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 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, 2022b; Dicio, 2022a

Joshi *et al.*, 2016

(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 verbais. Wallace *et al.*, 2014

(Humans Require Context to Infer Ironic Intent (so Computers Probably do, too))

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

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 acha que essa sequência contém ou não sarcasmo.

Veale e Hao, 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 e Hao, 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 e Greenwood, 2014 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 *et al.*, 2015 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*.

Referências

[Bharti *et al.* 2015] Santosh Kumar Bharti, Korra Sathya Babu e Sanjay Kumar Jena. "Parsing-based sarcasm sentiment recognition in twitter data". Em: *2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. 2015, pgs. 1373–1380. doi: 10.1145/2808797.2808910 (citado na pg. 8).

[Dicio 2022a] Dicio. *Ironia - Dicio, Dicionário Online de Português*. url: https://www.dicio.com.br/ironia/ (acesso em 19/09/2022) (citado na pg. 3).

[Dicio 2022b] Dicio. *Sarcasmo - Dicio, Dicionário Online de Português*. url: https://www.dicio.com.br/sarcasmo/ (acesso em 19/09/2022) (citado na pg. 3).

[Joshi *et al.* 2016] Aditya Joshi, Pushpak Bhattacharyya e Mark James Carman. *Automatic Sarcasm Detection: A Survey*. 2016. doi: 10.48550/ARXIV.1602.03426. url: https://arxiv.org/abs/1602.03426 (citado na pg. 3).

[Maynard e Greenwood 2014] Diana Maynard e Mark Greenwood. "Who cares about sarcastic tweets? investigating the impact of sarcasm on sentiment analysis." Em: *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*. Reykjavik, Iceland: European Language Resources Association (ELRA), mai. de 2014, pgs. 4238–4243. url: http://www.lrec-conf.org/proceedings/lrec2014/pdf/67_Paper.pdf (citado na pg. 7).

[Veale e Hao 2010] Tony Veale e Yanfen Hao. "Detecting ironic intent in creative comparisons". Em: *Proceedings of the 2010 Conference on ECAI 2010: 19th European Conference on Artificial Intelligence*. NLD: IOS Press, 2010, pgs. 765–770. isbn: 9781607506058 (citado na pg. 7).

[Wallace *et al.* 2014] Byron C. Wallace, Do Kook Choe, Laura Kertz e Eugene Charniak. "Humans require context to infer ironic intent (so computers probably do, too)". Em: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Baltimore, Maryland: Association for Computational Linguistics, jun. de 2014, pgs. 512–516. doi: 10.3115/v1/P14-2084. url: https://aclanthology.org/P14-2084 (citado na pg. 4).