Universidade de São Paulo
Instituto de Matemática e Estatística
Bacharelado em Ciência da Computação

Detecção de sarcasmo em redes sociais utilizando DeBERTa

Lucas Paiolla Forastiere

Monografia Final

mac 499 — Trabalho de Formatura Supervisionado

Supervisor: Prof. Dr. Ricardo Marcondes Marcacini

São Paulo
2017

Resumo

Lucas Paiolla Forastiere. **Detecção de sarcasmo em redes sociais utilizando DeBERTa**. Monografia (Bacharelado). Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2017.

Elemento obrigatório, constituído de uma sequência de frases concisas e objetivas, em forma de texto. Deve apresentar os objetivos, métodos empregados, resultados e conclusões. O resumo deve ser redigido em parágrafo único, conter no máximo 500 palavras e ser seguido dos termos representativos do conteúdo do trabalho (palavras-chave). Deve ser precedido da referência do documento. Texto texto. Texto texto.

Palavras-chave: Palavra-chave1. Palavra-chave2. Palavra-chave3.

Abstract

Lucas Paiolla Forastiere. **Sarcasm detection in social media using decoding-enhanced BERT with disentangled attention**. Capstone Project Report (Bachelor). Institute of Mathematics and Statistics, University of São Paulo, São Paulo, 2017.

Elemento obrigatório, elaborado com as mesmas características do resumo em língua portuguesa. De acordo com o Regimento da Pós-Graduação da USP (Artigo 99), deve ser redigido em inglês para fins de divulgação. É uma boa ideia usar o sítio www.grammarly.com na preparação de textos em inglês. Text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text. Text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text.

Keywords: Keyword1. Keyword2. Keyword3.

Lista de Abreviaturas

CFT	Transformada contínua de Fourier (*Continuous Fourier Transform*)
DFT	Transformada discreta de Fourier (*Discrete Fourier Transform*)
EIIP	Potencial de interação elétron-íon (*Electron-Ion Interaction Potentials*)
STFT	Transformada de Fourier de tempo reduzido (*Short-Time Fourier Transform*)
ABNT	Associação Brasileira de Normas Técnicas
URL	Localizador Uniforme de Recursos (*Uniform Resource Locator*)
IME	Instituto de Matemática e Estatística
USP	Universidade de São Paulo

Lista de Símbolos

ω	Frequência angular
ψ	Função de análise *wavelet*
Ψ	Transformada de Fourier de ψ

Lista de Figuras

Lista de Tabelas

Lista de Programas

Sumário

Capítulo 1

Introduction

Sarcasm detection is an important aspect of many natural language processing (NLP) systems, with many implications in natural language understanding, dialog systems, and data mining. However, sarcasm detection is difficult because it is infrequent in many conversations and, many times, it is difficult even for humans to discern.

Many studies have been made in the area and many datasets have been proposed with either *balanced* or *unbalanced* data. Also many of these datasets use humans to annotate sarcastic statements.

In this paper, we use the Self-Annotated Reddit Corpus (SARC), which is a large corpus for sarcasm detection created using Reddit posts to get labels automatically, to train, evaluate, and compare many NLP models.

Capítulo 2

Fundamentos e Trabalhos Relacionados

2.1 Detecção de Sarcasmo

De acordo com o dicionário Dicio, sarcasmo é uma zombaria que busca ofender, enquanto ironia é a ação de dizer o oposto do que se deseja expressar. Ainda segundo ele, a diferença entre esses dois termos se dá no fato de que sarcasmo é um dito ácido que pode ou não ser expresso por meio de uma ironia e essa, por sua vez, pode ou não ser utilizada para ofender. **dicio_sarc**; **dicio_irony**

(R. Giora, On Irony and Negation)

(H. Paul Grice. 1975. Logic and Conversation)

(Irony and Sarcasm: Corpus Generation and Analysis Using Crowdsourcin)

O termo detecção de sarcasmo refere-se à determinação de se há ou não sarcasmo em uma porção de texto verbal. E o termo detecção automática de sarcasmo refere-se a métodos computacionais de se resolver o problema acima mencionado. Computacionalmente, podemos definir esse problema como uma ***classificação binária***, termo explicado mais a frente.

Entretanto, na literatura é bastante comum também se definir detecção de sarcasmo como a determinação de se há ou não sarcasmo ou ironia verbal em uma porção de texto verbal. Portanto, em geral, ao se falar de sarcasmo, a literatura engloba tanto sarcasmo quanto ironia como se fossem a mesma coisa. Este texto também não fará discriminação entre sarcasmo ou ironia.

Em geral, esse problema é difícil, pois, por vezes, nem humanos conseguem perceber essa figura de linguagem (e.g. "*Oba, hoje está tão ensolarado, que vontade de ir para a escola.*"). Além disso, o contexto tende a importar muito. Muitas características externas ao texto podem servir para descriminar se uma pessoa está ou não sendo sarcástica. Entre alguns exemplos estão intonação, locutor, interlocutor, conhecimento prévio sobre a fala, tempo e espaço em que se fala e elementos não ver-

bais. **wallace-etal:2014:ironic-context**

2.2 Classificação binárias

O problema de detecção de sarcasmo pode ser tratado como uma classificação binária. Esse tipo de problema é muito estudado no campo do aprendizado de máquina e envolve classificar os elementos de um conjunto em dois grupos chamados de classes. Por classificar entende-se a determinação de um elemento do conjunto entre pertencente à primeira ou segunda classe.

No caso da detecção de sarcasmo, os elementos são os pedaços de texto e as duas classes são *ser sarcástico* e *não ser sarcástico*. Em termos matemáticos, tem-se um conjunto de exemplos denotado pela matriz X de dimensões $n \times m$, onde n é o número de exemplos e m é o número de características que se usa para descrever cada um desses exemplos. Cada linha de X é denotada por $x^{(i)}$ e chamada de *exemplo i* ou *instância i*.

A linha $x^{(i)}$ é um vetor de m posições em que cada posição é uma característica que descreve a entrada. Por exemplo, na detecção de sarcasmo, o primeiro valor pode ser a quantidade de palavras lidas, o segundo valor pode ser a quantidade de caracteres, a terceira pode ser a soma de positividade do texto (definida por algum critério específico) e assim por diante. É importante que a mesma posição em diferentes instâncias tenha o mesmo significado. Portanto, caso se defina que a primeira posição represente o número de palavras, todas as instâncias devem seguir essa regra.

Cada instância i pertence ou à primeira classe ou à segunda, modela-se isso por um valor $y^{(i)}$ chamado de i-ésimo rótulo (do inglês, *label*). E escreve-se $y^{(i)} \in \{ 0, 1 \}$, onde cada valor representa uma das classes. No caso de sarcasmo, pode-se representar por *não sarcasmo* o valor 0 e *sarcasmo*, o valor 1. Com esses valores $y^{(i)}$ constrói-se um vetor y onde $y_i = y^{(i)}$.

Ao final, o problema é descobrir uma função f que mapeie bem X em y. Note, entretanto, que n, o número de exemplos, é possivelmente infinito, pois sempre se pode achar novos exemplos e testar se f os mapeia corretamente. Ou seja, tenta-se achar essa função f conhecendo parcialmente o conjunto das instâncias.

2.2.1 Métricas de avaliação

Para saber se a função f mapeia bem X em y, é preciso definir uma métrica de avaliação, que é uma função matemática bem definida que indica quanto de erro se comete em determinado conjunto X.

Dado um conjunto de entrada X com n instâncias, seja y o vetor que representa se cada instância é ou não sarcástica. Seja então f a função sob avaliação (ou seja, a função de modelagem que recebe os textos e determina se são ou não sarcásticos). Seja $\hat{y} = f(X)$ os resultados gerados por essa função.

Note que para cada exemplo i, podemos ter duas opções:

1. $\hat{y}_i = y_i$ (*acerto*)

2. $\hat{y}_i \neq y_i$ (*erro*)

Além disso, pode-se note que o par $(y_i,\ \hat{y}_i)$ pode ter quatro valores:

1. $(1,\ 1)$, caso *verdadeiro positivo* (também conhecido como *true positive* e denotado por tp).
2. $(0,\ 1)$, caso *falso positivo* (também conhecido como *false positive* e denotado por fp).
3. $(0,\ 0)$, caso *verdadeiro negativo* (também conhecido como *true negative* e denotado por tn).
4. $(1,\ 0)$, caso *falso negativo* (também conhecido como *false negative* e denotado por fn).

Definidos esses valores, pode-se definir algumas métricas que ajudam a perceber quão boa a função f é. Dessa forma, pode-se comparar duas funções f.

Acurácia Definida simplesmente como a quantidade total de acertos divido pela quantidade total de instâncias. Seja n' a quantidade de acertos, então:

$$\texttt{Acurácia} = \frac{n'}{n}$$

É comum também definir a acurácia em termos dos vamos acima listados:

$$\texttt{Acurácia} = \frac{tp + tn}{tp + tn + fp + fn}$$

Note que $tp + tn$ representa justamente quando $\hat{y}_i = y_i$ e $fp + fn$, $\hat{y}_i \neq y_i$.

Precisão Definida como:

$$\texttt{Precisão} = \frac{tp}{tp + fp}$$

é a taxa de verdadeiros positivos em relação a todos os positivos. Portanto, mede a fração de quantos positivos foram acertados em relação a todos os que existiam no conjunto observado.

Revocação Muitas vezes chamada por seu nome em inglês, *recall*, é definida como:

$$\texttt{Revocação} = \frac{tp}{tp + fn}$$

é a taxa de verdadeiros positivos em relação todas as instâncias preditas como positivas. Portanto, a fração entre todos as instâncias previstas como positivas e as instâncias que realmente eram positivas.

A precisão e a revocação guardam uma relação chave entre si e geralmente se quer ter um bom balanço entre as duas. Observe essa relação a partir do seguinte exemplo.

Comece com a função

$$f\left(x^{(i)}\right) = 1 \quad \forall i \tag{2.1}$$

Como ela sempre considera a entrada como positiva, então os únicos casos existentes serão de verdadeiros positivos e falso positivos. Logo:

$$\texttt{Precisão} = \frac{tp}{tp + fp} = \frac{tp}{tp + (n - tp)} = \frac{tp}{n}$$

e

$$\texttt{Revocação} = \frac{tp}{tp + fn} = \frac{tp}{tp + 0} = 1$$

Portanto, a revocação terá o maior valor possível, enquanto a precisão será exatamente a fração de valores positivos que o conjunto observado possui. Como, semanticamente, a revocação é penalizada sempre que um elemento é positivo, mas foi previsto como negativo, então faz sentido que se tenha uma revocação de 100%, já que não se comete esse tipo de erro.

Agora, tome a função

$$f\left(x^{(i)}\right) = 0 \quad \forall i \tag{2.2}$$

Como ela sempre considera a entrada como negativa, então os únicos casos existentes serão de verdadeiro negativo e falso negativo. Logo:

$$\texttt{Precisão} = \frac{tp}{tp + fp} = \frac{0}{0 + fp} = 0$$

e

$$\texttt{Revocação} = \frac{tp}{tp + fn} = \frac{0}{0 + fn} = 0$$

Portanto, como a função nunca tenta marcar uma instância como positiva, ela nunca acerta os verdadeiros positivos e comete 100% de erro.

Métrica F_1 Por esses e outros motivos, é interessante agregar a precisão e revocação em um único valor, que permite fácil comparação entre dois modelos. É possível utilizar uma média simples desses dois valores, mas é muito mais comum a utilização da métrica F_1 (ou, do inglês, F_1 *score*).

Ela é definida como:

$$F_1 = 2 \cdot \frac{\texttt{precisão} \cdot \texttt{revocação}}{\texttt{precisão} + \texttt{revocação}}$$

2.3 Trabalhos Relacionados

Nessa seção abordar-se-á os trabalhos já realizados na área e as diferentes abordagens. Pode-se caracterizar os trabalhos por três principais tipos de abordagens: as baseadas em regras, baseadas em métodos de aprendizado e baseadas em contexto. Além disso,

falar-se-á um pouco sobre as principais características extraídas dos textos que têm sido utilizadas para melhorar a eficácia das soluções.

2.3.1 Abordagens Baseadas em Regras

Abordagens baseadas em regras são aquelas que usam regras fixas para determinar se uma sequência de palavras contém ou não ironia. Para criar essas regras, várias características do texto podem ser utilizadas, como as classes sintáticas das palavras, se são palavras de cunho positivo ou negativo, se há a presença ou não de certas palavras em uma ordem, entre qualquer outro tipo de regra imaginável.

Essa abordagem é computacional, porque as regras formam um algoritmo que pode ser implementado por uma linguagem de computação. Dessa forma, qualquer sequência de palavras pode ser dada como entrada para o algoritmo e ele retornará se ele acredita que essa sequência contém ou não sarcasmo.

A principal vantagem desse tipo de abordagem é que ela, além de detectar o sarcasmo, ajuda a estudá-lo. Ao criar uma regra do tipo *se o texto possui essas características, então ele é sarcástico* se cria uma explicação para as principais características presentes em um texto sarcástico.

Entretanto, as metodologias utilizadas são bastante específicas para cada tipo de texto e cada conjunto de dados. Por exemplo, as regras criadas para um determinado conjunto da rede social Twitter podem não funcionar em outras redes sociais como o Facebook, o Instagram ou o WhatsApp, pois cada uma dessas redes sociais possui um contexto e formato de conversação muito diferente. Portanto, são modelos que nos permitem explicar o processo de decisão, mas não permitem fácil generalização.

veale:2010 investigam em seu artigo sequências da forma "*as * as a **" ("tão * quanto um(a) **"), consideradas como analogias entre o que os autores chamam de *base* (*ground*) e *meio* (*vehicle*). Eles utilizam a API do Google para coletar 45021 instâncias do padrão "*about as * as **" e filtram os resultados na mão para chegar em 20299 instâncias que de fato são consideradas analogias. Eles então anotam manualmente os rótulos para essas instâncias e encontraram que 15502 casos (76%) são irônicos e apenas 4797 (24%) são não irônicos.

Então, dado uma sequência "*as * as a **", os autores utilizam mecanismos de busca na rede como a API do Google para distinguir entre três casos: os casos em que essa sequência nunca foi usada como "*about*"; aqueles que já foram, mas não frequentemente; e aqueles que que são frequentemente usados com essa marcação. Segundo os próprios autores, essas três categorias proveem, respectivamente, evidencia fraca contra ironia, evidencia fraca a favor da ironia e evidência forte para ironia. Ou seja, o caso em que temos mais certeza de que é uma ironia é o caso em que a sequência é frequentemente utilizada junto com a palavra "*about*".

Assim sendo, **veale:2010** criam um sequência de nove passos para classificar uma frase desse tipo entre as classes *ironia* e *não ironia*. Esses passos são bastante claros e precisos, podendo ser, portanto, implementados por um programa de computador e caracterizando, por conseguinte, uma detecção automática de sarcasmo.

Nessas regras, os autores utilizam o fato de que analogias mais frequentemente utilizadas são menos prováveis de serem irônicas, pois a ironia é utiliza bastante a criatividade do locutor para fazer analogias não usuais.

Devido à natureza das regras propostas pelos autores, eles conseguem medir a precisão e revocação obtida por cada uma das regras. No geral, o modelo atinge uma acurácia de 88%, o que é um valor bastante alto para esse tipo de problema. Entretanto, note-se que os autores se limitaram a um escopo muito fechado (das frases do tipo "*as * as a **").

maynard-greenwood:2014:cares exploram o uso de regras baseadas em hashtags presentes em postagens da rede social Twitter (`https://twitter.com/`) e sua aplicação para detecção de sarcasmo em um contexto mais geral de análise de sentimento.

Em seu artigo, eles utilizam um algoritmo de tokenização de hashtags para transformá-las em palavras com as quais eles conseguem extrair informação. No caso, eles utilizam as palavras contidas nas hashtags para avaliar se o sentimento é positivo ou negativo e inverter o sentido original da frase caso detectem sarcasmo nas hashtags. Por exemplo, a hashtag *#notreally* é tokenizada em *not* e *really* e identificada como uma tag sarcástica de acordo com regras definidas pelos autores.

Então, eles aplicam cinco regras que podem determinar o sentimento de uma postagem como *negativo* ou *positivo*, ou então trocar o sentimento pré-determinado da postagem. Os autores, portanto, fazem uso da detecção de sarcasmo utilizando hashtags para resolver um outro problema mais geral que é definir o sentimento predominante de um texto.

Como ponto negativo de sua técnica, está o fato de que nem sempre o tokenizador de hashtags funciona de forma adequada. Por exemplo, *#greatstart* pode ser dividida de duas formas: *greats* e *tart* ou *great* e *start*. Um ser humano provavelmente saberia que a segunda opção é a mais provável, mas o algoritmo não sabe fazer essa distinção e acaba ficando com o primeiro conjunto e palavras. Apesar disso, esse sistema de tokenização possui um $F1$ de $97,25\%$ e os autores obtiveram 91% de precisão e revocação no conjunto de dados utilizado por eles.

bharti-etal:2015:parsing-sarcasm apresentam duas abordagens baseadas em regras. A primeira é através da análise morfossintática (do inglês, *Part-of-Speech Tagging*) e criação de árvores sintáticas (do inglês, *parse-trees*) que identificam a positividade do sentimento e situação predominantes em tweets. A segunda é através da análise sintática de tweets que começam com interjeições.

Em sua primeira abordagem, os autores utilizam dicionários e algoritmos pré-criados para encontrar as classes morfológicas e sintáticas das palavras. Então, baseando-se nisso, eles utilizam um algoritmo criado por eles para encontrar um valor de positividade para o sentimento do texto e para a situação retratada e, com esses valores, caso seus valores tenham sinais contrários (sentimento positivo e situação negativa ou vice-versa), eles consideram o texto como sarcástico. Por exemplo, em "*I hate Australia in cricket, because they always win*" ("eu odeio a Australia no críquete, porque eles sempre ganham"), as palavras "*I hate*" apresentam um sentimento negativo, enquanto as palavras "*they always win*" retratam uma situação positiva, e, portanto, essa frase seria classificada como sarcástica em sua primeira abordagem.

Em sua segunda abordagem, os autores utilizam novamente algoritmos de *POS Tagging* pré-criados para achar as classes morfológicas e sintáticas de palavras em tweets que começam com interjeições, como *wow*, *oh*, *wow*, *aha*, *yay*, *yeah*, *nah*, etc. Em seu algoritmo, eles classificam como sarcásticos os textos que começam por interjeições e possuem um adjetivo ou advérbio imediatamente após a interjeição, ou então possuem, em algum após a interjeição, ou um advérbio seguido de adjetivo ou um adjetivo seguido de um substantivo ou um advérbio seguido de um verbo. Essa abordagem é bastante interessante, pois ela é bastante simples e, ainda assim, consegue um ótimo resultado de 0.90 F_1 no subconjunto de tweets marcado com a hashtag *sarcasm*.

2.3.2 Conjuntos de Características

Antes de prosseguir com as abordagens, essa seção mostra um pouco dos principais conjuntos de características usadas por essas abordagens. Abordagens baseadas em regras utilizam determinadas características como principalmente as classes morfossintáticas das palavras. Entretanto, as abordagens listadas a seguir fazem uso de conjuntos muito maiores de características retiradas do texto e vários trabalhos foram realizados para expandir a quantidade de características possíveis de se utilizar.

Como dito anteriormente, ao criar a matriz de exemplos de entrada X, cada coluna representa uma característica do texto. Dessa forma, o texto fica agregado em um conjunto de valores numéricos que são acessíveis para a máquina utilizar e comparar. Ao invés de se comparar dois texto distintos com números de caracteres e palavras diferentes, utiliza-se um algoritmo para extrair características (do inglês, "*features*") comuns entre os textos e compará-los por meio dessas características.

Um exemplo é o método chamado *bag-of-words* (em tradução literal "sacola-de-palavras") que cria um vetor com, por exemplo, 512 posições de valores naturais onde cada posição do vetor representa quantas vezes uma determinada palavra apareceu no texto. Cada posição do vetor é associada a uma palavra e, muitas vezes, se utiliza uma outra posição especial para denotar qualquer palavra que não está no dicionário de palavras selecionadas previamente. Por exemplo, a primeira posição pode representar quantas vezes a palavra "*a*" aparece no texto, a segunda posição pode representar quantas vezes a palavra "*the*" aparece, e assim por diante. Se a palavra "Lucas"for lida, então será adicionado um à última posição do vetor, que é utilizado para contar qualquer palavra que não é uma das outras 511.

(imagem ilustrando BOW)

Esse tipo de abordagem fica entre o baseado em regras e abordagens baseadas em aprendizado no quesito de explicabilidade dos modelos. Isso porque os autores, ao invés de criar regras específicas, utilizam métodos de extração de características, criadas especificamente com o intuito de fazer a detecção de sarcasmo.

A vantagem dessa abordagem é que geralmente os modelos são mais generalistas do que os baseados em regras, que são bastante fixas e não se adaptam bem a diferentes conjuntos de dados. Nesse caso, as mesmas metodologias de extração de características podem ser aplicadas em um conjunto de dados de textos extraídos de contextos diferentes (como

redes sociais, análises de produtos vendidos online, aplicativos de troca de mensagens ou falas transcritas).

Entretanto, como desvantagem, o fator manual ainda é muito presente nesse tipo de abordagem, assim como os métodos baseados em regras. Ao criar um modelo, um determinado conjunto de características é escolhido *a priori* arbitrariamente, assim como era o caso das regras no primeiro tipo de abordagem.

reyes:2012:from-humor exploram quatro grupos de características dentro de um contexto mais geral de detecção de humor e ironia. Esses grupos são: ambiguidade, através dos níveis estrutural, morfossintático e semântico; polaridade, através de palavras que possuam semântica positiva ou negativa; imprevisibilidade, através de desbalanceamentos contextuais entre o significado literal das palavras; e os cenários emocionais, através das emoções passadas por cada palavras.

liebrecht:2013:perfect-solution utilizam uni-, bi- e trigramas como características do texto. Um n-grama é uma sequência de n palavras do texto. Portanto, nesse tipo de característica, o unigrama seria o *bag-of-words* clássico citado acima, com as contagens de cada palavra, já o bigrama se trata das contagens de duas palavras seguidas. Por exemplo, sempre que as palavras "*so nice*" aparecerem em sequência, adiciona-se um à coluna que se refere a essa sequência.

reyes:2013:multidimensional-approach utilizam uma abordagem parecida com **reyes:2012:from-humor**, com quatro grupos de características: assinaturas, imprevisibilidade, estilo e cenários emocionais. Cada um dos grupos foca em várias características. Dentro das assinaturas, por exemplo, há o foco em pontuações específicas, emojis, citações e palavras em maiúsculas; há o foco em marcas implícitas que geram oposição entre partes do texto; e há o foco na compressão temporal, que identifica oposição temporal entre elementos relacionados.

A grande diferença desse trabalho em relação aos demais está, entretanto, no grupo de características de estilo. Esse grupo tenta modelar o estilo de escrita do texto para, assim, fazer a diferenciação entre estilos sarcásticos e não sarcásticos. Para fazer isso, eles utilizam três tipos de sequências textuais: n-gramas no nível de caracteres (ao invés de palavras) (abreviado pelos autores por *c-grams*), *skip-grams* (*s-grams*) e *skip-grams* de polaridade (*ps-grams*).

Os *c-grams* capturam a frequência de sequências de informação morfológica como sufixos e prefixos (como *-ly*, *-ing*, *dis-*, *un-*). Os *s-grams* funcionam como n-gramas, mas com alguns pulos entre palavras e são utilizados para obter relações entre palavras não adjacentes no texto. Por exemplo, na sentença "Hoje está chovendo, quero tanto ir à praia", um bigrama é representado pela sequência "Hoje está", enquanto um bigrama com pulo de uma palavra seria "Hoje chovendo". Por fim, os *ps-grams* criam uma sequência de rótulos de positividade baseado nos *s-grams*. Por exemplo, "Hoje chovendo" seria rotulado como "pos-neg" (positivo e negativo).

barbieri:2014:modelling-sarcasm, por sua vez, usam sete grupos de características léxicas que abstraem o uso de termos específicos e tornam o processo de detecção de sarcasmo mais genérico e fácil de implementar, não modelando padrões de palavras como em trabalhos anteriores. Esses grupos são: frequência, que mede o quão comum é cada

palavra utilizada no texto; escrito-falado (da tradução literal de *written-spoken* utilizado pelos autores em seu artigo), que funciona como a frequência, mas a frequência que a palavra é usada em diálogos escritos ou falados; intensidade, que mede o uso de adjetivos e advérbios para aumentar ou diminuir a intensidade semântica de outras palavras no texto; estrutura, que mede, por exemplo, o número de caracteres usados, o número de palavras, o tamanho médio das palavras, o número de verbos, de substantivos, o número total de pontuações, o número de risadas, o número de vírgulas, pontos finais, sinais de exclamação, o número de emojis (ou emoticons), o número de organizações, pessoas, títulos e datas citados pelos texto; sentimento, que mede valores de positividade entre as palavras e usa algumas métricas como soma dos valores positivos, soma dos negativos, média da diferença entre positivos e negativos, entre outros; sinônimos, que calcula métricas relacionadas aos sinônimos das palavras usadas no texto, como a quantidade de sinônimos que uma determinada palavra tem e que possuem frequência de utilização menor do que ela; e ambiguidade, que mede a quantidade de sentidos possíveis que uma palavra pode ter.

joshi:2015:context-incongruity propõem uma nova abordagem baseada em uma teoria linguística chamada de incongruência de contexto. Eles se baseiam no fato de que o tempo para um ser humano processar um texto sarcástico depende do grau de incongruência presente nele e criam características o que chamam de incongruências explícitas e implícitas. A primeira é caracterizada por palavras de sentimentos opostos e a segunda é caracterizada por frases de sentimento implícito, que não se pode detectar através de uma única palavra.

mishra:2016:harnessing-cognitive seguem por uma abordagem bastante diferente das demais, que utiliza características baseadas no movimento produzido pelos olhos de pessoas lendo textos sarcásticos. Algumas das características propostas por eles são a fixação da visão em pontos do texto e saltos mais rápidos entre duas porções do texto do que em relação às demais. Além disso, eles utilizam um grafo de informação estrutural ao utilizar as palavras como vértices do grafo e os saltos de visão entre as palavras como arestas.

2.3.3 Abordagens Baseadas em Métodos de Aprendizado

Métodos de aprendizado são aqueles que usam modelos baseados em aprendizado de máquina. Dentro desse campo do conhecimento, existem alguns tipos de aprendizado. Dentre eles, os principais são o aprendizado supervisionado, o aprendizado semi-supervisionado, o aprendizado não supervisionado e o aprendizado por reforço. Pesquisadores já fizeram uso dos três primeiros tipos de aprendizados listados, mas nesta monografia ater-se-á apenas aos trabalhos envolvendo aprendizado supervisionado, pois são os mais similares às técnicas aqui utilizadas. Para uma listagem mais vasta dos trabalhos correlacionados, recomenda-se a leitura dos trabalhos de **joshi:2017:sarcasm-detection-survey** e **yaghoobian:2021:sarcasm-detection-comparative-study**.

Dentro do campos do algoritmos de aprendizado supervisionado estão os algoritmos de classificação, que como dito anteriormente, trata-se de achar uma função f que mapeie bem os exemplos X em seus respectivos rótulos y. A matriz X contém contém diversas características extraídas do texto, como as exploradas em **??**. E o vetor de rótulos y contém números associados às classes de classificação. No caso da classificação binária, são

apenas duas classes (que, em detecção de sarcasmo, são *sarcasmo* e *não sarcasmo*), mas em contextos mais gerais, podem haver várias classes, como em análise de sentimento. Nesse caso, as classes podem ser o sentimento predominante no texto e sarcasmo é apenas mais uma entre outras classes. Isso depende de como cada autor quer modelar o problema e quais aplicações se espera ter de resultado.

Em abordagens clássicas utilizando aprendizado de máquina, muitos trabalhos se baseiam em modelos bem conhecidos e estudados como *decision trees* (DT), *random forecasts* (RF), *support vector machines* (SVM), *naive Bayes* (NB) e *Neural Networks* (NN) (**yaghoobian:2021:sarcasm-detection-comparative-study**). Esses modelos são utilizados em várias áreas nas quais se aplica o aprendizado de máquina e são aplicados por muitos autores sem colocar seu foco principal em modificar o modelo ou criar modelos específicos para a detecção de sarcasmo. Em alguns outros trabalhos, entretanto, os autores procuram alternativas menos conhecidas ou modelos híbridos para conseguir melhores resultados.

riloff:2013:sarcasm-constract utilizam uma abordagem híbrida que combina as previsões de uma SVM com um método baseado em contraste. Dessa forma, os autores combinam uma abordagem baseada em aprendizado com uma baseada em regras para criar um novo modelo. Nesse modelo, sempre que qualquer uma das duas abordagens classificar o texto como sarcástico, então o texto é classificado como sarcástico pelo modelo final.

liebrecht:2013:perfect-solution fazem uso de um algoritmo menos conhecido chamado de *balanced winnow*, que é similar ao algoritmo *perceptron*. Esse algoritmo permite a investigação de quais características foram mais importantes para fazer a classificação. Dessa forma, é possível procurar quais os padrões mais importantes para o classificador e tentar entender por que esse é o caso.

Árvores de decisão (do inglês *decision trees*) foram experimentadas por **reyes:2013:multidimensiona** para poder investigar a ordem de decisões tomadas pela árvore para ver também quais características fazem a maior diferença e explicam mais o resultado. Além disso, eles usam também o modelo *naive Bayes*, pois seu artigo utiliza diversas características booleanas (com valores zero ou um).

joshi-etal:2016:harnessing partem para um estudo e abordagem diferentes. Ao invés de utilizar bases de dados de redes sociais, como o frequentemente utilizado Twitter, eles exploram a detecção de sarcasmo em diálogos da séries de TV "Friends". E argumentam que o problema de detecção de sarcasmo de diálogos é melhor modelado como uma tarefa de rotulação de sequências ao invés de uma tarefa de classificação de cada rótulo isoladamente.

Nessa modelagem, a ordem em que as falas de um diálogo aparecem é levada em conta. Ao invés de tomar uma fala isoladamente e classificá-la, o modelo deve classificá-la utilizando todas as falas que vieram antes dela. Eles, então, utilizam dois modelos de aprendizado de máquina, o SEARN e o SVM-HMM e mostram que esses modelos de sequência performam melhor que os classificadores tradicionais.

liu-etal:2014:imbalanced-classification apresentam uma nova estratégia chamada por eles de *multi-strategy ensemble learning approach* (MSELA). Em seu trabalho, eles

exploram a detecção de sarcasmo em um conjunto de dados bastante desbalanceado (ou seja, com uma proporção pequena de textos sarcásticos em relação aos não sarcásticos). Para lidar com esse problema, eles propõem essa estratégia.

A MSELA é constituída de três partes diferentes. A primeira é chamada de *sample-ensemble strategy* (em tradução livre, "estratégia de conjunto de amostragens") e consiste em fazer amostragens nos dados originais de forma a criar N conjuntos da dados balanceados. Cada um desses sub-conjuntos de dados é então usado para treinar pequenos classificadores, que consistem da segunda parte: a *classifier-ensemble strategy* (em tradução livre, "estratégia de conjunto de classificadores"). Nessa etapa, cada os classificadores de cada sub-conjunto de dados não necessariamente são os mesmos. Os autores empregam três tipos de classificadores (sendo que, em teoria, pode-se utilizar mais) e, para cada sub-conjunto, escolhem um classificador aleatório para ser treinado. Por fim, a terceira etapa serve para juntar cada uma das classificações em uma única e é chamada de *weighted voting strategy* (em tradução livre, "estratégia de voto ponderado"). Nessa última estratégia, as decisões de cada classificador são ponderadas por uma medida de entropia da informação e, dessa forma, unidas para resultar em uma última decisão de se o texto é ou não sarcástico.

Além do modelos de aprendizado tidos como mais "clássicos", modelos de aprendizado profundo (ou seja, aqueles que utilizam grandes redes neurais) também foram utilizados. Esse tipo de modelo costuma resultar em melhores métricas, mas também exige uma quantidade muito grande de dados. Isso por muitas vezes é um problema, pois a maioria dos conjuntos de dados é rotulado por seres humanos que decidem se um texto é ou não sarcástico utilizando seus próprios conhecimentos.

ghosh-veale:2016:fracking-sarcasm-nn utilizam três tipos de redes neurais em conjunto e comparam seu desempenho em um conjunto de dados extraído do Twitter. Eles utilizam *long short term memories* (LSTMs), *convolutional neural networks* (CNNs) e *deep neural networks* (DNNs), combinando esses modelos entre si para conseguir melhores performances.

van-hee-etal:2018:semeval mostram em seu artigo os resultados da terceira tarefa da SemEval-2018, um *workshop* internacional de PLN com o objetivo de avançar as fronteiras do estado-da-arte em determinadas tarefas. Nesse artigo, eles mostram que as melhores soluções (em termos da métrica F_1) propostas pelos times utilizam modelos baseados em aprendizado profundo. Eles fazem principalmente o uso de redes do tipo LSTMs, CNNs e DNNs. Além disso, é presente o uso de *word embeddings* como o GloVe, que retratam cada uma das palavras presentes em um texto como vetores nos quais palavras de sentidos parecidos possuem vetores parecidos.

A tarefa proposta pela SemEval-2018 era dividida em duas sub-tarefas. A primeira era uma detecção de sarcasmo comum, onde bastava o modelo distinguir entre *tweets* sarcásticos e não sarcásticos. A segunda tarefa, entretanto, era mais complexa, pois exigia que o modelo também classificasse o tipo de sarcasmo que ocorria no texto. Os resultados mostraram que mesmo modelos de aprendizado profundo bastante sofisticados ainda estavam bastante longe de resultados promissores. Enquanto na primeira sub-tarefa o melhor time obteve uma métrica F_1 de 0.705, o melhor time na segunda sub-tarefa obteve apenas 0.507 pontos dessa mesma métrica.

2.3.4 Abordagens Baseadas em Contexto

Até então, foram exibidos trabalhos que se restringiram apenas ao texto de interesse. Entretanto, muitos outros trabalhos partem de outra abordagem, que é utilizar informações além do texto para se determinar se um texto é ou não sarcástico, chamadas de contexto.

wallace-etal:2014:ironic-context mostram em seu artigo que humanos obtém melhores resultados para detectar se um texto é ou não sarcástico quando eles recebem contexto além do texto original. Em seu trabalho, eles exploram a rede social Reddit e pedem para seres humanos decidirem se uma postagem nessa rede é ou não sarcástica e qual o seu grau de confiança nessa decisão. O humano pode pedir por contexto quando estiver em dúvida e os autores, então, apresentam qual era o tópico em que a postagem estava inserida e uma lista de outras postagens do mesmo usuário.

Eles então mostram que serem humanos acertam consistentemente mais quando obtém esse contexto extra, além de ficarem mais confiantes em sua decisão. Não só isso, mas os autores também mostram que um modelo *bag-of-words* costuma errar nesses casos em que humanos precisam de contexto extra. Portanto, os autores argumentam que máquinas provavelmente também precisam de contexto extra para conseguir melhores resultados. Como veremos, muitos outros autores criaram modelos que utilizam o contexto de uma postagem e provam empiricamente que o contexto pode ajudar.

Há muitas formas de se adicionar informações extras referentes ao contexto, **hazarika-etal:2018:cascade** criam um método para criar representar *embeddings* de usuários, que capturam o estilo de descrita e personalidade para criar uma representação que pode ser utilizada em adição ao texto escrito por aquele usuário.

wang-etal:2015:context-twitter utilizam contexto da conversa e do tópico, utilizando os tweets que vieram antes do tweet de interesse, outros tweets do mesmo autor e outro tweets que tenham as mesmas *hashtags* do tweet de interesse (ou seja, que pertençam ao mesmo tópico).

O melhor resultado obtido por eles é quando utilizam o histórico de tweets do autor. Eles experimental com o último tweet do autor, os três últimos e os cinco últimos, obtendo o melhores resultados conforme aumentavam esse número. Além disso, utilizar o contexto da conversa (ou seja, o histórico de respostas de que levou até o tweet de interesse) possui também um ganho nas métricas, mas variar entre 1, 3 ou 5 tweets não muda muito. Isso mostra que se o modelo tivesse acesso apenas ao tweet de interesse (que queremos determinar sarcasmo) e o tweet que ele respondeu (o anterior a ele na conversa) já possui a maioria do ganho que obteríamos ao olhar para a conversa. Intuitivamente, isso faz sentido, pois quando se dá uma resposta sarcástica, essa é uma resposta àquilo que foi falado imediatamente antes na vasta maioria dos casos. Por fim, eles mostram que a métrica F_1 cai ao adicionar contexto do tópico e que, portanto, adicionar outros tweets com a mesma *hashtag* do de interesse apenas atrapalha.

ghosh:2018:sarcasm-conversation-context também utilizam contexto da conversa em conjunto com LSTMs e mostram que ao adicionar a postagem que foi respondida pela postagem de interesse melhora os resultados. Eles utilizam vários conjuntos de dados e várias formas diferentes de LSTMs, assim, eles testam diversas formas de se acoplar

o comentário anterior ao de interesse e mostram que adicionar o contexto referente à postagem anterior sempre melhora os resultados.

jena-etal:2020:cnet criam um novo modelo chamado por eles de C-Net que foca também em utilizar o contexto gerado pelos comentários anteriores de um comentário de interesse. Dado que eles tenham $n - 1$ comentários anteriores, eles criam n modelos BERT, um para cada comentário. Portanto, em uma cadeia de n comentários (onde o n-ésimo é o comentário de interesse, o $n - 1$-ésimo é o comentário anterior a ele e assim por diante), o i-ésimo BERT é treinado utilizando apenas o i-ésimo comentário e deve retornar uma porcentagem y_i de que o comentário i leve a uma resposta sarcástica. Depois, eles concatenam esses resultados dos n modelos utilizando uma camada de fusão (do inglês, *fusion layer*) que utiliza algum modelo que recebe n valores e retorna uma probabilidade final y_{n+1} que será utilizada para dizer se o último comentário foi ou não sarcástico. Para fazer essa concatenação, eles utilizam um modelo de séries temporais chamado de *simple exponential smoothing* (SES).

kolchinski-potts:2018:representing-social-media-users exploram outra forma de se criar *embeddings* para os usuário de uma rede social. Para cada autor nos dados de treinamento, ele criam um vetor de 15 valores randômicos. Então esses vetores são passados como entrada para um modelo de rede neural recorrente (RNN) bidirecional e são atualizados durante o treinamento, com o objetivo de aprender características importantes do usuário. Então se aquele usuário aparece novamente na hora de fazer uma predição fora dos dados de treinamento, seu vetor de *embeddings* é utilizado. Contudo, caso o autor não tenha aparecido no treinamento, um vetor randômico é utilizado.

Por fim, há outras duas formas de se adicionar contexto à tarefa. A primeira delas é utilizando *word embeddings*, pois elas permitem aos modelos que as utilizam fazer comparações entre palavras e compreender a semântica passada por elas. Alguns exemplos são `Word2vec` (**mikolov-etal:2013:word2vec**), `GloVe` (**pennington-etal:2014:glove**), `fastText` (**bojanowski-etal:2016:fasttext**), `ELMo` (**peters-etal:2018:elmo**), `BERT` (**devlin-etal:2018:bert**). A segunda é utilizando uma técnica chamada de transferência de aprendizado (ou aprendizado por transferência) (do inglês, *transfer learning*). que consistem de um conjunto de dados realmente grande. Depois disso, os parâmetros do modelo são refinados na tarefa específica. Dentro do contexto de PLN, os modelos mais utilizados são os conhecidos como *transformers*, um tipo de modelo que será explicado no capítulo AAAA. Alguns exemplos são `GPT-3` (**brown-etal:2020:gpt3**), `XLNet` (**yang-etal:2019:xlnet**), `BERT` (**devlin-etal:2018:bert**), `RoBERTa` (**liu-etal:2019:roberta**) e `DeBERTa` (**he-etal:2020:deberta**), que é o foco deste trabalho.

Entre os trabalhos que utilizam *transformers*, um dos mais bem sucedidos é o de **potamias-etal:2020:transform-sarcasm**, que criam um modelo que utiliza um RoBERTa em conjunto com uma LSTM bidirecional (BiLSTM). Os autores argumentam que o RoBERTa é capaz de mapear as palavras para um espaço vetorial rico em semântica e que outros modelos de redes neurais podem usar esses valores para capturar dependências tempo-espaciais entre as palavras. Eles, portanto, removem a última camada do modelo pré-treinado RoBERTa e passam os valores da sua última camada escondida (do inglês, *hidden layer*) para uma BiLSTM que, por sua vez, tem seus resultados concatenados por

uma camada completamente conectada e passados para uma camada *softmax*.