UNIVERSIDADE DE CAXIAS DO SUL
ÁREA DE CONHECIMENTO DE CIÊNCIAS EXATAS E ENGENHARIA
BACHARELADO EM SISTEMAS DE INFORMAÇÃO

GUILHERME HENRIQUE SANTOS ANDREATA

O Uso de Processamento de Linguagem Natural para a Análise de Sentimentos na Rede Social Reddit.

André Luis Martinotto
Orientador

Caxias do Sul, Maio de 2017

O Uso de Processamento de Linguagem Natural para a Análise de Sentimentos na Rede Social Reddit.

por

Guilherme Henrique Santos Andreata

Projeto de Diplomação submetido ao curso de Bacharelado em Sistemas de Informação da área de conhecimento de ciências exatas e engenharia, como requisito obrigatório para graduação.

Projeto de Diplomação

Orientador: André Luis Martinotto
Banca examinadora:

André Gustavo Adami
CCTI/UCS
Carlos Eduardo Nery
CCTI/UCS

Projeto de Diplomação apresentado em
5 de Dezembro de 2013

Daniel Luís Notari
Coordenador

SUMÁRIO

LISTA DE ACRÔNIMOS

NLP	*Natural Language Processing*
NLTK	*Natural Language Toolkit*
MaxEnt	*Maximum Entropy*
RNTN	*Recursive Neural Tensor Networks*
VADER	*Valence Aware Dictionary and sEntiment Reasoner*
JSON	*JavaScript Object Notation*
POJO	*Plain Old Java Objects*

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

Palavras-chave: Kinect, Blender, Animação 3D.

1 INTRODUÇÃO

A linguagem é a forma com que nós nos comunicamos, seja ela escrita ou falada. De fato, a linguagem é a forma como expressamos nossas idéias, sentimentos e experiências. O Processamento de Linguagem Natural, é o termo utilizado para descrever um software ou componente de hardware que tem como função analisar a linguagem escrita ou falada (JACKSON; MOULINIER, 2007).

Existem duas abordagens de Processamento de Linguagem Natural, sendo que a primeira delas é chamada de simbólica ou racionalista e a outra de empírica. A primeira abordagem consiste em uma série de regras para a manipulação de símbolos, como as regras gramaticais que permitem identificar se uma frase está malformada ou não. A abordagem empírica está centrada na análise estatística da linguagem através de grandes quantidades de textos, como por exemplo, a utilização de modelos de Markov para reconhecer padrões na escrita (JACKSON; MOULINIER, 2007).

Existem diversos *frameworks open source* que facilitam o desenvolvimento de *softwares* para o Processamento de Linguagem Natural, sendo que entre esses destacam-se o *Stanford's Core NLP Suite* (Stanford CoreNLP, 2017), *Natural Language Toolkit* (Natural Language Toolkit, 2017), *Apache OpenNLP* (Apache OpenNLP, 2017) e *Spacy* (Spacy, 2017). Esses *frameworks* nos permitem, entre outras coisas, efetuar análise de sentimentos, identificar tópicos e conteúdos.

A rede social Reddit é o vigésimo terceiro *website* mais acessado na internet e o sétimo mais acessado nos Estados Unidos da América (Alexa, 2017). Através deste *website*, seus usuários podem criar ou se inscrever em comunidades, também conhecidas como *subreddits*. Uma vez que as comunidades são criadas pelos próprios usuários, podemos encontrar comunidades sobre todos os assuntos, sejam notícias do mundo, comunidades partidárias, comunidades criadas para pessoas de uma mesma localidade, comunidades de imagens engraçadas, etc.

Nestas comunidades é possível visualizar e comentar *links* enviados por outros usuários. Além disso, o usuário pode efetuar um voto de forma positiva, caso acredite que aquele *link* é útil para a comunidade. Caso contrário, é possível efetuar um voto negativo. Uma vez que os próprios usuários podem submeter *links*, os eventos e

notícias de todo o mundo são reportados no *website*, como exemplo, pode-se citar as eleições ocorridas no ano de 2016 nos Estados Unidos e o tiroteio ocorrido em Paris.

Neste trabalho será desenvolvido um software que permita realizar a análise dos comentários do *website* Reddit. Mais especificamente os comentários do Reddit serão analisados com o objetivo de identificar padrões de sentimentos, ou seja, determinar se a opinião expressada com relação a um determinado tópico é neutra, positiva ou negativa.

1.1 Objetivos do Trabalho

Este trabalho tem como objetivo a análise dos comentários disponíveis no *website* Reddit, identificando padrões de sentimentos entre os usuários de suas comunidades. De forma a atingir o objetivo principal desse trabalho, os seguintes objetivos específicos devem ser realizados:

- Desenvolver uma ferramenta para o Processamento Natural de Linguagem através de *frameworks* já existentes.
- Construção de uma base de dados a partir do *website* Reddit.
- Efetuar o processamento da base de dados utilizando a ferramenta desenvolvida.

1.2 Estrutura do Trabalho

2 PROCESSAMENTO DE LINGUAGEM NATURAL

O objetivo da área de Processamento de Linguagem Natural é analisar a linguagem natural, ou seja, a linguagem utilizada pelo seres humanos não importando se essa é escrita ou falada (MANNING; SCHüTZE, 1999).

O Processamento de Linguagem Natural é uma área antiga, sendo anterior a invenção dos computadores modernos. De fato, sua primeira grande aplicação foi um dicionário desenvolvido no Birkbeck College em Londres no ano de 1948. Por ser uma área complexa, seus primeiros trabalhos foram notavelmente falhos o que causou uma certa hostilidade por parte das agências formentadoras de pesquisas.

Os primeiros pesquisadores eram muitas vezes bilíngues, como por exemplo, nativos alemães que imigraram para os Estados Unidos. Acreditava-se que pelo fato desses terem conhecimento de ambas as linguas, Ingles e Alemão, eles teriam capacidade de desenvolver programas de computadores que efetuariam a tradução das linguas de modo satisfatório. Uma vez que esses encontraram muitas dificuldades, ficou claro que o maior problema não era o conhecimento de ambas as línguas e sim como expressar esse conhecimento na forma de um programa de computador (HANCOX, 2017).

Para que um computador seja capaz de interpretar uma língua, necessitamos antes entender como nós efetuamos essa interpretação. Por isso, uma parte considerável do Processamento de Linguagem Natural está apoiado na área de Linguística.

2.1 Linguística

O objetivo da Linguística é compreender como os humanos adquirem, produzem e entendem as diversas línguas, ou seja, a forma com que conversamos, a nossa escrita e outras mídias de comunicação (MANNING; SCHüTZE, 1999).

Na linguagem tanto escrita, como na falada, existem regras que são utilizadas para estruturar as expressões. Uma série de dificuldades no Processamento de Linguagem Natural são ocasionadas pelo fato de que as pessoas constantemente mudam essas regras para satisfazerem suas necessidades de comunicação (MANNING;

SCHüTZE, 1999). Uma vez que as regras são constantemente modificadas pelo loucutor, se torna extremamente difícil a criação de um software ou hardware efetue a interpretação de uma língua.

2.2 Métodos de Processamento de Linguagem Natural

O *Natural Language Processing* (NLP) tem como objetivo a execução de diferentes tarefas, como por exemplo, a categorização de documentos, a tradução e a geração de textos a partir de um banco de dados, etc. Podemos citar duas classes de métodos para a execução deste tipo de tarefas, que são os métodos simbólicos e os métodos estatísticos.

Nos final dos anos 50 e 60, existiam excelentes métodos esatatísticos, que foram desenvolvidos durante a segunda guerra mundial, para a solução de problemas científicos (SHANNON; WEAVER, 1948). Porém, no ano de 1957, Chomsky publicou o trabalho intitulado de *"Syntactic Structures"* onde descreve a teoria de gramática gerativa, que considera a gramática como um conjunto de regras. Essa abordagem através de um conjunto de regras, ao invés de um modelo matemático, entra em conflito com os trabalhos anteriores, criando duas comunidades no campo de Linguística. Como reflexo dessas duas comunidades, a área de NLP que crescia em paralelo a linguística também foi dividida em duas áreas. A primeira dessas áreas que fazia uso de métodos baseados em regras (simbólico) e a segunda que fazia o uso de métodos quantitativos (estatísticos).

Nesta seção será apresentado um exemplo de método simbólico e de um método estatístico. Destaca-se que essa descrição apresenta como objetivo somente diferenciar ambas as classes de métodos, através de seus requisitos e forma de execução. Destaca-se ainda que os métodos apresentados não são utilizados na análise de sentimentos, sendo que os métodos específicos que são utilizados para identificação de sentimentos serão descritos no Capítulo 3.

2.2.1 Método Simbólico

O método simbólico ou racionalista está baseado no campo da Linguística e faz o uso da manipulação dos símbolos, significados e das regras de um texto. Um exemplo de método simbólico é o método de Brill (BRILL, 1992). Por exemplo, no método de Brill a frase "João pintou a casa de branco", será separada em palavras que serão classificadas através de um dicionário pré-definido, como:

Palavra	João	pintou	a	casa	de	branco
Classificação:		Verbo	Artigo	Substantivo	Preposição	Adjetivo

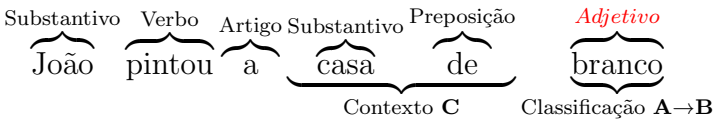
Observa-se que algumas palavras não foram identificadas, como "João" ou clas-

sificadas de forma incorreta, como “branco”. Desta forma, o método utiliza-se de outras duas regras para a classificação. A primeira regra classifica todas as palavras desconhecidas que iniciam em maiúsculo como substantivos, por exemplo, a palavra “João”. Já a segunda regra, atribui para a palavra desconhecida a mesma classificação de outras palavras que terminam com as mesmas três letras. Por exemplo, supondo que a palavra “pintou” não fosse encontrada no dicionário, essa seria associada a outras palavras terminadas com o sufixo “tou”, ou seja, essa seria classificada como verbo.

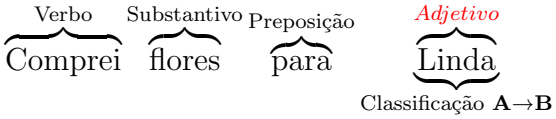
Palavra	João	pintou	a	casa	de	branco
Classificação:	**Substantivo**	Verbo	Artigo	Substantivo	Preposição	Adjetivo

Após essa classificação inicial, o método executa o seguinte conjunto de regras, ou ainda, regras derivadas dessas:

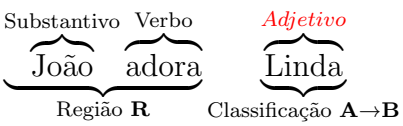
- Se uma palavra tem a classificação **A** e está no contexto **C** então a sua classificação deverá ser mudada para **B**. Por exemplo, se uma palavra **A** (branco no exemplo) é um adjetivo e uma das duas palavras anteriores é uma preposição (“de” no contexto **C**), mude para sua classificação para substantivo (classificação **B**).



- Se uma palavra tem a classificação **A** e tem uma propriedade **P** então a sua classificação deverá ser alterada para **B**. Por exemplo, se uma palavra **A** (“Linda”) foi classificada como um adjetivo e é iniciada com uma letra maiúscula (propriedade **P**), sua classificação deverá ser alterada para substantivo (classificação **B**).



- Se uma palavra tem a classificação **A** e uma palavra com a propriedade **P** está na região **R**, sua classificação deverá ser **B**. Por exemplo, se uma das duas palavras anteriores (“João adora”na região **R**) iniciam com letra maiúscula (propriedade **P**), sua classificação deverá ser alterada para substantivo (classificação **B**).



2.2.2 Método Estatístico

Um método estatístico utiliza-se de uma grandes quantidade de texto, procurando por padrões e associações a modelos, sendo que esses padrões podem ou não estar relacionados com regras sintáticas ou semânticas.

Os métodos estatísticos baseia-se na utilização de um sistema de aprendizado supervisionado, ou seja, a classificação é feita a partir de um conjunto de dados já classificado, que é chamado de *training set*. Um exemplo de método estatístico é a utilização de Modelos de Markov com a aplicação do algoritmo de Viterbi (MANNING; SCHüTZE, 1999).

Em um Modelo de Markov, a classificação da frase "João comprou um carro"é feita a partir de um *training set* que pode, por exemplo, ser composto por textos retirados de *web-sites*, sendo que as palavras destes textos já devem estar classificadas. A partir deste *training set*, as palavras "João", "comprou" e "carro" seriam classificadas como substantivo, verbo e substantivo, respectivamente. Já a palavra "um" apresenta uma ambiguidade uma vez que pode ser classificada como um artigo (ART), ou um substantivo (SM) ou um pronome (PRO). A Figura 2.1 representa o conjunto de possibilidades que o classificador pode utilizar para a classificação completa da frase.

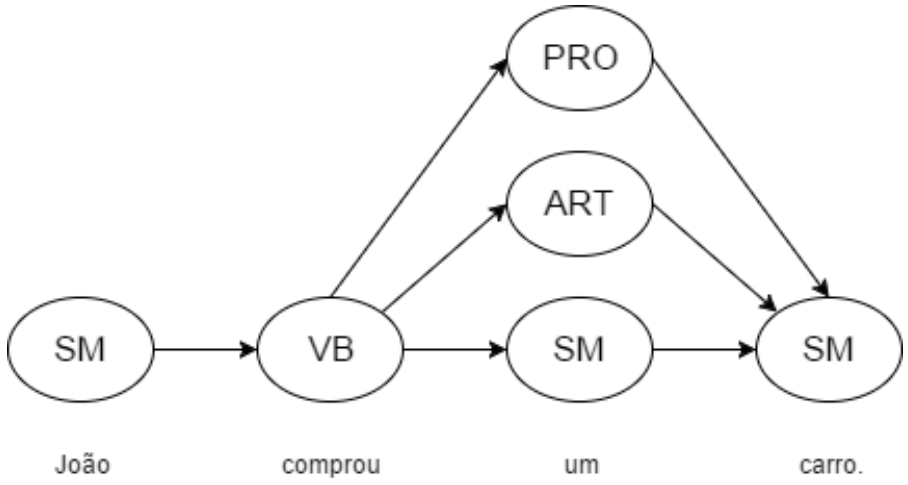


Figura 2.1: Caminhos possíveis de classificação

A idéia central da utilização de Modelos de Markov é escolher, entre esses caminhos (Figura 2.1), o caminho de maior probabilidade. Para tanto, se faz necessário calcular a probabilidade de todos os caminhos seguindo o Modelo de Markov e após este cálculo, utilizar o Algoritmo de Viterbi para definir qual caminho é o mais provável.

O Modelo de Markov irá utilizar-se do *training set* para inferir a classificação da palavra "um". Para este exemplo, vamos considerar um *training set* hipotético com as seguintes características: 10000 Substantivos aonde 150 são a palavra "um",

10 são a palavra “João” e 50 são a palavra “carro”, 20000 Artigos aonde 500 são a palavra “um”, 12000 Verbos aonde 50 são a palavra “comprou” e 15000 Pronomes aonde 50 são a palavra “um”. Neste caso, a probabilidade da palavra ser um substantivo é dada pela equação:

$$P(palavra|classe) = \frac{C(classe, palavra)}{C(classe)}$$

$$P(um|SM) = \frac{C(SM, um)}{C(SM)} = \frac{150}{10000} = 0.015$$

Da mesma forma, é calculada a probabilidade de “um” pertencer as demais classes, neste caso, pronome ou artigo. Por exemplo, a probabilidade da palavra “um” ser um pronome seria 0.0033 e a probabilidade da palavra “um” ser um artigo seria 0.025. Esse cálculo é realizado para todas as palavras da frase que está sendo classificada e todas as classes possíveis. A Tabela 2.1 apresenta o cálculo realizado anteriormente para as demais palavras e classes.

	João	comprou	um	carro
Substantivo	0.001	0	0.015	0.005
Verbo	0	0.0042	0	0
Artigo	0	0	0.025	0
Pronome	0	0	0.0033	0

Tabela 2.1: Tabela de Probabilidades de Associação

Além da probabilidade de associação a uma determinada classe, é calculada a probabilidade de transição de uma classe para a outra. Neste caso, o *training set* hipotético apresenta as seguintes características:

- De 20000 frases, 2500 iniciam com um substantivo, 5000 iniciam com um verbo, 5000 iniciam com um artigo e 5000 iniciam com um pronome.
- De 10000 substantivos, 10000 são seguidos por verbos.
- De 12000 verbos, 3000 são seguidos por um substantivo, 2000 são seguidos por um outro verbo, 5000 são seguidos por um artigo e 2000 são seguidos por um pronome.
- De 20000 artigos, 20000 são seguidos por um substantivo.
- De 15000 pronomes, 10000 são seguidos por um substantivo e 5000 são seguidos por um verbo.

Neste caso, a probabilidade de transição de um verbo para um substantivo é dada pela equação:

$$P(transicao|classe) = \frac{C(classe, transicao)}{C(classe)}$$

$$P(SM|VB) = \frac{C(VB, SM)}{C(VB)} = \frac{3000}{12000} = 0.25$$

Da mesma forma, a probabilidade de transição é cálculada para todas as demais classes. Por exemplo, a probabilidade de transição de um verbo para outro verbo é 0.17, de um verbo para um artigo é 0.42 e de um verbo para um pronome é 0.17. Também, é utilizada a mesma equação para o cálculo da probabilidade da frase iníciar com determinada classe. A Tabela 2.2 tem-se a probabilidade de transição onde foi feito o cálculo anterior para todas as classes possíveis para classificação do nosso *training set*.

	Substantivo	Verbo	Artigo	Pronome
Início	0.125	0.25	0.25	0.25
Substantivo	0.0	1.0	0.0	0.0
Verbo	0.25	0.17	0.42	0.17
Artigo	1.0	0.0	0.0	0.0
Pronome	0.67	0.33	0.0	0.0

Tabela 2.2: Tabela de Probabilidade de Transição

A partir das probabilidades anteriores cálculadas através do Modelo de Markov, é utilizado o algoritmo de Viterbi para determinar o caminho mais provável. Esse cálculo é feito começando pelo início da frase, calculada através de:

$$v_t(j) = v_{t-1} a_{ij} b_j(o_t)$$

Aonde que v_t é a probabilidade do caminho atual, v_{t-1} é o resultado do caminho anterior, a_{ij} é a probabilidade de transição e $b_j(o_t)$ é a probabilidade de associação.

Portanto a palavra "João", v_{t-1} é representada pelo valor 1, visto que essa é a primeira palavra e não foram feitos cálculos anteriores, a_{ij} é a probabilidade de transição entre "Início" e um substantivo, disponível na tabela 2.2 e $b_j(o_t)$ é a probabilidade de associação da palavra João com substantivo, disponível na tabela 2.1:

$$v_t(j) = 1 * 0.125 * 0.001 = 0.000125$$

Para "comprou", além dos valores retirados das tabelas 2.2 e 2.1, v_{t-1} é representado pelo cálculo do caminho anterior, ou seja, 0.000125:

$$v_t(j) = 0.000125 * 1 * 0.0042 = 0.000000525$$

Ao efetuar o cálculo de todos os caminhos, para determinar qual a classificação correta de uma palavra, é escolhido o caminho que tem maior probabilidade, no caso apresentado, a palavra "um" é classificada como artigo.

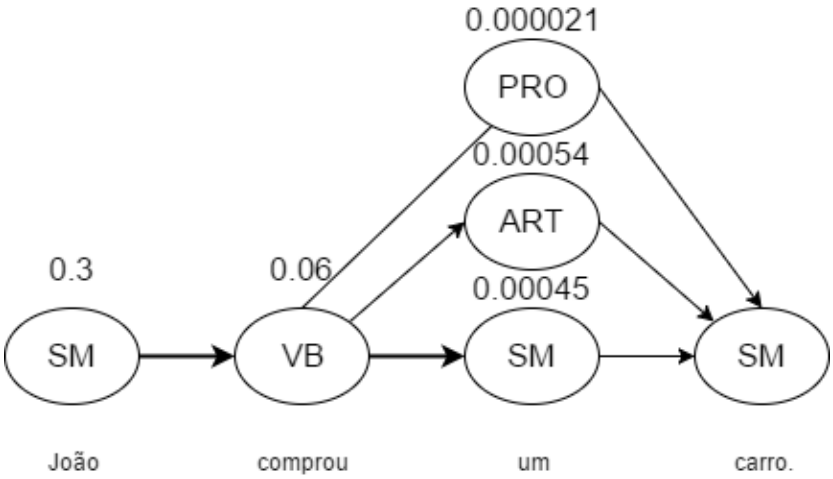


Figura 2.2: Caminhos já decididos de classificação

Como visto, o método simbólico para resolver problemas de Processamento de Linguagem Natural faz uso da criação de regras baseadas no conhecimento humano, enquanto o método estatístico, decide através de cálculos probabilísticos apoiados em estatísticas de um banco de dados para a resolução correta do problema.

3 MÉTODOS ESTATÍSTICOS E SIMBÓLICOS APLICADOS NA ANÁLISE DE SENTIMENTOS

3.1 Naive Bayes

O classificador Naive Bayes é um classificador baseado no teorema de Bayes com independencia entre seus atributos.

O teorema de Bayes é representado da seguinte forma:

$$P(c|d) = \frac{P(d|c)P(c)}{P(d)}$$

Supondo que precisamos determinar se o carro que João comprou na frase "João comprou um Focus." é o modelo sedan ou hatch.

- P(c|d) é a probabilidade de **d** pertencer a classe **c**. Ou seja, a probabilidade do carro Focus ser um sedan.
- P(d|c) é a probabilidade da classe **c** ser **d**. Ou seja, dentre todas as sedans, a probabilidade de um sedan ser um Focus.
- P(c) é a probabilidade da classe **c**. Ou seja, a frequência que sedans aparecem no nosso banco de dados.
- P(d) é a probabilidade de **d**. Ou seja, a frequência que Focus aparecem no nosso banco de dados.

Levando em consideração que temos o banco de dados representado pela tabela abaixo:

Probabilidade do Focus ser sedan:

$$P(Sedan|Focus) = \frac{P(Focus|Sedan)P(Sedan)}{P(Focus)}$$

$$P(Sedan|Focus) = \frac{2/3 * 3/10}{5/10} = \frac{0,2}{0,5} = 0,4$$

Probabilidade do Focus ser um hatch:

Carro	Categoria
Focus	Sedan
Gol	Hatch
Focus	Hatch
Focus	Sedan
Focus	Hatch
Fox	Hatch
Fiesta	Hatch
Cruze	Sedan
Focus	Hatch

Tabela 3.1: Tabela de Carro e Categoria.

$$P(Hatch|Focus) = \frac{P(Focus|Hatch)P(Hatch)}{P(Focus)}$$

$$P(Hatch|Focus) = \frac{3/6 * 7/10}{5/10} = \frac{0,35}{0,5} = 0,7$$

No caso utilizado como exemplo, o Focus(Atributo ou *Feature*) é um hatch (Rótulo ou *Label*). Porém, caso tenhamos mais um atributo para utilizar na classificação o classificador Naive Bayes não considera nenhuma dependência. Como por exemplo a frase "O carro era ano 2010 e tinha 2 portas" com os seguintes atributos:

Carro	Ano	Portas	Categoria
Focus	2010	2	Sedan
Gol	2010	4	Hatch
Focus	2011	4	Hatch
Focus	2011	2	Sedan
Focus	2011	2	Hatch
Fox	2012	4	Hatch
Fiesta	2012	2	Hatch
Cruze	2013	4	Sedan
Focus	2013	2	Hatch

Tabela 3.2: Tabela de anos, carros, portas e categorias.

O cálculo será feito da seguinte forma:

$$P(d|c) = P(d_1|c) * P(d_2|c) * \ldots * P(d_n|c)$$

$$P(Sedan|Focus) = P(Portas = 2|Sedan) * P(Ano = 2010|Sedan) \ldots *$$

$$P(Sedan|Focus) = 2/3 * 1/3 \ldots *$$

Por assumir independência entre seus atributos, os valores obtidos nós cálculos podem ser armazenados no banco de dados e reaproveitados. Por isso, sua performance é considerada incrívelmente boa até mesmo para casos aonde temos forte dependência de atributos (DOMINGOS; PAZZANI, 1997).

3.2 *Maximum Entropy*

O classificador *Maximum Entropy* (MaxEnt) tem como característica principal a preferência por modelos de dados uniformes sem efetuar nenhuma suposição injustificada.

Podemos utilizar um exemplo similar ao anterior para demonstrar a lógica do classificador MaxEnt.

"João comprou um carro.".

Supondo que temos que classificar o tipo de carro comprado em três categorias:

- Hatch.
- Sedan.
- Cupê.

Podemos afirmar que:

$$P(Hatch) + P(Sedan) + P(Cupe) = 1$$

Como só temos essas três possibilidades de classificação no nosso exemplo, o carro só pode ser classificado em uma dessas três possibilidades, ou seja, a soma das três probabilidades deve ser 100% ou 1 essa é a primeira restrição ou *constraint*. Abaixo duas tabelas que satisfazem essa restrição:

Tipo	%
Sedan	33%
Hatch	33%
Cupê	33%

Tabela 3.3: Tabela de Probabilidades A

Tipo	%
Sedan	50%
Hatch	50%
Cupê	0%

Tabela 3.4: Tabela de Probabilidades B

Sem nenhum conhecimento prévio da distribuição desses carros, ou seja, a quantidade de carros comprados por tipo, o classificador assume uma distribuição uniforme das probabilidades, portanto, com maior entropia.

- Hatch - 33%.
- Sedan - 33%.
- Cupê - 33%.

Agora, supondo que a partir do nosso banco de dados conseguimos verificar que em 80% dos casos o veículo comprado era um sedan ou hatch, temos uma nova restrição:

$$P(Hatch) + P(Sedan) = 0.8$$

Podemos novamente ter n distribuições diferentes, porém a distribuição mais uniforme que satisfaz as nossas duas restrições são:

- Hatch - 40%.
- Sedan - 40%.
- Cupê - 30%.

Esse é o princípio da Máxima Entropia utilizado nessa forma de classificação. Primeiro é descoberta a frequência de cada atributo, depois é procurada a distribuição que máximiza a entropia, ou seja, a mais uniforme.

3.3 *VADER*

O *Valence Aware Dictionary and sEntiment Reasoner* (VADER) é um dicionário e classificador de sentimentos que se baseia em regras, portanto, um método de classificação simbólico. Ele é especialmente ajustado para funcionar em redes sociais aonde temos um contexto vago e pouca quantidade de texto, nesse contexto, ele é extremamente eficaz, podendo se comparar a classificação feita por humanos (HUTTO; GILBERT, 2014).

Esse método faz uso de um dicionário que foi construído levando em consideração gírias e emoticons utilizados em redes sociais. Neste dicionário as palavras estão previamente associadas a uma polaridade de sentimento (positivo e negativo) e intensidade em uma escala de -4 até +4, como por exemplo, a palavra *great* tem a intensidade de 3.1 e *horrible* -2.5. Essa associação foi construída utilizando o método de *"wisdom of the crowd"* aonde um grupo de pessoas atribuiu os valores para cada palavra ao invés de somente uma pessoa especializada ou uma classificação automática através de estatística.

Ele faz uso de cinco regras gerais:

- Pontuação. O ponto de exclamação (!) aumenta a magnitude da intensidade sem modificar a orientação semântica. Como por exemplo, *"This place is great!!!"* é mais intenso que *"This place is great"*.
- Capitalização. Especificamente, uma palavra que é relevante para a análise de sentimentos, quando essa é escrita em letras maiúsculas, é aumentada a magnitude da intensidade do sentimento sem modificar a orientação semântica. Como por exemplo, na frase *"This place is GREAT"*, temos a palavra *"GREAT"* (Ótimo) que está relacionada com o sentimento positivo. Neste caso aonde ela está escrita em letras maiúsculas, ela é mais intensa que *"This place is great"*.
- Advérbios intensificadores. Estes impactam a intensidade do sentimento aumentando ou diminuindo a intensidade do sentimento. Na frase *"This place is extremelly good"* o advérbio *extremelly* (extremamente) aumenta a intensidade do sentimento expresso pela frase (*good* ou bom), enquanto na frase *"This place is marginally good"*, a palavra *"marginally" ou marginalmente* acaba diminuindo a intensidade do sentimento expresso.
- A palavra *"but"*. Essa palavra indica uma troca no sentimento da frase expressa aonde que o texto seguinte a ela expressa um sentimento mais dominante. Por exemplo, a frase *"This place is great but today, the service was horrible"* convém um sentimento misto.
- Por fim, ao examinar as três palavras anteriores, o método consegue identificar 90% dos casos aonde uma negação inverte a polaridade de um texto. Como por exemplo, na frase *"This place isn't that great"*, a palavra *great* demonstra um sentimento positivo, porém, ao analisar as três palavras anteriores *"place isn't that"* encontramos uma negação, mudando o sentimento expresso da frase de positivo para negativo.

4 FRAMEWORKS

4.1 Natural Language Toolkit

O *Natural Language Toolkit* (NLTK) é um *Framework* para Python criado em 2001 na Universidade de Pensilvânia. Ele contém mais de 50 dicionários e modelos já treinados incluindo:

- *Sentiment Polarity Dataset Version 2.0* - Conjunto de dados já classificados que contém mais de 1000 filmes avaliados de forma positiva e 1000 filmes avaliados de forma negativa.
- *SentiWordNet* - Provém um dicionário com as palavras extraídas do WordNet já classificadas em positividade, negatividade e objetividade.
- *VADER Sentiment Lexicon* - Dicionário especificamente ajustado para análise de sentimentos expressos em mídias sociais.

4.1.1 Análise de Sentimentos

Para a análise de sentimentos, o NLTK já possui implementado os três classificadores citados anteriormente, *Naive Bayes*, MaxEnt e também VADER.

Podemos utilizar o classificador Naive Bayes a partir da classe **nltk.classify.naivebayes.NaiveBayesClassifier** através dos seguintes métodos:

- *classify(featureset)* - Classifica a partir de um conjunto de atributos.
- *most_informative_features(n=100)* - A partir de um classificador treinado, retorna os atributos mais relevantes.
- *train(trainingset)* - Treina um classificador a partir de um *training set*.

Podemos utilizar o classificador MaxEnt a partir do módulo **nltk.classify.maxent** através dos seguintes métodos:

- *train(train_toks, algorithm=None, trace=3, encoding=None, labels=None, gaussian_prior_sigma=0, **cutoffs)* - Treina um classificador MaxEnt a partir de um *training set*.

  - *train_toks* - *Training set.*
  - *algorithm* - Algorítmo a ser usado para treinar o classificador.
  - *trace* - Nível de detalhe utilizado no log.
  - *encoding*
  - *labels* - Uma lista de possíveis rótulos, se nenhuma for especificada, todos os labels do *training set* serão utilizados.
  - *gaussian_prior_sigma=0* - Somente utilizado no LM-BFGS.
  - *cutoffs* - Argumentos que especificam condições em que o processo será terminado.

- *classify(featureset)* - Classifica a partir de um conjunto de atributos.
- *explain(featureset, columns=4)* - Mostra uma tabela demonstrando os efetiso de cada atributo e como eles combinam para determinar a probabilidade de cada rótulo.
- *show_most_informative_features(n=10, show='all')* - A partir de um classificador treinado, retorna os atributos mais relevantes.

Para utilização do VADER é utilizada a classe *SentimentIntensityAnalyzer* do módulo *vaderSentiment* através do método *polarity_scores*. Este método recebe uma frase e retorna um objeto contendo a intensidade positiva, neutra e negativa da frase.

O *framework* também contém um pacote contendo classes úteis para a análise de sentimentos chamado de *nltk.sentiment*. Nesse pacote temos os seguintes módulos:

- Classe *nltk.sentiment.sentiment_analyzer.SentimentAnalyzer* - Ferramentas para facilitar e implementar análise de sentimentos, especialmente para demonstrações e ensino.
- Módulo *nltk.sentiment.util* - Contém diversas classes de demonstrações e utilitários como conversão de *json* para *csv*.

4.2 Stanford CoreNLP

O Stanford CoreNLP é um conjunto de ferramentas escrito em Java para processamento de linguagem natural. Dentre essas ferramentas, estão incluídos: *Part-of-Speech Tagging* ou classificação gramatical, reconhecimento de entidade e análise de sentimentos. Também possui suporte a diversas linguas além do inglês, como: árabe, chinês, francês, alemão e espanhol.

4.2.1 Análise de Sentimentos

A análise de sentimentos do Stanford CoreNLP é realizada através de um novo modelo de rede neural construído em cima de estruturas gramaticais chamado de

Recursive Neural Tensor Networks (RNTN). Seu modelo é treinado a partir do *Sentiment Treebank*, um banco de dados que possui 215.154 orações distribuidas em 11.855 árvores de frases com sentimentos já classificados.

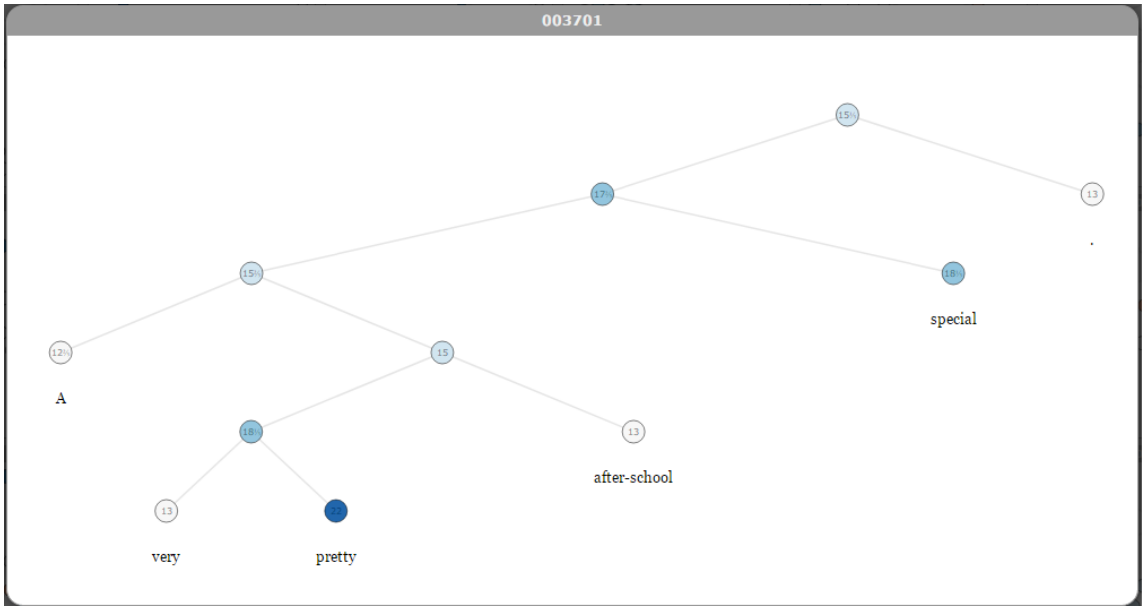


Figura 4.1: Frase já classificada disponível no Sentiment Treebank

A sua utilização pode ser feita de diversas formas, como linha de comando, através de um servidor *web* e através de sua API java:

```
1  public static void main(String[] args) throws IOException {
2          String text = "This World is an amazing place";
3          Properties props = new Properties();
4          props.setProperty("annotators", "sentiment");
5          StanfordCoreNLP pipeline = new StanfordCoreNLP(props);
6
7          Annotation annotation = pipeline.process(text);
8          List<CoreMap> sentences = annotation.get(CoreAnnotations.SentencesAnnotation.class);
9          for (CoreMap sentence : sentences) {
10             String sentiment = sentence.get(SentimentCoreAnnotations.SentimentClass.class);
11             System.out.println(sentiment + "\t" + sentence);
12         }
13 }
```

Figura 4.2: Exemplo de implementação

Como resultado, o console java irá imprimir que a frase é muito positiva ou *Very positive*.

5 REDE SOCIAL REDDIT

O *website* Reddit teve seu início em 2005 como um agregador de conteúdo e atualmente é o vigésimo terceiro *website* mais acessado na internet e o sétimo mais acessado nos Estados Unidos da América (Alexa, 2017). Seus usuários podem enviar links para conteúdos externos ao Reddit ou também mensagens de texto. A partir desse conteúdo enviado, seja ele uma mensagem de texto no próprio Reddit quanto um link a um *website* externo, seus usuários podem votar para cima (*upvote*) ou para baixo *downvote*, influenciando a sua posição no *website*. Esse algoritmo de ordenação de conteúdo é fechado portanto não está disponível para consulta. Além de votar no conteúdo, seus usuários podem enviar comentários como forma de expressar sua opinião.

Esse conteúdo é distribuido em *subreddits* que funcionam como comunidades que abordam certos assuntos. Os usuários podem se inscrever nesses *subreddits* para que seu conteúdo apareça na página inicial. Dentre os *subreddits* mais notáveis se encontram:

- */r/AskReddit* - Local para fazer perguntas gerais para outros usuários. Atualmente com 16.941.544 de inscritos.
- */r/worldnews* - Notícias do mundo. Atualmente com 16.570.606 de inscritos.
- */r/IAmA* - IAmA é um estilização de 'I am a' ou 'Eu sou um'. Local aonde os usuários podem fazer perguntas e respostas ao criador do tópico que se identifica por algo notável, como uma profissão ou algum feito. Atualmente com 16.941.544 de inscritos.

Dentre esses *subreddits* podemos destacar alguns dos tópicos mais acessados no ano de 2016:

- */r/IAmA - We're NASA scientists & exoplanet experts. Ask us anything about today's announcement of seven Earth-size planets orbiting TRAPPIST-1!* - Tópico de perguntas e respostas com cientistas da NASA após a descoberta dos planetas que orbitavam a estrela TRAPPIST-1.

- */r/IAmA - I'm Bill Gates, co-chair of the Bill & Melinda Gates Foundation. Ask Me Anything.* - Tópico de perguntas e respostas com Bill Gates.
- */r/worldnews - Fidel Castro is dead at 90.* - Link para anúncio da morte de Fidel Castro.
- */r/AskReddit - [Serious]South Koreans of Reddit, how did they teach you about the existence of North Korea in School when you were young?serious replies only* - Tópico perguntando para os usuários sul coreanos como que foi ensinado para eles sobre a existência da Coreia do Norte.

5.1 API

O *website* possui uma API *open source* localizada em https://github.com/reddit/. Sua documentação é gerada de forma automática a partir do código fonte e podemos encontrar ela em: https://www.reddit.com/dev/api/.

6 EXTRAÇÃO DE DADOS

Para a extração dos dados para a análise de sentimentos foi criado um *crawler* ou robô de navegação. Esse robô tem como objetivo a navegação automática no conteúdo web do Reddit, extraindo os dados referentes a tópicos e a comentários e persistindo esses em um banco de dados.

6.1 *Crawler*

O *Crawler* foi escrito na linguagem Java por se tratar de uma linguagem com uma grande quantidade de bibliotecas disponíveis e também sua facilidade de implementação. A Figura 6.1 representa a arquitetura utilizada para desenvolvimento deste software.

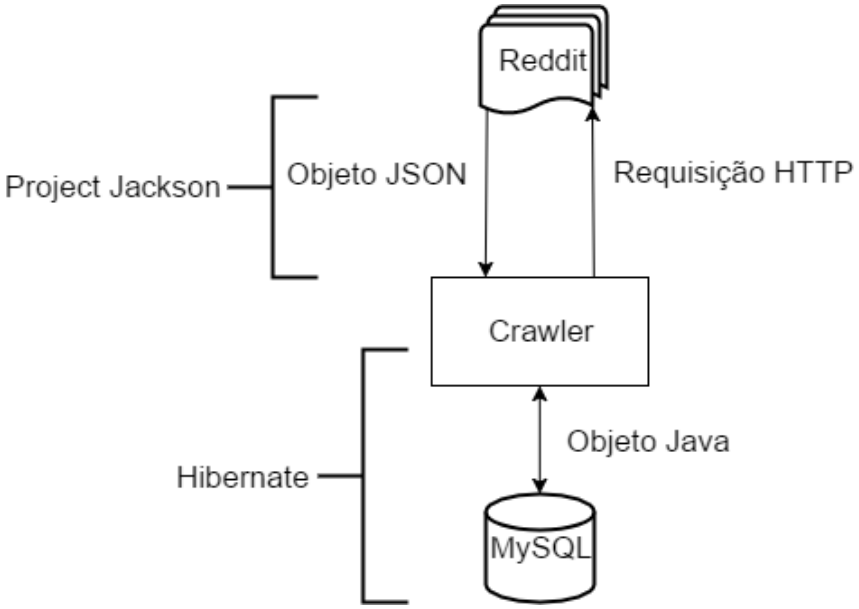


Figura 6.1: Arquitetura do *Crawler*

A partir de um *link* para um tópico, o software tem como tarefa, a extração e pesquisa de dados relacionados com o tópico em questão. Isso se faz da seguinte forma, primeiramente, é enviada uma requisição para o *link* utilizando o sufixo

".json", de forma demonstrada na API, a partir dessa requisição, o *website* retorna um objeto *JavaScript Object Notation* (JSON).

Como o JSON possui 68 campos que assim como seus tipos de dados, não se encontram em nenhuma documentação, foi utilizado o *website* jsonschema2pojo[1] para mapear o JSON retornado em um objeto Java. Esse *website* tem como objetivo a conversão de um esquema JSON ou o próprio JSON para um *Plain Old Java Objects* (POJO) ou Os Singelos Clássicos Objetos Java, permitindo o *download* da classe para utilização, já com as anotações "@JsonProperty" utilizadas na biblioteca Jackson. Este objeto disponibilizado a partir do *website* foi renomeado para *RedditPost* e adicionado ao código fonte da aplicação.

A anotação "@JsonProperty" é relativa ao mapeamento do objeto Java com relação ao JSON e nos permite com intermédio da classe *Object Mapper* instanciar o objeto *RedditPost* a partir de um objeto *String* em formato JSON.

```
RedditPost post =
    objectMapper.readValue(iteratorPost.get("data").toString(),
RedditPost.class);
```

Utilizando o método *readValue* que tem como retorno um *Object*, é informado um objeto *String*, neste caso *"iteratorPost.get("data").toString()"* e uma classe mapeada, neste caso *RedditPost*.

Porém, para o correto funcionamento, deve ser feita a seguinte mudança: O campo *edited* representando se foi editado o comentário, retornado no JSON apresenta um tipo de dado ambíguo aonde que caso o comentário não tenha sido editado, ele apresenta o valor *booleano* de *false*, porém, ao ter sido editado, ele apresenta seu valor em um formato decimal. Este e demais objetos que apresentavam o tipo de dado de forma ambígua foram transformados em objetos *String*.

A partir deste objeto Java, foi utilizado o *framework* Hibernate para a criação do banco de dados, assim como persistência destes. Através da anotação "@Entity", adicionada também na classe RedditPost. O Hibernate mapeia essa entidade junto ao banco de dados, neste caso, MySQL.

Para criação das tabelas do banco de dados, foi utilizada a propriedade *hibernate.hbm2ddl.auto* do Hibernate. Essa propriedade quando instânciada uma nova sessão do *framework* no Java executa as seguintes ações dependendo de seus valores informados:

- *validate*: Não efetua mudanças no banco de dados, somente valida.
- *update*: Atualiza o esquema do banco de dados conforme os objetos mapeados na camada Java.

[1] http://www.jsonschema2pojo.org/

- *create*: Cria o esquema contendo tabelas e campos a partir dos objetos Java, destruindo dados anteriores.
- *create-drop*: Cria o esquema da mesma que o *create*, porém, ao termino da sessão, remove o esquema criado.

No primeiro momento, a propriedade obteve o valor *create*, para fins de criação e validação do esquema criado e após isso, foi informado *update* como seu valor para tornar reflexo as alterações feitas na camada Java.

Portanto, a execução do *Crawler* funciona da seguinte forma, é enviada uma requisição para o *website* através da URL do tópico em questão com o sufixo "*.json*" no final. O *website* retorna um *JSON* com os dados referentes ao tópico solicitado e aos comentários deste tópico. Este objeto JSON é convertido em um POJO através da biblioteca Jackson e persistida no banco de dados através do *framework* Hibernate. Como a API do Reddit possui uma restrição do número de comentários disponibilizados, são efetuadas novas requisições para a seção *"more"* disponível no JSON de retorno.

REFERÊNCIAS

Alexa. **Alexa Top 500 Global Sites**. <Disponível em: http://www.alexa.com/topsites/>. Acesso em: 27 de Fevereiro de 2017.

Apache OpenNLP. **Apache OpenNLP**. <Disponível em: https://opennlp.apache.org/>. Acesso em: 27 de Fevereiro de 2017.

BRILL, E. A Simple Rule-based Part of Speech Tagger. In: THIRD CONFERENCE ON APPLIED NATURAL LANGUAGE PROCESSING, Stroudsburg, PA, USA. **Proceedings...** Association for Computational Linguistics, 1992. p.152–155. (ANLC '92).

DOMINGOS, P.; PAZZANI, M. J. On the Optimality of the Simple Bayesian Classifier under Zero-One Loss. **Machine Learning**, [S.l.], v.29, n.2-3, p.103–130, 1997.

HANCOX, P. J. **A brief history of Natural Language Processing**. <Disponível em: http://www.cs.bham.ac.uk/~pjh/sem1a5/pt1/pt1_history.html/>. Acesso em: 02 de Abril de 2017.

HUTTO, C. J.; GILBERT, E. VADER: a parsimonious rule-based model for sentiment analysis of social media text. In: ICWSM. **Anais...** The AAAI Press, 2014.

JACKSON, P.; MOULINIER, I. **Natural Language Processing for Online Applications**: text retrieval, extraction and categorization. [S.l.]: John Benjamins Publishing Company, 2007.

MANNING, C. D.; SCHüTZE, H. **Foundations of Statistical Natural Language Processing**. [S.l.]: MIT Press, 1999.

Natural Language Toolkit. **Natural Language Toolkit**. <Disponível em: http://www.nltk.org/>. Acesso em: 27 de Fevereiro de 2017.

SHANNON, C. E.; WEAVER, W. A Mathematical Theory of Communication. **The Bell System Technical Journal**, [S.l.], v.27, p.379–423,623–656, July, October 1948.

Spacy. **Spacy**. <Disponível em: `https://spacy.io/`>. Acesso em: 27 de Fevereiro de 2017.

Stanford CoreNLP. **Stanford CoreNLP**. <Disponível em: `http://stanfordnlp.github.io/CoreNLP/`>. Acesso em: 27 de Fevereiro de 2017.