Normalização de Texto

Processamento de Linguagem Natural

Prof. Leandro Alvim, D. Sc.

Agenda

- O que é Normalização?
- Exemplos de Normalização
- Implementações
- Bibliotecas
- Considerações Finais

O que é normalização de texto?

- Normalização de texto é o processo de transformar texto em uma única forma canônica
- Normalizar texto antes de resolver uma tarefa permite a filtragem de termos relevantes, remoção de termos irrelevantes ou transformação destes
- A normalização de texto requer estar ciente de qual tipo de texto deve ser normalizado e como ele deve ser processado posteriormente
- Não há nenhum procedimento de normalização que sirva para qualquer problema. Cada procedimento deve ser adequado ao seu problema.

Exemplos de Normalização

- Acrônimos
  - Países
    - *US -> USA*
    - *U.S.A -> USA*
  - *Organizações*
    - *UN -> United Nations*
- Acentuações/Umlauts
  - *naïve -> naive*
  - *ímã -> ima*

Exemplos de Normalização

- Letras maiúsculas
  - Produto -> produto
  - **João da Silva -> joao da silva**
- Tomar cuidado porque para certas tarefas como Reconhecimento de Entidades Nomeadas como entidade Nome, palavras que iniciam com maiúsculas são uma dica para Nome
- Nem sempre esta transformação lhe auxiliará. Analise bem a tarefa a ser resolvida.

Exemplos de Normalização

- Remoção de Espaços
  - *produto vendido por R$ 1.000,00* -> produto vendido por R$ 1.000,00

- Remoção de *Stop Words*
  - Termos frequentes, porém pouco informativos
    - *Preposições, artigos, ...*
  - Necessita da resolução da tarefa POS-TAG para identificar a gramática
  - Nem toda tarefa se beneficia desta remoção

Exemplos de Normalização

- Valores
  - Telefones
    - +55(21)92762-5209 -> 5521927625209
  - Datas
    - 11 de Jan de 2019, 11/01/2019, ... -> 20190111
  - Moedas
    - R$ 20,00 -> 20 reais
  - Endereços
    - Rua Dias da Cruz número 25 apto 100 -> rua dias da cruz num 25 ap 100

- Valores muitas vezes, como são bem distintos, podem aumentar muito a quantidade de atributos
  - Uso excessivo de memória
  - Tempo alto de processamento do preditor
- Transformar num padrão comum
  - *20 reais* -> DINHEIRO
  - *email@gmail.com* -> EMAIL
  - *20190111* -> DATA
  - *5521927625209* -> TELEFONE

Exemplos de Normalização

- *Lemmatization*
  - Manter o *lemma*
    - *produção, produzir, produz, produzindo -> produz*
  - Significado
  - Necessita de uma dicionário
- *Stemming*
  - Removendo sufixos
    - economia, economizar, economizando, econômico -> econom

Exemplos de Normalização

- Correção de escrita
  - Modelos de Linguagem
    - *beringela -> berinjela*
    - *cachorrro, cachoro -> cachorro*
    - *sombrancelha -> sobrancelha*

Exemplos de Normalização

- Normalização Fonética
  - BRI e Buscadores
  - Auxilia na recuperação de palavras com som parecido
  - Codifica a palavra de acordo com a pronúncia
  - Exemplos
    - Herman -> H655
    - Veronika, Veronique -> V652

- Soundex Code
  - Cada palavra é comprimida em quatro caracteres
- Algoritmo
  - Manter a primeira letra
  - Remover a, e, i, o, u, y, h, w.
  - Substituir consoantes com sons similares por dígitos
    - bfpv -> 1;
    - cgjkqsxz -> 2;
    - dt -> 3;
    - l -> 4;
    - mn -> 5;
    - r -> 6;
  - Remover ocorrências de dígitos similiares em sequência
  - Manter apenas os quatro primeiros códigos (colocar zeros no final se necessário)

Implementações

- Simples remoção de pontuação
  - Útil em classificação de texto
  - Ruim para tarefas de reconhecimento de entidades, pos-tagging, ...

```
>>> import string
...
... input_str = "Este &é [um] exemplo? {de} string. com.? pontuações!!!!" # Sample string
... result    = input_str.translate(str.maketrans('', '', string.punctuation))
...
... print(result)
Este é um exemplo de string com pontuações
```

Implementações

- Remoção simples de valores

```
>>> import re
...
... input_str = """Comprei 3 ingressos para o filme."""
... result = re.sub(r"\d+", "", input_str)
...
... print(result)
...
Comprei  ingressos para o filme.
```

Implementações

- Substituição simples de valores

```
>>> import re
...
... input_str = """Comprei 3 ingressos para o filme."""
... result = re.sub(r"\d+", "NUMERO", input_str)
...
... print(result)
...
Comprei NUMERO ingressos para o filme.
```

Implementações

- Substituição simples de valores (muito usado em classificação de texto)

```
>>> results = """Comprei 3 ingressos para o filme que custaram cada R$ 30,00."""
... results = re.sub(r"((?:R\$)?\s+(?:(\d{1,3}\.|){1,}\d{1,3}\,\d{1,2}))", "DINHEIRO", results)
... results = re.sub(r"\d+", "NUMERO", results)
...
... print(results)
...
Comprei NUMERO ingressos para o filme que custaram cada DINHEIRO.
```

Implementações

- Substituição simples de valores
  - Cuidado, a ordem importa!

```
>>> results = """Comprei 3 ingressos para o filme que custaram cada R$ 30,00."""
... results = re.sub(r"\d+", "NUMERO", results)
... results = re.sub(r"((?:R\$)?\s+(?:(\d{1,3}\.|){1,}\d{1,3}\,\d{1,2}))", "DINHEIRO", results)
...
... print(results)
Comprei NUMERO ingressos para o filme que custaram cada R$ NUMERO,NUMERO.
```

Implementações

- Substituição de valores para um padrão (muito usado em recuperação da informação)

```
>>> input_str = """O telefone é +55(21 )98273-4212 ."""
... padrao_regex = r"((?:\+?\s*(\d{2})\s*)?(?:\(\s*(\d{2})\s*\))\s*(\d?\d{4})(?:\-|\s*)?(\d{4})\s*)"
... results = re.search(padrao_regex,input_str)
...
... s = [g for g in results.groups()]
... novo_padrao = "".join(s[1:])
...
... results = re.sub(padrao_regex,novo_padrao,input_str)
...
... print(results)
...
O telefone é 5521982734212.
```

Implementações

- Colocando em letras minúsculas

```
>>> input_str = "Os 5 maiores países em população de 2017 são China, India, Estatados Unidos, Indonesia e Brasil."
... result    = input_str.lower()
...
... print(result)
os 5 maiores países em população de 2017 são china, india, estatados unidos, indonesia e brasil.
```

Implementações

- Removendo *Stop Words*
  - *Frequência alta, importância baixa*

```
>>> from nltk.tokenize import word_tokenize
... from nltk.corpus import stopwords
...
... input_str      = "NLTK is a leading platform for building Python programs to work with human language data."
... word_list      = word_tokenize(input_str)
... filtered_words = [word for word in word_list if word not in stopwords.words('english')]
... print(filtered_words)
['NLTK', 'leading', 'platform', 'building', 'Python', 'programs', 'work', 'human', 'language', 'data', '.']
```

Implementações

- Código Soundex

```
>>> import fuzzy
...
... soundex = fuzzy.Soundex(4)
... codigo = soundex("Marcel")
... print(codigo)
...
M624
```

Implementações

- Stemmer

```
>>> from nltk.stem import PorterStemmer
... from nltk.tokenize import word_tokenize
... stemmer    = PorterStemmer()
... input_str = "There are several types of stemming algorithms."
... input_str = word_tokenize(input_str)
... for word in input_str:
...     print(stemmer.stem(word))
...
there
are
sever
type
of
stem
algorithm
.
```

- Lemma

```
>>> from nltk.stem import WordNetLemmatizer
... from nltk.tokenize import word_tokenize
... lemmatizer = WordNetLemmatizer()
... input_str = "been had done languages cities mice"
... input_str = word_tokenize(input_str)
... for word in input_str:
...     print(lemmatizer.lemmatize(word))
...
been
had
done
language
city
mouse
```

Implementações

- Corretor de texto

```
>>> from spellchecker import SpellChecker
... spell = SpellChecker()
...
... # find those words that may be misspelled
... misspelled = spell.unknown(['something', 'is', 'hapenning', 'here'])
... for word in misspelled:
...     # Get the one `most likely` answer
...     print(spell.correction(word))
...     # Get a list of `likely` options
...     print(spell.candidates(word))
...
happening
{'penning', 'henning', 'happening'}
```

Bibliotecas

- NLTK
  - http://www.nltk.org/
- SPACY
  - https://spacy.io/
- TextBlob
  - https://textblob.readthedocs.io/en/dev/
- CLiPS
  - https://www.clips.uantwerpen.be/pages/pattern-en
- Gensim
  - https://radimrehurek.com/gensim/

Considerações Finais

- A normalização é uma parte trabalhosa, porém crucial parte de pre processamento do texto. Uma normalização mal feita ou mesmo a não utilização desta, pode acarretar em perda de qualidade na resolução do problema
- O subconjunto de tarefas de normalização a serem utilizadas dependerá do problema a ser resolvido. Cabe também a você testar e pensar na utilidade de cada uma para o problema
- Hoje em dia há diversas bibliotecas de qualidade para cada uma das tarefas de normalização em várias línguas. Não se prenda a somente a uma biblioteca.