto posto, é mister analisar as diversas técnicas utilizadas para a detecção destas entidades automatizadas. Neste aspecto, Ji et al. (2016) apresenta uma classificação das técnicas em dois grandes conjuntos a saber:

a) Detecção **Host-side** - conjunto de técnicas de detecção de *social bots* que identificam modificações suspeitas no ambiente dos terminais finais (cujas contas utilizadas neste sistema são manipuladas por bots), sobretudo no que tange aos recursos computacionais gerenciados pelo sistema operacional, por exemplo: identificam mudanças quanto à lista de serviços ou *daemons* carregados no momento que o sistema operacional é iniciado, verificam o conjunto de portas TCP/IP abertas para comunicação em um determinado intervalo de tempo, endereços IP acessados, etc.

b) Detecção **Server-side** - conjunto de técnicas que identificam contas automatizadas utilizando padrões comportamentais de utilização e navegação, principalmente no que está relacionado às contas que promovem sites e blogs externos.

De modo análogo, no mesmo expediente, Ferrara et al. (2016) estabelece uma outra classificação para técnicas de detecção, voltadas para o que Ji et al. (2016) classifica como abordagens *Server-side*:

a) Abordagem Topológica (***Graph-based***): utilizam atributos relacionados a topologia da rede, por exemplo: coeficiente de clusterização, medidas de centralidade e distribuição dos graus;

b) Abordagem baseada em Atributos Comportamentais (***Feature-based***): utilizam atributos relacionados ao padrão de utilização da rede social. São exemplos de atributos: idade da conta, localização do usuário, tempo entre a geração de mensagens, taxa de envio de solicitação de amizades;

c) Abordagem (***Crowd-sourcing***): compreende o mecanismo de rotulação manual das contas, altamente custoso devido à grande quantidade de dados a serem analisados;

d) Abordagem Híbrida (***Hybrid-based***): conjunto de abordagens que emprega qualquer combinação a partir das abordagens anteriores.

2.1.1 ABORDAGENS HOST-SIDE

O conjunto de técnicas nesta abordagem identifica mudanças no contexto dos recursos computacionais dos sistemas terminais, ou seja, nos sistemas computacionais dos usuários

das redes sociais. Por exemplo, o trabalho de Chu et al. (2012) utiliza padrões biométricos, incluindo padrões de utilização de dispositivos de entrada (mouse e teclado) para caracterização de contas automatizadas. Para tal, empregam uma página Web para coletar estes dados relacionados aos padrões de navegação, de modo a caracterizar e discriminar contas, permitindo ou não a entrada de um usuário.

Exemplos de trabalhos nesta mesma área: Kartaltepe et al. (2010), Nagaraja et al. (2011), Tan et al. (2012).

2.1.2 ABORDAGENS SERVER-SIDE

Neste grupo de técnicas, as abordagens discriminam contas automatizadas utilizando informações contextuais no nível das redes sociais. Deste modo, neste expediente, diversos tipos de atributos podem ser utilizados para depreender um comportamento suspeito, tais como: quantidade de amigos ou seguidores, tempo entre postagens, quantidade de urls nas postagens, número de solicitações de amizade enviadas, número de visitas a perfis de usuários, etc.

2.1.2.1 ABORDAGEM TOPOLÓGICA

Neste grupo residem os trabalhos que utlizam eminentemente os dados acerca da topologia da rede. Esta rede pode ser uma abstração de parte da rede social, respeitando algumas restrições, ou pode compreender outro tipo de abstração. Sob este paradigma técnicas baseadas em *Random Walks* figuram como as mais proeminentes (FERRARA et al., 2016).

2.1.2.2 ABORDAGEM BASEADA EM ATRIBUTOS COMPORTAMENTAIS

A grande de maioria de trabalhos desenvolvidos para detecção de *bots* em redes sociais encontra-se neste grupo (FERRARA et al., 2016). Como exemplos de atributos utilizados tem-se: número de seguidores ou amigos na rede, número de mensagens postadas, idade da conta, etc.

O conteúdo textual é considerado como um atributo de uma conta, desta forma, a grande maioria das abordagens utilizam técnicas balizadas na extração de estatisticas sobre estes textos.

2.1.2.3 ABORDAGEM *CROWD-SOURCING*

Esta abordagem utiliza a rotulação manual das contas utilizando diversas fontes de informação da conta no contexto da rede social em questão (FERRARA et al., 2016).

Wang et al. (2013) advoga a favor desta abordagem, afirmando que a rotulação de contas automatizadas figura como uma tarefa adequada para seres humanos, sobretudo para depreender particularidades subjetivas em relação às mensagens propagadas, tais como: ironia, sarcasmo, etc. Ou até mesmo para identificar anomalias ou padrões na rede social. Por outro lado, mostram evidências de que este tipo de classificação de contas apresenta uma queda significativa de performance após um determinado tempo. Stein et al. (2011) figura como exemplo de trabalho utilizando esta abordagem.

2.1.2.4 ABORDAGEM HÍBRIDA

Ferrara et al. (2016) preconiza que as abordagens híbridas utilizam uma combinação de abordagens para a solução do problema de detecção de contas automatizadas. Ou seja, como exemplo, uma solução sob o grupo de abordagens híbridas pode utilizar técnicas voltadas para o emprego de dados topológicos e dados comportamentais sobre as contas relacionadas.

2.2 REDES NEURAIS

Redes Neurais são modelos matemáticos inspirados nos princípios de funcionamento dos neurônios biológicos e na estrutura do cérebro (GOLDSCHMIDT, 2010). Tal como o cérebro estes modelos possuem capacidade de adquirir, armazenar e utilizar o conhecimento e buscam representar de forma computacional competências inerentes aos seres humanos tais como aprendizado, generalização, associação, e abstração (GOLDSCHMIDT, 2010).

Uma rede neural é um processador paralelo e distribuído composto por unidades de processamento simples, que possuem propensão natural para armazenar conhecimento experimental, dispondo o mesmo para utilização (HAYKIN, 2009). Segundo Hastie et al. (2009), uma rede neural é um modelo de estatístico não linear capaz de empreender tanto uma classificação como uma regressão, tipicamente representado por um diagrama de rede como na figura 2.1.

De acordo com Haykin (2009), Redes Neurais são semalhantes ao cérebro humano por dois aspectos:

- O conhecimento é adquirido pela rede através de um processo de aprendizado.
- A força das conexões inter-neurônio (conhecida como pesos sinápticos) é utilizada para armazenar o conhecimento adquirido.

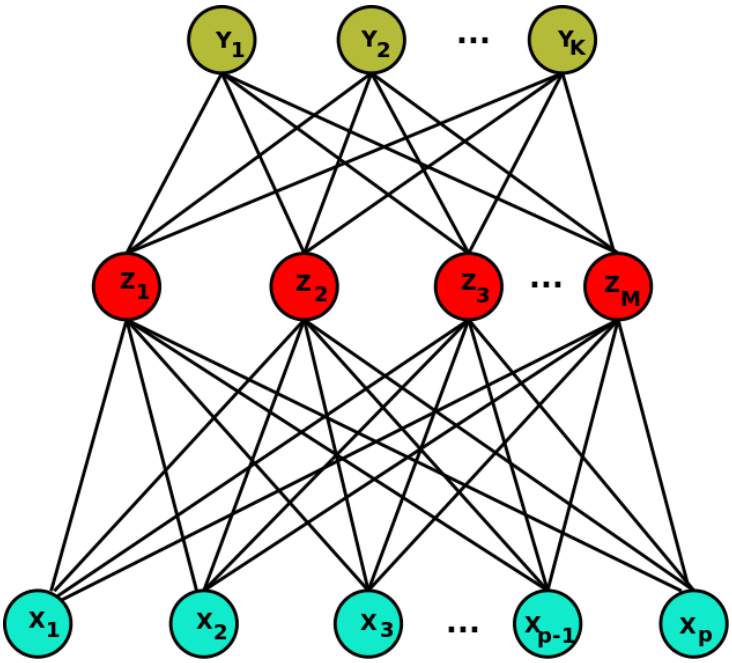

FIG. 2.1: Esquema de uma rede neural contendo apenas uma camada escondida. Adaptado de (HASTIE et al., 2009)

O procedimento para obtenção do conhecimento é chamado algoritmo de aprendizado: a fundção responsável por ajustar os pesos sinápticos da rede para responder de forma adequanda um dado objetivo (HAYKIN, 2009). Importante ressaltar que é possível para uma rede neural modificar sua própria topologia, o que é motivado pelo fato dos neurônios no cérebro humano poderem morrer e novas conexões sinápticas surgirem (HAYKIN, 2009).

Em uma regressão, o valor de K (número de neurônios na saída de rede) é igual a 1, porém estas redes são capazes de manipular múltiplas respostas quantitativas (HASTIE et al., 2009).

Para uma classificação envolvento K classes, será necessário utilizar K neurônios na última camada, de modo que o K-ésima unidade modela a probabilidade da ocorrência da classe K. Desta forma, existem K valores alvo Y_k, tal que $k = 1, ..., K$, cada saída codificada no intervalo $[0, 1]$ (HASTIE et al., 2009).

Atributos derivados Z_m (figura 2.1) são criados a partir de combinações lineares das entradas da rede, e então o valor alvo em Y_k é modelado como uma função da combinação linear de Z_m, de acordo com Hastie et al. (2009):

$$Z_m = \sigma(\alpha_{0m} + \alpha^T X), m = 1, ..., M,$$
$$T_k = \beta_{0k} + \beta_k^T Z, \ K = 1, ..., K,$$
$$f_k(X) = g_k(T), k = 1, ..., K,$$

tal que $Z = (Z_1, Z_2, ..., Z_M)$, e $T = (T_1, T_2, ...T_K)$.

Desta forma, a função de ativação $\sigma(v)$ geralmente é uma função sigmóide $\sigma(v) = 1/_{1+e^{-v}}$ (HASTIE et al., 2009).

2.2.1 ARQUITETURAS

2.2.1.1 REDES *MULTILAYER FEEDFORWARD*

Modelos que utilizam uma ou mais camadas escondidas, onde os respectivos neurônios são chamados de unidades escondidas; o termo escondido refere-se ao fato que esta parte da rede é oculta para entrada e saída da rede neural (HAYKIN, 2009).

A adição de uma ou mais camadas escondidas permite a rede extrair estatísticas de ordem superior a partir do padrão de entrada, de acordo com Haykin (2009). A rede da figura 2.1 contém apenas uma camada escondida composta pelos neurônios $Z_1, ..., Z_M$. Ademais, tipicamente os neurônios em cada camada da rede possuem como entrada as saídas dos neurônios da camada anterior apenas (HAYKIN, 2009).

2.2.1.2 REDES RECORRENTES

As redes recorrentes são diferentes pois possuem pelo menos uma conexão do tipo *feedback* (criando assim um cíclo). Como ilustração, uma rede recorrente pode consistir de uma rede de apenas uma camada de neurônios, com cada neurônio alimentando as conexões de entrada dos demais neurônios tal como a figura 2.2. Importante notar que estas conexões cíclicas utilizam um elemento responsável por atrasar o tempo (Z^{-1} na figura), com o objetivo de apresentar um comportamento dinâmico não linear (HAYKIN, 2009).

2.3 APRENDIZADO PROFUNDO

Algoritmos tradicionais de Aprendizado de Máquina são limitados no que tange à capacidade de processar dados em forma bruta (*raw data*), de acordo com LeCun et al. (2015). Ainda neste contexto, tal trabalho ainda preconiza que as técnicas de Aprendizado Profundo (do inglês - *Deep Learning*) não demandam processos de Engenharia de Atributos e Redução de Dimensionalidade. Um comparação entre modelos de Aprendizado Profundo e os demais modelos de aprendizado está apresentada na figura 2.3. Importante notar que estes processos são muito custosos e tomam a maior parte de um projeto de Aprendizado de Máquina (DOMINGOS, 2012).

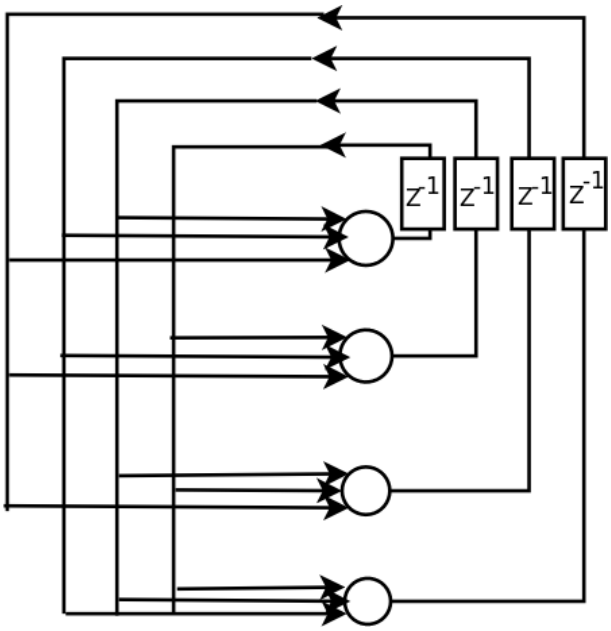

FIG. 2.2: Esquema de uma rede neural recorrente. Adaptado de (HAYKIN, 2009)

As técnicas de Aprendizado Profundo têm obtido muita visibilidade atualmente, sobretudo por exibir resultados promissores em frentes de pesquisa que figuravam como grande desafio para as técnicas tradicionais de Aprendizado de Máquina. Resultados promissores em reconhecimento de imagem em problemas de Visão Computacional tal como em Krizhevsky et al. (2012) e Farabet et al. (2013) e reconhedimento de voz (MIKOLOV et al., 2011) e (HINTON et al., 2012).

Além dos resultados promissores, existem fatores importantes que corroboram a importância das técnicas de Aprendizado Profundo sobre os métodos tradicionais. Segundo LeCun et al. (2015), modelos tradicionais demandam conhecimento prévio sobre os dados e os processos relacionados a estes dados. Esse conhecimento muitas vezes é incipiente ou inexistente, além disso, muito deste conhecimento não pode ser formalizado, o que pode figurar como conhecimento tácito (POLANYI, 2009). Por fim, mas não menos importante, talvez o fator mais importante acerca da relevância e justificativa da utilização destes modelos, em detrimento dos modelos de Aprendizado de Máquina tradicionais, baseia-se no fato de o cérebro humano possuir arquitetura profunda, ou seja, também exibe uma arquitetura de multiplas camadas (WANG; RAJ, 2015).

Arquiteturas profundas são capazes de depreender funções mais complexas e/ou relações ainda não vistas por modelos tradicionais de Aprendizado de Máquina (LECUN et al., 2015). Assim sendo, Wang e Raj (2015) explora fortes aspectos acerca dos modelos *Deep Learning* ao exemplificar que funções que requerem O (2^n) unidades de processa-

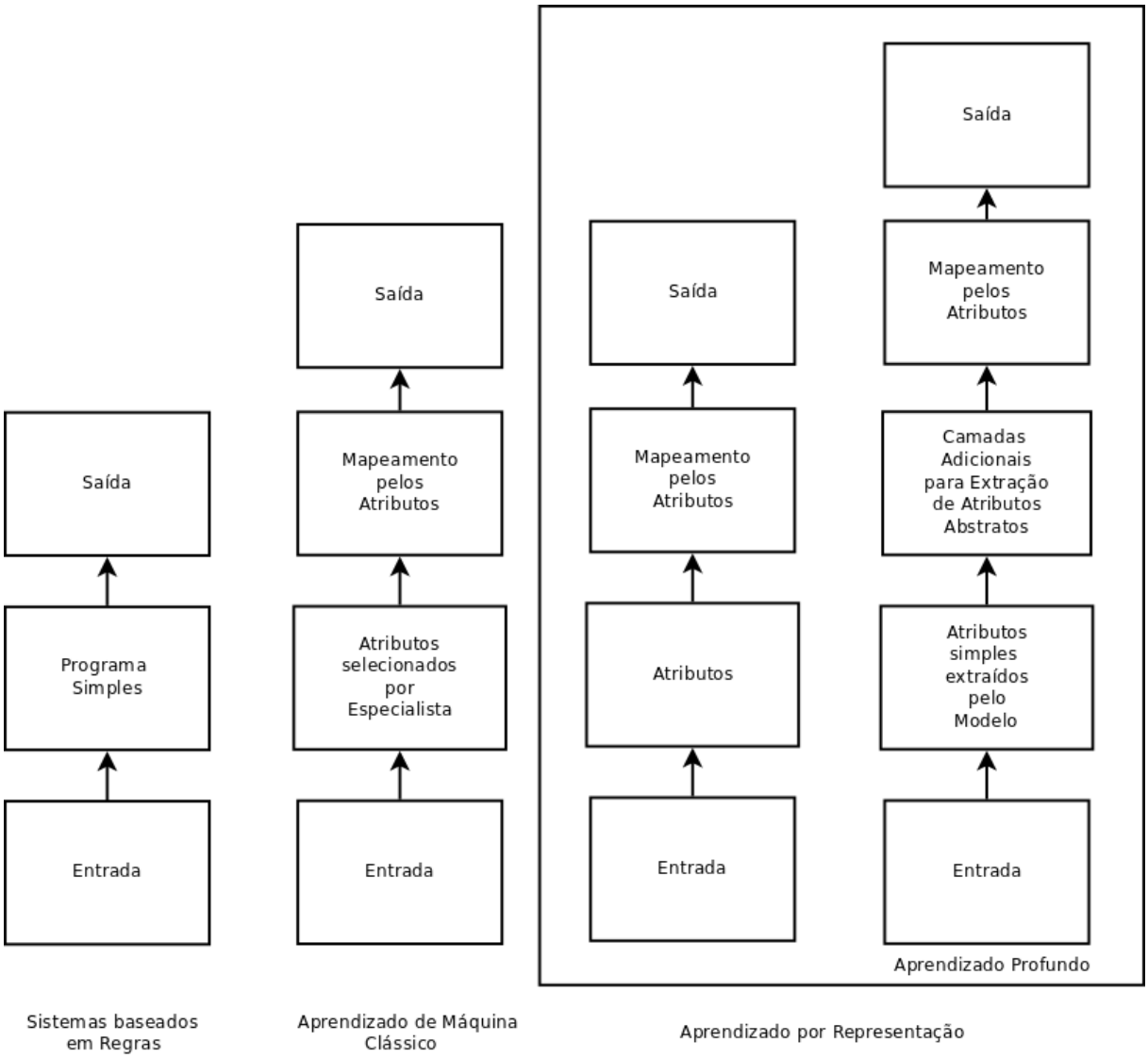

FIG. 2.3: Comparação entre Métodos de Aprendizado - Adaptado de Goodfellow et al. (2016)

mento em uma arquitetura tradicional, podem ser representadas por $O(n^2)$ unidades numa arquitetura profunda contendo O(log n) camadas.

2.3.1 REDES NEURAIS CONVOLUCIONAIS

Inspiradas no modelo biológico da visão e idealizada por Le Cun et al. (1990). De acordo com LeCun et al. (2015), estas redes obtiveram uma grande quantidade de publicações a partir de 2006, e isso de fato ocorreu devido à diminuição dos preços das GPUs (Graphics Process Unit), o que acabou fomentando, desta foma, a popularização de técnicas de programação voltadas para a arquitetura destas unidades. As redes convolucionais (Fig. 3.2) são Redes Neurais com múltiplas camadas escondidas, que descrevem apenas duas

grandes operações: extração de atributos e classificação.

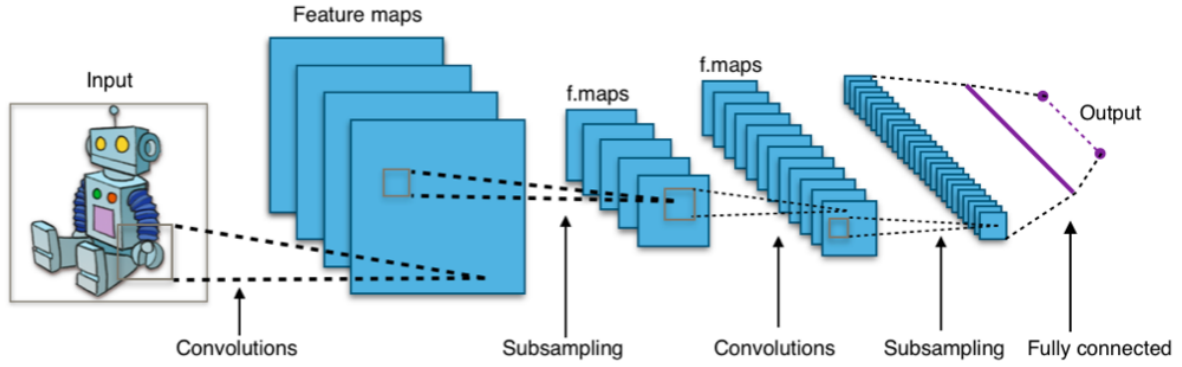

FIG. 2.4: Rede Neural Convolucional Típica - Autor: Aphex34 - 2015 | CC-BY-SA.4.0

Na etapa de extração de atributos, apenas 2 operações são realizadas: *subsampling*, que é a redução de dimensionalidade propriamente dita e convolução, que é uma operação linear entre os componentes de uma matriz quadrada (geralmente de ordem 3 ou 5), que também é conhecida como *kernel*. Desta forma, estas operações de convolução são as responsáveis pela extração dos melhores atributos a partir de um exemplo ou padrão apresentado à rede.

Uma Rede Neural Convolucional é uma rede neural MLP (do inglês - *Multilayer Perceptron*) desenvolvida especificamente para reconhecer formas de duas dimensões com alto grau de invariância à translação, alteração de tamanho, distorção e outras formas de distorção (HAYKIN, 2009). Desta forma, a tarefa de reconhecimento de formas ou imagens é aprendida pelo modelo através de um aprendizado supervisionado, dividido em algumas etapas a saber:

a) Extração de atributos (do inglês - *Feature extraction*) - cada neurônio obtém um entrada sináptica de um campo receptivo local da camada anterior, deste modo empreende a extração de atributos locais (ou *local features* do inglês). A partir do momento que um atributo ou traço é extraído, a localização exata do mesmo torna-se menos importante, de forma que sua posição relativa aos outros atributos ou traços é praticamente preservada.

b) Mapeamento de atributos - Cada camada da rede é composta por múltiplos mapas, de modo que cada mapa de atributos figura tal como um plano, onde neurônios individuais são separados para compartilhar o mesmo conjunto de pesos sinápticos.

c) *Subsampling* - cada camada convolucional é seguida por uma camada que empreende

as operações de: local *averaging* e *subsampling*, de modo que a resolução do mapa de atributos ou traços é reduzido.

Diante disso, é importante reiterar que todos os pesos em todas as camadas de um rede neural convolucional são aprendidos através da fase de treinamento. Ademais, a rede *aprende* seus próprios atributos ou traços de forma automática.

Ou seja, uma rede neural convolucional pode vir a exibir múltiplas camadas de convolução e *subsampling*, camadas estas responsáveis por extrair os atributos ou traços mais relevantes para a tarefa de aprendizado. A camada de convolução é composta por diversos neurônios, de modo que cada um é responsável por aplicar um filtro em uma parte específica de uma imagem ou da matriz que representa ou figura como o padrão de entrada. Desta forma, cada neurônio está conectado a um conjunto de pixels da camada anterior e cada uma destas conexões, produz uma saída que é então passada para a camada subsequente. Ademais, importante notar que os pesos das conexões podem ser interpretados como uma matriz que figura tal como um filtro nas operações de convolução, também conhecida como *kernel*.

Os *kernels* são os responsáveis diretos pela extração dos atributos (ou *features*, do inglês). Numa rede convolucional a extração do conhecimento ocorre em operações sucessivas de convolução e *subsampling* realizadas em camadas sucessivas, respectivamente. Por sua vez, os *kernels* geralmente são matrizes quadradas de ordem 3, 5 ou 7. Estas matrizes percorrem toda a matriz de entrada ou imagem, extraindo assim os traços ou assinaturas. Posteriormente, uma camada se *subsampling* é utilizada para reduzir a dimensão do objeto gerado a partir da convolução, imediatamente anterior, de acordo com a ilustração contida na FIG. 2.3.

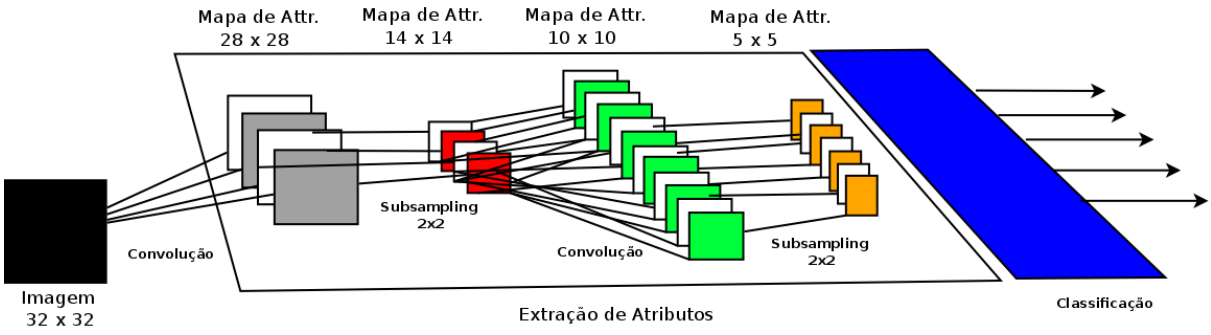

FIG. 2.5: Rede Neural Convolucional - Exemplo de Arquitetura

2.3.1.1 CONVOLUÇÃO

De acordo com Goodfellow et al. (2016), convolução é uma operação que envolve duas funções, de modo que em tarefas de aprendizado de máquina, a entrada comumente figura como uma matriz multidimensional e o *kernel* também é uma matriz multidimensional de parâmetros. Logo, tais parâmetros se adaptam no decorrer do aprendizado, tendo seus respectivos valores iniciais escolhidos de forma randômica pelo algoritmo de treinamento.

Por exemplo, ainda segundo Goodfellow et al. (2016), caso seja apresentada uma imagem de duas dimensões como entrada para uma rede convolucional, normalmente seria esperado que o *kernel* também seja uma matriz quadrada, tal que a operação pode ser definida tal como:

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n)$$

A figura 2.6 ilustra um exemplo prático de uma operação de convolução envolvendo uma matriz quadrada de ordem 3 como entrada e um *kernel* de ordem 2, considerando como parâmetros *stride* (1,1), ou seja, o valor do *stride* informa que o *kernel* passará pela matriz de entrada utilizando apenas um passo na horizontal ou um passo na vertical, sem utilizar uma técnica conhecida do inglês como *zero padding*, que nada mais é o preenchimento com zeros para completar a matriz de entrada para a operação de convolução. A figura geométrica resultante na parte inferior representa a matriz resultante após a operação de convolução. Importante notar a redução de dimensionalidade que ocorre após a operação de convolução. Ademais, a figura exibe uma operação de convoluçao sem transbordamento (do inglês, *flipping*), ou seja, o kernel ocupa todas as subregiões da matriz de entrada. Esse tipo de técnica é também conhecido como *valid convolution*, do inglês.

2.3.1.2 *POOLING*

Após a obtenção dos atributos na etapa de convolução, a operação de *pooling* segue. Esta operação é responsável por trocar os valores da matriz de saída, a mesma obtida após a operação de convolução, por uma representação estatística destes valores. Como ilustração, uma possibilidade tem-se a operação chamada *max pooling*, que efetua o cálculo do maior valor dentro de uma região delimitada por um "retângulo". Este "retângulo"nada mais é que um delimitador de uma área de itens vizinhos da matriz de entrada, que passa pela matriz, efetuando o cálculo e modificando a mesma por esta representação.

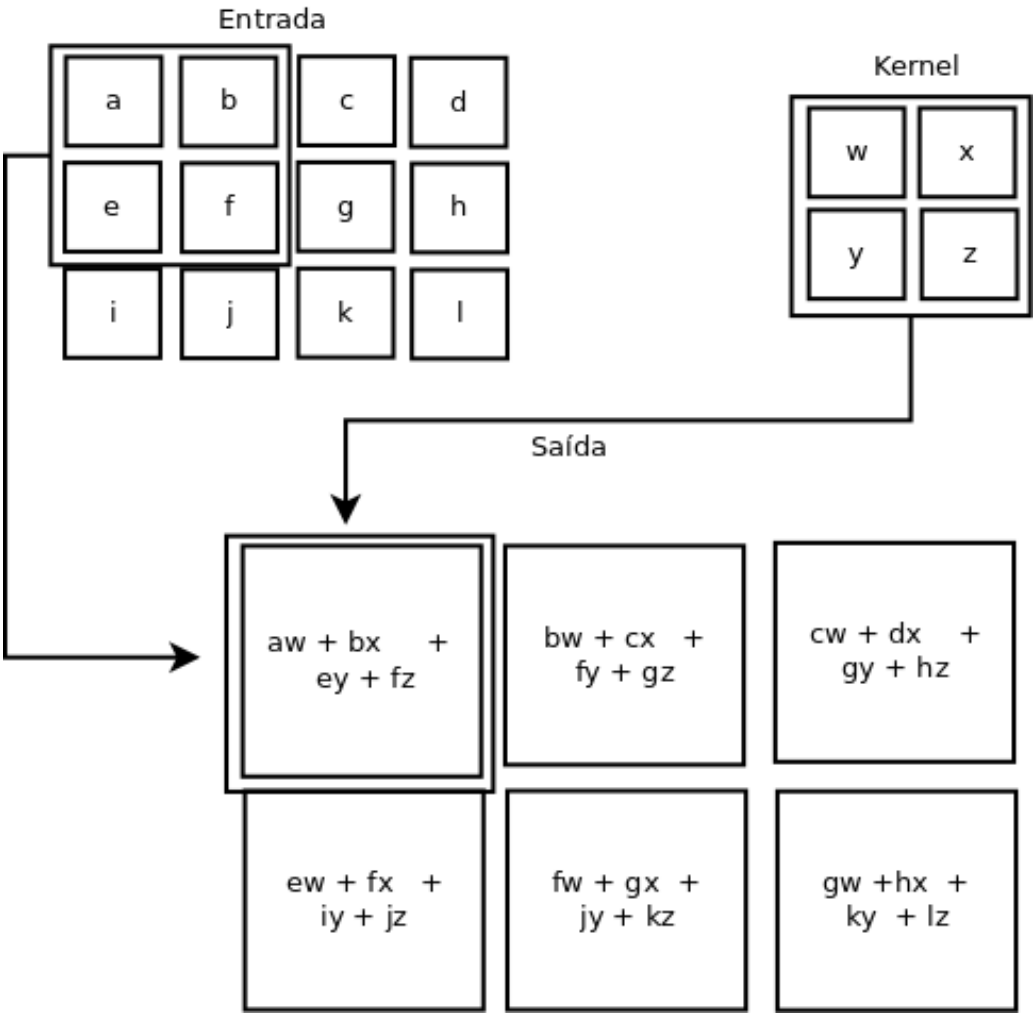

FIG. 2.6: Exemplo de Convolução 2D

2.4 ÁRVORES DE DECISÃO

Uma árvore de decisão utiliza a estratégia dividir para consquista para resolver um problema de decisão (CARVALHO et al., 2011). Um problema difícil pode ser dividido em problemas mais simples, aos quais recursivamente é a aplicada a mesma estratégia para resolução. As soluções dos subproblemas podem ser combinadas, na forma de uma árvore, para produzir uma solução do problema difícil (CARVALHO et al., 2011). Árvores de decisão são modelos de aprendizados atraentes pois possuem treinamentos diretos e extremamente velozes (HO, 1995). Tais árvores são grafos acíclicos direcionados (figura 2.7, de modo que cada nós ou é um nó de divisão, com dois ou mais sucessores, ou é um nó folha (CARVALHO et al., 2011). De acordo com Carvalho et al. (2011) os nós figuram como:

a) Nó folha - rotulado com uma função. Geralmente são considerados apenas os valores

da variável alvo nos exemplos que chegam a um nó folha. No caso mais simples, a função é a constante que minimiza a função custo.

b) Nó de divisão - contém um teste condicional baseado nos valores do atributo. De modo padrão, os testes são univariados: os testes envolvem apenas um único atributo e valores do domínio do mesmo.

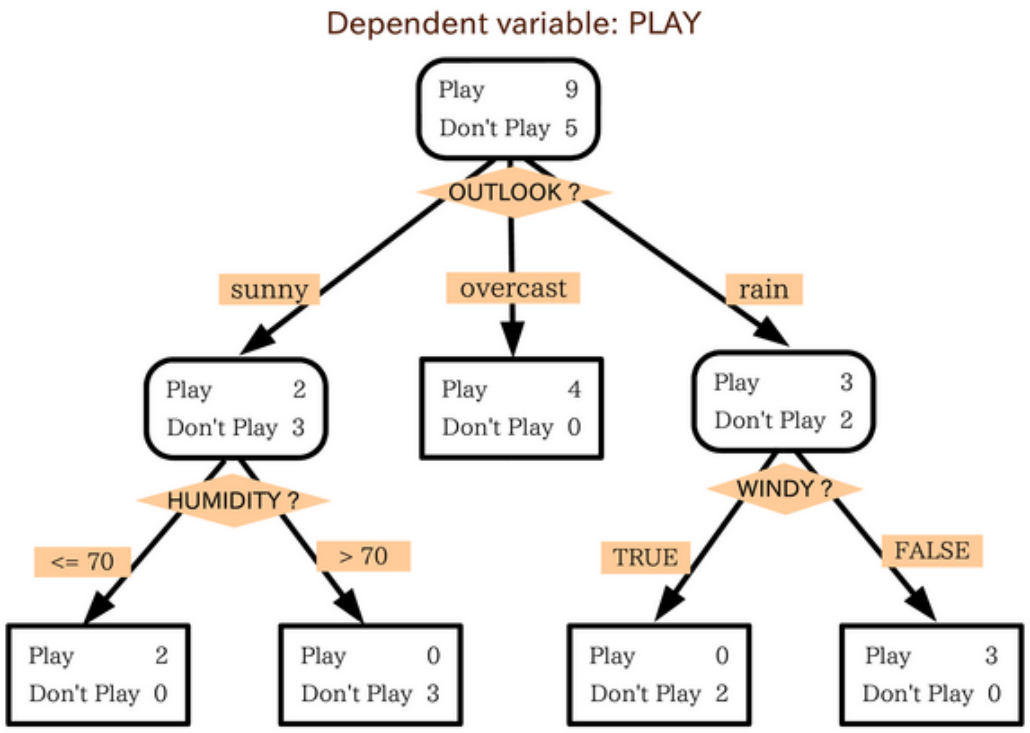

FIG. 2.7: Exemplo de árvore de Decisão - Magnus Manske - (CCO) - Wikimedia Commons

Árvores de Decisão são mais interpretáveis que outros modelos de classificação como redes neurais, porque elas combinam questões simples acerca dos dados de modo mais intelegível (KINGSFORD; SALZBERG, 2008).

2.4.1 RANDOM FORESTS

Random Forests é o modelo representado pela criação de múltiplas árvosres de decisão construídas para subespaços selecionados aleatoriamente, a partir do espaço do atributos (HO, 1995). Por exemplo, para um dado espaço de m dimensões, existem 2^m subespaços nos quais uma árvore de decisão pode ser construída (HO, 1995).

Segundo Ho (1995), uma árvore de decisão é construída para cada subespaço selecionado utilizando todo o conjunto de treinamento, empregando como algoritmo de treinamento:

- Projeção do Eixo Central - O objetivo do algoritmo é separar pelo menos 2 classes para cada nó de divisão. Primeiramente, encontra as duas classes cujas médias são as mais distantes (distância Euclidiana). Deste modo, as médias destas classes estarão conectadas por uma linha reta (eixo central) e todas as instâncias de dados são então projetados nesta linha. A busca então começa neste intervalo entre as médias a um passo fixo, e uma função de erro é calculada para cada hiperplano que passa pelos pontos finais do intervalo ortogonal à linha ou

- Treinamento Perceptron - Semelhante ao primeiro algoritmo, este utiliza incrementos fixos para escolher o hiperplano para cada nó não terminal. Trata-se de uma otimização iterativa que redunda em uma substancial redução dos tamanhos das árvores.

Ou seja, para construir as árvores de decisão, amostras de dados são selecionadas de forma aleatória (com substituição) a partir do conjunto de dados original. Se o número de amostras é N, então N amostras são retiradas com reposição. Deste modo, para cada nó folha ou nó de divisão, o algoritmo determina o melhor atributo a ser utilizado para particionar um subespaço (escolhido de forma aleatória) contendo m dimensões ou atributos, tal que $m << M$. A partir disso, cada árvore na floresta cresce o máximo possível sem ser recortada. Sendo assim, para classificar uma nova instância, cada árvore da floresta fornece uma classificação, mas o resultado final é fornecido pelo resultado do voto da maioria das árvores da floresta (CHEN; LIU, 2005).

2.5 PRÉ-PROCESSAMENTO E REPRESENTAÇÃO DE DADOS TEXTUAIS

O processo de pré-processamento figura como uma etapa de grande importância para todo o projeto de Aprendizado de Máquina (DOMINGOS, 2012). Isso pode ser justificado pelo fato de que, na grande maioria das vezes, o dado não está preparado para ser utilizado e apresentado a um modelo de aprendizado. Ademais, é importante ressaltar que cada modelo espera por um determinado tipo de formatação para o seu padrão de entrada. Outrossim, segundo Witten et al. (2016) procedimentos de pré-processamento dos dados brutos[6] objetivam preparar o dado para que os diversos algoritmos de aprendizado de máquina possam efetivamente construir um modelo, uma descrição estrutural da informação que está contida de forma implícita nos dados. Ainda, de acordo com os

[6]Dados dispostos tal como foram obtidos ou extraídos em processos de coleta, sem tratamento, consolidação ou agregação.

autores, os procedimentos realizados durante a fase de pré-processamento podem ainda incluir atividades de construção e calibragem do modelo de aprendizado.

Diante do exposto, tem-se que a etapa de pré-processamento figura como um passo de grande importância em um projeto de aprendizado de máquina ou mineração de dados. Marsland (2015) preconiza que os algoritmos de aprendizado de máquina tendem a aprender de forma mais eficiente, se os padrões de entrada forem ajustados de forma adequada antes do modelo iniciar o processo de treinamento. Por exemplo, caso uma rede neural - dado que seja um projeto de aprendizado supervisionado, tal que o dado fornece a quais classes pertencem cada uma das instâncias da amostra, forneça resposta binária 0 e 1, os atributos que figuram como variável resposta precisam estar dispostos da mesma forma, portanto, caso não estejam, estes atributos precisam ser ajustados para que estejam no mesmo espectro de valor: 0 e 1.

Além disso, segundo Han et al. (2011) existem várias técnicas de pré-processamento de dados, a saber:

a) Limpeza de Dados (do inglês, *Data Cleaning*) - aplicada para remover ruídos e insconsistências nos dados;

b) Integração de Dados (do inglês, *Data Integration*) - unifica dados de múltiplas fontes em um repositório coerente, tal como um *data warehouse*[7]

c) Redução de Dados (do inglês, *Data Reduction* - técnica capaz de reduzir o volume dos dados, por agregação, eliminação de atributos redundantes ou clusterização.

d) Tranformações dos Dados (do inglês, *Data Transformations* - pode ser aplicada para normalizar os dados ou consolidar os mesmos, visando aprimorar a acurácia aferida junto aos algoritmos de aprendizado utilizados.

Ainda, consoante com Han et al. (2011), a limpeza dos dados é um dos processos mais comuns em um projeto de aprendizado ou mineração de dados, desta forma, portanto, objetiva preencher valores faltantes, atenuar ruídos ou identificar *outliers*[8]. No que tange ao tratamento de valores ausentes, os autores sugerem as seguintes possibilidades:

a) **Ignorar registro:** este método retira o registro da base de dados, o que pode redundar em perda de informação útil, caso os outros atributos estejam preenchidos, podendo incorrer em perda de informações úteis.

[7]Base de dados eminentemente destinada para para operações do tipo OLAP - *Online Analytical Processing*, comumente não relacionais.

[8]Valor aberrante ou atípico em uma série de dados, de forma que possui grande afastamento dos demais pontos.

b) **Preencher manualmente:** abordagem demasiadamente custosa no que tange ao tempo, podendo ser inexequível caso a base seja muito grande.

c) **Preencher usando valor constante:** substituir os valores ausentes por valores constantes, tal como algum rótulo ou *label*. Embora seja uma abordagem simples, isso pode impactar de forma negativa o método de aprendizado.

d) **Usar métricas de centralidade:** substituir valores numéricos ausentes (que exibem distribuição normal) pela média. Por outro lado, para distribuições assimétricas (do inglês, *skewed*) os autores recomendam a utilização da mediana.

e) **Utilizar o valor mais provável:** neste caso o valor mais provável pode ser determinado através de uma regressão ou árvore de decisão. Por exemplo, utilizando os demais atributos de um registro, na base de dados que possui algum campo sem valor expresso, é possível modelar uma árvore de decisão para indicar o valor para preenchimento.

Ademais, outra tarefa de grande importância, sobretudo no que diz respeito ao tratamento de textos, especialmente os que são produzidos na Web, figura na detecção de idioma, tendo um dado texto como entrada, o que pode configurar uma dificuldade adicional, Eisenstein (2013) exorta que, com o crescimento das mídias sociais, é possível ainda encontrar construções frasais incomuns e termos (palavras) que não figuram na maioria dos corpus linguísticos [9] mais conhecidos para estudos e análises. O autor apresenta alguns exemplos de textos (em inglês) que ilustram quão complexo pode vir a ser, trabalhar com textos produzidos no contexto das mídias sociais:

a) "Boom! Ya ur website suxxx bro." (Sarah Silverman)

b) "...dats why pluto is pluto it can neva b a star" (Shaquille O'Neil)

c) "Work on farm Fri. Burning piles of brush WindyFire got out of control. Thank God for good naber He help get undr control Pants-BurnLegWound." (Senador Charles Grassley)

Os textos acima representam alguns exemplos do tipo de desafio linguístico, isto porque: fazem uso de pontuação fora do padrão gramatical e evidenciam dissonância no que tange à sintaxe, vocabulário e ortografia (EISENSTEIN, 2013). Por conta disso, alguns

[9]Conjunto de textos escritos em um dado idioma, que serve como base de análise linguística (JURAFSKY; MARTIN, 2014).

trabalhos apontam que soluções que vigoram como o estado da arte em Processamento de Linguagem Natural - PLN podem ter baixo desempenho ao utilizar textos deste tipo para uma gama importante de tarefas tais como: análise de sentimento, *POS-Tagging*[10], etc. (TOUTANOVA et al., 2003) e (EISENSTEIN, 2013).

2.5.1 REPRESENTAÇÃO DE DADOS TEXTUAIS

De acordo com Rossi (2016) muitas técnicas vêm sendo propostas nas últimas décadas, visando extrair informação útil de grandes volumes de dados textuais. Sendo assim, neste expediente, a representação textual figura como um problema de grande relevância em áreas tais como: Busca e Recuperação da Informação (SCHÜTZE et al., 2008) (do inglês, *Information Retrieval*) e Processamento de Linguagem Natural (JURAFSKY; MARTIN, 2014).

Modelos de Busca e Recuperação da Informação (BRI) são eminentemente baseados em textos, ou seja, figuram como modelos que utilizam textos dos documentos para a confecção de rankings de acordo com uma dada consulta (do inglês, *query*) (BAEZA-YATES; RIBEIRO-NETO, 2008). Outrossim, de acordo com os autores, o processo de BRI pode ser dividido em 2 grandes atividades: (a) concepção de um *framework* lógico para representar documentos (e seus respectivos contéudos textuais) e consultas e (b) a definição de uma função para compor o ranking para cada documento a respeito de uma dada consulta.

No que tange ao Processamento de Linguagem Natural, a grande maioria dos processos visa identificar quão similar são duas ou mais *strings*. Segundo Chowdhury (2003), o processamento de Linguagem Natural (do inglês, *Natural Language Processing*) é a área que pesquisa e explora métodos que objetivam fazer os computadores entenderem e manipularem informação textual e, a partir disso, realizar tarefas úteis. Liddy (2001) preconiza que o Processamento de Linguagem Natural figura como um conjunto de técnicas computacionais para análise e representação de textos em um ou mais níveis de análise linguística, cujo propósito é alcançar e reproduzir o comportamento humano para um vasto espectro de tarefas e aplicações. Ainda segundo Liddy (2001), existem vários níveis de processamento, quando humanos compreendem ou representam uma linguagem. Importante notar, de acordo com os autores, Busca e Recuperação da Informação figura como uma subárea de PLN. Sendo assim, outros exemplos de aplicações em PLN seguem:

a) tradução automática (realizada por computador);

[10]Processo de rotulação de vocábulos em um texto, de acordo com sua respectiva classe gramatical.(JURAFSKY; MARTIN, 2014)

b) sumarização e processamento de texto em linguagem natural;

c) análise de sentimentos;

d) reconhecimento de autoria de textos, etc.

Liddy (2001) indica que uma das abordagens mais antigas para representar textos é o modelo conhecido como Bolsa de Palavras (do inglês, *Bag of Words* (BOW)). BOW utiliza TF-IDF[11], deste modo, em conformidade com a definição de Schütze et al. (2008), seja $\vec{V}(d)$ o vetor derivado do documento d, com um componente vetorial para cada termo ou vocábulo relevante. Tais componentes são computados usando o esquema TF-IDF(ver tabela 2.1). Desta forma, para falar sobre o modelo TD-IDF é mister abordar a definição formal do termo IDF, anteriormente. Seja K_i o r-ésimo termo mais frequente, ou seja, $n(r) = n_i$. Desta forma, associado com o termo K_i, o IDF_i é dado segundo Baeza-Yates e Ribeiro-Neto (2008) e Schütze et al. (2008) por:

$$idf_i = \log \frac{N}{n_i}$$

tal que N figura como o número total de documentos e n_i é o número de documentos onde o termo i ocorre. Deste modo, combinando o conceito de frequência do termo[12] (do inglês, *term frequency* - tf) com a definição anterior de IDF, tem-se:

$$tf\text{-}idf_{t,d} = tf_{t,d} \times idf_t$$

Em suma, $tf\text{-}idf_{t,d}$ designa ao termo t um peso no documento d que significa, segundo Schütze et al. (2008):

a) maior quando t ocorre muitas vezes dentro de um pequeno número de documentos;

b) menor quando o termo t ocorre poucas vezes em um documento ou em vários documentos;

c) menor possível quando o termo ocorre praticamente em todos os documentos do corpus, ou seja muito frequente.

[11]Do inglês, *Term Frequency-Inverse Document Frequency*, maiores detalhes em (SALTON; BUCKLEY, 1988)

[12]De acordo com Schütze et al. (2008) o indicador termo frequência pode ser computado de várias formas, mas a abordagem mais simples e utilizada é a simples frequência do termo em um documento.

Diante do exposto, um conjunto de documentos pode ser visto como um conjunto de vetores no espaço vetorial, de forma que existe um eixo para cada termo relevante ou frequente, de acordo com limites superiores e inferiores fornecidos como entrada ao algoritmo. Sendo assim, nesta representação, cada elemento do vetor informa a relevância de um termo ou palavra no contexto do conjunto de textos utilizados (SALTON; BUCKLEY, 1988). Em síntese, o modelo TF-IDF representa uma medida estatística que reflete o grau de importância de uma palavra no contexto do conjunto de documentos[13].

Por outro lado, consoante com Liddy (2001), o modelo BOW possui uma limitação importante no modo como representa os textos, pois este modelo retém apenas a frequência das palavras em um documento, determinando assim sua relevância, porém não informa a sequência da ocorrência dos mesmos, perdendo desta forma um traço de informação que pode ser importante na representação de um documento.

Termos	Doc1	Doc2	Doc3
car	0.88	0.09	0.58
auto	0.1	0.71	0
insurance	0	0.71	0.7
best	0.46	0	0.41

TAB. 2.1: Exemplo de valores para um modelo tf-idf para documentos. Termos frequentes terão valores próximos de 0.

Cavnar et al. (1994) e Croft e Lafferty (2013) apresentam um modelo estatístico baseado em N-Gramas para representação e modelagem de dados textuais[14]. Esta representação constrói vetores compostos de *strings* de N palavras (letras, fonemas, etc.) consecutivas extraídas dos documentos ou coleção de textos. Segundo Liu et al. (2005) este tipo de arranjo figura como uma representação textual eficaz, mantendo a ordem das palavras ou termos do texto, porém são inviáveis para a utilização em certas técnicas[15], por conta do grande número de dimensões geradas a partir de bases de dados de grande volume.

Diante disso, resumidamente, N-gramas faz uso de um modelo probabilístico de linguagem, que pode ser utilizado para predizer o próximo item de um trecho textual ou, de forma equivalente, conceber a representação de um texto através de uma compressão (CROFT; LAFFERTY, 2013). Isso é feito através da estimação da entropia dos textos

[13]Dicionários de vocábulos ou termos.

[14]Trata-se de uma sequência contígua de N itens de uma amostra textual. Estes itens podem ser sílabas, fonemas, palavras, letras ou palavras.

[15]Tais como *Principal Component Analysis*(WOLD et al., 1987) e *Latent Semantic Indexing*(BAEZA-YATES; RIBEIRO-NETO, 2008)

em inglês, através de experimentos realizados por Shannon (1951). A ideia por trás dos experimentos reside em uma questão: dada uma sentença de letras, qual é a probabilidade das próximas letras? Shannon (1951) preconiza que a partir da base de dados de treino, é possível derivar a distribuição de probabilidade para a próxima letra, dado um histórico de n letras: a=0.4, b=0.002, c=0.0000001, etc. Ou seja, o somatório de todas as probabilidades no que tange às possibilidades de letras para a sequência perfaz 1.

Desta maneira, na tarefa de modelagem de linguagem, assume-se a independência dos termos, seja cada palavra ou letra dependendo apenas das $n-1$ anteriores. Como ilustração, uma cadeia de Markov[16] de ordem $n-1$ seria uma aproximação possível do modelo, capaz de representar a linguagem em questão.

Consoante com Croft e Lafferty (2013), modelos baseados em N-gramas são incipientes para capturar os mais importantes relacionamentos entre termos em uma dada linguagem. De acordo com os autores, pois modelos baseados em N-gramas fazem fortes e falsas suposições acerca das localização dos termos do texto, pois

Uma alternativa é apresentada em Mikolov et al. (2011), um modelo chamado *Word2Vec*. Trata-se de um modelo criado por Mikolov no *Google*, que recebe como entrada um conjunto de documentos textuais e produz um conjunto de vetores no hiperespaço, de forma que a cada termo ou vocábulo único é designado um vetor correspondente único. Em geral, este modelo é composto por uma rede neural contendo 2 camadas escondidas, que são treinadas para reconstruir contextos linguísticos das palavras do corpus. Deste modo, palavras semanticamente próximas também estão proximamente localizadas no hiperespaço.

A utilização de diferentes parâmetros pode afetar de forma cabal a qualidade do modelo. Por conta disso, a acurácia pode ser aprimorada por diversos procedimentos, tais como: escolha do modelo de arquitetura (*CBOW* ou *Skip-Gram*), aumento da base de dados, utilização de thesaurus[17], aumento da base de dados de treino, aumento do número de dimensões do vetor, etc. De acordo com Mikolov et al. (2011), esses aprimoramentos demandam maior custo computacional.

Word2Vec é baseado em um modelo neural para processamento de linguagem natural balizado pelo que preconiza Bengio et al. (2003). Em suma, seja um conjunto de treino composto por uma sequência de palavras $w_1 ... w_t$ tal que $w_t \in V$, V sendo um vocabulário ou *corpus* grande, mas finito. Desta forma, de acordo com os autores, o modelo neural para

[16]Modelo estocástico utilizado para modelar sistemas que mudam de forma randômica(TIERNEY, 1994).

[17]Dicionário de sinônimos

linguagem natural objetiva aprender a função $f(w_t, ..., w_{t-n+1}) = \hat{P}(w_t|w_1^{t-1})$[18]. Segundo Bengio et al. (2003), esta função possui duas partes:

a) Um mapeamento C para qualquer elemento i de V para um vetor $C(i) \in \mathbb{R}^m$. Este mapemanto representa o vetor distribuído de atributos (do inglês, *distributed feature vector*) associado a cada palavra do vocabulário. C é representado por uma matriz $|V|$ x m de parâmetros livres.

b) Um função de probabilidade sobre as palavras expressas por C: uma função g mapeia uma sequência de entrada composta por um vetor de atributos para palavras no contexto, $(C(w_{t-n+1}), ..., C(w_{t-1}))$, para uma distribuição de probabilidade condicional no conjunto de palavras no vocabulário V para a próxima palavra w_t da sequência. Logo, de acordo com o que preconizam os autores, a saída de g é um vetor cujo o i-ésimo elemento estima a probabilidade $\hat{P}(w_t = i|w_1^{t-1})$ tal como na figura 2.8.

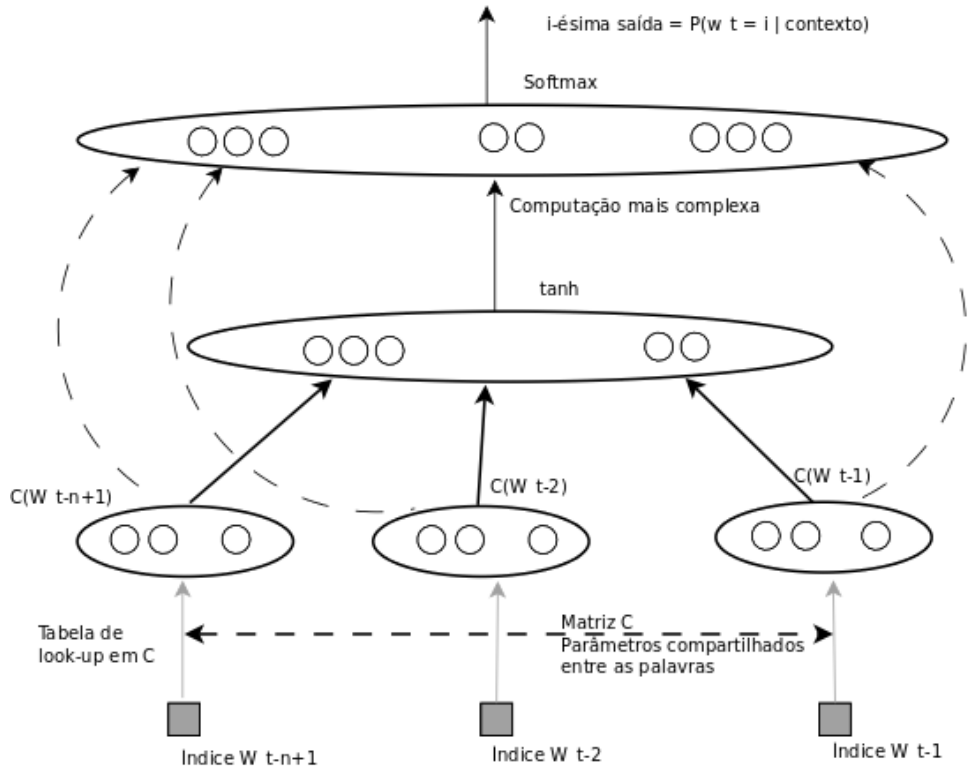

FIG. 2.8: Arquitetura neural $f(i, w_{t-1}, ..., w_{t-n+1}) = g(i, C(w_{t-1}), ..., C(w_{t-n+1}))$, tal que g é a rede neural e C é o i-ésimo vetor atributo da palavra. (BENGIO et al., 2003)

Consequentemente, a função f é a composição dos dois mapeamentos C e g, de forma

[18] $\hat{P}$ é um estimador, ou seja, uma estimativa para uma quantidade ou valor mensurados a partir dos dados observados. Este estimador é uma função de variáveis aleatórias.(JAYNES, 2003).

que C é compartilhado entre todas as palavras no contexto, onde cada uma destas duas partes possui parâmetros associados. No que tange à C os parâmetros são simplesmente os vetores, representados pela matriz C $|V|$ x m, tal que a linha i figura o vetor $C(i)$ para a i-ésima palavra. Por fim, g pode ser implementada por uma rede *feed-forward* ou uma rede recorrente. O conjunto de parâmetros pode ser descrito como: $\Theta = (C, \omega)$ e o respectivo treinamento está balizado na busca pelo Θ que maximiza a verossimilhança, tal como exorta Bengio et al. (2003):

$$L = \frac{1}{T} \sum_t \log f(w_t, w_{t-1}, ..., w_{t-n+1}, \Theta) \; + \; R(\Theta),$$

tal que $R(\Theta)$ é o termo de regularização, penalidade do decaimento do peso aplicado apenas aos pesos da rede neural e a matriz C, excetuando assim o *bias*[19].

CBOW (do inglês, *Continuous Bag of Words*) é um dos modelos neurais *Word2Vec* que prediz o valor da palavra corrente a partir de uma janela contendo palavras de contexto similar. Neste modelo, a ordem das palavras na janela não influencia a predição. Por outro lado, já o modelo *Skip-Gram* faz o inverso. utiliza a palavra corrente para predizer a janela de palavras de mesmo contexto (MIKOLOV et al., 2013a). De acordo com os autores, *CBOW* é mais rápido, mas em contrapartida o modelo *Skip-Gram* performa melhor no que tange às palavras menos frequentes (MIKOLOV, 2014). Importante notar, portanto, que *Word2Vec* figura como uma técnica para representação de dados textuais sem perda de informação.

CBOW foi a primeira arquitetura proposta (MIKOLOV et al., 2013a), de forma que as camadas escondidas (natureza não-linear) são removidas e a camada de projeção é compartilhada por todas as palavras e não somente compartilhada com a matriz de projeção. Como dito anteriormente, este modelo proposto é baseado no que apresenta (BENGIO et al., 2003), cuja arquitetura consiste das camadas de entrada, projeção, escondida e camada de saída. Portanto, como informam os autores em Mikolov et al. (2013a), na camada de entrada as N palavras anteriores são codificadas usando uma codificação de tamanho $|V|$, sendo V o vocabulário. A camada de entrada é então projetada para a camada de projeção P, que possui dimensionalidade N, utilizando a matriz compartilhada de projeção C. Como em qualquer momento apenas N entradas estarão ativas, por conta da quantidade N de palavras anteriores sendo consideradas, a composição da matriz de projeção é uma operação não muito custosa, de forma que a computação mais custosa figura nas operações entre as camadas de projeção e escondida, pois os valores na camada

[19]Bengio et al. (2003) aborda que o *bias* figura como os parâmetros aditivos da rede neural

de projeção são densos, relembrando que a camada escondida objetiva calcular a distribuição de probabilidade de todas as palavras do vocabulário V redundando em uma saída na última camada com dimensão V (MIKOLOV et al., 2013a). Logo, a complexidade da computação de treinamento para cada conjunto é dado pela seguinte equação, segundo Mikolov et al. (2013b) e Mikolov et al. (2013a):

$$Q = N \times D + N \times D \times H + H \times V$$

Skip-Gram representa outro modelo neural de linguagem natural (*Word2Vec*) proposto por Mikolov et al. (2013a), como dito anteriormente. Agora em vez de prever a próxima palavra, baseando-se no contexto, este modelo busca maximizar a classificação de um vocábulo utilizando como base outra palavra na mesma sentença. Ou seja, Mikolov et al. (2013a) utiliza cada palavra corrente da sentença como um padrão de entrada para um classificador log-linear, e busca predizer palavras dentro de uma certa faixa antes e depois da palavra corrente. Os autores ainda informam que o aumento da faixa constitui um *trade-off* entre aumento da eficiência na representação e o respectivo custo computacional. Palavras distantes são menos relacionadas com a palavra corrente do que as palavras mais próximas, portanto, um peso é utilizado para quantificar esta força do relacionamento. A complexidade desta arquitetura, segundo (MIKOLOV et al., 2013a) pode ser representada como:

$$Q = C \times (D + D \times \log_2(V)),$$

de forma que C representa a maior distância entre as palavras. Seja $C = 5$, por exemplo, o algoritmo selecionará de forma randômica um número R entre $[1, 5]$, e utilizará este valor R de palavras anteriores e subsequentes à palavra corrente para usar como rótulos corretos para classificação. Isso demandará $2R$ classificações com a palavra corrente como entrada do modelo e cada uma das $R + R$ palavras como saída do modelo.

Por fim, como outra alternativa para representação de textos, Zhang et al. (2015) expõe uma técnica para a formatação de textos sem perda (do inglês, lossless), em nível de caractere, que utiliza uma técnica chamada *char quantization*. Esta técnica mapeia cada caractere que compõe uma mensagem para um vetor de *string* binária, informando no mapeamento o caractere corrente encontrado, de acordo com o mapa de carcateres utilizado. Desta forma, usando um mapa de caracteres distintos específico, cada caractere é representado por um vetor de $1 \times N$, tal que N representa o número de caracteres distintos representados pelo mapa. Zhang et al. (2015) exibe excelentes resultados utilizando a técnica proposta baseada em Redes Neurais Convolucionais aplicadas em classificação de textos.

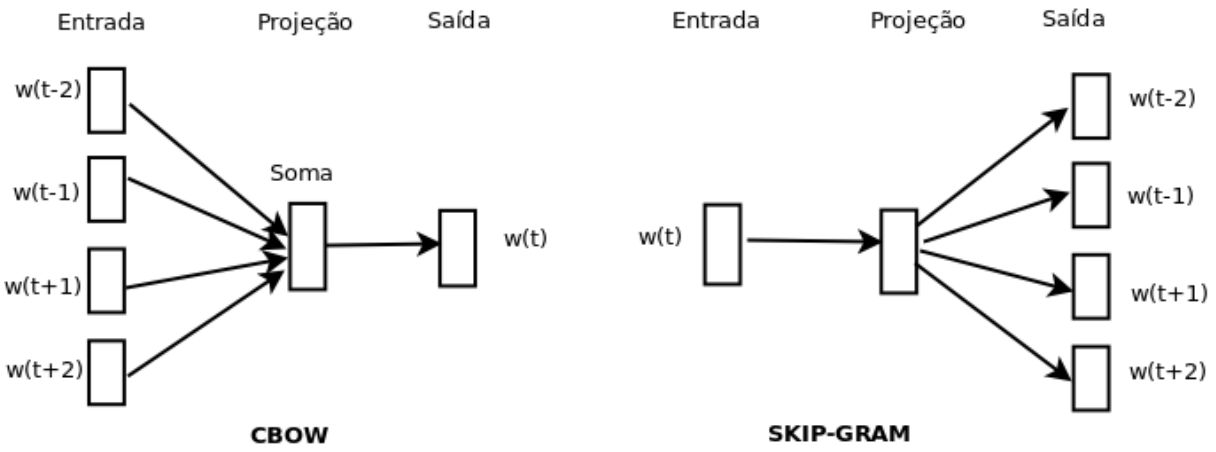

FIG. 2.9: Arquiteturas *CBOW* e *SKIP-GRAM* (MIKOLOV et al., 2013a)

2.6 CLUSTERIZAÇÃO DE DADOS

Clusterização é a tarefa de descobrir agrupamentos (do inglês - *clusters*) naturais para um conjunto de padrões, pontos ou objetos. Jain (2010) define clusterização como uma técnica estatística de classificação que visa descobrir se indivíduos de uma população são lançados em grupos diferentes quando são realizadas comparações quantitativas de múltiplas características. Ou seja, a clusterização figura como um processo geral de aprendizado não supervisionado, que busca segmentar indivíduos ou instâncias de uma amostra em grupos (ou *clusters*), sem conhecer algum atributo resposta (ou classe), ou seja, trata-se de uma tarefa geral que apreende um determinado padrão ou similaridade de uma instância, olhando eminentemente para os dados da amostra. Marsland (2015) preconiza que clusterização é o processo de identificar similaridades entre instâncias do conjunto de dados de entrada, que porventura pertençam a um mesmo cluster ou agrupamento, de forma que para tal não é possível utilizar qualquer fonte de informação para validar a pertinência destas instâncias a este ou qualquer outro cluster identificado. Sendo assim, o processo descobre por ele mesmo a informação de pertinência utilizando tão somente os dados de entrada.

Carvalho et al. (2011) prescreve a existência de diversos algoritmos de agrupamento, expondo que cada algoritmo emprega um critério específico de agrupamento, impelindo assim uma estrutura de dados. Desta maneira, caso os dados estejam em conformidade com as exigências da técnica empregada, a estrutura verdadeira dos clusters pode ser encontada, de acordo com os autores.

A noção de *cluster* não é definida de forma consensual ou uniforme, pois cada algoritmo de clusterização preconiza uma definição distinta, por exemplo Estivill-Castro

(2002) declara que objetos podem ser agrupados baseados pelo princípio de maximização da similaridade inter-classe ou pela minimização da similaridade intra-classe. Em Carvalho et al. (2011) os algoritmos de clusterização são classificados de acordo com a técnica utilizada para definir os clusters, tais como[20]:

a) Hierárquicos - algoritmos que geram uma sequência de partições aninhadas a partir de uma matriz de proximidade.

b) Particionais - algoritmos que otimizam o critério de agrupamento utilizando uma técnica iterativa baseada em erro quadrático.

c) Baseados em Densidade - grupo de técnicas que assumem que os *clusters* são regiões de alta densidade de objetos, separadas por regiões com baixa densidade no espaço de objetos.

d) Baseados em Grafo - técnicas que usam teoria dos grafos, de modo que a base de dados é representada por um grafo de proximidade.

e) Basedos em Redes Neurais - utilizam redes neurais artificiais para definir os agrupamentos.

f) Baseados em Grid - algoritmos que definem um reticulado (do inglês, *grid*) para o espaço de dados e realiza todas as operações nesse espaço reticulado.

Algoritmos hierárquicos podem ser divididos em dois grupos: aglomerativo, que começa com n *clusters* com um único objeto e forma a sequência de partições agrupando os clusters sucessivamente, e o divisivo, que inicia o algoritmo com o cluster contendo todos os objetos e forma a sequência dividindo os *clusters* sucessivamente. Sendo assim, um algoritmo hierárquico aglomerativo gera uma sequência de partições de n objetos de k clusters, de forma que o nível 1 apresenta n *clusters* de um objeto e o nível n apresenta um *cluster* com todos os objetos (CARVALHO et al., 2011).

Algoritmos particionais iniciam através da criação de uma partição inicial. Em seguida, os objetos são movidos de um *cluster* para outro com o objetivo de melhorar o valor do critério de agrupamento (erro quadrático, o que garante a compactação dos *clusters*). Esses algoritmos são eficientes, mas podem convergir para um ótimo local, obejtivando minimizar o erro quadrático para um número fixo de clusters (CARVALHO et al., 2011).

[20]Importante notar que um algoritmo pode ser enquadrado em mais de uma categoria.

Técnicas de clusterização baseadas em densidade são capazes de obter clusters de formas arbitrárias, pois um cluster definido como um componente denso conectado pode crescer em qualquer direção dada pela densidade. Diante disso, o cálculo da função densidade global requer a soma das funções "influência"associada a todos os pontos, logo, para a definição da função influência, utiliza-se uma função distância qualquer, que seja reflexiva e simétrica. DBSCAN (do inglês, *Density-Based Spatial Clustering of Applications with Noise*, apresentado em Ester et al. (1996)) figura como um exemplo nesta vertente (CARVALHO et al., 2011).

Algoritmos baseados em grafos utilizam os dados representados tal como um grafo de proximidade. De maneira mais simples, cada nó figura como objeto e é conectado com os $n-1$ nós restantes, redundando em uma grafo completo. Assim sendo, os pesos nas arestas informam a similaridade ou distância entre os objetos (CARVALHO et al., 2011). Portanto, segundo os autores, os métodos de agrupamento decompõem o grafo em componentes conectados pela remoção de arestas inconsistentes, ou, podem vir a inserir alguma aresta, balizado por algum critério. No fim, cada um destes componentes resultantes do processo de agrupamento vai representar um *cluster* (CARVALHO et al., 2011).

Estratégias baseadas em redes neurais utilizam modelos de redes neurais artificiais voltadas para o aprendizado não-supervisionado. Dentre as abordagens que figuram neste grupo, a rede SOM (do inglês, Self-Organizing Map (KOHONEN, 2013)) é a mais proeminente (CARVALHO et al., 2011). As redes SOM possuem neurônios dispostos em um reticulado (do inglês, *grid*) uni ou bidimensional. Deste modo, cada um dos neurônios desta rede está ligado a todas as respectivas entradas. Esta rede possui geralmente apenas uma camada computacional (camada escondida). Para cada instância do padrão de entrada apresentado, os neurônios da camada de saída competem para terem seus respectivos pesos ajustados. Diante disso, o neurônio com o maior valor de ativação figura como o vencedor. Segundo Carvalho et al. (2011) a localização espacial de uma vizinhança topológica de neurônios ativados é determinada e centrada no neurônio vencedor. Deste modo, o próximo passo busca a adaptação dos pesos, de forma que que o neurônio vencedor e seus vizinhos, que tiverem seus pesos ajustados, aumentam seus valores de ativação para futuras instâncias similares à instância corrente (CARVALHO et al., 2011).

Algoritmos baseados em grid definem um reticulado para o espaço de dados e neste espaço realiza suas operações, na medida que, em termos gerais, essa estratégia mostra-se muito eficiente para um grande conjunto de dados, conseguindo lidar bem com eventuais ocorrências de *outliers* (CARVALHO et al., 2011). CLIQUE (AGRAWAL et al., 1998)

(do inglês, *Clustering in Quest*) e MAFIA (NAGESH et al., 2001) (do inglês, *Merging of Adaptative Finite Intervals*) são exemplos de algoritmos baseados em reticulado.

Diante do exposto, em suma, clusterizar ou agrupar objetos, instâncias ou entidades demanda uma noção de medida para determinar se estes pontos são similares entre si. De acordo com Rokach e Maimon (2005), existem dois principais tipos de medida: medidas de distância e medidas de similaridade. Os autores ainda indicam que a distância entre dois pontos ou instâncias x_i e x_j como: $d(x_i, x_j)$, e para ser válida, esta medida precisa ter natureza simétrica, sendo infinitamente próxima de zero para os casos de similaridade máxima.

2.6.1 K-MEANS

O *K-means* figura como um método (algortimo guloso) de particionamento dos dados no expediente de análise dos dados, de forma que os dados apresentados ao algoritmo (conjunto de entrada) são objetos dispostos de acordo com suas respectivas localizações, tendo suas respectivas distâncias calculadas para todas as demais observações que compõem o conjunto de entrada apresentado ao algoritmo (GHOSH; DUBEY, 2013). Ou seja, o *K-means* empreende o particionamento destes objetos em agrupamentos (*clusters*, do inglês) mutuamente disjuntos, de modo que cada objeto em um mesmo cluster está disposto de forma aproximada de seus pares, porém, em contrapartida, distantes dos demais objetos que não fazem parte deste *cluster* ou agrupamento. Seguindo a taxonomia proposta em Carvalho et al. (2011). *K-means* faz parte do grupo de algoritmos baseados em erro quadrático.

No contexto do algoritmo, cada *cluster* é representado pelo seu elemento mais ao centro, denominado como centróide. Sendo assim, o algoritmo K-means objetiva encontrar uma partição tal que os objetos de cada cluster elencado, possua uma distância mínima do centróide deste cluster. Portanto, segundo Jain (2010) o erro quadrático entre o centróide c_k e os demais pontos do cluster K pode ser definida como:

$$J(c_k) = \sum_{x_i \in \{C_K\}} ||x_i - c_k||^2$$

Portanto, em outras palavras, o objetivo do algoritmo K-means é minimizar a soma dos erros quadráticos sobre todos os K clusters elencados pelo algoritmo, desta forma a soma pode ser expressa tal como:

$$J(C) = \sum_{k=1}^{K} \sum_{x_i \in \{C_K\}} ||x_i - c_k||^2$$

De acordo com Jain e Dubes (1988) o algoritmo *K-means* pode ser descrito pelos seguintes passos:

a) Ajustar o valor K para o valor fornecido ao algoritmo para o número de clusters desejado;

b) Escolher de forma aleatória K pontos iniciais que serão usados como estimativas iniciais para os K centróides;

c) Gerar uma nova partição ao designar cada objeto ou dado da amostra para o centróide gerado mais próximo;

d) Recálculo dos K centróides;

e) Os passos c e d serão repetidos até que nenhuma observação mude seu cluster designado ou até que os K centróides não mudem mais.

Conforme dito antetiormente, o *K-means* é baseado no erro quadrático, ou seja, o algoritmo minimiza a distância entre cada objeto ou instância observada e o centróide do *cluster* ao qual ele pertence, de forma que essa função objetivo é minimizada por clusters de formato globular (hiperesférico) do mesmo raio ou *clusters* bem separados (CARVALHO et al., 2011).

Importante notar que o algoritmo é sensível à escolha inicial dos centróides e da sua forma de atualização, pois dependendo da escolha dos centróides, o algoritmo pode vir a convergir para um ótimo local (CARVALHO et al., 2011). Ademais, dependendo do tipo de distância utilizada, é restrita a objetos em espaços euclidianos e em geral os clusters deste algoritmo são desbalanceados[21], conforme sugerem os autores.

2.6.2 DBSCAN

O *DBSCAN* é um algoritmo de clusterização baseado em densidade, ou seja, assume que os clusters são regiões de alta densidade, no espaço de objetos (CARVALHO et al., 2011). Deste modo, um cluster definido como um componente denso conectado cresce em qualquer direção dada pela densidade, desta forma os *clusters* obtidos podem ter formas arbitrárias (ESTER et al., 1996).

A ideia principal do algoritmo é que para cada ponto de um *cluster*, a vizinhança (obtida por um determinado raio) precisa conter um número mínimo de pontos, ou seja, a densidade da vizinhança precisa ser superior a um determinado limiar (ESTER et al.,

[21] Exibem distribuição de elementos não uniforme entre os *clusters*.

1996). Logo, segundo os autores, o formato da vinzinhança é determinado pela escolha da distância entre dois pontos p e q, denotando $dist(p, q)$. Por exemplo, ao utilizar a distância de Manhatan em um espaço de duas dimensões, o formato da vizinhança é retangular (ESTER et al., 1996).

2.6.3 AFFINITY PROPAGATION

O *Affinity Propagation* é uma técnica de clusterização criada por Frey e Dueck (2007). Este algoritmo utiliza um mecanismo de troca de mensagens para dividir os dados em subgrupos de interesse utilizando o conceito de troca de mensagens entre dados ou instâncias (do inglês, *data points*). Atuando de forma diversa de outros algoritmos conhecidos de clusterização tal como *k-means*, por exemplo, *Affinity Propagation* não requer que o número de *clusters* seja passado como parâmetro. Ou seja, o algoritmo não força um número pré-estabelecido de *clusters*, mas sim obtém o número a partir da base de dados fornecida como entrada, e pode ser ajustado através de parâmetros, de acordo com a conveniência e do tipo do problema.

Ainda de acordo com Frey e Dueck (2007), *Affinity Propagation* recebe um número real $S(k, k)$ para cada ponto k. Desta forma, cada um destes pontos cujos valores sejam maiores que $S(k, k)$ possuem maior probabilidade de serem escolhidos como **exemplares** (*clusters*). Estes valores $S(k, k)$ figuram como um parâmetro do algoritmo conhecido como **preferências**. Desta forma, o número de exemplares identificados, ou o número de *clusters* é diretamente influenciado pelo valor do parâmetro preferência. Caso todos os pontos possuam a mesma probabilidade para serem eleitos como exemplares, o valor do parâmetro preferência pode ser ajustado para um valor comum, de forma que o ajuste deste venha propiciar diferentes números de clusters como saída do algoritmo. Como ilustração, o valor comum do parâmetro preferência pode ser a mediana das similaridades do padrão de entrada, resultando em um número moderado de clusters ou o menor valor das similaridades, o que resultaria em um número menor de clusters, segundo Frey e Dueck (2007).

Affinity Propagation utiliza dois tipos de mensagens trocadas entre os pontos de dados, de acordo com Frey e Dueck (2007). As mensagens podem ser utilizadas a qualquer momento para decidir quais pontos serão eleitos exemplares e para cada ponto qualquer ou instância, a qual exemplar ou cluster este elemento pertence. Logo, a responsabilidade $R(i, k)$, enviada do ponto i para o candidato ao cluster k, outro parâmetro conhecido utilizado pelo algoritmo, denota a evidência acumulada do quão apropriado o ponto k

seria ao ser escolhido como exemplar para o ponto ou instância i, levando em consideração todos os outros candidatos a exemplares. Além disso, a disponibilidade $A(i,k)$ representa também a evidência acumulada de quão apropriado seria para o ponto i escolher o ponto k como seu exemplar ou cluster, levando em consideração o suporte oriundo de todos os demais pontos, dos quais k poderia ser um exemplar. Deste modo, $A(i,k)$ é inicializado com o valor zero. Portanto, as responsabilidades são calculadas utilizando a seguinte regra:

$$R(i,k) = S(i,k) - max_{k's.t.k' \neq k} A(i,k') + S(i,k')$$

No começo do algoritmo, todas as disponibilidades $A(i,k)$ são ajustadas para o valor zero, $R(i,k)$ é configurado para a similaridade de entrada entre os pontos i e k, seu respectivo exemplar, menos a maior similaridade entre o ponto i e os demais candidatos a exemplar de i. Sendo assim, a atualização destas variáveis de interesse é orientada pelos dados ou instâncias da base, e não leva em consideração, portanto, demais pontos e candidatos, respectivamente, cada um leva em consideração para ser eleito como um exemplar. A posteriori, ao fim do algoritmo, quando alguns pontos são designados como exemplares ou clusters, seus respectivos valores de disponibilidade $A(i,k)$ caem para valores abaixo de zero, de acordo com a regra de atualização do algoritmo, removendo estes pontos do certame.

Diante do exposto, seja ainda para $k = i$, o parâmetro chamado responsabilidade $R(k,k)$, que é ajustado para o valor de preferência que o ponto K possui de ser escolhido exemplar, $S(k,k)$, subtraindo-se o maior valor das similaridades entre o ponto i e todos os demais candidatos a exemplar. Logo, de acordo com Frey e Dueck (2007), este valor de responsabilidade apreende matematicamente a evidência acumulada de que o ponto ou instância k possui de poder vir a ser eleito exemplar, baseado tão somente em seu valor de preferência de entrada, modificado pelo fator que informa quão inapropriado seria que este ponto seja designado como outro exemplar. Embora, as atualizações das respectivas responsabilidades permitam que todos os candidatos a exemplar possam competir pela posse de um dado ou ponto, a seguinte relação de atualização coleta indícios dos dados, para obter a informação se cada candidato a exemplar seria de fato um bom exemplar ou cluster, caso fosse selecionado:

$$A(i,k) = min(0, r(k,k) + \sum_{i's.t.i' \notin \{i,k\}} \{max\{0, R(i',k)\}\})$$

Assim sendo, a disponibilidade $A(i,k)$ é ajustada para o valor de responsabilidade própria $R(k,k)$ adicionado ao somatório das responsabilidades, maiores que zero, o candidato a exemplar k é notificado por outros ou pontos. Isto se deve ao fato de ser apenas

necessário para um bom cluster ou exemplar, explicar bem algumas instâncias (responsabilidades positivas), a despeito de quão ruim ele é ao explicar outras instâncias ou pontos (responsabilidade negativa). Entretanto, caso a responsabilidade $r(k, k)$ seja menor que zero, isso indica que o ponto k está mais ajustado a figurar como componente de um outro cluster ou exemplar do que pertencer ao cluster descrito por si. Ademais, se outros pontos obtiverem valores positivos de responsabilidade para k, este pode vir a figurar como um exemplar desses respectivos pontos.

De modo a limitar a influência mensagens contendo grandes valores positivos de responsabilidade, o somatório de todos valores de responsabilidade entrantes é limitado a zero. Por outro lado, a disponibilidade própria de k, representada por $A(k, k)$ é atualizada de forma diversa, de acordo com a seguinte equação:

$$A(k, k) = \sum_{i' s.t. i' \notin \{i,k\}} \{max\{0, R(i', k)\}\})$$

Esta equação denota o que a mensagem evidencia de forma acumulada sobre quão apropriado é o ponto k, para ser eleito como o respresentante do cluster, baseado exclusivamente nas mensagens de responsabilidades recebidas e oriundas dos demais pontos. Importante ressaltar que as mensagens são trocadas apenas por pares de pontos ou instâncias, cujas similaridades são conhecidas de antemão.

Diante do exposto, os valores de responsabilidade e disponibilidade serão combinados com o intuito de identificar os exemplares. Seja um dado ponto t, o valor de k que maximiza a expressão: $A(t, k) + R(t, k)$ pode indicar dois cenários a saber:

a) caso k = t, identificar t como um exemplar;

b) identificar o ponto ou instância que figurará como exemplar e, portanto, irá conter t em seu *cluster*.

O algoritmo termina depois de alguma iterações, ou após as atualizações nas mensagens trocadas entre os pontos ficarem abaixo de um determinado limiar, ou após as decisões locais (na perspectiva do ponto) permanecerem constantes após um certo número de iterações do algoritmo.

Desta forma, em suma, *Affinity Propagation* é um algoritmo que objetiva encontrar um conjunto ótimo de pontos (exemplares) da base de dados (do inglês, *dataset*) para respresentar os clusters, e consequentemente, obter a soma das similaridades ou distâncias de todos os pontos da base até os seus respectivos exemplares, tal que esta soma seja máxima (ou mínima). Seguindo uma estratégia de troca de mensagens entre os pontos,

que objetiva unicamente atualizar toda a rede até chegar a um estado de convergência. *Affinity Propagation* requer apenas cálculos simples e locais (na perspectiva do ponto) e, por conta disso, é considerado mais eficaz que os demais algoritmos baseados em k-centróides (FREY; DUECK, 2007).

3 TRABALHOS RELACIONADOS

A disseminação de spam no Twitter vem ganhando destaque (aparente) sobre outros métodos mais conhecidos de disseminação, tal como: *email*, mensagens instantâneas, etc. Além da disseminação de spam, diversas atividades atribuídas aos *bots* também vêm ganhando volume no que tange às ocorrências no Twitter, tais como: manipulação de *trending topics* (BENEVENUTO et al., 2010), (RATKIEWICZ et al., 2011a), (AIELLO et al., 2012) e (BOLLEN et al., 2011); *link farming* - uma forma de adulterar os índices nos sistemas de busca com a utilização de *links* externos, forjando relevância momentânea no *PageRank*, (GHOSH et al., 2012); *phishing*[22] (BOSHMAF et al., 2013) e contas associadas a *bots* ou a atividades ilegais nos trabalhos de (THOMAS et al., 2013), (TIMBERG, 2013) (CHU et al., 2012) e (BERGER; MORGAN, 2015).

Desta forma, com o aparente crescimento dos trabalhos de pesquisa neste tema (FERRARA et al., 2016), os *social bots* têm exibido um aumento de sofisticação, assim também como um aprimoramento nas técnicas de evasão que visam ofuscar sua presença nas redes sociais (WANG et al., 2013). Por conta disso, as maiores empresas (Facebook, Twitter, MySpace, etc.) fornecedoras de serviços de redes sociais (do inglês, *OSN - Online Social Networks*) estão aumentando o direcionamento de seus esforços para aprimorar seus respectivos sistemas de defesa contra estas entidades indesejadas, sobretudo por conta da propagação de boatos e informações falsas (do inglês, termo mais comumente empregado é *fake news*) e manipulação dos conhecidos *trending topics* (JI et al., 2016). Isso tudo em um cenário, onde dados estatísticos confirmam o forte impacto dos *social bots*: um levantamento feito no Twitter mostra que 8,5% das contas existentes são controladas por *social bots*[23].

Baltazar et al. (2009) figura como o primeiro trabalho a reportar a atuação de um *socialbot* em uma rede social em meados de 2008, batizado como *koobface*, e na época rotulado como um *malware* revolucionário, pois ele era capaz de: furtar informações dos usuários das redes, enviar *spams* e adulterar estatísticas voltadas para os mecanismos de busca. Ademais, abordando ainda os primórdios da atuação dos *social bots*, Naz foi o primeiro *sociabot* descoberto no *Twitter* em 2009 (NAZARIO, 2009). Este *bot* utilizava

[22]Técnicas de engenharia social e de subterfúgios voltadas para obtenção de dados de identidade e finanças dos usuários (ALSHARNOUBY et al., 2015).

[23]http://time.com/3103867/twitter-bots/

os protocolos do *Twitter* para estabelecer comunicação de comando e controle (C&C) para furto de informações dos diversos usuários da rede.

É mister apontar que a grande maioria dos trabalhos no expediente de detecção de *social bots* utiliza técnicas baseadas em atributos comportamentais (FERRARA et al., 2016). Outrossim, por outro lado, as abordagens que utilizam informação da topologia rede social, advogam que a utilização de técnicas de detecção de comunidades pode expor a presença destas entidades, porque admitem como premissa que os *socialbots* estabelecem conexões com outros *bots* de modo a evadir dos mecanismos de defesa dos serviços de redes sociais. Contudo, isso não é verdadeiro sempre, e isto é preconizado por Ferrara et al. (2016) e Ji et al. (2016). Por outro lado, quando presente em uma composição de solução, na grande maioria das vezes a abordagem topológica atua de forma bastante eficaz. Existem diversos trabalhos que abordam esse tipo de cenário, tais como: Ji et al. (2016), Yang et al. (2015), Beutel et al. (2013), Alvisi et al. (2013), Wang et al. (2016), Boshmaf et al. (2015).

Os *socialbots* mais modernos têm exibido grau crescente de sofisticação e ,atualmente, estas entidades têm se mostrado capazes de criar perfis contendo vasta quantidade de informação, exibindo fotos de jovens atraentes, objetivando assim atrair a atenção de perfis autênticos. Com isso, estas contas automatizadas conseguem aumentar o seu índice de aceitação à solicitação de ligação a usuários terceiros. Por conta disso, por conseguinte, elas têm obtido grande êxito ao dirimir o efeito dos mecanismos de detecção ao ocultar traços já conhecidos dos *social bots*. [24]

Outrossim, acerca da crescente modernização dos *socialbots*, Nagaraja et al. (2011) reporta a utilização de mecanismos de esteganografia (técnica que oculta códigos de instruções ou mensagens em arquivos de imagem ou áudio) para o estabelecimento de comunicação e sinalização entre os *bots* e seus respectivos *bot masters*. Desta forma, os *socialbots* fazem *download* das imagens enviadas pelo *botmaster* e extraem os comandos ocultos no aquivo de imagem. Após a execução destes comandos, os *bots* ocultam as mensagens coletadas em imagens e fazem *upload* das mesmas. Por sua vez, os *botmasters* coletam estas imagens e extraem todo o conteúdo oculto (FERRARA et al., 2016).

A Tabela TAB 3.1 exibe um panorama sumarizado acerca dos trabalhos relacionados (em ordem alfabética), divididos pelos tipos de abordagem e pela forma de tratamento realizada no que tange ao conteúdo textual, ou seja, a utilização do conteúdo textual original

[24]Ligações ou amizades com contas autênticas figuravam em baixa monta, pois praticamente suas respectivas redes eram compostas por outros *bots*, mas esse traço é administrado de forma inteligente pelos *social bots* (YANG et al., 2014).

ou estatísticas relacionadas aos textos gerados pelas contas contempladas no *dataset*.

Além disso, o presente trabalho apresenta uma taxonomia na figura 3.1 sobre os trabalhos relacionados, de acordo com as duas classificações abordadas, anteriormente na seção 2.1.

Sob o prisma das abordagens topológicas, Beutel et al. (2013) é voltado para o ambiente do Facebook, onde os autores apresentam um algoritmo de detecção de comunidades para identificar *social bots* ou *sybils*, partindo do princípio que estas contas estão mais fortemente conectadas a outros *bots*, do que ligados a contas legítimas. Cao et al. (2012) apresenta uma abordagem topológica baseada em técnicas de *Random Walks* (LAMBIOTTE et al., 2014). Davis et al. (2016) apresenta um rotulador de contas automatizadas utilizando diversos atributos, dos mais variados tipos. Dados topológicos tais como: medidas de centralidade, distribuição de graus e medidas de centralidade são também utilizados.

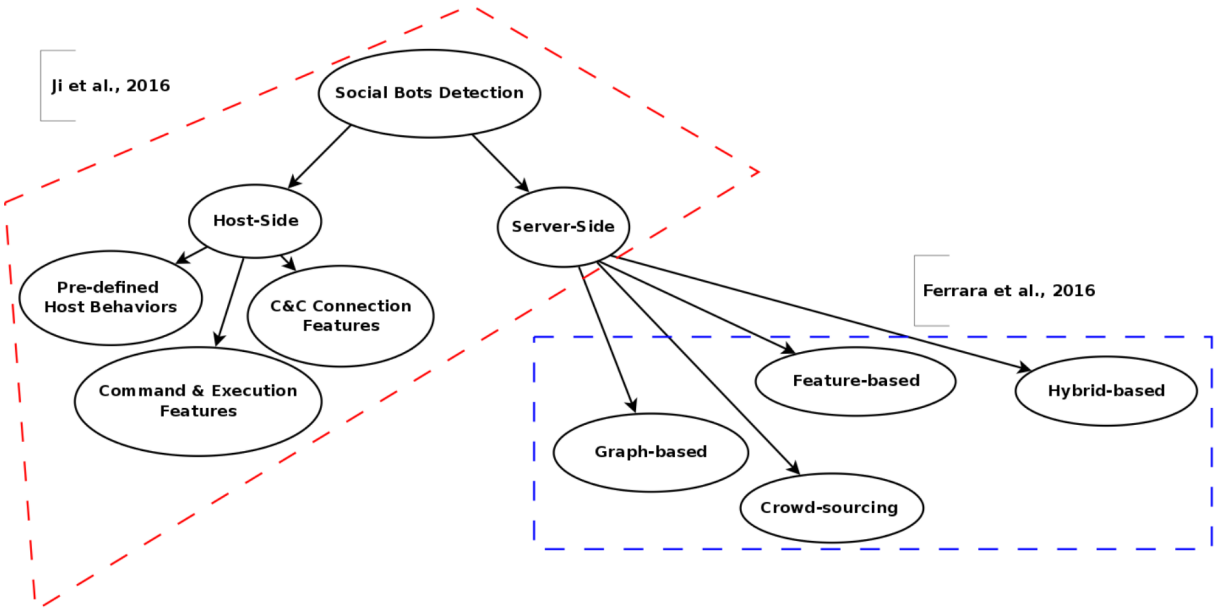

FIG. 3.1: Proposta de combinação de taxonomias, em vermelho Ji et al. (2016) e em azul Ferrara et al. (2016)

A abordagem baseada em atributos comportamentais (do inglês, *feature based*) figura como a abordagem mais utilizada pela maioria dos trabalhos de detecção de *socialbots* (FERRARA et al., 2016). Badri Satya et al. (2016) utiliza técnicas de classificação de contas utilizando dados acerca dos perfis e padrão comportamental no que tange aos *likes* e demais dados de comportamento no ambiente do Facebook. Ao fim do estudo, promove um *ranking* de 30 técnicas de aprendizado supervisionados no que tange à acurácia deste modelos. Lee et al. (2011) utiliza um mecanismo de rotulação de contas baseado em

diversas informações sobre contas do Twitter. Utilizam diversas técnicas de aprendizado supervisionado, de modo a apresentar um rotulador baseado *Random Forest* exibindo o maior acurácia, superando modelos baseados em Redes Neurais (*MLP*) (ROSENBLATT, 1961) e *SVM* (CORTES; VAPNIK, 1995). Xiao et al. (2015) apresenta uma abordagem para o Linkedin visando a detecção de *socialbots* no momento do cadastro de uma conta. Ou seja, o objetivo é não permitir que nenhuma conta automatizada entre na rede. Utilizam como atributos informações de cadastro de conta, endereço IP utilizado no cadastro, etc. (YANG et al., 2014) desenvolveu um método de detecção de *sybils* para a segunda maior rede social da China: RenRen. Para tal, fizeram uso de uma base de dados proprietária contendo o padrão de navegação dos usuários. Outrossim, implementaram um detector offline utilizando um mecanismo de detecção de similaridade textual dentro de uma janela de tempo, objetivando identificar contas que replicam conteúdo similar de mensagens na rede. Outros exemplos de trabalhos nesta abordagem: (BEUTEL et al., 2013), (LEE et al., 2011) e (YANG et al., 2015), (BRAZ; GOLDSCHMIDT, 2017),(HWANG et al., 2012), (WANG et al., 2013).

Sob o prisma das técnicas baseadas na abordagem híbrida (*hybrid-based*), onde mais de uma fonte de informação é utilizada, de acordo com Ferrara et al. (2016), Yang et al. (2014) apresenta uma solução para detecção de contas automatizadas no contexto da RenRen[25] (segunda maior rede social na China) utilizando dados da topologia da rede e padrões de click (do inglês, *clickstream* de cada conta de usuário). Este trabalho exibe um estudo bastante aprofundado sobre o comportamento dos *social bots* (os autores chamam estas entidades de *sybils*), com enfoque às ligações que os *sybils* realizam para aumentar seu número de seguidores ou amigos. Portanto, preconizam que os *sybils* não formam comunidades entre si sempre, de forma a dirimir os efeitos dos mecanismos de detecção da Renren baseados em detecão de comunidades. Esta decisão é tomada de forma mais oportuna, caso a conta tenha um baixo nível de aceitação no que tange às solicitações enviadas aos demais usuários. Ademais, corroboram que os *sybils* podem cooperar mutuamente para empregar ataques tal como propagação de mensagens indesejadas (do inglês, *spam*) e manipulação de tendências na rede.

(DAVIS et al., 2016) apresenta um sistema sob o prisma das abordagens híbridas e figura hoje como estado da arte em detecção de *social bots* (GILANI et al., 2017). Este sistema utiliza diversas fontes de informação tais como: dados topológicos, estatísticas acerca das contas, estatísticas dos textos propagados, *pos-tagging*[26], etc. Sendo assim,

[25]http://www.renren-inc.com/en/

[26]Processo de rotulação de palavras quanto à sua classe gramatical, tal como: verbo, substantivo,

além do estudo aprofundado acerca destas fontes de informação, os autores criaram uma solução (*Bot Or Not*) *online* baseada em *Random Forest* (o modelo que apresentou a maior acurácia) para classificar contas do Twitter. A despeito de utilizar uma abordagem híbrida, este trabalho não utiliza o conteúdo textual das mensagens propagadas, em vez disso utiliza estatísticas relacionadas aos textos, o que por si só incorre em perda de informação (BEAUDRY; RENNER, 2012). Ademais, mesmo sendo considerado um trabalho referência para a detecção de *social bots*, o sistema *Bot Or Not* apresenta muitos erros de classificação de contas automatizadas (GILANI et al., 2017).

Por outro lado, quando o conteúdo textual é utilizado diretamente na detecção de contas automatizadas, a grande maioria das abordagens modela suas respectivas soluções na extração de estatisticas sobre os textos propagados pelas contas. Lee et al. (2011), Davis et al. (2016), Igawa et al. (2015) e Barbon et al. (2017) figuram como exemplos de trabalhos que retiram dados estatísticos do conteúdo textual, tais como: quantidade de URL's no texto, quantidade menções únicas a contas de terceiros, número de palavras em um tweet, número de *POS tags* em um *tweet*, entropia de palavras utilizadas no tweet, etc. Logo, quando estatísticas dos textos são utilizadas como atributos para o treinamento de um modelo, fica configurada a perda de informação, segundo Beaudry e Renner (2012).

(KERETNA et al., 2013) apresenta um trabalho de detecção de *social bots* apenas buscando identificar a autoria dos textos propagados e comparando o conteúdo textual, descrito por estatísticas linguíticas dos textos, de modo a identificar os diversos estilos de escrita nos textos, fazendo uso de aprendizado supervisionado. Para tal utilizaram 1000 mensagens do *Twitter* de 30 contas distintas. Todavia, não informa os resultados obtidos nos experimentos.

Até onde foi possível verificar, poucos trabalhos capitanearam a utilização de técnicas com o aproveitamento do texto integral das mensagens propagadas pelas contas. O trabalho de Igawa et al. (2016) figura como uma exceção ao apresentar uma técnica que utiliza o conteúdo textual original dos *tweets*, transformando os textos em sinais, fazendo uso de *wavelets*. Estas atuam como descritores da distribuição dos termos-chave encontrados nos *tweets* gerados pelas contas relacionadas. Desta forma, cada conjunto contendo todos os textos propagados de cada usuário, figura como um documento relacionado àquele usuário. Deste modo, o modelo proposto exibe uma acurácia competitiva de 94% na classificação das contas automatizadas, apenas buscando por traços e assinaturas identificadas nos textos, sem utilizar outras fontes de informação. Uma observação adicional sobre Igawa et al. (2016) está no que tange à utilização de um *dataset* muito pequeno,

etc.(JURAFSKY; MARTIN, 2014)

contendo aproximadamente 100 contas.

Braz e Goldschmidt (2017) e Kudugunta e Ferrara (2018) são os primeiros trabalhos a apresentar técnicas baseadas em *deep learning* para a detecção de contas automatizadas. Kudugunta e Ferrara (2018) apresenta uma técnica que utiliza uma rede neural recorrente (LSTM - *Long Short-Term Memory*) em conjunto com redes convolucionais sobre o conteúdo textual e sobre as estatísticas relacionados aos mesmos. Os autores advogam que o modelo proposto atinge por volta de 99% de acurácia (AUC), figurando assim como o estado da arte.

	Host-side	Atributos Topológicos	Conteúdo Textual	Atributos Comportamentais	Estatísticas Textuais
(ALVISI et al., 2013)		X			
(BADRI SATYA et al., 2016)				X	
(BARBON et al., 2017)		X		X	
(BEUTEL et al., 2013)		X		X	
(BOSHMAF et al., 2015)		X		X	
(BRAZ; GOLDSCHMIDT, 2017)			X	X	
(CAO et al., 2012)		X			
(CHU et al., 2012)				X	
(DAVIS et al., 2016)		X		X	X
(GILANI et al., 2017)		X		X	X
(IGAWA et al., 2016)			X		
(HWANG et al., 2012)				X	
(KERETNA et al., 2013)					X
(KUDUGUNTA; FERRARA, 2018)			X		X
(TAN et al., 2012)	X				
(XIAO et al., 2015)				X	
(YANG et al., 2014)			X	X	X
(YANG et al., 2015)				X	
(WANG et al., 2013)				X	
(WANG et al., 2016)				X	

TAB. 3.1: Panorama Sumarizado - Trabalhos a partir de 2013

4 MÉTODOS PROPOSTOS

4.1 CONSIDERAÇÕES INICIAIS

O presente trabalho apresenta dois métodos voltados para a detecção de *bots* sociais a saber: DetBot Alfa e DetBot Beta. Desta forma, busca-se demonstrar que a utilização do texto original das mensagens pode viabilizar a criação de modelos aprimorados para esta tarefa frente aos modelos que utilizam apenas atributos comportamentais ou estatísticas acerca destas contas. Isto posto, o DetBot Alfa figura como o processo de classificação de contas, tal como foi concebido inicialmente primeira vez em Braz e Goldschmidt (2017). Este método preconiza a utilização de um classificador de mensagens para identificar traços de mensagens geradas por *bots* sociais. Para tal, este modelo é submetido a um processo de treinamento, de forma que amostras aleatórias de mensagens propagadas são aplicadas. A partir disso, tendo o conjunto de contas que confeccionaram as mensagens do passo anterior, outras amostras de mensagens destas contas (ou seja, mensagens nunca antes apresentadas ao modelo) são então aplicadas ao modelo treinado, para que cada conta receba um valor percentual relativo ao grau de suspeição depreendido pelo classificador, no que tange à autoria da mensagem ser atribuída a um *bot* social. Com esse dado gerado pelo classificador de *tweets*, a base de dados é enriquecida para então ser submetida a um modelo de classificação de contas. A descrição deste método (DetBot Alfa) é apresentada em detalhes na seção 4.3.

Ademais, visando ainda avaliar o potencial de generalização dos métodos propostos, o presente trabalho ainda apresenta o método DetBot Beta, método este que representa uma evolução em relação ao método anterior: DetBot Alfa. O DetBot Beta tem como objetivo apoiar e estender a busca por evidências experimentais de que a utilização dos textos originais das mensagens postadas pode contribuir na construção de modelos para detecção de *bots* sociais mais precisos do que aqueles que se baseiam exclusivamente nos atributos comportamentais dessas contas. Por outro lado, diferentemente do DetBot Alfa que seleciona amostras de contas e mensagens de forma aleatória, o DetBot Beta direciona o processo de seleção de forma a obter amostras representativas da rede completa, ou seja, mensagens de todo o *dataset* estão sendo consideradas. Para tanto, agrupa as mensagens por similaridade de conteúdo a fim de obter amostras que representem cada *cluster* identificado. Em seguida, de forma similar ao método DetBot Alfa, o DetBot Beta

treina um modelo para analisar o conteúdo original dessas mensagens a fim de identificar quais são suspeitas de terem sido produzidas por *bots* sociais. Por fim, assim como no DetBot Alfa, aplica um modelo que busca classificar cada conta v como *bot* ou não *bot* com base nos atributos comportamentais e no percentual de mensagens de v que tenham sido consideradas suspeitas pela etapa anterior. Maiores detalhes do método DetBot Beta na seção 4.4.

4.2 FORMALISMOS BÁSICOS

Desta forma, seja N uma rede social representada por um grafo dirigido $G(V, E)$, onde cada vértice $v \in V$ representa uma conta (ou usuário) de N e cada aresta $e \in E$ corresponde a um par ordenado entre duas contas u e v, representado por $e = (u, v)$, que indica que u segue v. Suponha que cada conta u possui um conjunto de mensagens $M_u = \{m_{u,1}, m_{u,2}, ..., m_{u,u_k}\}$ postadas por ela em N. O conjunto de todas as mensagens de N é representado por $M_N = \bigcup_{i=1}^{|V|} M_{u_i}$, onde $|V|$ corresponde à cardinalidade de V. Considere ainda que cada conta u possui uma lista ordenada de atributos $(u.a_1, u.a_2, ..., u.a_r, u.c)$, sendo $u.a_i$ comportamentais e $u.c$ um atributo binário que informa se u é *bot* ou não.

4.3 MÉTODO DETBOT ALFA

O método proposto possui quatro etapas conforme ilustrado na Figura 4.2. A descrição de cada uma delas encontra-se detalhada a seguir:

4.3.1 ETAPA 1 - SELEÇÃO DE MENSAGENS E CONTAS

A Seleção de Mensagens e Contas é o primeiro processo do método e é responsável por construir três conjuntos que serão utilizados pelas demais etapas: V', M'_{Treino} e M'_{Teste}. Inicialmente, tem-se $V' = \emptyset$, $M'_{Treino} = \emptyset$ e $M'_{Teste} = \emptyset$. A construção dos conjuntos é realizada da seguinte forma: executa-se uma seleção aleatória sem reposição de x_{treino} mensagens de M_N a partir de um conjunto de contas em V, também selecionado de forma randômica, tal que cada mensagem $m \in M_u$, e atualiza-se $V' = V' \cup \{u\}$. Por conseguinte, tais mensagens são armazenadas em M'_{Treino}. Após a formação de V' e M'_{Treino}, inicia-se a construção de M'_{Teste}. Para cada $u \in V'$, são selecionadas aleatoriamente x_{teste} mensagens $m \in M_N$ tais que $m \notin M'_{Treino}$. Tais mensagens são incluídas em M'_{Teste}. Desta forma, ao final do procedimento, tem-se que $M'_{Treino} \cap M'_{Teste} = \emptyset$. Cabe ressaltar que tanto

x_{treino} quanto x_{teste} são variáveis definidas pelo usuário do processo. Em seguida, toda mensagem $m \in M'_{Treino} \cup M'_{Teste}$ é enriquecida com a classificação da conta u responsável por sua propagação, i. e., $u.c$. Tal informação será utilizada na etapa 2.

Ou seja, um conjunto de mensagens é selecionado para compor o treinamento do modelo, e a partir destas são coletados os seus respectivos autores. Por conseguinte, destes autores são selecionados um determinado número de mensageens (igual para cada conta) para composição do conjunto de teste do modelo de classificação de mensagens. Portanto, o conjunto de teste é disjunto do conjunto de treino.

4.3.2 ETAPA 2 - CLASSIFICAÇÃO DE MENSAGENS

A Classificação de Mensagens é a segunda etapa e subdivide-se em dois passos. No primeiro, treina-se uma rede convolucional com o conjunto M'_{Treino}. Para tanto, a rede recebe como entrada a codificação sem perda do texto bruto de cada mensagem m, sendo a classificação da mensagem a saída desejada $u.c$ a ser aprendida pela rede. Uma vez concluído o treinamento da rede neural, o segundo passo consiste em aplicar o modelo aprendido pela rede a cada uma das mensagens contidas em M'_{Teste}, contabilizando, ao final, para cada conta u, o percentual de mensagens classificadas como sendo suspeitas de terem sido propagadas por *bots* sociais.

É importante ressaltar que a formatação dos textos nesta etapa representa um ponto relevante do presente trabalho. Para tal, a formatação do conteúdo textual das mensagens utiliza o que preconica Zhang et al. (2015). Dito isto, esta técnica permite transformar uma mensagem em uma matriz binária de *strings*, de acordo com um mapa de caracteres. A figura 4.1 ilustra como é realizada esta formatação. Logo, cada linha da matriz mapeia a ocorrência de um, e somente um determinado caractere, dispondo na respectiva coluna da matriz os valores '1' em caso de ocorrência do caracatere e '0' para demais casos, de modo que cada linha da matriz representa apenas um caractere de um texto. Portanto, um texto contendo 200 caracteres demanda uma matriz de 200 linhas, contendo um número de colunas exatamente igual ao número de caracteres que podem ser mapeados, dependendo do caso. Deste modo, textos menores que um determinado limite fornecido, são preenchidos com linhas contendo apenas '0' até alcançar o limite.

Portanto, esta técnica é capaz de converter um dado texto em uma matriz de *bits*, tal como a representação matricial de uma imagem qualquer. Deste modo, tem-se uma

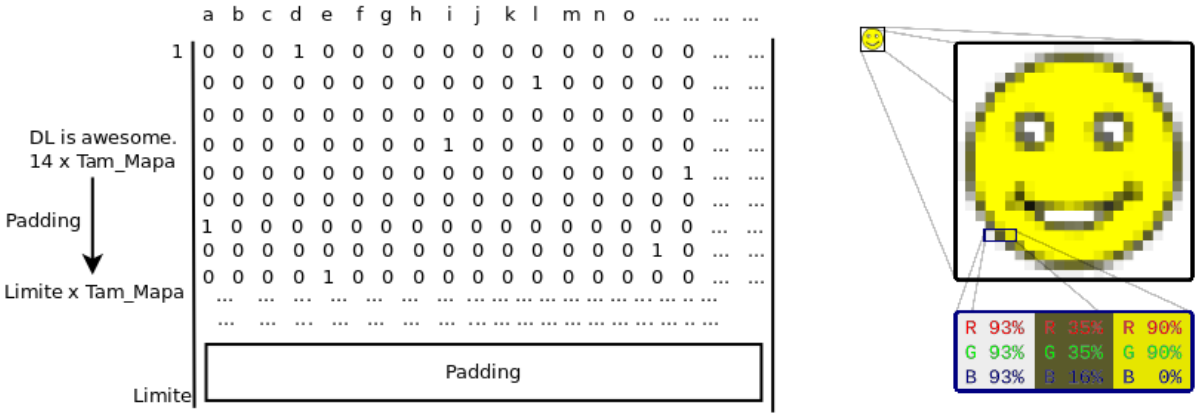

FIG. 4.1: Ilustração da técnica de codificação aplicada (ZHANG et al., 2015).

relação análoga entre *pixels* e caracteres no que tange ao padrão de entrada para os diversos modelos de aprendizado (ZHANG et al., 2015).

4.3.3 ETAPA 3 - ENRIQUECIMENTO DE V'

Enriquecimento de V' é a etapa responsável por acrescentar um novo dado a cada uma das contas $u \in V'$. Assim, a lista ordenada de atributos que descreve u passa a ser $(u.a_1, u.a_2, ..., u.a_r, u.c, u.susp)$, onde $u.susp$ contém o percentual de mensagens de u classificadas como suspeitas na etapa anterior.

Desta maneira, cada conta selecionada terá o percentual de mensagens depreendidas como suspeitas pelo modelo, enriquecendo desta forma o conjunto de dados sobre estas contas.

4.3.4 ETAPA 4 - CLASSIFICAÇÃO DE CONTAS

A Classificação de Contas inicia a partir de um dado algoritmo de classificação S, deste modo esta etapa realiza um processo de validação cruzada com k conjuntos sobre V'. Este processo consiste em dividir V' em k conjuntos de contas para, em seguida, realizar k iterações. Em cada iteração, um dos k conjuntos é utilizado como conjunto de teste e os $k-1$ restantes utilizados para treinamento de S. O desempenho do modelo de classificação construído a cada iteração é armazenado. O processo se repete até que todos os k conjuntos tenham sido utilizados uma vez como conjunto de teste. Ao final do processo, é obtido o desempenho médio dos k modelos gerados. Em todas as iterações do processo, os atributos $a_1, a_2, ..., a_r, susp$ foram entradas e o atributo c a saída de S.

Deste modo, a classificação de contas é processada utilizando a técnica da validação

cruzada *K-Fold* para cada uma das contas como sendo do tipo *bot* ou não *bot*.

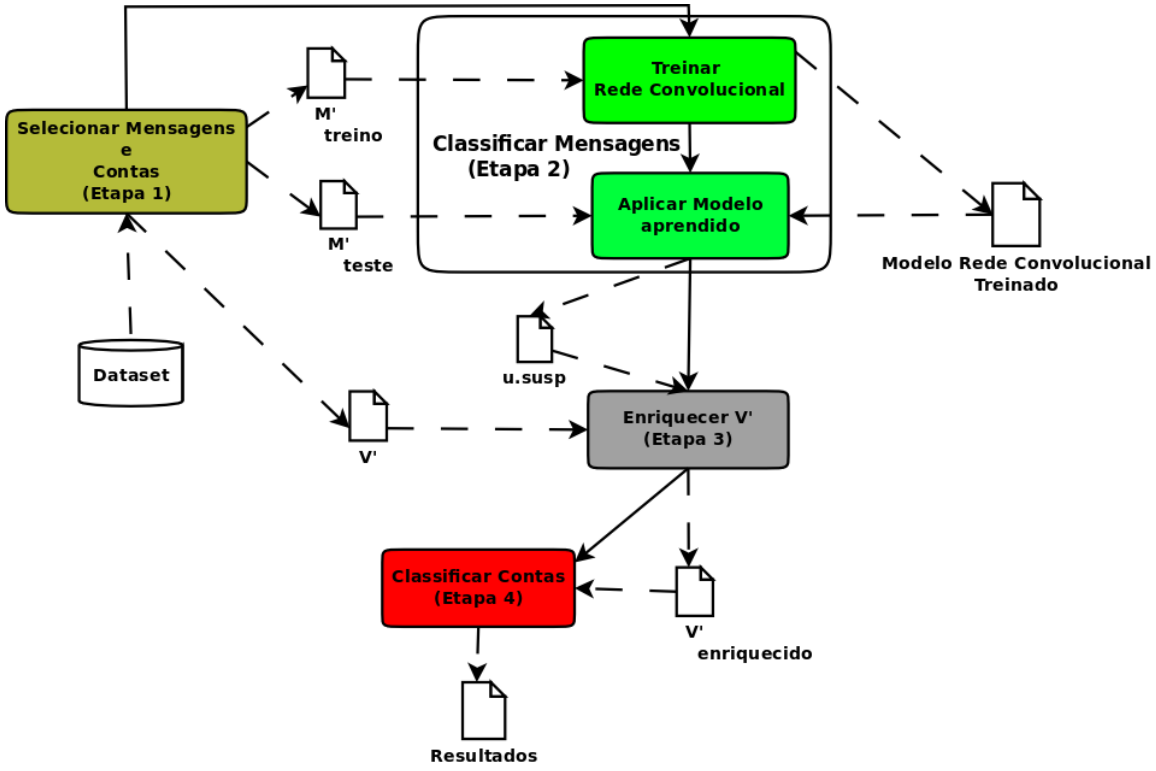

FIG. 4.2: Visão Geral das Etapas do Método DetBot Alfa.

4.4 MÉTODO DETBOT BETA

O método DetBot Beta possui cinco etapas conforme ilustrado na Figura 4.3. As etapas 3, 4 e 5 são análogas, respectivamente, às etapas 2, 3 e 4 do método Det Bot Alfa descrito na seção 4.3.

4.4.1 ETAPA 1 - CLUSTERIZAÇÃO DE MENSAGENS

A Clusterização de Mensagens é responsável por construir subconjuntos de mensagens que serão utilizados pela etapa de seleção de amostras. Inicialmente, considera-se $V' = \emptyset$, $V'_{Temp} = \emptyset$, $M'_{Treino} = \emptyset$ e $M'_{Teste} = \emptyset$. A construção dos subconjuntos é realizada da seguinte forma: executa-se uma *clusterização* do conjunto completo de mensagens M_N, o que contempla todo o conteúdo textual da rede N. Como saída, as mensagens acabam particionadas em k *clusters* disjuntos, representados por $C_1, C_2, ..., C_k$, onde k é a quantidade de conjuntos gerados.

4.4.2 ETAPA 2 - SELEÇÃO DE MENSAGENS

A Seleção de Mensagens segue da seguinte maneira: para cada *cluster* C_i são selecionadas aleatoriamente p mensagens de C_i e armazenadas em M_i''. Após a seleção das $p \times k$ mensagens, constroi-se M'_{Treino}, da seguinte maneira: $M'_{Treino} = \bigcup_{i=1}^{k} M_k''$. Para cada mensagem $m \in M'_{Treino}$, recupera-se a conta u tal que $m \in M_u$ e atualiza-se $V'_{Temp} = V'_{Temp} \cup \{u\}$. Após a construção de V'_{Temp} e M'_{Treino}, V' é formado por $|V'_{Temp}|$ contas u_i selecionadas aleatoriamente, de tal forma que $u_i \notin V'_{Temp}$. Por fim, para cada $u_i \in V'$, são selecionadas aleatoriamente p'' amostras de M_{u_i}. Em seguida tais amostras são incluídas em M'_{Teste}. Assim sendo, ao final do procedimento, tem-se que $M'_{Treino} \cap M'_{Teste} = \emptyset$. Em seguida, toda mensagem $m \in M'_{Treino} \cup M'_{Teste}$ é enriquecida com a classificação da conta u responsável por sua propagação, i. e., *u.c.* Tal informação é utilizada na etapa 3.

Ou seja, como ilustração, sejam $p = 10$ e $k = 5$ os valores iniciais. Desta forma, têm-se 10 mensagens para cada um dos 5 clusters, perfazendo deste modo 50 mensagens para treinamento. A partir disso, sejam $p'' = 5$ e $V'_{Temp} = 30$. Logo, tem-se $|M'_{Teste}| = 150$ mensagens.

4.4.3 ETAPA 3 - CLASSIFICAÇÃO DE MENSAGENS

A Classificação de Mensagens, terceira etapa do processo é idêntica à classificação de mensagens do método anterior, apresentado na seção 4.3.2, e tal como este subdivide-se em dois passos: o primeiro consiste no treinamento da rede neural convolucional e o segundo é a aplicação deste modelo treinado para teste. A seção 4.3.2 figura como referência.

4.4.4 ETAPA 4 - ENRIQUECIMENTO DE V'

Denominada Enriquecimento de V', a etapa de número 4 do processo é responsável por acrescentar uma novo dados a cada uma das contas e é idêntica à etapa 3 da seção 4.3.

4.4.5 ETAPA 5 - CLASSIFICAÇÃO DE CONTAS

Dado um algoritmo de classificação S, a quinta e última etapa, Classificação de Contas, realiza um processo de validação cruzada com q conjuntos sobre V'. Este processo consiste em dividir V' em q conjuntos de contas tal como é realizado na etapa 4 da seção 4.3.

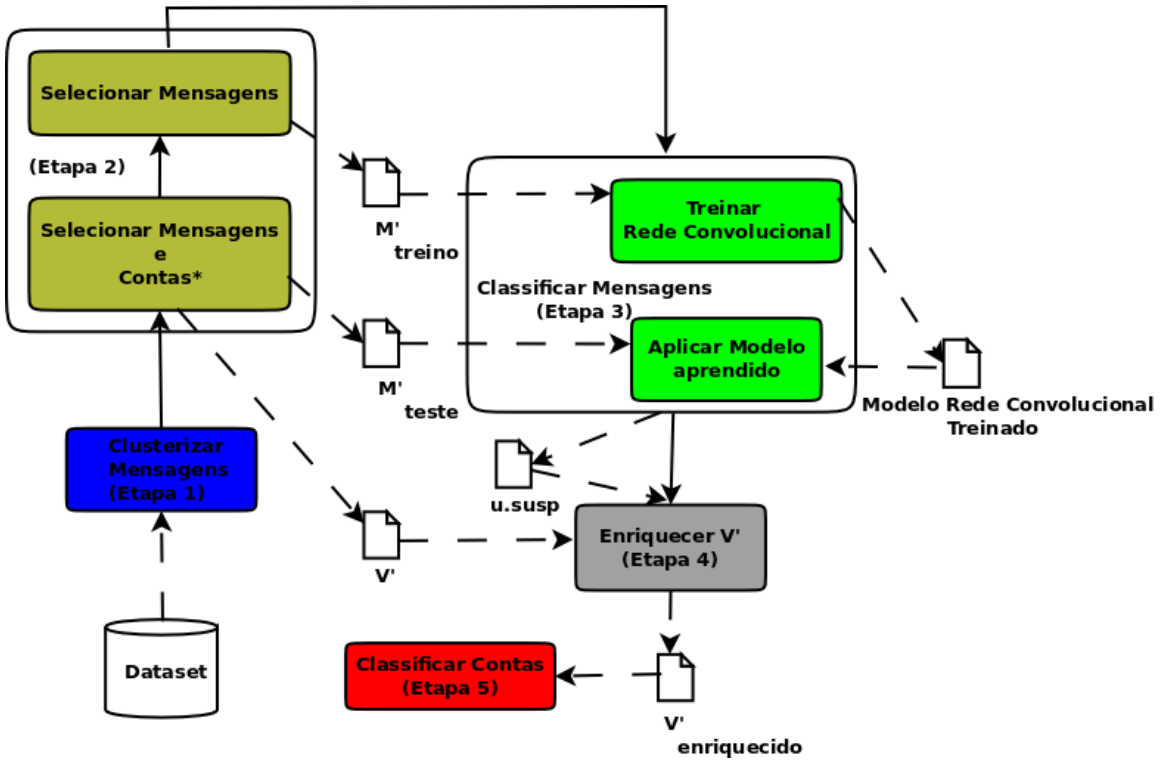

FIG. 4.3: Visão Geral das Etapas do Método DetBot Beta.

5 EXPERIMENTOS E RESULTADOS

5.1 CONSIDERAÇÕES GERAIS

Para a exposição da técnica proposta, a fim de provar a hipótese do presente trabalho, buscou-se criar um modelo estatístico para rotular as contas automatizadas. Para tal, criou-se um classificador de *spam* para fornecer uma nova fonte de informação para propiciar a criação de rotuladores mais precisos. Neste expediente utilizou-se uma rede neural convolucional (GOODFELLOW et al., 2016) para figurar como o classificador de mensagens geradas pelos usuários contidos no dataset apresentado em Lee et al. (2011).

O dataset utilizado (LEE et al., 2011) é composto por dados de contas (e suas respectivas mensagens) do *Twitter* relativos ao período de 30 de dezemdo de 2009 a 2 de agosto de 2010. O mesmo possui as seguintes características principais:

a) 22.2K contas *bots* e 19.2K contas de humanos legítimos e seus respectivos atributos comportamentais;

b) 2.3M de tweets gerados pelas referidas contas *bot* e 3.2M de tweets gerados por contas dos usuários legítimos relacionados;

c) Os tweets possuem ao menos 7 idiomas distintos: inglês, francês, alemão, mandarim, japonês, italiano e espanhol;

Primeiramente, foi necessário criar um processo para identificação de idiomas[27], de modo a manter no dataset para os processos de aprendizado apenas mensagens em um mesmo idioma: inglês. Para tal, utilizou-se um *toolkit* em Python contendo um dicionário[28] de palavras para o inglês contendo tanto termos tradicionais como termos voltados para a Web. Desta forma, após a aplicação desta filtragem o dataset passou a conter 3.71M de mensagens: 1.93M geradas por contas *bot* e 1.78M de mensagens criadas por usuários legítimos. Ou seja, o filtro foi capaz de retirar 33,7% do total de mensagens, estas criadas em idiomas diferentes do Inglês. Análises sobre o conjunto de mensagens filtrado comprovaram a eficácia do filtro aplicado.

[27]Para tal, criou-se um script em Python para filtrar mensagens, mantendo apenas as escritas em inglês
[28]NLTK - https://www.nltk.org/

Isto posto, faz-se necessário ressaltar que a técnica *char quantization* (ZHANG et al., 2015) apresentou resultados muito promissores, comprovando a boa adequação de modelos baseados em redes convolucionais para analisar e depreender similaridades textuais, desta forma, colocando os caracteres para os textos tal como os *pixels*[29] para as imagens, buscando assim identificar na análise dos textos os resultados expressivos obtidos por estes modelos no tratamento e análise de imagens (KRIZHEVSKY et al., 2012). Portanto, por conta destes resultados apresentados em Zhang et al. (2015), a técnica *char quantization* foi avaliada em experimentos preliminares em Braz e Goldschmidt (2017). Deste modo, por conseguinte, por conta da adequação da técnica comprovada através de experimentos em Zhang et al. (2015) e pelos resultados preliminares apresentados em Braz e Goldschmidt (2017), o presente trabalho seguiu na utilização deste modelo de aprendizado profundo para a tarefa de classificação de mensagens, a despeito da utilização pouco expressiva no que tange à utilização de modelos CNN para análise de conteúdo textual (ZHANG et al., 2015).

Não obstante, para avaliar e realizar os processos de treinamento, foi planejado incialmente utilizar o dataset (LEE et al., 2011) em sua íntegra, separando parte para treino e parte para validação (seja por *holdout* ou por validação cruzada), porém problemas relativos ao tempo excessivamente longo para treinamento e necessidade de recurso computacional mais robusto impossibilitaram este tipo de tratamento. Com o recurso computacional disponível para o presente trabalho (2 núcleos a 2GHZ com 64GB de memória principal rodando um Ubuntu 16.04 LTS - kernel: 4.13.0-43), não foi possível treinar o modelo CNN utilizando mais do que 50K mensagens, o que representa uma fração de apenas 0,8 % do contingente total de mensagens contidas no *dataset*.

Desta maneira, o método DetBot Beta representa um aprimoramento em relação ao método anterior chamado DetBot Alfa, pois em vez de buscar amostras raondômicas de mensagens para treinar o modelo CNN, o método DetBot Beta utiliza amostras representativas de toda a base de dados ao empregar amostras de agrupamentos de mensagens depreendidas através de um clusterização de toda a base de mensagens do *dataset*. Foram utilizados diversos algoritmos de clusterização na busca de coletar amostras mais representativas acerca da base. Maiores detalhes seguem na seção 4.4.

[29]Menor elemento em uma tela ou dispositivo de exibição - https://pt.wikipedia.org/wiki/Pixel

5.2 EXPERIMENTOS COM MÉTODO DETBOT ALFA

A fim de avaliar o método proposto, foi realizado um experimento inicial com o mesmo *dataset* adotado em (LEE et al., 2011). Trata-se de uma base de dados do *Twitter* com informações referentes ao período de dezembro de 2009 a agosto de 2010[30]. A base contém os atributos comportamentais indicados na Tabela 5.1, além dos textos de todas as mensagens publicadas no período. No total, o *dataset* contém aproximadamente 40 mil contas e 5,5 milhões postagens, sendo 87 postagens por conta, em média, aproximadamente. A escolha deste *dataset* deveu-se basicamente à disponibilidade dos atributos comportamentais em conjunto com os textos publicados pelas respectivas contas. Contudo, conforme informado na seção 5.1, foi aplicado um filtro para empreender o treino do modelo CNN apenas com *tweets* escritos em Inglês. Com isso o volume total de mensagens caiu de 5.5M para 3.71M.

TAB. 5.1: Atributos comportamentais disponíveis no *dataset* do experimento.

Atributos	**Descrição**
# usuários seguindo	número de usuários que a conta segue
#tweets	número de tweets postados pela conta
razão de #seguindo por #seguidores	razão de #usuários seguidos por #seguidores
#usuários seguidores	número de seguidores

Ainda assim, diante do elevado volume de contas e mensagens disponíveis no *dataset* completo, o que demanda grande poder computacional e elevado tempo para o treinamento de modelos de aprendizado de máquina, optou-se por utilizar subconjuntos menores na realização do treinamento do modelo para a geração dos resultados preliminares. Desta forma, foram avaliados quatro cenários: no primeiro foram utilizadas 2000 mensagens publicadas, no segundo 6000 mensagens, 20000 mensagens no terceiro e, por fim, 100000 mensagens para o treinamento e teste do classificador de mensagens. Outrossim, a seleção de mensagens e contas de todos os cenários seguiu o procedimento descrito na etapa 1 do método proposto na seção 4.3. Assim sendo, a Tabela 5.2 apresenta um breve detalhamento estatístico sobre os cenários do experimento[31].

A etapa de classificação de mensagens foi implementada em todos os cenários com uma rede neural convolucional cuja configuração e arquitetura estão descritas na Tabela 5.3. Importante notar que o otimizador utilizado foi o *ADAM* (KINGMA; BA, 2014). A

[30]Neste *dataset*, cada conta contém um *label* que indica se a conta é falsa (*bot*) ou legítima.

[31]É válido ressaltar que foi adotada, em ambos os cenários, a mesma distribuição de classes encontradas no *dataset* original, em relação ao número de contas: 50% de contas do tipo *bot* e 50% de contas legítimas.

TAB. 5.2: Dados estatísticos sobre os cenários do experimento

	x_{treino}	x_{teste}	$x_{treino} + x_{teste}$	$\|V'\|$	Média Msg/Conta	Contas *bot*
Cenário 1	1000	1000	2000	53	37.7	50%
Cenário 2	3000	3000	6000	160	37.5	50%
Cenário 3	10000	10000	20000	533	37.5	50%
Cenário 4	50000	50000	100000	2659	37.6	50%

implementação deste modelo foi feita utilizando *Python* e *Keras* (CHOLLET et al., 2015).

TAB. 5.3: Arquitetura da rede convolucional

	Filtros	Kernel	Função de Ativ.	Stride
2 Camadas	150	7X7	RELU	1
4 Camadas	150	3X3	RELU	1
Fully Connected	31 neurônios	–	RELU	–
Última camada	2 neurônios	–	SOFTMAX	–

As mensagens precisaram ser formatadas a fim de serem submetidas à rede na etapa 2. Para esta formatação foi utilizada a técnica *Char Quantization* proposta por (ZHANG et al., 2015)[32]. Desta forma, cada texto de mensagem publicada foi transformado em uma matriz binária de dimensão 150x64, uma vez que 150 é o limite máximo de caracteres para cada postagem no Twitter e 64 é a quantidade de caracteres considerados para mapeamento. A Figura 5.1 apresenta o conjunto de caracteres considerados para fins de mapeamento. Assim, as matrizes geradas foram esparsas. Cada linha de uma dada matriz correspondeu a um e, somente um, caracter da mensagem associada e, portanto, teve o *bit* setado para 1 exatamente na coluna relativa ao caracter identificado e 0 para as demais colunas da linha em questão.

```
['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v',
 'w', 'x', 'y', 'z', '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', '-', ',', ';', '.', '!', '?', ':', '/',
 '\',',_', '@', '#', '$', '%', '^', '&', '*', '+', '-', '=', '<', '>', '(', ')', '[', ']', '{', '}']
```

FIG. 5.1: Conjunto de caracteres usado para mapeamento das mensagens.

Ainda na etapa de Classificação de Mensagens foi executado o procedimento de *hold-dout* (com 2/3 de M'_{Treino} para treino e 1/3 de M'_{Treino} para validação), para cada um dos cenários. Deste modo, com o classificador treinado utilizando M'_{Treino}, fez-se a apresentação de M'_{Teste} para o modelo efetuar a classificação de cada mensagem m e, por conseguinte, viabilizar o cálculo do percentual de mensagens classificadas como sendo

[32] A escolha por este tipo de formatação deveu-se basicamente pelo bom desempenho proporcionado por ela nos experimentos de classificação de texto relatados pelos autores.

geradas por *bots* para todas as contas da amostra (*u.susp*). Deste modo, com as taxas geradas, pode-se enriquecer V', finalizando assim a execução da Etapa 3.

Para classificação das contas foram utilizados dois algoritmos de classificação: *Random Forest* (HO, 1995) e MLP (*Multilayer Perceptron*) (ROSENBLATT, 1961). Em cada cenário, cada algoritmo foi submetido a um processo de validação cruzada com 10 conjuntos. Os algoritmos executados foram instanciados com suas configurações *default*, utilizando Weka (WITTEN et al., 2016). Importante notar que foi adotada a mesma parametrização dos algoritmos durante todo o processo. A Tabela 5.4 apresenta os parâmetros adotados por cada algoritmo.

TAB. 5.4: Configuração utilizada para cada algoritmo de classificação

Algoritmo	**Configuração**
MLP	hidden layers= 4, learning rate=0.3, momentum=0.2, epochs=500
Random Forest	bag size percent = 100, batch size = 100, num iterations = 100

A fim de comparar a influência dos textos brutos no processo de detecção de *bots* sociais, os mesmos algoritmos de classificação utilizados na etapa 4 foram aplicados sobre o conjunto V' sem o enriquecimento realizado pelo método proposto. Também neste caso, foi realizado um processo de validação cruzada com 10 conjuntos.

A Tabela 5.5 mostra as acurácias obtidas pelos dois algoritmos na classificação de contas em duas situações: uma com V' contendo apenas os atributos comportamentais da Tabela 5.1 e a outra com V' enriquecido pelo método proposto para conter também o percentual de mensagens suspeitas associado a cada conta. Pode-se perceber que, de uma forma geral, os algoritmos de classificação aplicados em V' enriquecido apresentaram desempenho superior aos mesmos algoritmos aplicados em V' sem enriquecimento, em ambos os cenários. Os resultados obtidos são, portanto, evidências experimentais de que a utilização de textos brutos de mensagens de contas de redes sociais pode contribuir para produzir modelos de detecção de *bots* sociais mais precisos do que aqueles que se baseiam somente nos atributos comportamentais dessas contas. Ademais, aplicando os algoritmos de seleção de atributos RELIEF (KONONENKO, 1994) e CorrelationAttributeEval (WITTEN et al., 2016) (utiliza o coeficiente de Correlação de Pearson), verificou-se que o atributo criado pelo método proposto ficou em primeiro lugar no que se refere ao ganho de informação para a tarefa de detecção de *bots* em ambos os cenários. Este fato reforça a importância de se investigar alternativas de utilização do conteúdo textual original das mensagens na construção de modelos de detecção de *bots* sociais.

TAB. 5.5: Resultados do experimento: acurácia dos classificadores

Cenário	**Enriquecimento de V'**	**MLP**			**Random Forest**		
		Acc.	FPs	FNs	Acc.	FPs	FNs
1	Não	69.23%	11%	19%	**86.5**%	7%	5%
1	**DetBot Alfa**	84.61%	7%	7%	**90.3**%	5%	3%
2	Não	80.5%	13%	6%	**89.3**%	4%	6%
2	**DetBot Alfa**	81.76%	8%	9%	**90.5**%	3%	5%
3	Não	81.8%	12%	5%	**90.3**%	4%	5%
3	**DetBot Alfa**	81.6%	12%	6%	**90.1**%	4%	5%
4	Não	85.5%	7%	7%	**90.3**%	5%	3%
4	**DetBot Alfa**	86.76%	7%	5%	**92.5**%	3%	4%

5.3 EXPERIMENTOS COM MÉTODO DETBOT BETA

O método DetBot Beta foi aplicado em 3 cenários. A tabela 5.6 apresenta detalhes estatísticos sobre a composição dos 3 cenários[33].

A seleção de mensagens e contas dos cenários 1 a 3 seguiu o procedimento detalhado na seção 4.4. No entanto, antes de iniciar o processo de clusterização das mensagens disponíveis no *dataset*, foi necessário realizar algumas operações de pré-processamento dos textos. Operações estas também aplicadas na aplicação do método DetBot Alfa da seção anterior.

Desta forma, faz-se necessário salientar que o referido *dataset* contém *tweets* em diversos idiomas e uma mesma conta pode ter postado mensagens em mais de um idioma. A primeira operação realizada foi uma filtragem das mensagens em idioma distinto da língua inglesa. Com isso, das cerca de 5.5 milhões de mensagens inicialmente existentes, restaram 3.6 milhões. Para seleção e processamento destas mensagens em inglês, utilizou-se a biblioteca NLTK (BIRD et al., 2009).

TAB. 5.6: Dados estatísticos sobre os cenários do experimento - **DetBot Beta**

	x_{treino}	x_{teste}	$x_{treino} + x_{teste}$	Média Msg/Conta (Teste)	Contas *bot*
1	1000	1000	2000	100	50%
2	3000	3000	6000	200	50%
3	10000	10000	20000	200	50%

Logo, no tocante à clusterização das mensagens, do mesmo modo foi necessário representar cada um dos *tweets* como um vetor de palavras. A representação utilizada para

[33]É importante enfatizar que foi adotada a mesma distribuição de classes encontradas no *dataset* original, em relação ao número de contas: 50% de contas do tipo *bot* e 50% de contas legítimas, em todos os cenários.

tal foi *Bag of Words* (LIDDY, 2001).

Diante disso, os algoritmos de clusterização utilizados foram: *Affinity Propagation* (FREY; DUECK, 2007), *DB-Scan* (ESTER et al., 1996) e k-Means (JAIN; DUBES, 1988) empregando a distância euclidiana como métrica. Utilizando uma máquina contendo 2 núcleos a 2.4Ghz e 64GB de memória principal, com um Linux Ubuntu 16.04 LTS como sistema operacional hospedeiro, apenas o *K-Means* conseguiu de fato empreender a clusterização. Os demais algoritmos pararam de executar o processo após várias horas por falta de recursos computacionais, sobretudo no que tange à utilização de memória principal, bastante demandada por conta do grande contingente de mensagens na base. Maiores detalhes acerca do comportamento destes algoritmos implementados, quando apresentados a uma grande massa de dados podem ser obtidos em McInnes et al. (2016). A tabela 5.7 apresenta os tempos relativos de execução de cada algoritmo.

Para escolha do número de *clusters* k do *K-Means*, foram realizados testes onde o valor de k variou de 4 a 10. Para cada valor, foi calculado o coeficiente de *Jaccard* (SALTON; MCGILL, 1986) entre os conjuntos de palavras associados aos *clusters* (dois a dois) a fim de se verificar o grau de interseção entre eles. Deste modo, o valor selecionado ($k = 6$) foi o que apresentou menor índice de *Jaccard* total entre todas as demais configurações. Por conseguinte, a tabela 5.8 mostra a distribuição dos *tweets* entre os clusters, evidenciando uma concentração deveras acentuada no *cluster* 3, frente aos demais *clusters*. Importante notar que esse comportamento foi verificado para todos os valores utilizados de K nos experimentos.

TAB. 5.7: Tempos de execução dos algoritmos de clusterização

	Tempo	Concluído?
Affinity Propagation	16h	Não
DB-Scan	15h	Não
K-Means	9,5h	Sim

Ademais, as figuras 5.2, 5.3, 5.4, 5.5, 5.6 e 5.7 exibem *wordclouds* relativas a cada um dos clusters identificados pelo *K-Means*, para o valor de $K = 6$, utilizado para dividir os *tweets*. Essas imagens evidenciam que os *tweets* e seus respectivos termos estão distribuídos entre os agrupamentos elencados. No *cluster* 5 cujos termos mais relevantes estão dispostos na imagem 5.7, com destaque para termos tais como: ver ou olhar (do inglês, look), *sound*, *"lol"* (termo utilizado para expressar risadas ou grande contentamento na *WEB*), *say*, *friend*, etc. Já o *cluster* 4, ilustrado na figura 5.6, destaca palavras voltadas para mídias digitais como: *Video Game*, *You Tube*, *chat*, etc. Para o *cluster* 3 ilustrado

pela imagem 5.5 os termos em destaque são: "OW", *Thank*, *make money*, entre outros. O *cluster* 2 cujo *wordcloud* está apresentado na imagem 5.4 destaca palavras do inglês como: *time*, *lol*, *really*, *real time*. O *cluster* 1 representado pelo na imagem 5.3 destaca vocábulos como: New York, New Moon, new blog, etc. Por fim, o *cluster* 0 representado pela imagem 5.2 destaca termos como: *just got*, *just*, *lol*, etc.

TAB. 5.8: Distribuição dos Tweets por Cluster

Cluster	Qtde de Tweets
0	154154
1	161158
2	87305
3	3130980
4	66731
5	116194

Após a clusterização, variou-se a quantidade de mensagens selecionadas de cada *cluster*, levando aos três cenários indicados. Assim, no primeiro cenário foram utilizadas 2.000 mensagens publicadas, no segundo, 6.000 mensagens e no terceiros, 20.000 mensagens para o treinamento e teste do classificador dos *tweets*. Diante disso, a tabela 5.6 apresenta os cenários utilizados nos experimentos do método DetBot Beta.

TAB. 5.9: Resultados: acurácia dos classificadores

Cenário	**Enriquecimento de V'**	**MLP**			**Random Forest**		
		Acc.	FPs	FNs	Acc.	FPs	FNs
1	Não	72.35%	10%	17%	**87.4**%	6%	5%
1	Estatísticas Textuais	79,67%	8%	12%	**90.8**%	5%	5%
1	**DetBot Beta**	81.15%	7%	8%	**91.5**%	5%	3%
2	Não	82.7%	11%	6%	**89.8**%	4%	5%
2	Estatísticas Textuais	83.69%	10%	6%	**90.4**%	3%	5%
2	**DetBot Beta**	83.74%	8%	8%	**90.9**%	3%	5%
3	Não	82.1%	12%	5%	**91.7**%	3%	5%
3	Estatísticas Textuais	82.5%	12%	5%	**92.0**%	4%	4%
3	**DetBot Beta**	82.5%	12%	5%	**92.1**%	3%	4%

A Tabela 5.9 mostra as acurácias obtidas pelos dois algoritmos na classificação de contas em três situações: uma com V' contendo apenas os atributos comportamentais da Tabela 5.1, outra com V' enriquecido com estatísticas do texto: quantidade de URL's na mensagem e quantidade de menções a terceiros, e por fim a última com V' enriquecido apenas com atributo gerado a partir do método proposto, para conter também o percentual de mensagens suspeitas associado a cada conta. Pode-se perceber que, de uma

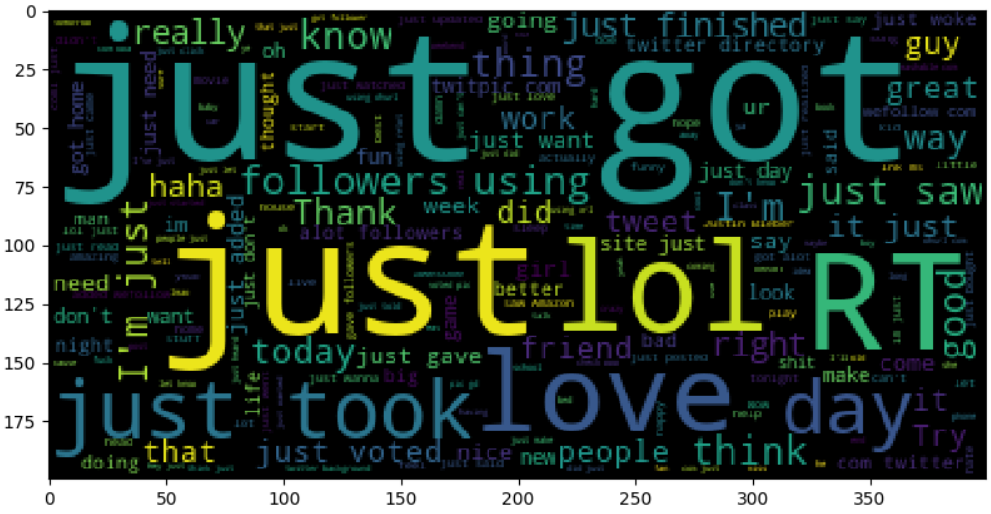

FIG. 5.2: Termos mais relevantes - Cluster 0

forma geral, os algoritmos de classificação aplicados em V' enriquecido com o atributo gerado pelo método DetBot Beta apresentaram desempenho superior aos mesmos algoritmos aplicados em V' sem enriquecimento e com utilização de estatísticas textuais, em todos os cenários, tal como foi evidenciado na seção 5.2.

Ademais, é possível verificar um aprimoramento no que tange aos resultados do método DetBot Beta em relação ao DetBot Alfa, isso comparando os cenários contendo a mesma quantidade de mensagens para ambos os métodos, tais como cenários 1, 2 e 3. Desta forma fica evidenciada a melhoria na qualidade das amostras coletadas pelo método DetBot Beta. Além disso, é possível verificar um aprimoramento das acurácias para ambos os métodos quando um número maior de mensagens é aplicado para o treinamento utilizando ambos os métodos. Isso confirma o que preconiza Goodfellow et al. (2016) no que tange à quantidade informação e a respectiva evolução dos modelos.

Isto posto, estes resultados figuram como evidências acerca da utilização do conteúdo textual das mensagens para a confecção de modelos aprimorados para a detecção de contas controladas pelos *bots* sociais, mesmo utilizando uma pequena fração de informação para o treinamento destes classificadores. Portanto, a utilização de uma quantidade maior de

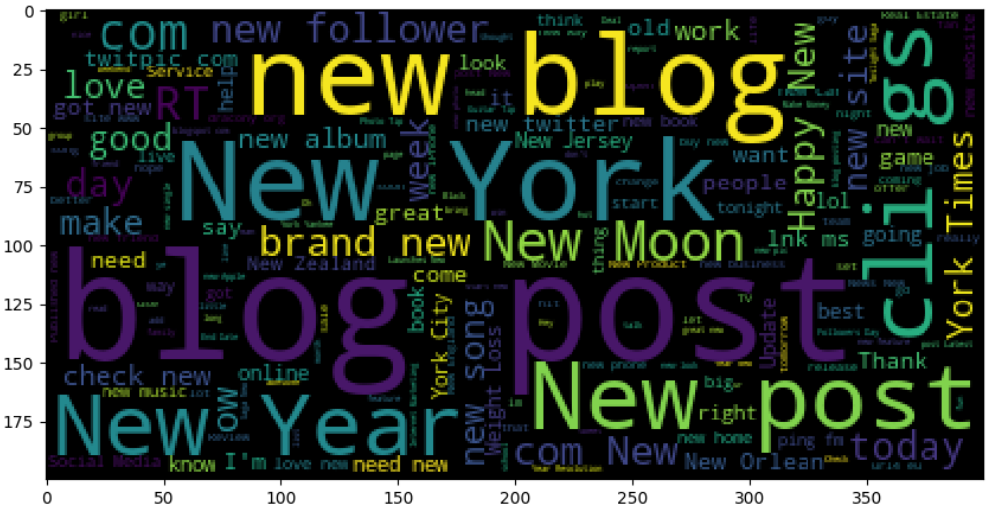

FIG. 5.3: Termos mais relevantes - Cluster 1

dados pode ainda tornar mais evidente o que está aqui exposto nestes resultados. Traços importantes das mensagens acerca do conteúdo e da forma são depreendidos pelo modelo de aprendizado de forma mais eficiente, propiciando assim sistemas de detecção mais eficientes para mitigar a presença e efeitos destas entidades indesejadas, na maior parte do tempo ou contexto.

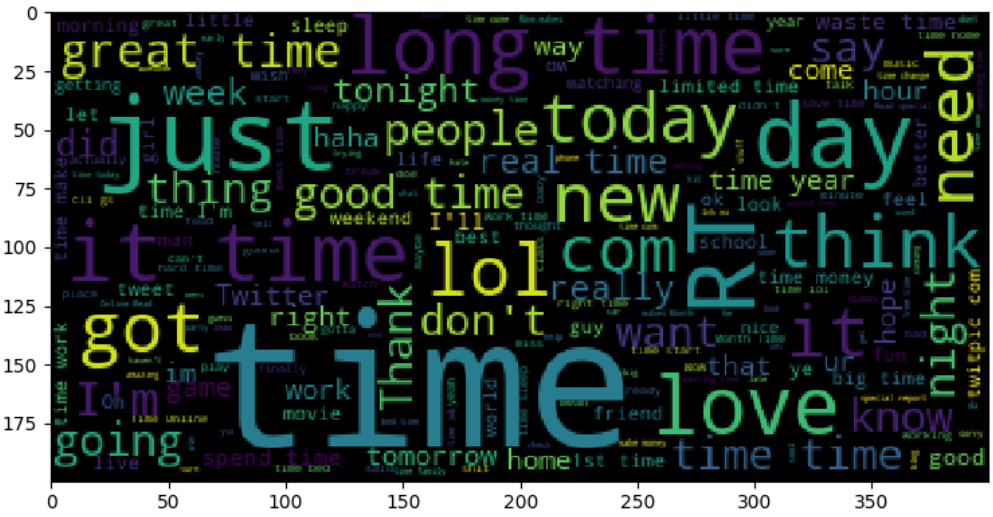

FIG. 5.4: Termos mais relevantes - Cluster 2

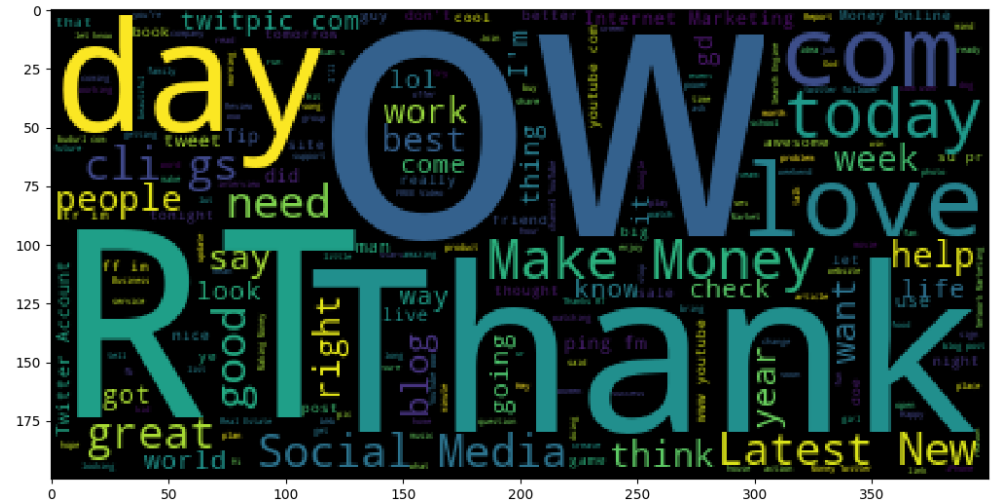

FIG. 5.5: Termos mais relevantes - Cluster 3

FIG. 5.6: Termos mais relevantes - Cluster 4

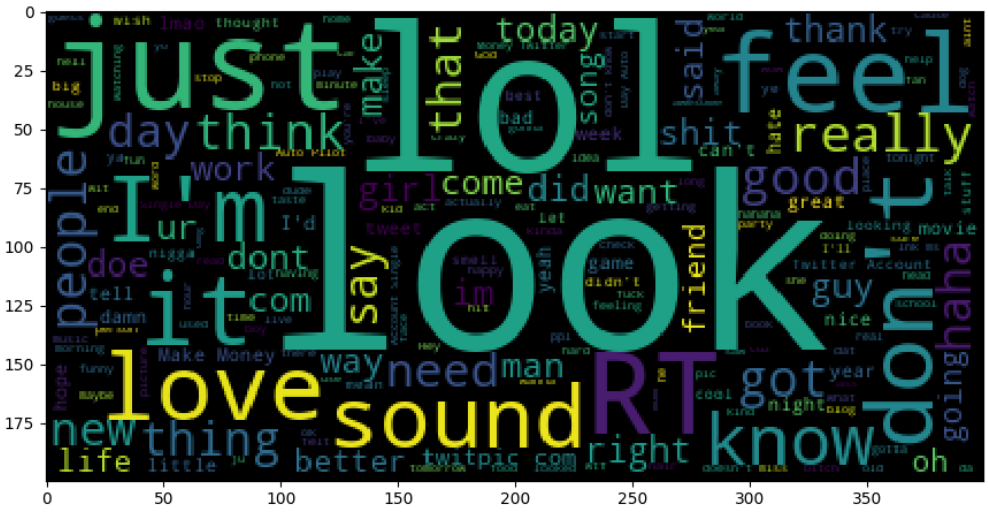

FIG. 5.7: Termos mais relevantes - Cluster 5

6 CONSIDERAÇÕES FINAIS

As redes sociais estão cada vez mais vulneráveis a ações dos *bots* sociais, contas automatizadas capazes de interagir com outros usuários e reproduzir o comportamento de uma conta legítima, sobretudo ao que tange à confecção de conteúdo de forma autônoma. Tais entidades são empregadas para manipulação de opiniões de usuários das redes sociais por meio de atividades maliciosas como disseminação de notícias falsas, adulteração de estatísticas de percepção pública, furto de contas, etc.

Deste modo, a detecção de *bots* sociais visa mitigar os efeitos destas entidades, buscando identificar suas respectivas atividades e eliminar a presença destas contas automatizadas. Neste expediente, após avaliar os principais trabalhos relacionados na detecção de *bots* sociais, foi possível verificar que a grande maioria dos trabalhos utiliza técnicas sob a perspectiva das abordagens comportamentais ou híbridas. Todavia, poucos trabalhos utilizam de fato o texto bruto original dos textos gerados pelas contas, com o intuito de identificar estes *bots*.

Isto posto, a extração de estatísticas do conteúdo textual das mensagens postadas norteia a grande maioria dos trabalhos. Não obstante, visto que esta extração pode incorrer na perda de informação, o presente trabalho objetivou exibir indícios experimentais de que a utilização dos textos originais das mensagens é capaz de aprimorar a precisão da detecção de *bots* sociais. Para tal, foram propostos dois métodos: DetBot Alfa e DetBot Beta, que aplicam uma rede neural convolucional para identificar mensagens suspeitas. baseando-se nos atributos comportamentais e no percentual de mensagens depreendidas como suspeitas pelo modelo, no contexto de cada conta, os métodos empregam amostras aleatórias de uma parte reduzida do *dataset* e também apresenta a aplicação de amostras mais representativas de mensagens do mesmo *dataset*, objetivando identificar características de autoria destas mensagens propagadas. Por fim, os métodos ainda aplicam um classificador binário para detectar a presença de *bots* sociais a partir destas informações já consolidadas. Desta forma, resultados obtidos a partir de uma base de dados do Twitter confirmam a adequação do método aqui proposto, evidenciando que o emprego do conteúdo textual original das mensagens publicadas pelas contas pode cooperar para produção de modelos mais eficazes do que aqueles que se baseiam apenas em atributos comportamentais e estatísticas.

Portanto, as contribuições obtidas neste trabalho são as seguintes:

(i) Melhorias no desempenho de classificadores de bots sociais construídos a partir dos métodos propostos.

(ii) Novas técnicas de detecção de *bots* sob o paradigma das abordagens ***feature-based*** de detecção de *bots* sociais baseada em técnicas de Aprendizado Profundo, utilizando o conteúdo textual original das mensagens propagadas.

(iii) Nova taxonomia e modelo comparativo acerca das técnicas utilizadas para detecção de *bots* sociais utilizando Aprendizado de Máquina.

Como trabalhos futuros, um estudo mais aprofundado dos clusters para avaliar o ganho de representatividade das mensagens selecionadas. Outrossim, ainda seria válido avaliar a semântica de todas as mensagens confeccionadas. Adicionalmente, seria interessante também avaliar a adequação de outras técnicas de aprendizado profundo para a depreender traços de autoria das mensagens na detecção de *bots* sociais, tais como redes neurais recorrentes do tipo LSTM (HOCHREITER; SCHMIDHUBER, 1997) em combinação com outros modelos de aprendizado profundo tal como exposto em Kudugunta e Ferrara (2018).

Além disso, seria oportuno avaliar a criação de modelos utilizando programação paralela e distribuída fazendo uso de programação *GPU* (do inglês, Graphical Process Unit), por exemplo, objetivando deste modo reduzir sobremaneira o longo tempo de treinamento destes modelos de aprendizado profundo.

Outrossim, por fim, sob o prisma das abordagens *host-side*, existem padrões no que tange aos fluxos de dados das botnets que podem ser avaliados para as redes sociais, sobretudo ao avaliar o que preconiza van Roosmalen et al. (2018).

7 REFERÊNCIAS BIBLIOGRÁFICAS

AGRAWAL, R.; GEHRKE, J.; GUNOPULOS, D. ; RAGHAVAN, P. Automatic subspace clustering of high dimensional data for data mining applications. In: PROCEEDINGS OF THE 1998 ACM SIGMOD INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA, 1., SIGMOD '98, –., 1998. **Anais**... New York, NY, USA: ACM, 1998, p. 94–105. Disponível em: <http://doi.acm.org/10.1145/276304.276314>. Acesso em: 10/12/2017.

AIELLO, L. M.; DEPLANO, M.; SCHIFANELLA, R. ; RUFFO, G. People are strange when you're a stranger: Impact and influence of bots on social networks. In: IN ICWSM, 2012., 2012. **Anais**... [S.l.: s.n.], 2012. Disponível em: <https://arxiv.org/abs/1407.8134>. Acesso em: 5/11/2016.

ALSHARNOUBY, M.; ALACA, F. ; CHIASSON, S. Why phishing still works: User strategies for combating phishing attacks. **International Journal of Human-Computer Studies**, v. 82, p. 69–82, 2015.

ALVISI, L.; CLEMENT, A.; EPASTO, A.; LATTANZI, S. ; PANCONESI, A. Sok: The evolution of sybil defense via social networks. In: SECURITY AND PRIVACY (SP), 2013 IEEE SYMPOSIUM ON, 1., 2013. **Anais**... [S.l.: s.n.], 2013, p. 382–396.

BADRI SATYA, P. R.; LEE, K.; LEE, D.; TRAN, T. ; ZHANG, J. J. Uncovering fake likers in online social networks. In: PROCEEDINGS OF THE 25TH ACM INTERNATIONAL ON CONFERENCE ON INFORMATION AND KNOWLEDGE MANAGEMENT, -., CIKM '16, -., 2016. **Anais**... New York, NY, USA: ACM, 2016, p. 2365–2370. Disponível em: <http://doi.acm.org/10.1145/2983323.2983695>. Acesso em: -.

BAEZA-YATES, R.; RIBEIRO-NETO, B. **Modern Information Retrieval: The Concepts and Technology Behind Search**. 2nd. ed. USA: Addison-Wesley Publishing Company, 2008. ISBN 9780321416919.

BALTAZAR, J.; COSTOYA, J. ; FLORES, R. The real face of koobface: The largest web 2.0 botnet explained. **Trend Micro Research**, v. 5, n. 9, p. 10, 2009.

BARBON, S.; IGAWA, R. A. ; ZARPELAO, B. B. Authorship verification applied to detection of compromised accounts on online social networks. **Multimedia Tools and Applications**, v. 76, n. 3, p. 3213–3233, 2017.

BEAUDRY, N. J.; RENNER, R. An intuitive proof of the data processing inequality. **Quantum Info. Comput.**, v. 12, n. 5-6, p. 432–441, 2012. Disponível em: <http://dl.acm.org/citation.cfm?id=2230996.2231000>. Acesso em: -.

BENEVENUTO, F.; MAGNO, G.; RODRIGUES, T. ; ALMEIDA, V. Detecting spammers on twitter. In: PROCEEDINGS OF THE 7TH ANNUAL COLLABORATION, ELECTRONIC MESSAGING, ANTI-ABUSE AND

SPAM CONFERENCE (CEAS), 2016., 2010. **Anais**... [S.l.: s.n.], 2010. Disponível em: <http://homepages.dcc.ufmg.br/ fabricio/download/sigirfp437-benevenuto.pdf>. Acesso em: 5/07/2016.

BENGIO, Y.; DUCHARME, R.; VINCENT, P. ; JAUVIN, C. A neural probabilistic language model. **Journal of machine learning research**, v. 3, n. Feb, p. 1137–1155, 2003.

BERGER, J.; MORGAN, J. The isis twitter census: Defining and describing the population of isis supporters on twitter. **The Brookings Project on US Relations with the Islamic World**, v. 3, n. 20, 2015. Disponível em: <https://www.brookings.edu/research/the-isis-twitter-census-defining-and-describing-the-population-of-isis-supporters-on-twitter/>. Acesso em: 1/10/2016.

BESSI, A.; FERRARA, E. Social bots distort the 2016 u.s. presidential election online discussion. **First Monday**, v. 21, n. 11, 2016. Disponível em: <http://firstmonday.org/ojs/index.php/fm/article/view/7090>. Acesso em: 5/06/2017.

BEUTEL, A.; XU, W.; GURUSWAMI, V.; PALOW, C. ; FALOUTSOS, C. Copycatch: Stopping group attacks by spotting lockstep behavior in social networks. In: PROCEEDINGS OF THE 22ND INTERNATIONAL CONFERENCE ON WORLD WIDE WEB, 2013., WWW '13, -., 2013. **Anais**... New York, NY, USA: ACM, 2013, p. 119–130. Disponível em: <http://doi.acm.org/10.1145/2488388.2488400>. Acesso em: -.

BIRD, S.; KLEIN, E. ; LOPER, E. **Natural language processing with Python: analyzing text with the natural language toolkit**. [S.l.]: "O'Reilly Media, Inc.", 2009.

BOLLEN, J.; MAO, H. ; ZENG, X. Twitter mood predicts the stock market. **Journal of Computational Science**, v. 2, n. 1, p. 1–8, 2011.

BOSHMAF, Y.; LOGOTHETIS, D.; SIGANOS, G.; LERÍA, J.; LORENZO, J.; RIPEANU, M. ; BEZNOSOV, K. Integro: Leveraging victim prediction for robust fake account detection in osns.. In: NDSS, 1., 2015. **Anais**... [S.l.: s.n.], 2015, p. 8–11.

BOSHMAF, Y.; MUSLUKHOV, I.; BEZNOSOV, K. ; RIPEANU, M. The socialbot network: when bots socialize for fame and money. In: PROCEEDINGS OF THE 27TH ANNUAL COMPUTER SECURITY APPLICATIONS CONFERENCE, 1., 2011. **Anais**... [S.l.: s.n.], 2011, p. 93–102.

BOSHMAF, Y.; MUSLUKHOV, I.; BEZNOSOV, K. ; RIPEANU, M. Design and analysis of a social botnet. **Comput. Netw.**, v. 57, n. 2, p. 556–578, 2013. Disponível em: <http://dx.doi.org/10.1016/j.comnet.2012.06.006>. Acesso em: 1/09/2016.

BRAZ, P.; GOLDSCHMIDT, R. Um método para detecção de bots sociais baseado em redes neurais convolucionais aplicadas em mensagens textuais. In: SBSEG 2017 (), 1., 2017. **Anais**... [S.l.: s.n.],

2017, p. 501–508. Disponível em: <"https://sbseg2017.redes.unb.br/wp-content/uploads/2017/04/20171109_ANAIS_SBSEG_2017_FINAL_E-BOOK.pdf». Acesso em: 10/11/2017.

CAO, Q.; SIRIVIANOS, M.; YANG, X. ; PREGUEIRO, T. Aiding the detection of fake accounts in large scale social online services. In: PRESENTED AS PART OF THE 9TH USENIX SYMPOSIUM ON NETWORKED SYSTEMS DESIGN AND IMPLEMENTATION (NSDI 12), 1., 2012. **Anais**... San Jose, CA: USENIX, 2012, p. 197–210. Disponível em: <https://www.usenix.org/conference/nsdi12/technical-sessions/presentation/cao>. Acesso em: 01/05/2017.

CARVALHO, A.; FACELI, K.; LORENA, A. ; GAMA, J. **Inteligencia Artificial uma abordagem de aprendizado de maquina**. [S.l.]: Rio de Janeiro: LTC, 2011.

CAVNAR, W. B.; TRENKLE, J. M. ; OTHERS. N-gram-based text categorization. **Ann arbor mi**, v. 48113, n. 2, p. 161–175, 1994.

CHEN, X.-W.; LIU, M. Prediction of protein–protein interactions using random decision forest framework. **Bioinformatics**, v. 21, n. 24, p. 4394–4400, 2005. Disponível em: <http://dx.doi.org/10.1093/bioinformatics/bti721>. Acesso em: –.

CHOLLET, F.; OTHERS. Keras. **N/A**, v. N/A, p. N/A, 2015. Disponível em: <https://github.com/fchollet/keras>. Acesso em: 11/11/2016.

CHOWDHURY, G. G. Natural language processing. **Annual review of information science and technology**, v. 37, n. 1, p. 51–89, 2003.

CHOWDURY, R.; ADNAN, M. N. M.; MAHMUD, G. A. N. ; RAHMAN, R. M. A data mining based spam detection system for youtube. In: EIGHTH INTERNATIONAL CONFERENCE ON DIGITAL INFORMATION MANAGEMENT (ICDIM 2013), -., 2013. **Anais**... [S.l.: s.n.], 2013, p. 373–378.

CHU, Z.; GIANVECCHIO, S.; WANG, H. ; JAJODIA, S. Detecting automation of twitter accounts: Are you a human, bot, or cyborg?. **IEEE Transactions on Dependable and Secure Computing**, v. 9, n. 6, p. 811–824, 2012.

CORTES, C.; VAPNIK, V. Support vector networks. **Machine Learning**, v. 20, p. 273–297, 1995.

CROFT, W. B.; LAFFERTY, J. **Language modeling for information retrieval**. [S.l.]: Springer Science & Business Media, 2013.

DAVIS, C. A.; VAROL, O.; FERRARA, E.; FLAMMINI, A. ; MENCZER, F. Botornot: A system to evaluate social bots. In: PROCEEDINGS OF THE 25TH INTERNATIONAL CONFERENCE COMPANION ON WORLD WIDE WEB, 2016., 2016. **Anais**... [S.l.: s.n.], 2016, p. 273–274.

DOMINGOS, P. A few useful things to know about machine learning. **Commun. ACM**, v. 55, n. 10, p. 78–87, 2012. Disponível em: <http://doi.acm.org/10.1145/2347736.2347755>. Acesso em: 3/12/2016.

EISENSTEIN, J. What to do about bad language on the internet. In: PROCEEDINGS OF THE 2013 CONFERENCE OF THE NORTH AMERICAN CHAPTER OF THE ASSOCIATION FOR COMPUTATIONAL LINGUISTICS: HUMAN LANGUAGE TECHNOLOGIES, 1., 2013. **Anais**... [S.l.: s.n.], 2013, p. 359–369.

ESTER, M.; KRIEGEL, H.-P.; SANDER, J. ; XU, X. A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise. In: PROCEEDINGS OF THE SECOND INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING, 1., KDD'96, –., 1996. **Anais**... [S.l.]: AAAI Press, 1996, p. 226–231. Disponível em: <http://dl.acm.org/citation.cfm?id=3001460.3001507>. Acesso em: 20/01/2018.

ESTIVILL-CASTRO, V. Why so many clustering algorithms: a position paper. **ACM SIGKDD explorations newsletter**, v. 4, n. 1, p. 65–75, 2002.

FARABET, C.; COUPRIE, C.; NAJMAN, L. ; LECUN, Y. Learning hierarchical features for scene labeling. **IEEE transactions on pattern analysis and machine intelligence**, v. 35, n. 8, p. 1915–1929, 2013.

FERRARA, E. Manipulation and abuse on social media. **SIGWEB Newsl.**, v. –, n. Spring, p. 4:1–4:9, 2015. Disponível em: <http://doi.acm.org/10.1145/2749279.2749283>. Acesso em: 14/04/2018.

FERRARA, E. Measuring social spam and the effect of bots on information diffusion in social media. **CoRR**, v. abs/1708.08134, 2017. Disponível em: <http://arxiv.org/abs/1708.08134>. Acesso em: 05/01/20118.

FERRARA, E.; VAROL, O.; DAVIS, C.; MENCZER, F. ; FLAMMINI, A. The rise of social bots. **Commun. ACM**, v. 59, n. 7, p. 96–104, 2016. Disponível em: <http://doi.acm.org/10.1145/2818717>. Acesso em: -.

FREY, B. J.; DUECK, D. Clustering by passing messages between data points. **science**, v. 315, n. 5814, p. 972–976, 2007.

GHOSH, S.; VISWANATH, B.; KOOTI, F.; SHARMA, N. K.; GAUTAM, K.; BENEVENUTO, F.; GANGULY, N. ; GUMMADI, K. Understanding and Combating Link Farming in the Twitter Social Network. In: PROCEEDINGS OF THE 21ST INTERNATIONAL WORLD WIDE WEB CONFERENCE (WWW'12), 2012., 2012. **Anais**... Lyon, France: [s.n.], 2012. Disponível em: <www.mpi-sws.org/ farshad/TwitterLinkfarming.pdf>. Acesso em: 2/07/2016.

GHOSH, S.; DUBEY, S. K. Comparative analysis of k-means and fuzzy c-means algorithms. **International Journal of Advanced Computer Science and Applications**, v. 4, n. 4, p. –, 2013.

GILANI, Z.; FARAHBAKHSH, R.; TYSON, G.; WANG, L. ; CROWCROFT, J. Of bots and humans (on twitter). In: PROCEEDINGS OF THE 2017 IEEE/ACM INTERNATIONAL CONFERENCE ON ADVANCES IN SOCIAL NETWORKS ANALYSIS AND MINING 2017, 1., 2017. **Anais**... [S.l.: s.n.], 2017, p. 349–354.

GOLDSCHMIDT, R. R. Uma introdução à inteligência computacional: fundamentos, ferramentas e aplicações. **Rio de Janeiro Brasil: IST-Rio**, v. 1, p. 32, 2010.

GOODFELLOW, I.; BENGIO, Y. ; COURVILLE, A. **Deep Learning**. [S.l.]: MIT Press, 2016.

HAN, J.; KAMBER, M. ; PEI, J. **Data Mining: Concepts and Techniques**. 3rd. ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2011. ISBN 0123814790, 9780123814791.

HASTIE, T.; TIBSHIRANI, R. ; FRIEDMAN, J. **The elements of statistical learning: data mining, inference and prediction**. 2. ed. [S.l.]: Springer, 2009.

HAYKIN, S. **Neural Networks and Learning Machines**. [S.l.]: Prentice Hall, 2009. ISBN 9780131471399.

HINTON, G.; DENG, L.; YU, D.; DAHL, G. E.; MOHAMED, A.-R.; JAITLY, N.; SENIOR, A.; VANHOUCKE, V.; NGUYEN, P.; SAINATH, T. N. ; OTHERS. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. **IEEE Signal Processing Magazine**, v. 29, n. 6, p. 82–97, 2012.

HO, T. K. Random decision forests. In: DOCUMENT ANALYSIS AND RECOGNITION, 1995., PROCEEDINGS OF THE THIRD INTERNATIONAL CONFERENCE ON, 1., 1995. **Anais**... [S.l.: s.n.], 1995, p. 278–282.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. **Neural computation**, v. 9, n. 8, p. 1735–1780, 1997.

HWANG, T.; PEARCE, I. ; NANIS, M. Socialbots: Voices from the fronts. **interactions**, v. 19, n. 2, p. 38–45, 2012.

IGAWA, R. A.; BARBON JR, S.; PAULO, K. C. S.; KIDO, G. S.; GUIDO, R. C.; JÚNIOR, M. L. P. ; SILVA, I. N. D. Account classification in online social networks with lbca and wavelets. **Inf. Sci.**, v. 332, n. C, p. 72–83, 2016. Disponível em: <http://dx.doi.org/10.1016/j.ins.2015.10.039>. Acesso em: 05/01/2017.

IGAWA, R. A.; DE ALMEIDA, A. M. G.; ZARPELÃO, B. B. ; BARBON JR, S. Recognition of compromised accounts on twitter. In: PROCEEDINGS OF THE ANNUAL CONFERENCE ON BRAZILIAN SYMPOSIUM ON INFORMATION SYSTEMS: INFORMATION SYSTEMS: A COMPUTER SOCIO-TECHNICAL PERSPECTIVE, 1., 2015. **Anais**... [S.l.: s.n.], 2015, p. 9–14.

JAGATIC, T. N.; JOHNSON, N. A.; JAKOBSSON, M. ; MENCZER, F. Social phishing. **Commun. ACM**, v. 50, n. 10, p. 94–100, 2007. Acesso em: 2/05/2017.

JAIN, A. K. Data clustering: 50 years beyond k-means. **Pattern recognition letters**, v. 31, n. 8, p. 651–666, 2010.

JAIN, A. K.; DUBES, R. C. **Algorithms for Clustering Data**. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1988. ISBN 0-13-022278-X.

JAYNES, E. T. **Probability theory: the logic of science**. [S.l.]: Cambridge university press, 2003.

JI, Y.; HE, Y.; JIANG, X.; CAO, J. ; LI, Q. Combating the evasion mechanisms of social bots. **Comput. Secur.**, v. 58, n. C, p. 230–249, 2016. Disponível em: <http://dx.doi.org/10.1016/j.cose.2016.01.007>. Acesso em: -.

JURAFSKY, D.; MARTIN, J. H. **Speech and language processing**. [S.l.]: Pearson London, 2014.

KARTALTEPE, E.; MORALES, J.; XU, S. ; SANDHU, R. Social network-based botnet command-and-control: emerging threats and countermeasures. In: APPLIED CRYPTOGRAPHY AND NETWORK SECURITY, 1., 2010. **Anais**... [S.l.: s.n.], 2010, p. 511–528.

KERETNA, S.; HOSSNY, A. ; CREIGHTON, D. Recognising user identity in twitter social networks via text mining. In: SYSTEMS, MAN, AND CYBERNETICS (SMC), 2013 IEEE INTERNATIONAL CONFERENCE ON, 1., 2013. **Anais**... [S.l.: s.n.], 2013, p. 3079–3082.

KINGMA, D. P.; BA, J. Adam: A method for stochastic optimization. **arXiv preprint arXiv:1412.6980**, v. 1, p. –, 2014.

KINGSFORD, C.; SALZBERG, S. L. What are decision trees?. **Nature biotechnology**, v. 26, n. 9, p. 1011, 2008.

KOHONEN, T. Essentials of the self-organizing map. **Neural networks**, v. 37, p. 52–65, 2013.

KONONENKO, I. Estimating attributes: analysis and extensions of relief. In: EUROPEAN CONFERENCE ON MACHINE LEARNING, 1., 1994. **Anais**... [S.l.: s.n.], 1994, p. 171–182.

KRIZHEVSKY, A.; SUTSKEVER, I. ; HINTON, G. E. Imagenet classification with deep convolutional neural networks. In: ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS, 1., 2012. **Anais**... [S.l.: s.n.], 2012, p. 1097–1105.

KUDUGUNTA, S.; FERRARA, E. Deep neural networks for bot detection. **arXiv preprint arXiv:1802.04289**, v. 1, p. –, 2018.

LAMBIOTTE, R.; DELVENNE, J.-C.; BARAHONA, M. ; OTHERS. Random walks, markov processes and the multiscale modular organization of complex networks.. **IEEE Trans. Network Science and Engineering**, v. 1, n. 2, p. 76–90, 2014.

LECUN, Y.; BENGIO, Y. ; HINTON, G. Deep learning. **Nature**, v. 521, n. 7553, p. 436–444, 2015.

LEE, K.; EOFF, B. D. ; CAVERLEE, J. Seven months with the devils: a long-term study of content polluters on twitter. In: IN AAAI INT'L CONFERENCE ON WEBLOGS AND SOCIAL MEDIA (ICWSM, 2011., 2011. **Anais**... [S.l.: s.n.], 2011. Disponível em: <http://digital.cs.usu.edu/Acesso em: -.

LIDDY, E. D. **Natural language processing**. [S.l.: s.n.], 2001.

LIU, N.; ZHANG, B.; YAN, J.; CHEN, Z.; LIU, W.; BAI, F. ; CHIEN, L. Text representation: From vector to tensor. In: DATA MINING, FIFTH IEEE INTERNATIONAL CONFERENCE ON, 1., 2005. **Anais**... [S.l.: s.n.], 2005, p. 4–pp.

MAHER, C. A.; LEWIS, L. K.; FERRAR, K.; MARSHALL, S.; DE BOURDEAUDHUIJ, I. ; VANDELANOTTE, C. Are health behavior change interventions that use online social networks effective? a systematic review. **Journal of medical Internet research**, v. 16, n. 2, p. –, 2014.

MARSLAND, S. **Machine learning: an algorithmic perspective**. [S.l.]: CRC press, 2015.

MCINNES, L.; OTHERS. Benchmarking performance and scaling of python clustering algorithms. **N/A**, v. N/A, p. N/A, 2016. Disponível em: <http://hdbscan.readthedocs.io/en/latest/performance$_a nd_s calability.html$ > .*Acessoem* : 20/02/2018.

MIKOLOV, T. Word2vec. –, v. –, 2014. Disponível em: <https://code.google.com/archive/p/word2vec/>. Acesso em: 31 nov. de 2016.

MIKOLOV, T.; CHEN, K.; CORRADO, G. ; DEAN, J. Efficient estimation of word representations in vector space. **arXiv preprint arXiv:1301.3781**, v. 1, 2013. Disponível em: <https://arxiv.org/abs/1301.3781>. Acesso em: 20/03/2018.

MIKOLOV, T.; DEORAS, A.; POVEY, D.; BURGET, L. ; ČERNOCKỲ, J. Strategies for training large scale neural network language models. In: AUTOMATIC SPEECH RECOGNITION AND UNDERSTANDING (ASRU), 2011 IEEE WORKSHOP ON, 1., 2011. **Anais**... [S.l.: s.n.], 2011, p. 196–201.

MIKOLOV, T.; SUTSKEVER, I.; CHEN, K.; CORRADO, G. S. ; DEAN, J. Distributed representations of words and phrases and their compositionality. In: ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS, 1., 2013. **Anais**... [S.l.: s.n.], 2013, p. 3111–3119.

NAGARAJA, S.; HOUMANSADR, A.; PIYAWONGWISAL, P.; SINGH, V.; AGARWAL, P. ; BORISOV, N. "stegobot: A covert social network botnet". In: FILLER, T.; PEVNÝ, T.; CRAVER, S. ; KER, A. (Org.). **Information Hiding: 13th International Conference, IH 2011, Prague, Czech Republic, May 18-20, 2011, Revised Selected Papers**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011. p. 299–313. ISBN 978-3-642-24178-9.

NAGESH, H.; GOIL, S. ; CHOUDHARY, A. Adaptive grids for clustering massive data sets. In: PROCEEDINGS OF THE 2001 SIAM INTERNATIONAL CONFERENCE ON DATA MINING, 1., 2001. **Anais**... [S.l.: s.n.], 2001, p. 1–17.

NAZARIO, J. Twitter-based botnet command channel. **-**, v. -, 2009. Disponível em: <https://www.arbornetworks.com/blog/asert/twitter-based-botnet-command-channel/>. Acesso em: Accessed: 2017-01-20.

POLANYI, M. **The tacit dimension**. [S.l.]: University of Chicago press, 2009.

RATKIEWICZ, J.; CONOVER, M. D.; MEISS, M.; FLAMMINI, A. ; MENCZER, F. Detecting and tracking political abuse in social media. In: IN PROCEEDINGS OF THE 5TH AAAI INTERNATIONAL CONFERENCE ON WEBLOGS AND SOCIAL MEDIA (ICWSM'11, 1., 2011. **Anais**... [S.l.: s.n.], 2011. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.646.5073>. Acesso em: 13/10/2017.

RATKIEWICZ, J.; CONOVER, M.; MEISS, M.; GONÇALVES, B.; PATIL, S.; FLAMMINI, A. ; MENCZER, F. Truthy: Mapping the spread of astroturf in microblog streams. In: PROCEEDINGS OF THE 20TH INTERNATIONAL CONFERENCE COMPANION ON WORLD WIDE WEB, 1., WWW '11, –., 2011. **Anais**... New York, NY, USA: ACM, 2011, p. 249–252. Disponível em: <http://doi.acm.org/10.1145/1963192.1963301>. Acesso em: 11/11/2016.

ROKACH, L.; MAIMON, O. Clustering methods. In: MAIMON, O.; ROKACH, L. (Org.). **Data Mining and Knowledge Discovery Handbook**. Boston, MA: Springer US, 2005. p. 321–352. ISBN 978-0-387-25465-4.

ROSENBLATT, F. **Principles of neurodynamics. perceptrons and the theory of brain mechanisms**. [S.l.: s.n.], 1961. (Relatório Técnico).

ROSSI, R. G. **Classificação automática de textos por meio de aprendizado de máquina baseado em redes**. 2016. Tese (N/D) – Universidade de São Paulo, São Paulo, 2016.

SALTON, G.; BUCKLEY, C. Term-weighting approaches in automatic text retrieval. **Information processing & management**, v. 24, n. 5, p. 513–523, 1988.

SALTON, G.; MCGILL, M. J. **Introduction to Modern Information Retrieval**. New York, NY, USA: McGraw-Hill, Inc., 1986. ISBN 0070544840.

SCHÜTZE, H.; MANNING, C. D. ; RAGHAVAN, P. **Introduction to information retrieval**. [S.l.]: Cambridge University Press, 2008.

SHANNON, C. E. Prediction and entropy of printed english. **Bell Labs Technical Journal**, v. 30, n. 1, p. 50–64, 1951.

STEIN, T.; CHEN, E. ; MANGLA, K. Facebook immune system. In: PROCEEDINGS OF THE 4TH WORKSHOP ON SOCIAL NETWORK SYSTEMS, 1., 2011. **Anais**... [S.l.: s.n.], 2011, p. 8.

TAN, E.; GUO, L.; CHEN, S.; ZHANG, X. ; ZHAO, Y. Spammer behavior analysis and detection in user generated content on social networks. In: 2012 IEEE 32ND INTERNATIONAL CONFERENCE ON DISTRIBUTED COMPUTING SYSTEMS, 1., 2012. **Anais**... [S.l.: s.n.], 2012, p. 305–314.

THOMAS, K.; GRIER, C.; SONG, D. ; PAXSON, V. Suspended accounts in retrospect: An analysis of twitter spam. In: PROCEEDINGS OF THE 2011 ACM SIGCOMM CONFERENCE ON INTERNET MEASUREMENT CONFERENCE, 1., IMC

'11, –., 2011. **Anais**... New York, NY, USA: ACM, 2011, p. 243–258. Disponível em: <http://doi.acm.org/10.1145/2068816.2068840>. Acesso em: 3/05/2017.

THOMAS, K.; MCCOY, D.; GRIER, C.; KOLCZ, A. ; PAXSON, V. Trafficking fraudulent accounts: The role of the underground market in twitter spam and abuse. In: PRESENTED AS PART OF THE 22ND USENIX SECURITY SYMPOSIUM (USENIX SECURITY 13), 2013., 2013. **Anais**... [S.l.: s.n.], 2013, p. 195–210. Acesso em: 10/06/2017.

TIERNEY, L. Markov chains for exploring posterior distributions. **Annals of Statistics**, v. 22, p. 1701–1762, 1994.

TIMBERG, C. Research in india suggests google search results can influence an election. **-**, v. 1, 2013. Disponível em: <->. Acesso em: 4/12/2016.

TOUTANOVA, K.; KLEIN, D.; MANNING, C. D. ; SINGER, Y. Feature-rich part-of-speech tagging with a cyclic dependency network. In: PROCEEDINGS OF THE 2003 CONFERENCE OF THE NORTH AMERICAN CHAPTER OF THE ASSOCIATION FOR COMPUTATIONAL LINGUISTICS ON HUMAN LANGUAGE TECHNOLOGY-VOLUME 1, 1., 2003. **Anais**... [S.l.: s.n.], 2003, p. 173–180.

VAN ROOSMALEN, J.; VRANKEN, H. ; VAN EEKELEN, M. Applying deep learning on packet flows for botnet detection. In: PROCEEDINGS OF THE 33RD ANNUAL ACM SYMPOSIUM ON APPLIED COMPUTING, 1., SAC '18, –., 2018. **Anais**... New York, NY, USA: ACM, 2018, p. 1629–1636. Disponível em: <http://doi.acm.org/10.1145/3167132.3167306>. Acesso em: 28/07/2018.

WANG, G.; KONOLIGE, T.; WILSON, C.; WANG, X.; ZHENG, H. ; ZHAO, B. Y. You are how you click: Clickstream analysis for sybil detection. In: PRESENTED AS PART OF THE 22ND USENIX SECURITY SYMPOSIUM (USENIX SECURITY 13), 1., 2013. **Anais**... [S.l.: s.n.], 2013, p. 241–256.

WANG, G.; ZHANG, X.; TANG, S.; ZHENG, H. ; ZHAO, B. Y. Unsupervised clickstream clustering for user behavior analysis. In: PROCEEDINGS OF THE 2016 CHI CONFERENCE ON HUMAN FACTORS IN COMPUTING SYSTEMS, 1., CHI '16, –., 2016. **Anais**... New York, NY, USA: ACM, 2016, p. 225–236. Disponível em: <http://doi.acm.org/10.1145/2858036.2858107>. Acesso em: 5/10/2016.

WANG, H.; RAJ, B. A survey: Time travel in deep learning space: An introduction to deep learning models and how deep learning models evolved from the initial ideas. **arXiv preprint arXiv:1510.04781**, v. 1, 2015. Disponível em: <https://arxiv.org/abs/1510.04781>. Acesso em: 17/2/2017.

WITTEN, I. H.; FRANK, E.; HALL, M. A. ; PAL, C. J. **Data Mining: Practical machine learning tools and techniques**. [S.l.]: Morgan Kaufmann, 2016.

WOLD, S.; ESBENSEN, K. ; GELADI, P. Principal component analysis. **Chemometrics and intelligent laboratory systems**, v. 2, n. 1-3, p. 37–52, 1987.

XIAO, C.; FREEMAN, D. M. ; HWA, T. Detecting clusters of fake accounts in online social networks. In: PROCEEDINGS OF THE 8TH ACM WORKSHOP ON ARTIFICIAL INTELLIGENCE AND SECURITY, 2015., AISEC

'15, -., 2015. **Anais**... New York, NY, USA: ACM, 2015, p. 91–101. Disponível em: <http://doi.acm.org/10.1145/2808769.2808779>. Acesso em: 1/12/2016.

YANG, Z.; WILSON, C.; WANG, X.; GAO, T.; ZHAO, B. Y. ; DAI, Y. Uncovering social network sybils in the wild. **ACM Transactions on Knowledge Discovery from Data (TKDD)**, v. 8, n. 1, p. 2, 2014.

YANG, Z.; XUE, J.; YANG, X.; WANG, X. ; DAI, Y. Votetrust: Leveraging friend invitation graph to defend against social network sybils. **-**, v. -, 2015. Disponível em: <http://ieeexplore.ieee.org/document/6567045>. Acesso em: 16/07/2016.

ZHANG, X.; ZHAO, J. ; LECUN, Y. Character-level convolutional networks for text classification. In: ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS, 1., 2015. **Anais**... [S.l.: s.n.], 2015, p. 649–657.