Processamento de Linguagem Natural (PLN)

Alvaro José Lopes
Davi Fagundes Ferreira da Silva

Data - ICMC

@AlvaroJL @davi_fagundes

30 de janeiro de 2023

Sumário

Sumário

Introdução

O que é PLN?

- Sigla para "Natural Language Processing"ou processamento de linguagem natural
- Por mais que venha em mente que essa tecnologia lida só com texto, na realidade é com um conceito mais abstrato e portanto tem várias áreas de atuação e aplicações.
- Sua origem tem a ver com o famoso teste de turing e um experimento em georgetown, no anos 50.

Aplicações

- Filtragem de emails;
- Assistentes inteligentes (Tipo a Siri ou Google Assistente);
- Resultados de pesquisa (como a pesquisa do google, claro que ela tem mais coisas, mas PLN age, por exemplo, na área de "você quis dizer isso?"
- Geração e predição de textos;
- Tradução de linguagens;
- Chatbots;
- Reconhecimento de voz e por aí vai.

Principais Termos

Corpus

- é um conjunto de dados linguísticos, sistematizados segundo determinados critérios, suficientemente extensos em amplitude e profundidade, de maneira que sejam representativos da totalidade do uso linguístico ou de algum de seus âmbitos, dispostos de tal modo que possam ser processados por computador, com a finalidade de propiciar resultados vários e úteis para a descrição e análise (SANCHEZ, 1995, pp. 8-9)
- É como se fosse um dataset, traduzindo para nossa língua, mas tem esse nome por ser do campo da linguística, e um conjunto de corpus forma um corpora.

Documento

- Documento é um objeto do corpus, como por exemplo, o capítulo seria um objeto de um livro.
- Exemplos: um romance; uma mensagem de texto no celular; uma publicação.

Sentença

- É um objeto do documento, como por exemplo, um parágrafo ou uma frase seria um objeto de um capítulo.

Token

- É um objeto do seu texto, podendo ser ou uma palavra ou uma sentença (geralmente é uma palavra).

```
['Artificial', 'Intelligence', '(', 'AI', ')', 'is', 'likely', 'to',
'be', 'either', 'the', 'best', 'or', 'the', 'worst', 'thing', 'to',
'happen', 'to', 'humanity', '.']
```

Figura: Exemplo de token

Pré-processamento

Normalization

- É o processo de transformar o texto em sua forma canônica para poder realizar alguma análise ou tarefa.
- Não existe uma normalização geral para cada texto, afinal dependendo do que é necessário, precisa-se de processos diferentes. É um sistema chave-fechadura.

Lemmatização

- é um processo que, ao levar em consideração a classe gramatical da palavra (além de tirar as "stop-words"), remove as flexões dela, retornando-a a sua forma mais básica (sua forma no dicionário)
- Exemplo: "break", "breaks", "broke", "broken", "breaking"seriam lemmatizados para "break".
- É comumente usado em tecnologias como vetorização de palavras, TF-IDF e LDA para modelagem de tópicos.
- Problema: nem sempre reduz as palavras. (e.g.: break -¿ break)

Stemming

- É quase a mesma ideia, no entanto releva-se apenas a palavra e a reduz para a sua raiz (ou stem), podendo ser gramaticalmente incorreta, mas ainda com valor.
- Exemplo: automatic, automation = autom.
- Esse processo pode acarretar dois tipos de erros: ter um stem que perdeu toda sua informação e ficou uma raíz sem valor; ou ter dois stem com significados muito diferentes.
- Mesmo com esses problemas, ainda é bem útil para, por exemplo, mecanismos de busca para indexar palavras.

Segmentation

- É o processo de separar o texto em partes coesas (geralmente chamadas de "tokens"), dependendo do que seria necessário.

Sentence Segmentation

```
from nltk.tokenize import sent_tokenize
text = "God is Great! I won a lottery."
print(sent_tokenize(text))

Output: ['God is Great!', 'I won a lottery ']
```

Figura: Tokenização de um texto em sequências

Word Segmentation

- É o ato de separar todas as palavras em tokens.
- Exemplo:

```
['It', 'would', 'be', 'unfair', 'to', 'demand', 'that', 'people',
'cease', 'pirating', 'files', 'when', 'those', 'same', 'people', 'are',
'not', 'paid', 'for', 'their', 'participation', 'in', 'very',
'lucrative', 'network', 'schemes', '.', 'Ordinary', 'people', 'are',
'relentlessly', 'spied', 'on', ',', 'and', 'not', 'compensated', 'for',
'information', 'taken', 'from', 'them', '.', 'While', 'I', 'would',
'like', 'to', 'see', 'everyone', 'eventually', 'pay', 'for', 'music',
'and', 'the', 'like', ',', 'I', 'would', 'not', 'ask', 'for', 'it',
'until', 'there', 'is', 'reciprocity', '.']
```

Figura: Tokenização de um texto do Shakespeare

- Nesse caso, ainda estão os sinais de pontuação, mas é possível retirá-los.

Baseline Pipeline

Baseline Pipeline

- Após o pré-processamento, ainda estamos lidando com dados não estruturados na forma de texto.
- Essa forma não é adequada para os algoritmos de Machine Learning. Pois, são em sua grande maioria baseados em distância e por isso é necessário que os dados sejam representados como vetores.

Baseline Pipeline

- Após o pré-processamento, ainda estamos lidando com dados não estruturados na forma de texto.
- Essa forma não é adequada para os algoritmos de Machine Learning. Pois, são em sua grande maioria baseados em distância e por isso é necessário que os dados sejam representados como vetores.
- **Solução:** Construir um *Vector Space Model* para mapear um objeto (e.g texto) em um vetor.

Baseline Pipeline

- Após o pré-processamento, ainda estamos lidando com dados não estruturados na forma de texto.
- Essa forma não é adequada para os algoritmos de Machine Learning. Pois, são em sua grande maioria baseados em distância e por isso é necessário que os dados sejam representados como vetores.
- **Solução:** Construir um *Vector Space Model* para mapear um objeto (e.g texto) em um vetor.
- Encontrar maneiras não-supervisionadas para se aprender representações vetoriais de um input, ao invés de criar representações manualmente por *feature engineering*, é uma área de pesquisa denominada **Representation Learning**.

Baseline Pipeline

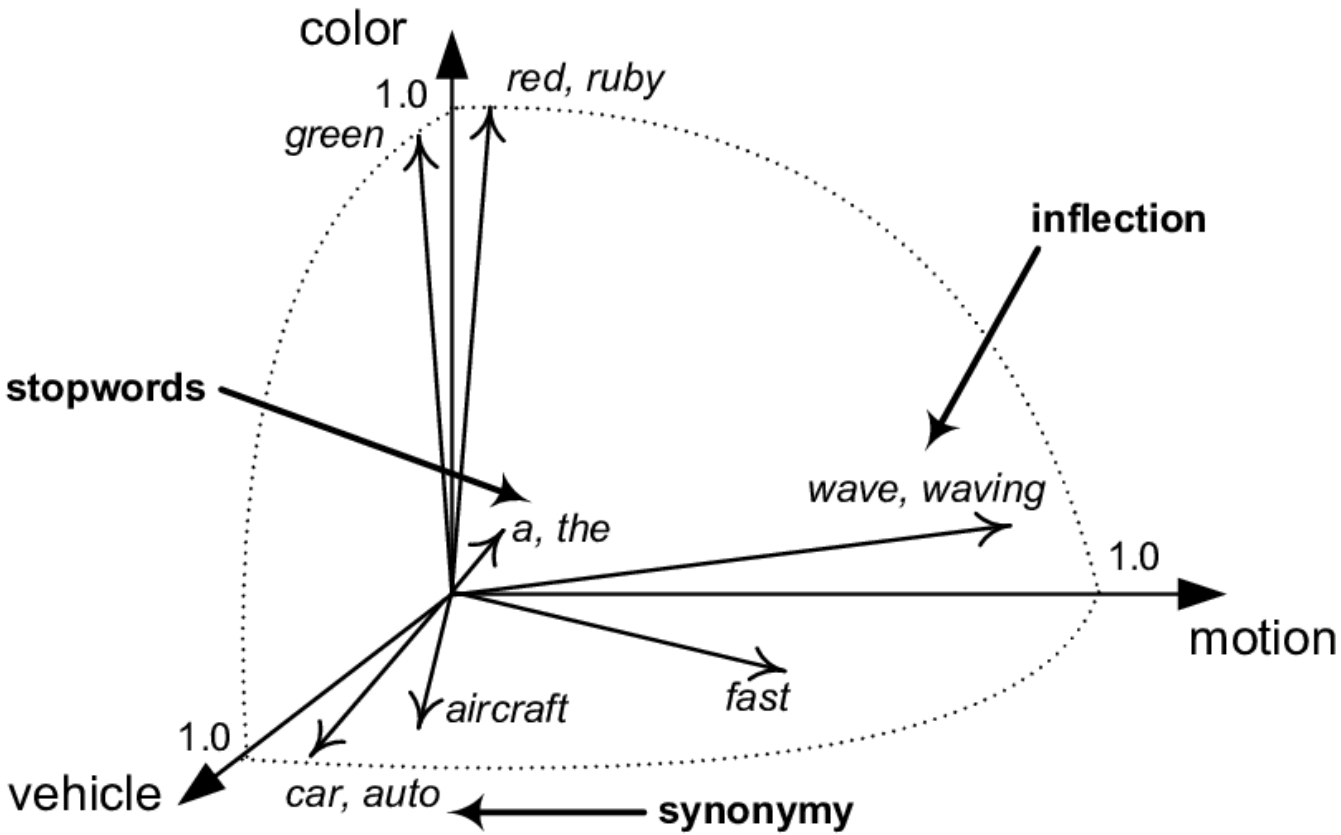


Figura: Modelo Espaço Vetorial baseado em tópicos. Cada tópico é um eixo e as palavras estão distribuídas nesse espaço.

Baseline Pipeline – Exemplo

Queremos que os vetores, de alguma forma, codifiquem o significado semântico dos nossos dados textuais.

Classe	**Palavra**	**excelente**	**terrível**
neg	horrível	6	113
neg	terrível	8	309
neg	vergonhoso	1	69
neg	decepcionante	19	29
pos	maravilhoso	91	6
pos	bom	21	9
pos	incrível	67	2
pos	legal	118	2

Tabela: VSM de palavras com base na co-ocorrência dessas palavras com *excelente* e *terrível* em um corpus.

Baseline Pipeline – Exemplo

Podemos analisar o significado "latente"da tabela anterior para construir uma regra de decisão:

- Se a palavra possui uma **maior co-ocorrência com excelente**, então será classificada como **positiva**;
- **Caso contrário**, será classificada como **negativa**.

$Pr(Class = pos)$	**Palavra**	**excelente**	**terrível**
≈ 0	w_1	4	82
≈ 0	w_2	5	84
≈ 1	w_3	49	3
≈ 1	w_4	41	5

Tabela: Classificação de palavras não vistas no Corpus como positivas ou negativas, com base em uma simples regra de decisão.

Hipótese Distribucional

O significado de uma palavra desconhecida pode ser inferido de seu contexto. Considerando o exemplo abaixo:

- Uma garrafa de tezgüino está na mesa.
- Todos gostam de tezgüino.
- Tezgüino te deixa bêbado.
- Fazemos tezgüino do trigo.

Hipótese Distribucional

O significado de uma palavra desconhecida pode ser inferido de seu contexto. Considerando o exemplo abaixo:

- Uma garrafa de tezgüino está na mesa.
- Todos gostam de tezgüino.
- Tezgüino te deixa bêbado.
- Fazemos tezgüino do trigo.

Os contextos em que a palavra tezgüino é utilizada sugerem que tezgüino pode ser um tipo de bebida alcoólica feita de trigo.

Hipótese Distribucional

"Diga-me com quem tu andas, e te direi quem tu és"

"You shall know a word by the company it keeps."

Firth (1957)

"The meaning of a word is it use in the language"

Wittgenstein (1953)

"If units of text have similar vectors in a text frequency matrix, then they tend to have similar meanings."

Turney and Pantel (2010)

Hipótese Distribucional

"Diga-me com quem tu andas, e te direi quem tu és"

"You shall know a word by the company it keeps."

Firth (1957)

"The meaning of a word is it use in the language"

Wittgenstein (1953)

"If units of text have similar vectors in a text frequency matrix, then they tend to have similar meanings."

Turney and Pantel (2010)

Intuição: Palavras que ocorrem/se distribuem nos mesmos contextos tendem a ter o mesmo sentido.

Hipótese Distribucional

"Diga-me com quem tu andas, e te direi quem tu és"

"You shall know a word by the company it keeps."

Firth (1957)

"The meaning of a word is it use in the language"

Wittgenstein (1953)

"If units of text have similar vectors in a text frequency matrix, then they tend to have similar meanings."

Turney and Pantel (2010)

Intuição: Palavras que ocorrem/se distribuem nos mesmos contextos tendem a ter o mesmo sentido.

O que significa estarem no mesmo contexto? No caso de palavras, significa possuírem palavras vizinhas similares ao longo do texto.

Hipótese Distribucional – Exemplo

- **Sinônimos** como **oculista** e **oftalmologista** ocorrem em contextos parecidos, por exemplo perto de palavras como **olho** e **examinado**.

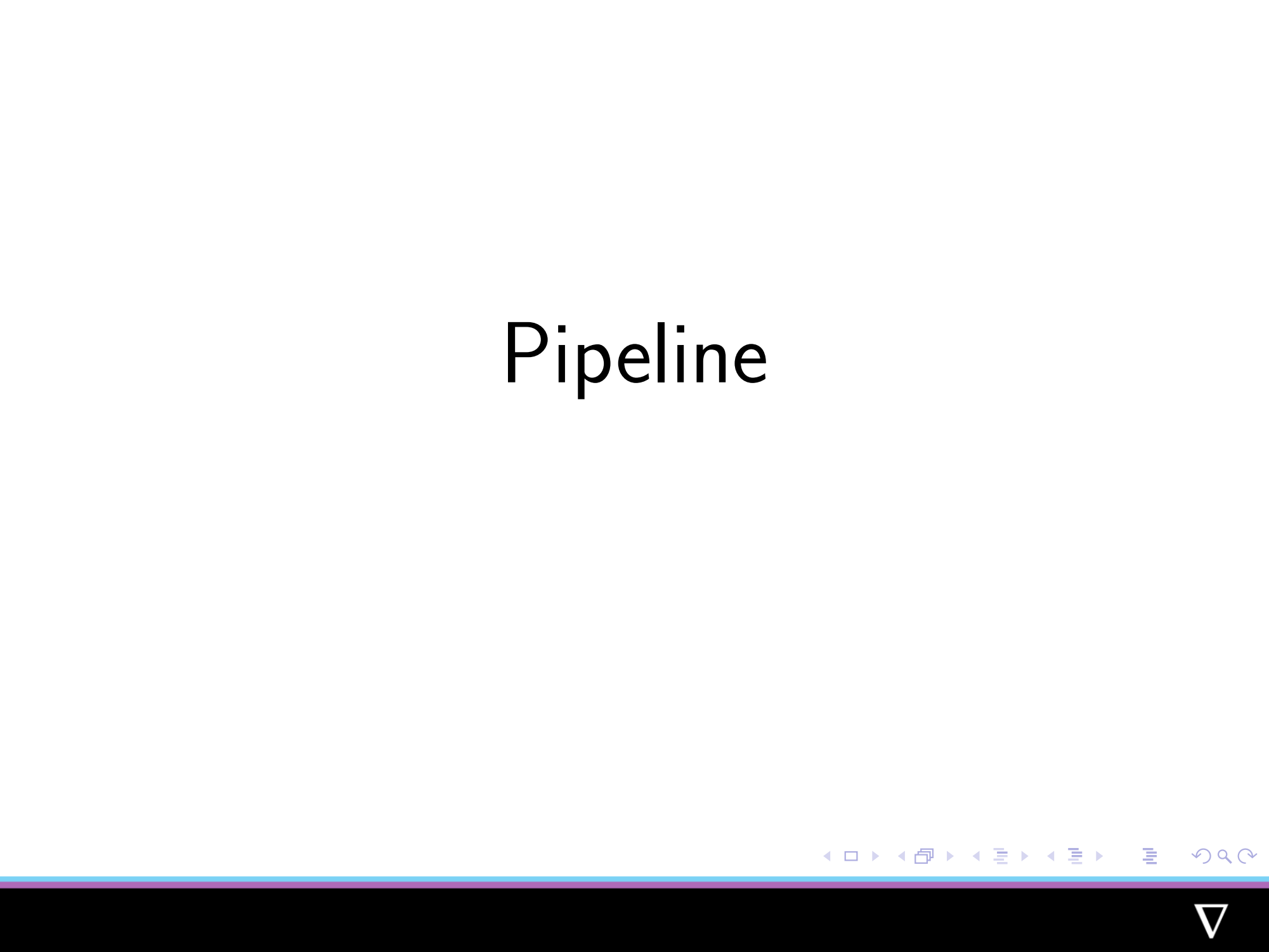
Hipótese Distribucional – Exemplo

- **Sinônimos** como **oculista** e **oftalmologista** ocorrem em contextos parecidos, por exemplo perto de palavras como **olho** e **examinado**.
- **Cachorro** e **gato** são palavras **similares**, pois compartilham muitas características (animais, domésticos, com orelhas).

Hipótese Distribucional – Exemplo

- **Sinônimos** como **oculista** e **oftalmologista** ocorrem em contextos parecidos, por exemplo perto de palavras como **olho** e **examinado**.
- **Cachorro** e **gato** são palavras **similares**, pois compartilham muitas características (animais, domésticos, com orelhas).
- Palavras **relacionadas** podem ser associadas por um determinado campo semântico.
  - As palavras **comida**, **prato**, **chefe**, **garçom** estão associadas a **restaurantes**.
  - As palavras **porta**, **cozinha**, **cama**, **família** estão associadas a **casa**.

Pipeline



Pipeline

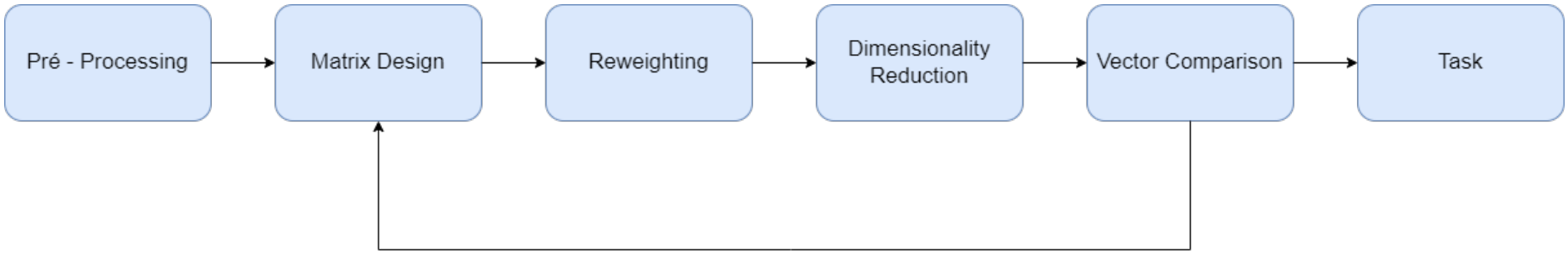


Figura: Pipeline clássico de PLN. O objetivo principal é fornecer um dado estruturado para realização da Task solicitada. Caso o Modelo Espaço Vetorial não seja o adequado para a task é possível replanejar.

Pipeline – Matrix Design

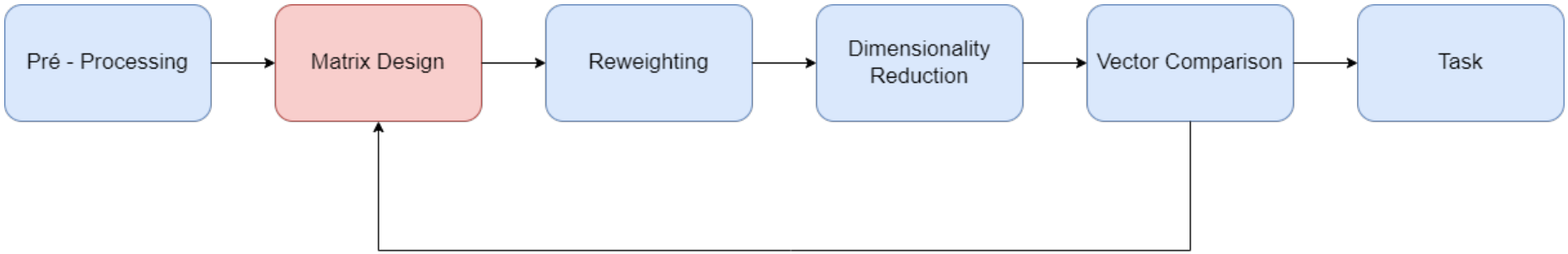


Figura: Nessa etapa do pipeline é preciso escolher o tipo de matriz mais adequado para os seus dados e para sua task.

As matrizes de contagem são construídas a partir do conceito de co-ocorrência:

- **word x document**: nro de vezes que uma palavra aparece em cada documento.
- **word x word**: nro de vezes na qual duas palavras ocorrem juntas em um documento.

Pipeline – Reweighting

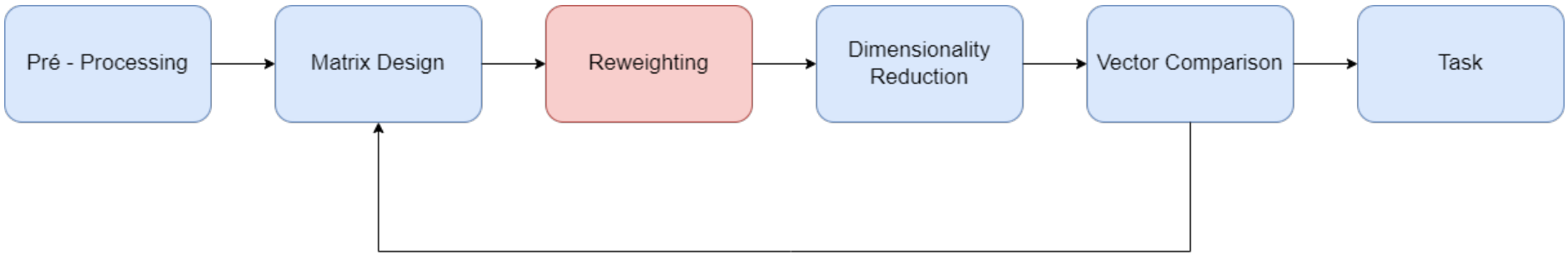


Figura: Nessa etapa do pipeline queremos ajustar os valores de contagem para obter uma melhor representação da semântica latente no corpus.

As matrizes de contagem puras não são tão boas representações. Podemos usar os seguintes métodos para amplificar o conhecimento "latente" nas contagens e reduzir o que não importa:

- **TF-IDF** (Term Frequency–Inverse Document Frequency)
- **PMI** (Pointwise Mutual Information)

Pipeline – Dimensionality Reduction

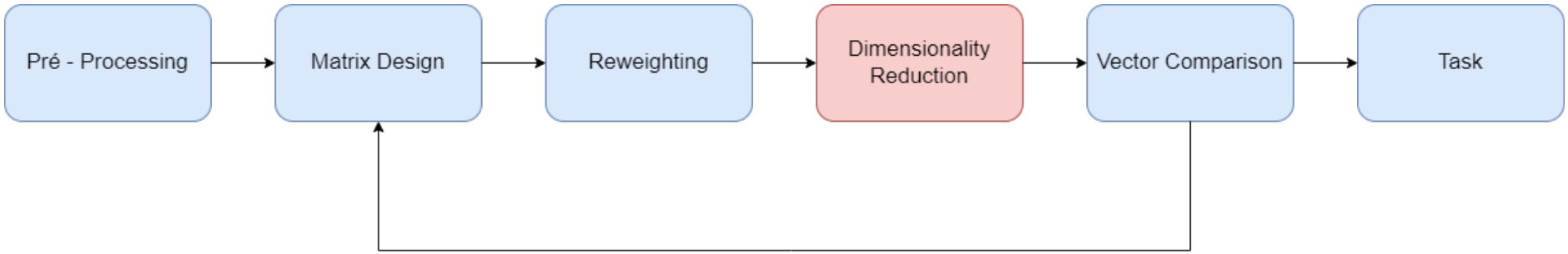


Figura: Nessa etapa do pipeline queremos reduzir a dimensão do vetor.

Como veremos, as matrizes de ocorrência são de alta dimensão e esparsas (grande quantidade de elementos nulos). Para reduzir o efeito de *curse of dimensionality* pode-se aplicar os seguintes métodos:

- **LSA** (Latent Semantic Analysis)
- **PCA e t-SNE** (t-Distributed Stochastic Neighbor Embedding)

Pipeline – Vector Comparison

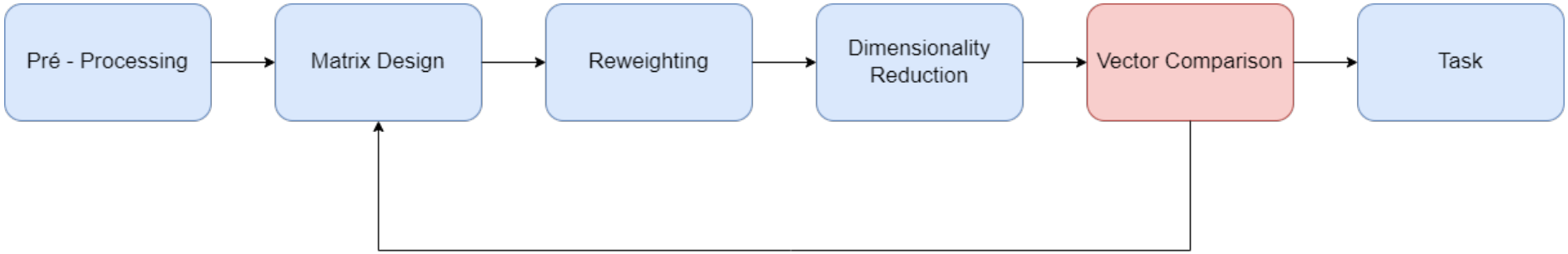


Figura: Nessa etapa do pipeline queremos operacionalizar a noção de similaridade através da comparação de vetores.

Munidos das representações vetoriais dos nossos objetos (e.g palavras, documentos), precisamos estabelecer uma medida de distância para calcular numericamente a similaridade entre eles. Algumas métricas são:

- **Euclideana**
- **Cosseno**

Matrix Design

Matrix Design – word x document

Matriz de ordem $|V|$ x $|D|$, em que cada linha representa uma palavra do vocabulário e cada coluna representa um documento de uma coleção de documentos.

Matrix Design – word x document

Matriz de ordem $|V|$ x $|D|$, em que cada linha representa uma palavra do vocabulário e cada coluna representa um documento de uma coleção de documentos.

	As You Like It	**Twelfth Night**	**Julius Caesar**	**Henry V**
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

Tabela: Matrix do tipo termo-documento de 4 palavras em 4 peças de Shakespeare. Cada célula contém o número de vezes que uma palavra ocorre em um determinado documento.

Matrix Design – word x document

Matriz de ordem $|V|$ x $|D|$, em que cada linha representa uma palavra do vocabulário e cada coluna representa um documento de uma coleção de documentos.

	As You Like It	**Twelfth Night**	**Julius Caesar**	**Henry V**
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

Tabela: Matrix do tipo termo-documento de 4 palavras em 4 peças de Shakespeare. Cada célula contém o número de vezes que uma palavra ocorre em um determinado documento.

Comumente denominada de Bag of Words (BoW).

Matrix Design – word x document

No exemplo anterior, podemos interpretar cada documento como um vetor de dimensão $|V|$ (tamanho do vocabulário) usando o vetor coluna da matriz correspondente ao documento.

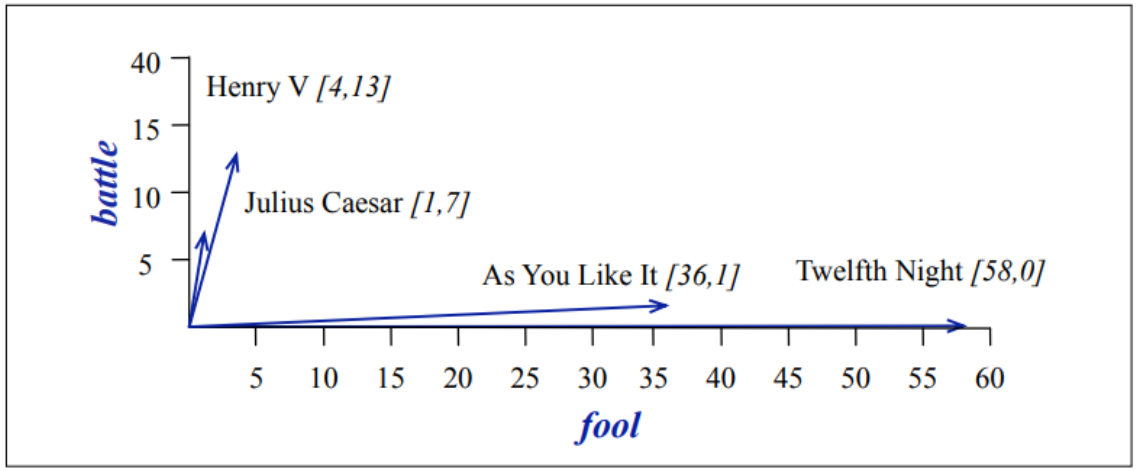


Figura: Representação visual dos vetores para cada documento, considerando apenas duas dimensões. As peças de comédia possuem altos valores ao longo da dimensão *fool* e baixos valores ao longo da dimensão *battle*

Matrix Design – word x document

Intuição sobre similaridade entre documentos: documentos similares tendem a ter vetores similares, já que esses documentos tendem a possuir palavras similares.

Matrix Design – word x document

Intuição sobre similaridade entre documentos: documentos similares tendem a ter vetores similares, já que esses documentos tendem a possuir palavras similares.
Da mesma forma, podemos utilizar o vetor linha para representar uma palavra.

	As You Like It	**Twelfth Night**	**Julius Caesar**	**Henry V**
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

Matrix Design – word x document

Intuição sobre similaridade entre documentos: documentos similares tendem a ter vetores similares, já que esses documentos tendem a possuir palavras similares.
Da mesma forma, podemos utilizar o vetor linha para representar uma palavra.

	As You Like It	**Twelfth Night**	**Julius Caesar**	**Henry V**
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

Intuição sobre similaridade entre palavras: palavras similares tendem a ter vetores similares porque tendem a ocorrer em documentos similares (tratam de assuntos similares, logo possuem palavras similares).

Matrix Design – word x word

Matriz de ordem $|V|$ x $|V|$ em que cada célula indica o número de vezes que uma palavra *target* e uma palavra de *context* co-ocorrem em um mesmo contexto ao longo do corpus.

word/context	**aardvark**	**computer**	**data**	**result**	**pie**	**sugar**
cherry	0	2	8	9	442	25
strawberry	0	0	0	1	60	19
digital	0	1670	1683	85	5	4
information	0	3325	3982	378	5	13

Tabela: Vetor de co-ocorrência para 4 palavras contidas no corpus do Wikipedia.

Matrix Design – word x word

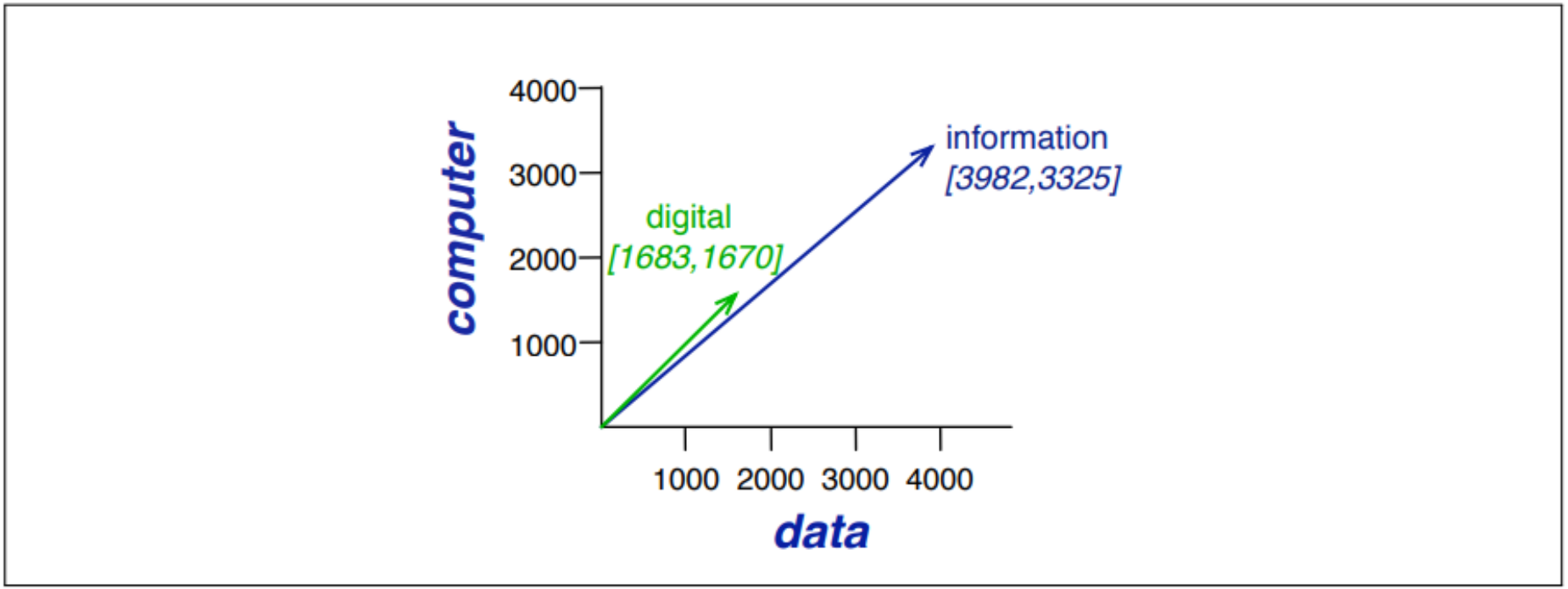


Figura: Representação espacial dos vetores correspondentes às palavras *digital* e *information*, considerando apenas duas dimensões.

Matrix Design – word x word

Para construir a matriz do tipo **word x context** precisamos definir o que significa dizer que duas palavras co-ocorrem em um mesmo contexto:

- **Documento:** Cada célula da matriz representa a quantidade de vezes que duas palavras aparecem em um mesmo documento. **Problema:** contexto muito grande, se o documento for extenso.

Matrix Design – word x word

Para construir a matriz do tipo **word x context** precisamos definir o que significa dizer que duas palavras co-ocorrem em um mesmo contexto:

- **Documento:** Cada célula da matriz representa a quantidade de vezes que duas palavras aparecem em um mesmo documento. **Problema:** contexto muito grande, se o documento for extenso.
- **Janela:** Cada célula da matriz representa a quantidade de vezes que palavra de contexto ocorre junto com a palavra analisada em uma janela de tamanho L.

Matrix Design – sliding window

Figura: Exemplo de uma *sliding window* de tamanho $L = 2$ sendo aplicada em um texto.

Matrix Design – sliding window

Na hora de escolher um valor de L deve-se considerar que para janelas:

- **Maiores**: será capturada mais informações semânticas (e.g sobre o tópico sendo tratado na sentença).
- **Menores:** será captura mais informações sintáticas (combinação usual de palavras, e.g substantivo + adjetivo).

Esperamos que, pela hipótese distribucional, palavras semelhantes co-ocorram com palavras vizinhas (de contexto) semelhantes.

Vector Comparison

Vector Comparison

Munidos das representações vetoriais dos nossos objetos (e.g palavras, documentos) queremos uma métrica que **recebe como argumento dois vetores** de mesma dimensão e **forneça** uma medida para a **similaridade** entre eles.

Vector Comparison

Munidos das representações vetoriais dos nossos objetos (e.g palavras, documentos) queremos uma métrica que **recebe como argumento dois vetores** de mesma dimensão e **forneça** uma medida para a **similaridade** entre eles.

	d_x	d_y
A	2	4
B	10	15
C	14	10

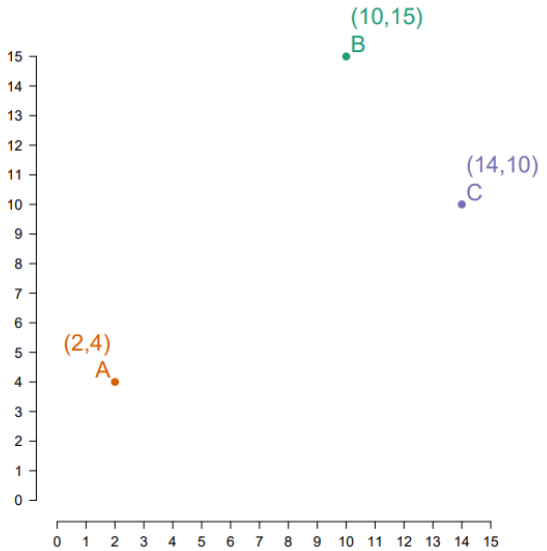


Figura: Diferentes medidas de distância fornecem diferentes resultados. B e C são mais similares considerando apenas a frequência absoluta d_x e d_y. Por outro lado, comparando as frequências de cada coordenada proporcionalmente, então A e B são mais semelhantes pois d_y é a coordenada mais frequente nesses vetores.

Vector Comparison – Distância Euclideana

$$\textbf{euclidean}(u, v) = \sqrt{\sum_{i=1}^{n} |u_i - v_i|^2}$$

	d_x	d_y
A	2	4
B	10	15
C	14	10

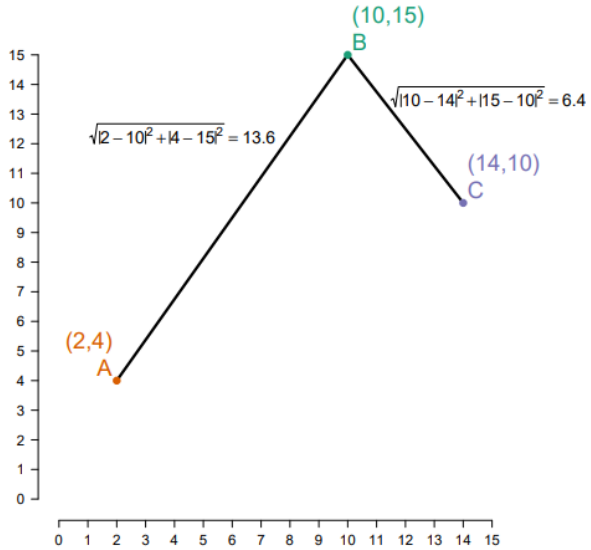


Figura: Usando distância euclideana no exemplo anterior, B e C são mais similares.

Vector Comparison – Cosseno

$$cos(\theta) = \frac{u \cdot v}{|u||v|} \in [0, 1]$$

Valores no intervalo $[0, 1]$. Em que:

- $0 \implies$ ortogonais.
- $1 \implies$ apontam para a mesma direção.

	d_x	d_y
A	2	4
B	10	15
C	14	10

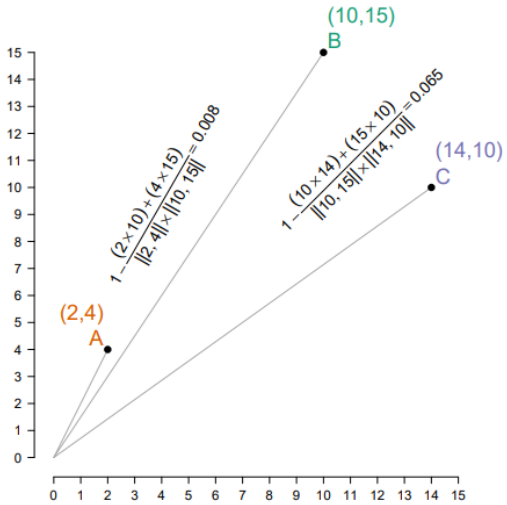


Figura: Usando cosseno no exemplo anterior ($(1 - cos(\theta))$, B é mais similar a A do que a C.

Vector Comparison – Qual usar?

A métrica de similaridade mais comumente utilizada é o cosseno entre dois vetores.

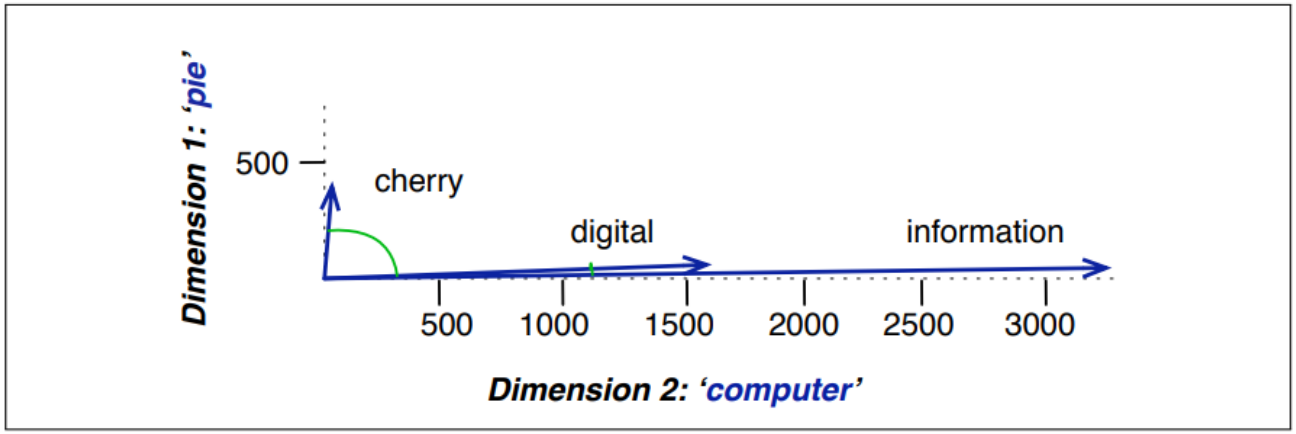


Figura: Representação gráfica do uso da métrica de similaridade cosseno. Os vetores das palavras *cherry*, *digital* e *information* em relação à contagem das palavras *computer* e *pie* no mesmo contexto.

Reweighting

Reweighting

Apenas a frequência absoluta entre palavras e documentos, ou palavras alvo e contexto, não é a melhor medida para avaliar a associação entre esses objetos, pois ela é bastante enviesada e pouco discriminativa.

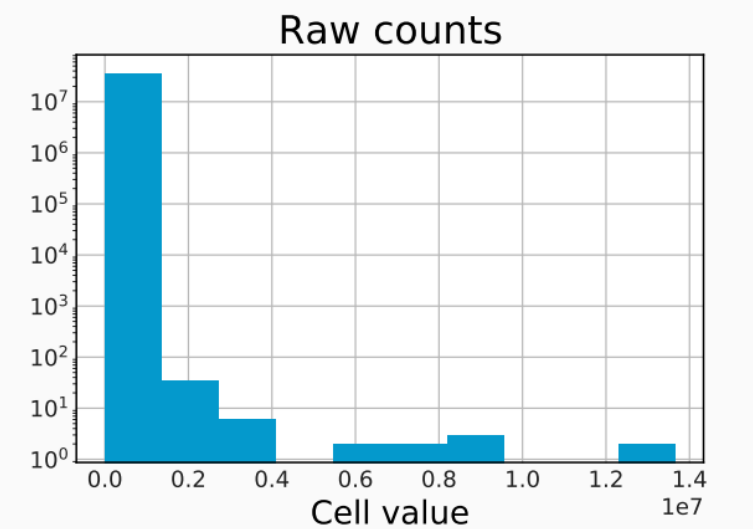


Figura: Distribuição da frequência absoluta em uma matriz de contagem.

Reweighting

Retomando o exemplo de uma matriz do tipo *word x document* de peças do Shakespeare.

	As You Like It	**Twelfth Night**	**Julius Caesar**	**Henry V**
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

Reweighting

Retomando o exemplo de uma matriz do tipo *word x document* de peças do Shakespeare.

	As You Like It	**Twelfth Night**	**Julius Caesar**	**Henry V**
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

- Palavras que são **muito frequentes** em todos os documentos, como *stopwords* e o *good* **não nos ajudam**.

Reweighting

Retomando o exemplo de uma matriz do tipo *word x document* de peças do Shakespeare.

	As You Like It	**Twelfth Night**	**Julius Caesar**	**Henry V**
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

- Palavras que são **muito frequentes** em todos os documentos, como *stopwords* e o *good* **não nos ajudam**.
- Por outro lado, palavras que possuem uma **frequência razoável** ao longo dos documentos **nos ajuda a discriminar** mais do que palavras que são muito **raras** em todos os documentos.

Reweighting – TF-IDF

- Normalmente aplicada a matrizes do tipo *word x document*. É o produto entre **term frequency** (TF) e **inverse document frequency** (IDF).
- Forma de reduzir o viés da frequência absoluta.

Reweighting – TF-IDF

- Normalmente aplicada a matrizes do tipo *word x document*. É o produto entre **term frequency** (TF) e **inverse document frequency** (IDF).
- Forma de reduzir o viés da frequência absoluta.
- **TF:** é a frequência de uma palavra t no documento d.

$$tf_{t,d} = count(t, d) = M_{t,d}$$

- **IDF:** é a **razão** entre o **número total de documentos** $|D| = N$ pela **quantidade de documentos em que uma palavra** t **ocorre**, também denominado de **document frequency** df_t. A ideia é que ele forneça um maior peso a palavras **nem raras, nem muito frequentes**.

$$idf_t = N/df_t$$

Reweighting – TF-IDF

- Normalmente aplicada a matrizes do tipo *word x document*. É o produto entre **term frequency** (TF) e **inverse document frequency** (IDF).
- Forma de reduzir o viés da frequência absoluta.
- **TF:** é a frequência de uma palavra t no documento d.

$$tf_{t,d} = count(t, d) = M_{t,d}$$

- **IDF:** é a **razão** entre o **número total de documentos** $|D| = N$ pela **quantidade de documentos em que uma palavra** t **ocorre**, também denominado de **document frequency** df_t. A ideia é que ele forneça um maior peso a palavras **nem raras, nem muito frequentes**.

$$idf_t = N/df_t$$

- O peso TF-IDF $w_{t,d}$ de uma palavra t em um documento d será dado por $w_{t,d} = tf_{t,d} \times idf_t$

Reweighting – TF-IDF

Aplicando TF-IDF para o exemplo das peças de Shakespeare teremos:

	As You Like It	**Twelfth Night**	**Julius Caesar**	**Henry V**
battle	0.074	0	0.22	0.28
good	0	0	0	0
fool	0.019	0.021	0.0036	0.0083
wit	0.049	0.044	0.018	0.022

Tabela: TD-IDF aplicado a matriz do tipo word-document contendo 4 palavras e 4 peças de Shakespeare. Note que o termo *idf* eliminou a importância de uma palavra onipresente como *good* e reduziu o impacto da palavra levemente discriminatória *fool*.

Reweighting – TF-IDF

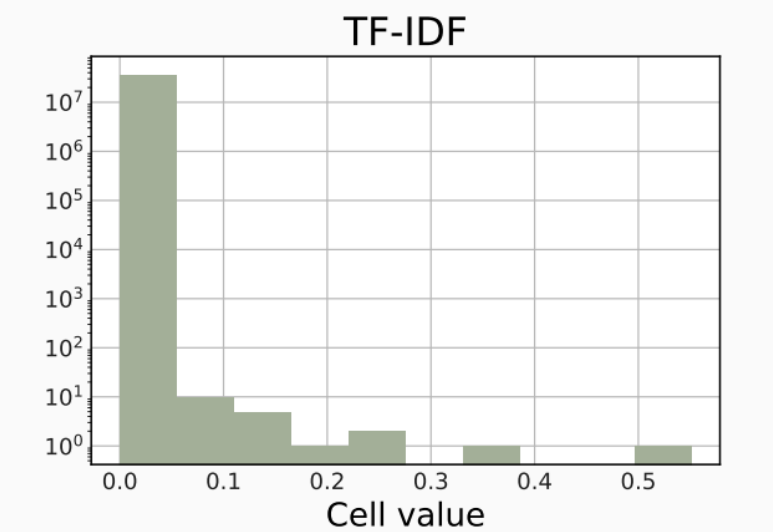


Figura: Distribuição dos valores TF-IDF ao longo das células de uma matriz do tipo *word x word*.

Reweighting – PMI

Pointwise Mutual Information é normalmente aplicada a matrizes do tipo *word x word* e mede o quão frequente dois eventos x e y ocorrem juntamente na amostra, em comparação com o que seria esperado se eles ocorressem independentemente.

$$\mathrm{PMI}(w, c) = \log\left(\frac{P(w, c)}{P(w)\,P(c)}\right)$$

- $P(w, c)$: Probabilidade conjunta de ocorrência (co-ocorrência) da palavra w com uma palavra de contexto c.
- $P(w)$: Probabilidade independente da ocorrência de uma dada palavra w.
- $P(c)$: Probabilidade independente da ocorrência de um dado contexto c.

Reweighting – PMI

Pointwise Mutual Information é normalmente aplicada a matrizes do tipo *word x word* e mede o quão frequente dois eventos x e y ocorrem juntamente na amostra, em comparação com o que seria esperado se eles ocorressem independentemente.

$$\mathrm{PMI}(w, c) = \log\left(\frac{P(w, c)}{P(w)\,P(c)}\right)$$

- $P(w, c)$: Probabilidade conjunta de ocorrência (co-ocorrência) da palavra w com uma palavra de contexto c.
- $P(w)$: Probabilidade independente da ocorrência de uma dada palavra w.
- $P(c)$: Probabilidade independente da ocorrência de um dado contexto c.

Intuição: O peso indica o quão mais frequente duas palavras co-ocorrem no corpus em relação ao acaso (iid).

Reweighting – PPMI

- Como não temos acesso às probabilidades, faremos um estimativa usando as contagens no nosso corpus.

word/context	computer	data	result	pie	sugar
cherry	0	0	0	4.38	3.30
strawberry	0	0	0	4.10	5.51
digital	0.18	0.01	0	0	0
information	0.02	0.09	0.28	0	0

Tabela: PPMI na matriz palavras do Wikipedia. *strawberry* e *cherry* bastante associados a *pie* e *sugar*, enquanto *data* levemente associadas a *information*.

Reweighting – PPMI

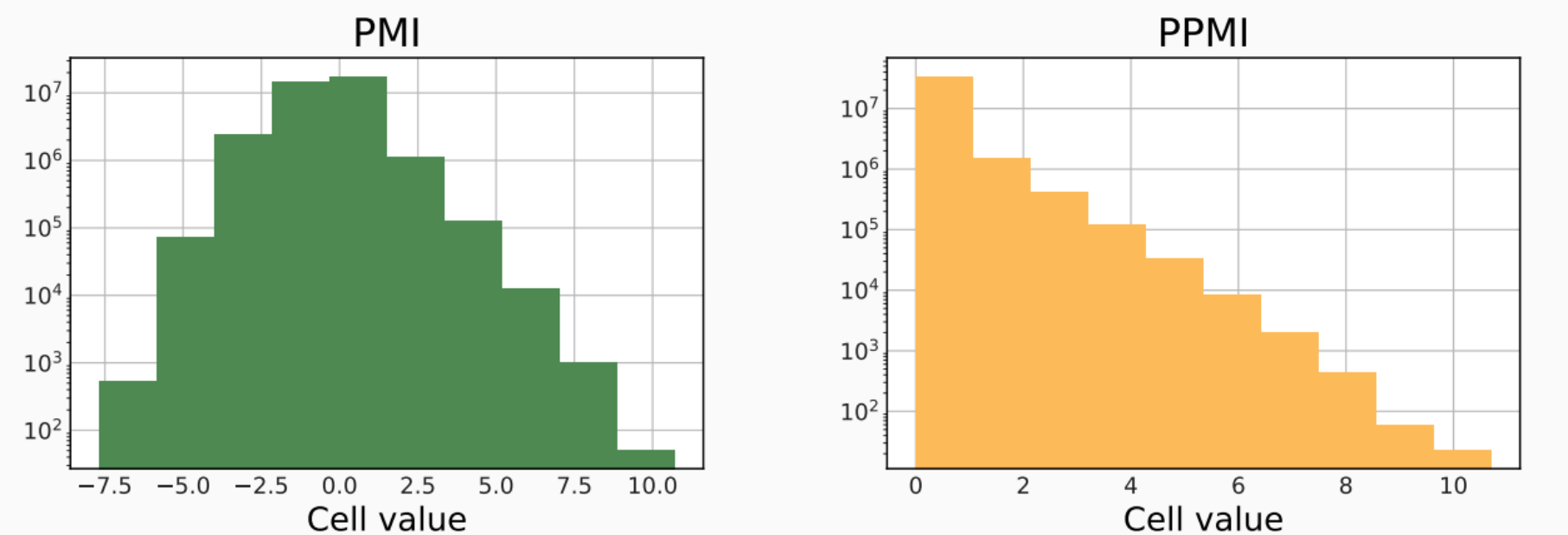


Figura: Distribuição dos valores PMI e PPMI ao longo das células de uma matriz do tipo *word x word* do dataset giga5.

Dimensionality Reduction

Dimensionality Reduction

- Até aqui no pipeline, conseguimos construir **vetores esparsos** (algumas palavras não aparecem ao lado das outras ou não aparecem em determinados documentos) e de **alta dimensão** (tamanho grande do vocabulário e grande quantidade de documento na coleção).

Dimensionality Reduction

- Até aqui no pipeline, conseguimos construir **vetores esparsos** (algumas palavras não aparecem ao lado das outras ou não aparecem em determinados documentos) e de **alta dimensão** (tamanho grande do vocabulário e grande quantidade de documento na coleção).
- Queremos algum método que encontre as dimensões mais importantes dos vetores (por exemplo, que preservam a maior variância) reduzindo ao máximo a perda de informação.
- Os novos vetores obtidos serão também denominados de **embedding**.

Dimensionality Reduction – Intuição

Intuição: Podemos traçar algumas retas (e.g por regressão linear) e projetar os pontos nessa reta.

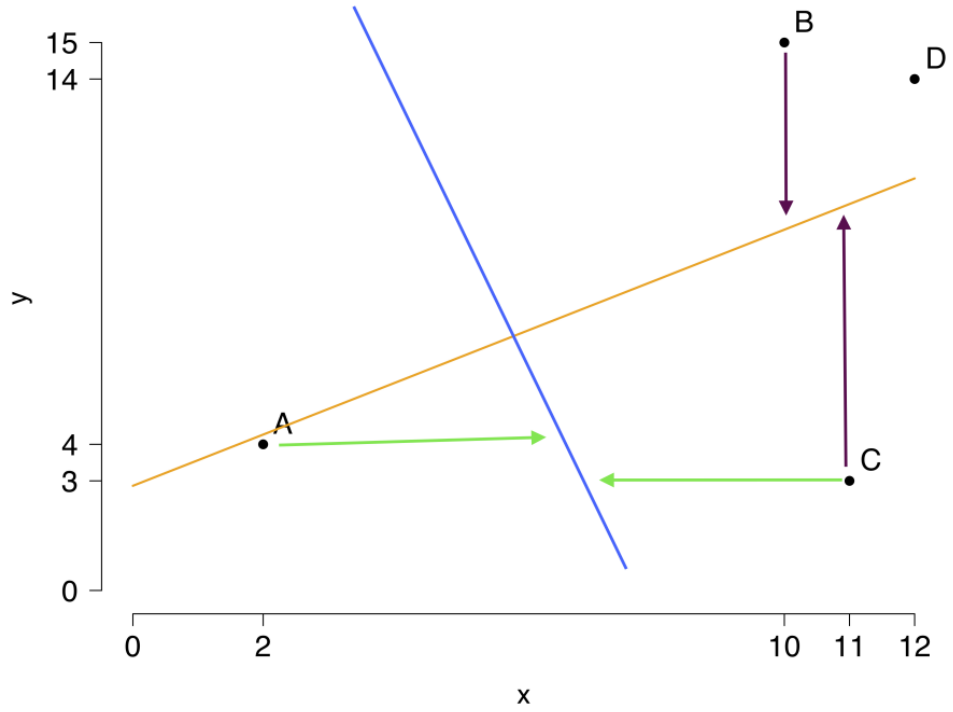


Figura: Diferentes reduções de dimensionalidades aplicadas aos pontos. Note que na projeção em **amarelo** o ponto mais próximo de C é B, enquanto na projeção em **azul** o ponto mais próximo é A.

Dimensionality Reduction – LSA

A Decomposição por Valores Singulares (*Singular Value Decomposition*, **SVD**) permite que qualquer matriz de números reais A, de dimensão mxn pode ser fatorada em três matrizes T, S e D da seguinte forma:

$$A_{mxn} = T_{mxm} S_{mxm} D^{\mathsf{T}}_{nxm}$$

- T_{mxm}: Matriz de termos, contendo vetores ortogonais.
- S_{mxm}: Matriz diagonal de valores singulares em ordem crescente (partindo do que mais preserva variância dos dados)
- D_{nxm}: Matriz de documento, contendo vetores ortogonais.

Dimensionality Reduction – LSA

A Decomposição por Valores Singulares (*Singular Value Decomposition*, **SVD**) permite que qualquer matriz de números reais A, de dimensão mxn pode ser fatorada em três matrizes T, S e D da seguinte forma:

$$A_{mxn} = T_{mxm} S_{mxm} D^{\mathsf{T}}_{nxm}$$

- T_{mxm}: Matriz de termos, contendo vetores ortogonais.
- S_{mxm}: Matriz diagonal de valores singulares em ordem crescente (partindo do que mais preserva variância dos dados)
- D_{nxm}: Matriz de documento, contendo vetores ortogonais.

Podemos usar **SVD** para aprender representações de dimensão reduzida ao selecionar quais termos e valores singulares incluiremos no modelo espaço vetorial. Esse método é denominado de **Truncated SVD**.

Dimensionality Reduction – LSA

	d1	d2	d3	d4	d5	d6
gnarly	1	0	1	0	0	0
wicked	0	1	0	1	0	0
awesome	1	1	1	1	0	0
lame	0	0	0	0	1	1
terrible	0	0	0	0	0	1

Distance from *gnarly*
1. gnarly
2. awesome
3. terrible
4. wicked
5. lame

⇓⇑

T(erm)					
gnarly	0.41	0.00	0.71	0.00	-0.58
wicked	0.41	0.00	-0.71	0.00	-0.58
awesome	0.82	-0.00	-0.00	-0.00	0.58
lame	0.00	0.85	0.00	-0.53	0.00
terrible	0.00	0.53	0.00	0.85	0.00

×

S(ingular values)				
2.45	0.00	0.00	0.00	0.00
0.00	1.62	0.00	0.00	0.00
0.00	0.00	1.41	0.00	0.00
0.00	0.00	0.00	0.62	0.00
0.00	0.00	0.00	0.00	-0.00

×

T

D(ocument)					
d1	0.50	-0.00	0.50	0.00	-0.71
d2	0.50	0.00	-0.50	0.00	0.00
d3	0.50	-0.00	0.50	0.00	0.71
d4	0.50	-0.00	-0.50	-0.00	0.00
d5	-0.00	0.53	0.00	-0.85	0.00
d6	0.00	0.85	0.00	0.53	0.00

gnarly	0.41	0.00
wicked	0.41	0.00
awesome	0.82	-0.00
lame	0.00	0.85
terrible	0.00	0.53

×

2.45	0.00
0.00	1.62

=

gnarly	1.00	0.00
wicked	1.00	0.00
awesome	2.00	0.00
lame	0.00	1.38
terrible	0.00	0.85

Distance from *gnarly*
1. gnarly
2. wicked
3. awesome
4. terrible
5. lame

Figura: Aplicando LSA para matriz do tipo *word x document*. Note que *gnarly* e *wicked* são gírias com conotação positiva que nunca ocorrem nos mesmos documentos, porém eles possuem um vizinho próximo no espaço: *awesome*. Note que o **Truncated SVD** com $k = 2$ foi capaz de capturar essa característica mais abstrata.

Dimensionality Reduction – LSA no pipeline

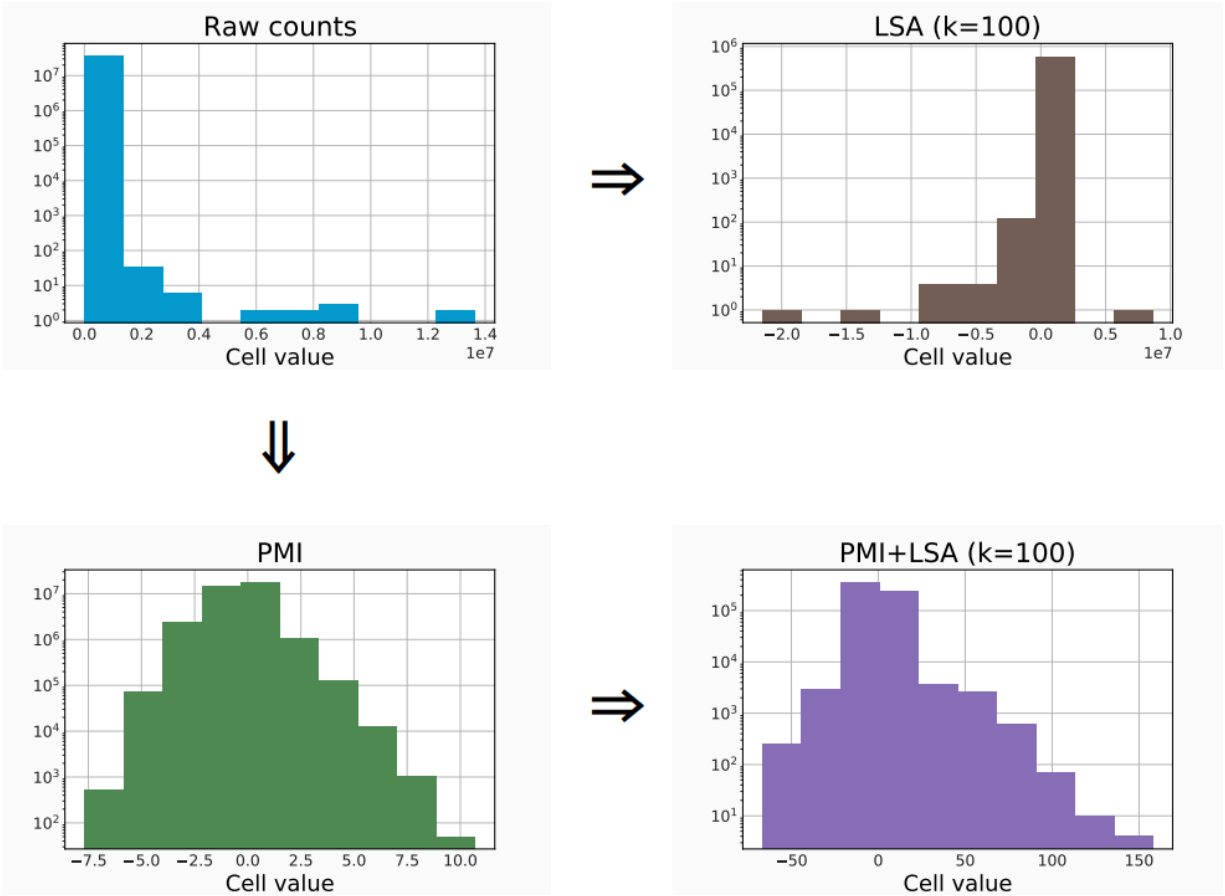


Figura: Distribuição dos valores das células de uma matriz do tipo *word x word* aplicando LSA somente e PMI+LSA. Note que PMI+LSA consegue preservar a distribuição obtida aplicando-se PMI.

Dimensionality Reduction – Visualização

Podemos aplicar outras técnicas de redução de dimensionalidade, como PCA e t-SNE, para tentarmos visualizar o resultado do modelo Espaço Vetorial aprendido:

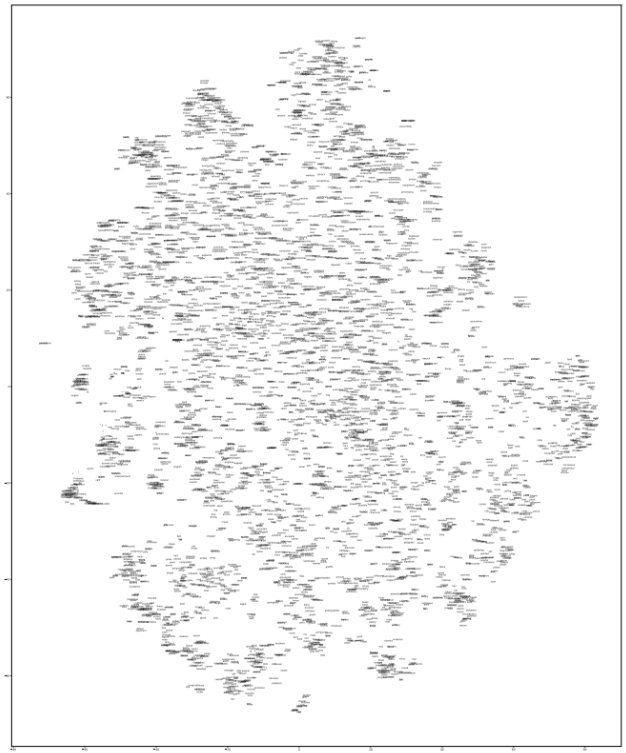
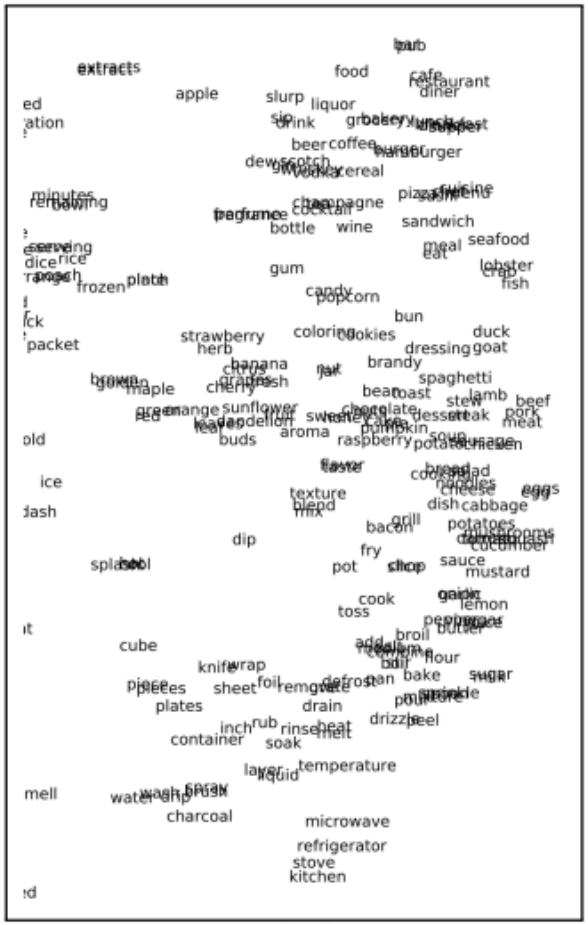
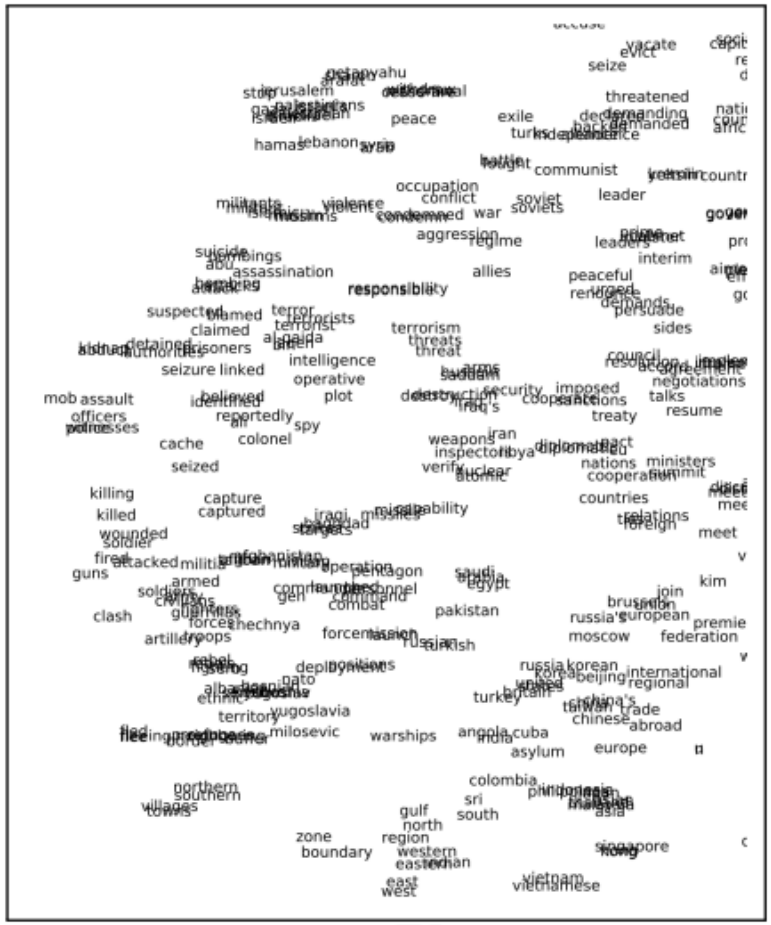


Figura: Resultado visual após aplicar t-SNE em uma matriz do tipo PPMI (giga20 dataset). Note os agrupamentos de palavras.

Dimensionality Reduction – Visualização



cooking



conflict

Figura: Zoom em clusters de palavras relacionadas a temas culinários e conflitos.

Dimensionality Reduction – Visualização

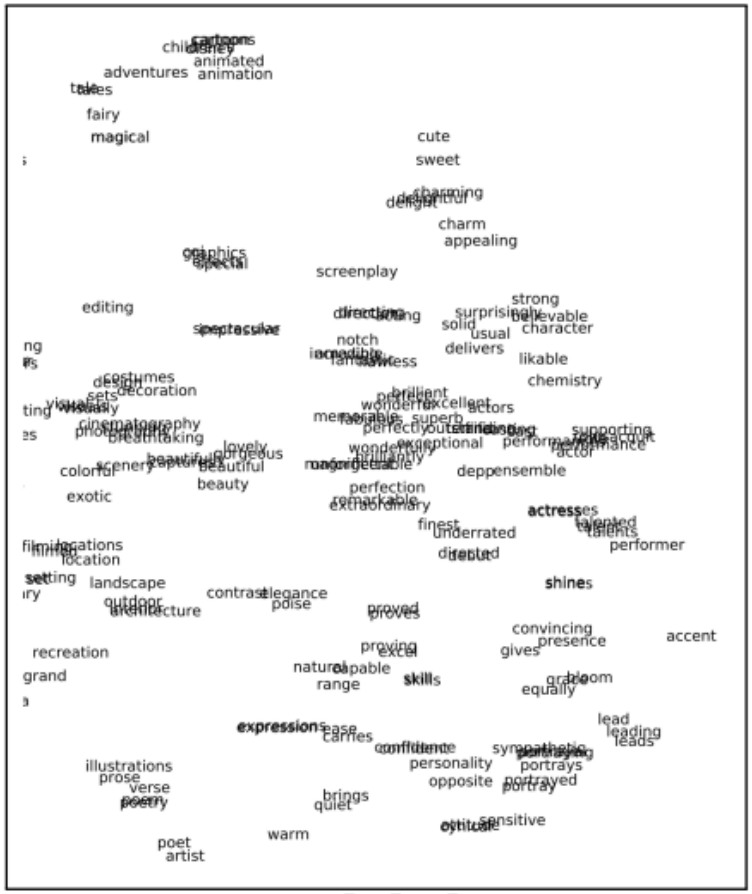


Figura: Zoom em clusters de palavras de conotações positivas e negativas

Referências

Daniel Jurafsky, James H. Martin (2019)
Speech and Language Processing
Capítulo 6 - "Vector Semantics and Embeddings"

Christopher Potts (2021)
Stanford CS224U Natural Language Understanding — Spring 2021
Vector Space Models classes

Jay Allamar
The Illustrated Word2vec

Obrigado!