Processamento de Texto com MATLAB

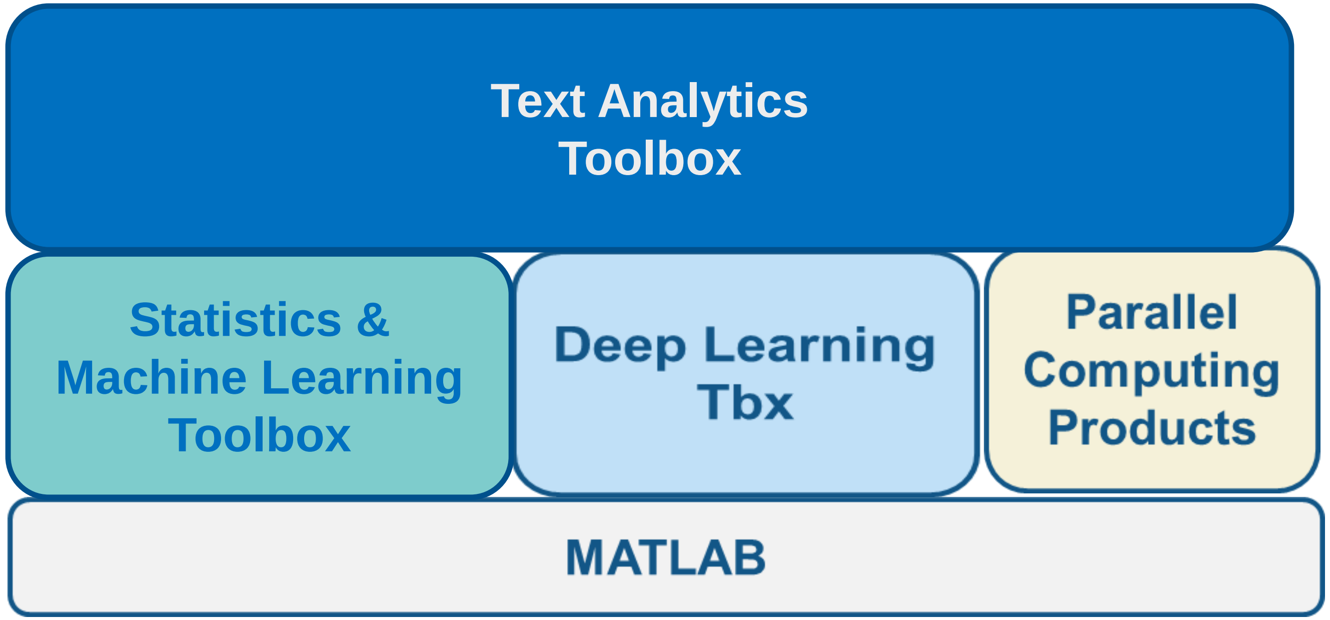
Oct/2019

Alberto Shimahara/Daniel Vieira
alberto.shimahara@opencadd.eng.br
Application Engineer

OPENCADD
MODEL - BASED DESIGN DRIVEN COMPANY
Modeling for life!

Text Analytics Toolbox inicio na R2017b

- Algoritmos para Analise de Texto
  - Extração
  - Visualização
  - N-Gram Counting
  - Bag of words
- Suporte de Machine Learning
  - * LSA,LDA, etc.
- Suporte de Deep Learning
  - Sentiment Analysis
  - LSTM's
- Suporte a Paralelização (PCT, MDCS)



Text Analytics Toolbox

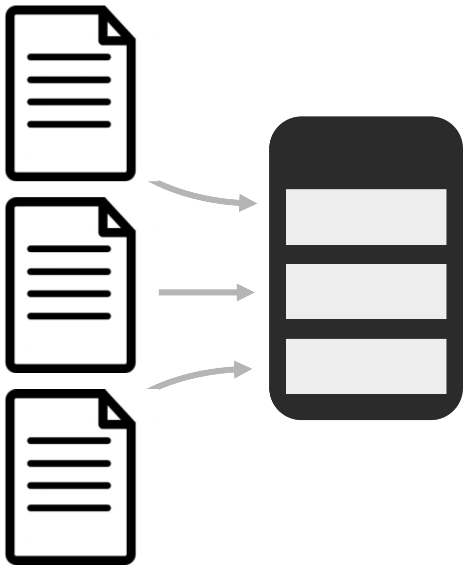
Acesse e Explore Dados

Preprocesse Dados

Desenvolva Modelos Preditivos

Limpar Texto

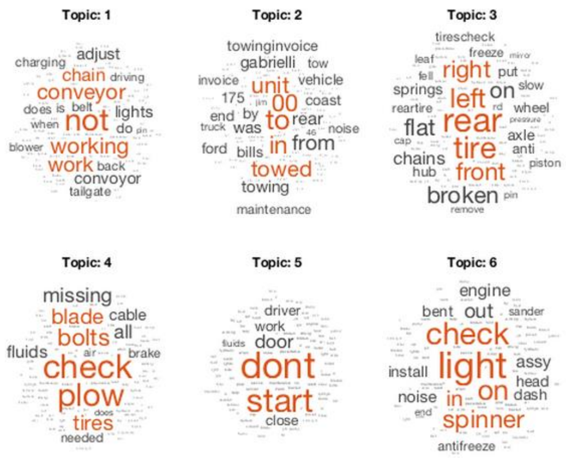
Converte para Numerico



Media reported two trees blown down along I-40 in the Old Fort area.

media report two tree blown down i40 old fort area

	cat	dog	run	two
doc1	1	0	1	0
doc2	1	1	0	1



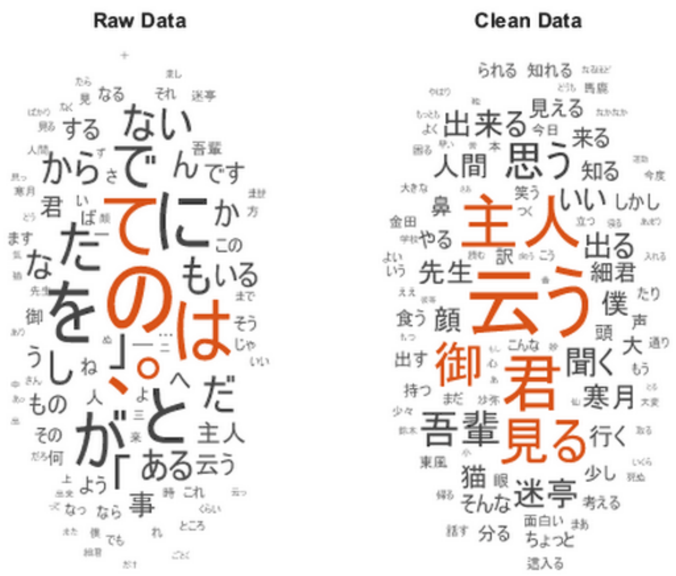
- Word Docs
- PDF's
- Texto
- HTML

- Stop Words
- Stemming
- Tokenization

- Bag of Words
- TF-IDF
- Word Embeddings

- Latent Dirichlet Allocation
- Latent Semantic Analysis

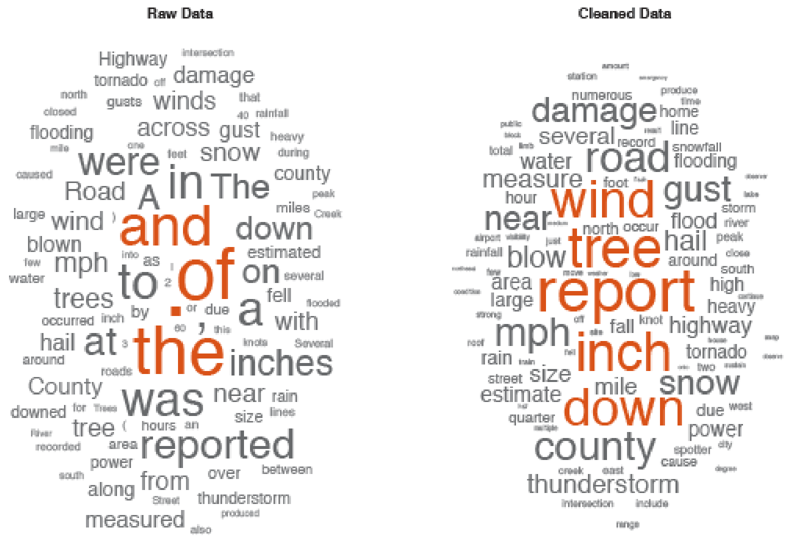
Suporte a Idiomas:



- Ingles
- Alemão
- Japones
- Koreano

Aplica-se outras linguagens, com algumas considerações:

- Tokenization
- Stop Word removal
- Sentence Detection
- Word Cloud
- Word embeddings



20 Newsgroups dataset: http://qwone.com/~jason/20Newsgroups/

Quase 19mil textos sobre 20 tópicos:

- comp.graphics
- comp.os.ms-Windows.misc
- comp.sys.ibm.pc.hardware
- comp.sys.mac.hardware
- comp.windows.x

- rec.autos
- rec.motorcycles

- rec.sport.baseball
- rec.sport.hockey

- sci.crypt
- sci.electronics
- sci.med
- sci.space

- misc.forsale

- talk.politics.misc
- talk.politics.guns
- talk.politics.mideast

- talk.religion.misc
- alt.atheism
- soc.religion.christian

20 Newsgroups dataset: http://qwone.com/~jason/20Newsgroups/

Quase 19mil textos sobre 20 tópicos:

- comp.graphics
- comp.os.ms-Windows.misc
- comp.sys.ibm.pc.hardware
- comp.sys.mac.hardware
- comp.windows.x

computers

- rec.autos
- rec.motorcycles

autos

- rec.sport.baseball
- rec.sport.hockey

sport

- sci.crypt
- sci.electronics
- sci.med
- sci.space

science

- misc.forsale

sales

- talk.politics.misc
- talk.politics.guns
- talk.politics.mideast

politics

- talk.religion.misc
- alt.atheism
- soc.religion.christian

religion

20 Newsgroups dataset: http://qwone.com/~jason/20Newsgroups/

Quase 19mil textos sobre 20 tópicos:

- comp.graphics
- comp.os.ms-Windows.misc
- comp.sys.ibm.pc.hardware
- comp.sys.mac.hardware
- comp.windows.x

computers

- rec.autos
- rec.motorcycles

autos

- rec.sport.baseball
- rec.sport.hockey

sport

- sci.crypt
- sci.electronics
- sci.med
- sci.space

science

- misc.forsale

sales

- talk.politics.misc
- talk.politics.guns
- talk.politics.mideast

politics

- talk.religion.misc
- alt.atheism
- soc.religion.christian

religion

Algumas características:

- E-mails
- Muitos erros de ortografia
- Muitos nomes próprios

20 Newsgroups dataset: http://qwone.com/~jason/20Newsgroups/

Quase 19mil textos sobre 20 tópicos:

- comp.graphics
- comp.os.ms-Windows.misc
- comp.sys.ibm.pc.hardware
- comp.sys.mac.hardware
- comp.windows.x

computers

- rec.autos
- rec.motorcycles

autos

- rec.sport.baseball
- rec.sport.hockey

sport

- sci.crypt
- sci.electronics
- sci.med
- sci.space

science

- misc.forsale

sales

- talk.politics.misc
- talk.politics.guns
- talk.politics.mideast

politics

- talk.religion.misc
- alt.atheism
- soc.religion.christian

religion

Algumas características:

- E-mails
- Muitos erros de ortografia
- Muitos nomes próprios

Exemplo:

From: cs012055@cs.brown.edu (Hok-Chung Tsang) Subject: Re: Saturn's Pricing Polic In article <C4vIr5.L3r@shuksan.ds.boeing.com>, fredd@shuksan (Fred Dickey) writes: |> CarolinaFan@uiuc (cka52397@uxa.cso.uiuc.edu) wrote: |> : I have been active in defending Saturn lately on the net and would |> : like to state my full opinion on the subject, rather than just reply to others' |> : points. |> : |> : The biggest problem some people seem to be having is that Saturn |> : Dealers make ~$2K on a car. I think most will agree with me that the car is |> : comparably priced with its competitors, that is, they aren't overpriced |> : compared to most cars in their class. I don't understand the point of |> : arguing over whether the dealer makes the $2K or not? |> |> I have never understood what the big deal over dealer profits is either. |> The only thing that I can figure out is that people believe that if |> they minimize the dealer profit they will minimize their total out-of-pocket |> expenses for the car. While this may be true in some cases, I do not |> believe that it is generally true. I bought a Saturn SL in January of '92. |> AT THAT TIME, based on studying car prices, I decided that there was |> no comparable car that was priced as cheaply as the Saturn. Sure, maybe I |> could have talked the price for some other car to the Saturn price, but |> my out-of-pocket expenses wouldn't have been any different. What's important |> to me is how much money I have left after I buy the car. REDUCING DEALER PROFIT |> IS NOT THE SAME THING AS SAVING MONEY! Show me how reducing dealer profit |> saves me money, and I'll believe that it's important. My experience has |> been that reducing dealer profit does not necessarily save me money. |> |> Fred Say, you ....

Preprocessamento

REGular EXPression

```
T=strjoin(regexp(T,'\S*@\S*','split'),' ');
```

Preprocessamento

```
T=strjoin(regexp(T,'\S*@\S*','split'),' ');
```

`\S*@\S*` = [Qualquer quantidade de não-espaços] + @ + [qualquer quantidade de não-espaços]

OPENCADD

Preprocessamento

```
T=strjoin(regexp(T,'\S*@\S*','split'),' ');
```

`\S*@\S*` = e-mails!!

Preprocessamento

```
T=strjoin(regexp(T,'\S*@\S*','split'),' ');
```

```
T = {'o que está antes de \S*@\S*'} {'o que está depois de \S*@\S*'}
```

Preprocessamento

```
T=strjoin(regexp(T,'\S*@\S*','split'),' ');
```

```
{'o que está antes'} {'o que está depois'}
```

Preprocessamento

```
T=strjoin(regexp(T,'\S*@\S*','split'),' ');
```

```
T = 'o que está antes o que está depois'
```

Apagou os endereços de e-mail da string

Preprocessamento

```
T=strjoin(regexp(T,'\S*@\S*','split'),' ');

T=strjoin(regexp(T,'\w*\d{1,}\w*','split'),' ');
```

Mesma coisa, para apagar números...

Preprocessamento

```
T=strjoin(regexp(T,'\S*@\S*','split'),' ');

T=strjoin(regexp(T,'\w*\d{1,}\w*','split'),' ');

T=lower(T);

T=erasePunctuation(tokenizedDocument(T));

T=normalizeWords(addPartOfSpeechDetails(addEntityDetails(T)));
```

Tokenização

Marca tokens que podem se referir a “Entidades” (nomes próprios)

Adiciona informação gramatical a cada token (verbos, substantivos, adjetivos, etc)

Radicalização: reduz diversas palavras que derivam de um mesmo radical ao seu radical comum

Ex: boy, boys, boyish → boy

Preprocessamento

```
T=strjoin(regexp(T,'\S*@\S*','split'),' ');
T=strjoin(regexp(T,'\w*\d{1,}\w*','split'),' ');
T=lower(T);
T=erasePunctuation(tokenizedDocument(T));
T=normalizeWords(addPartOfSpeechDetails(addEntityDetails(T)));
T=removeLongWords(removeShortWords(removeStopWords(T),4),20);
```

Remove palavras “inúteis” (preposições, etc)

Remove palavras mais curtas que X caracteres

Remove palavras mais longas que X caracteres

Preprocessamento

```
T=strjoin(regexp(T,'\S*@\S*','split'),' ');
T=strjoin(regexp(T,'\w*\d{1,}\w*','split'),' ');
T=lower(T);
T=erasePunctuation(tokenizedDocument(T));
T=normalizeWords(addPartOfSpeechDetails(addEntityDetails(T)));
T=removeLongWords(removeShortWords(removeStopWords(T),4),20);
```

L *Remove palavras "inúteis" (preposições, etc)*

Remove palavras mais curtas que X caracteres

Remove palavras mais longas que X caracteres

Pode tokenizar em qualquer língua, mas as funções marcadas com "L" só suportam:

Inglês, Alemão, Japonês, Koreano

R2019a R2018b R2019b

Preprocessamento

```
T=strjoin(regexp(T,'\S*@\S*','split'),' ');
T=strjoin(regexp(T,'\w*\d{1,}\w*','split'),' ');
T=lower(T);
T=erasePunctuation(tokenizedDocument(T));
T=normalizeWords(addPartOfSpeechDetails(addEntityDetails(T)));
T=removeLongWords(removeShortWords(removeStopWords(T)));
```

Exemplo de texto “limpo”:

```
t=doc2cell(T);
strjoin(t{1},' ')
```

"hokchung tsang subject saturns pricing policy article dickey writes wrote active defending saturn lately state opinion subject rather reply others points biggest problem people saturn dealers think agree comparably priced competitors overpriced compared class understand point arguing dealer makes never understood dealer profits thing figure people believe minimize dealer profit minimize total outofpocket expenses cases believe generally bought saturn january based studying prices decided comparable priced cheaply saturn maybe talked price saturn price outofpocket expenses wouldnt different important money reducing dealer profit thing saving money reducing dealer profit saves money believe important experience reducing dealer profit necessarily money bought saturn dealer profit dealer profit paying saving money moreover saturn really reduce dealer profit margin better deals price saturn already below market average class reduce dealer profit below market average attract people saturns money force competitors lower prices survive saturn owners benefit lower dealer profit buyers saving money ....”

Classificação de Texto com Bag of Words / Bag of N-Grams

Bag of Words / N-Grams

```
bagOfWords with properties:

        Counts: [18828×15847 double]
    Vocabulary: [1×15847 string]
      NumWords: 15847
  NumDocuments: 18828
```

Ordenado

`17 tokens: randy subject headlights problem thanks responded posting problem trucks headlights problem loose connection minority suggested thanks again`

Desordenado

Classificação de Texto com Bag of Words / Bag of N-Grams

Bag of Words / N-Grams

```
 200 tokens: subject re lexan polish from jonathan jefferies
5373 tokens: from dances with bikers subject faq what is the
 136 tokens: from ken buck subject re do trains have radar
 195 tokens: from herschel h mayo subject re braindead
 133 tokens: from ronald j deblock jr subject re changing
 196 tokens: from les bartel subject re aftermarket
 253 tokens: from herschel h mayo subject re braindead
 251 tokens: from trevor paquette subject my day in court
 236 tokens: from larry rogers subject re saturn manual
  43 tokens: from michael j sawicki cta subject regal
 280 tokens: from brent woody moss subject re do trains have
 280 tokens: from bill mayhew subject re electronic
 119 tokens: from john r daker subject re open letter to
 725 tokens: subject aftermarket cruise controls specific
 170 tokens: from daniel u holbrook subject re did us drive
  78 tokens: from john r daker subject re license plates
  78 tokens: from john r daker subject re isuzu amigo
  77 tokens: from john r daker subject re license plates in
 249 tokens: from charles parr subject re truck tailgates
 167 tokens: from srinagesh gavirneni subject chevy sprint
 308 tokens: from eric youngblood subject re old corvettes
 110 tokens: from steven j orlin subject re changing oil by
 199 tokens: from robert j dilmore subject re dumbest
 202 tokens: from rick open vms colombo subject re do trains
 166 tokens: from malcolm g costello subject re sprayedon
 605 tokens: from mark monninger subject car buying story
```

```
B=bagOfWords(T);
```

```
bagOfWords with properties:

       Counts: [18828×89033 double]
   Vocabulary: [1×89033 string]
     NumWords: 89033
 NumDocuments: 18828
```

```
B=removeInfrequentWords(B,10);
```

```
bagOfWords with properties:

       Counts: [18828×15847 double]
   Vocabulary: [1×15847 string]
     NumWords: 15847
 NumDocuments: 18828
```

Bag of Words / N-Grams

```
 200 tokens: subject re lexan polish from jonathan jefferies
5373 tokens: from dances with bikers subject faq what is the
 136 tokens: from ken buck subject re do trains have radar
 195 tokens: from herschel h mayo subject re braindead drive
 133 tokens: from ronald j deblock jr subject re changing oi
 196 tokens: from les bartel subject re aftermarket aircondi
 253 tokens: from herschel h mayo subject re braindead drive
 251 tokens: from trevor paquette subject my day in court re
 236 tokens: from larry rogers subject re saturn manual tran
  43 tokens: from michael j sawicki cta subject regal fiberg
 280 tokens: from brent woody moss subject re do trains have
 280 tokens: from bill mayhew subject re electronic odometer
 119 tokens: from john r daker subject re open letter to nis
 725 tokens: subject aftermarket cruise controls specific qu
 170 tokens: from daniel u holbrook subject re did us drive
  78 tokens: from john r daker subject re license plates nis
  78 tokens: from john r daker subject re isuzu amigo opinio
  77 tokens: from john r daker subject re license plates in
 249 tokens: from charles parr subject re truck tailgates mi
 167 tokens: from srinagesh gavirneni subject chevy sprint i
 308 tokens: from eric youngblood subject re old corvettes l
 110 tokens: from steven j orlin subject re changing oil by
 199 tokens: from robert j dilmore subject re dumbest automo
 202 tokens: from rick open vms colombo subject re do trains
 166 tokens: from malcolm g costello subject re sprayedon be
 605 tokens: from mark monninger subject car buying story wa
```

```
B=bagOfWords(T);
```

```
bagOfWords with properties:

       Counts: [18828×89033 double]
   Vocabulary: [1×89033 string]
     NumWords: 89033
 NumDocuments: 18828
```

```
B=removeInfrequentWords(B,10);
```

```
bagOfWords with properties:

       Counts: [18828×15847 double]
   Vocabulary: [1×15847 string]
     NumWords: 15847
 NumDocuments: 18828
```

Alternativamente...

```
B=bagOfNgrams(T,1:5);
B=removeInfrequentNgrams(T,10);
```

Classificação de Texto com Bag of Words / Bag of N-Grams

Bag of Words / N-Grams

```
bagOfWords with properties:

        Counts: [18828×15847 double]
    Vocabulary: [1×15847 string]
      NumWords: 15847
  NumDocuments: 18828
```

B.Counts: Ndocs x Nwords (double) $\longrightarrow$ *X*

labels: Ndocs x 1 (categorical) $\longrightarrow$ *Y*

Bag of Words / N-Grams

```
bagOfWords with properties:

       Counts: [18828×15847 double]
   Vocabulary: [1×15847 string]
     NumWords: 15847
 NumDocuments: 18828
```

B.Counts: Ndocs x Nwords (double) → *X*

labels: Ndocs x 1 (categorical) → *Y*

A partir daqui é "só" um problema de Machine Learning simples...
...de 15847 dimensões
Exemplo:

```
model=fitcnb(X,Y);          % treino
newY=predict(model,newX);   % uso
```

Novo texto preprocessado e tokenizado
da mesma forma que os textos de treino

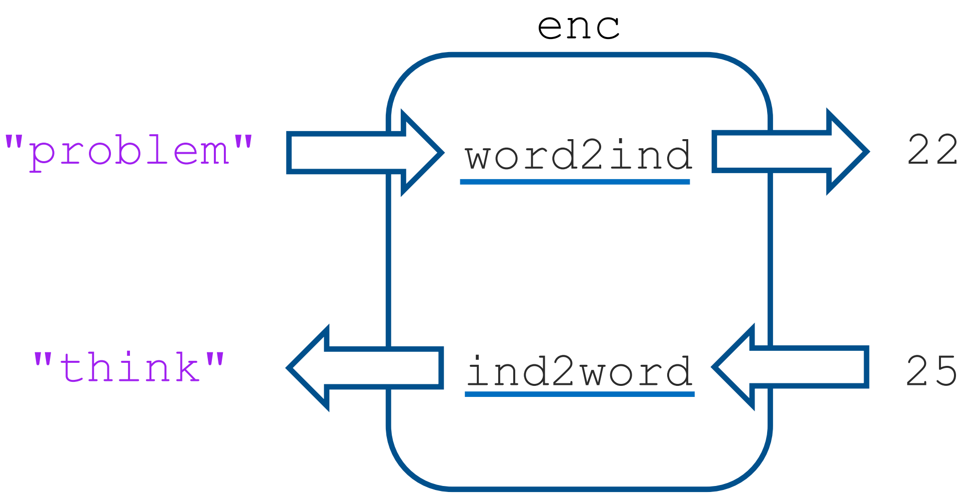
...mesmo dataset e preprocessamento similar...

Word Encoding

tokenizedDocument

```
enc=wordEncoding(T);
```

```
wordEncoding with properties:

    NumWords: 89033
  Vocabulary: [1×89033 string]
```




```
word2ind(enc,"problem")

ind2word(enc,25)
```

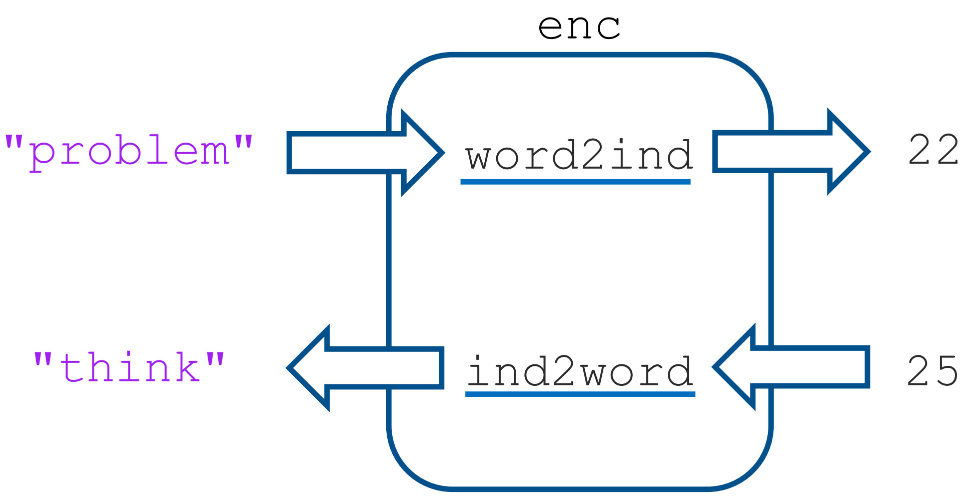
...mesmo dataset e preprocessamento similar...

Word Encoding

tokenizedDocument

```
enc=wordEncoding(T);
```

```
wordEncoding with properties:

     NumWords: 89033
   Vocabulary: [1×89033 string]
```




```
word2ind(enc,"problem")

ind2word(enc,25)
```

```
T(3)
17 tokens: randy subject headlights problem
thanks responded posting problem trucks
headlights problem loose connection minority
suggested thanks again
```

```
A=doc2sequence(enc,T(3))
A{1}
118    3   119   22   120   121   122   22   123   119 ...
```

Classificação de sequências

```
inputSize = 1;
embeddingDimension = 64;
numWords = enc.NumWords;
numHiddenUnits = 32;
numClasses = numel(categories(labels));

layers=[sequenceInputLayer(inputSize)
    wordEmbeddingLayer(embeddingDimension,numWords)
    lstmLayer(numHiddenUnits,'OutputMode','last')
    dropoutLayer(0.3)
    fullyConnectedLayer(numClasses)
    softmaxLayer
    classificationLayer];
```

Classificação de sequências

1 sequencia a ser classificada

```
inputSize = 1;
embeddingDimension = 64;
numWords = enc.NumWords;
numHiddenUnits = 32;
numClasses = numel(categories(labels));

layers=[sequenceInputLayer(inputSize)
    wordEmbeddingLayer(embeddingDimension,numWords)
    lstmLayer(numHiddenUnits,'OutputMode','last')
    dropoutLayer(0.3)
    fullyConnectedLayer(numClasses)
    softmaxLayer
    classificationLayer];
```

Classificação de sequências

```
inputSize = 1;
embeddingDimension = 64;
numWords = enc.NumWords;
numHiddenUnits = 32;
numClasses = numel(categories(labels));

layers=[sequenceInputLayer(inputSize)
    wordEmbeddingLayer(embeddingDimension,numWords)
    lstmLayer(numHiddenUnits,'OutputMode','last')
    dropoutLayer(0.3)
    fullyConnectedLayer(numClasses)
    softmaxLayer
    classificationLayer];
```

Word Embedding: essa camada tem a função de aprender uma representação de cada código (palavra) por um vetor de dimensão `embeddingDimension`

Uma ***boa*** *representação será capaz de expressar relações semânticas entre palavras de forma algébrica, por exemplo:*

$$king - man = queen - woman = monarch$$

$$france - paris = italy - rome$$

Classificação de sequências

```
inputSize = 1;
embeddingDimension = 64;
numWords = enc.NumWords;
numHiddenUnits = 32;
numClasses = numel(categories(labels));

layers=[sequenceInputLayer(inputSize)
    wordEmbeddingLayer(embeddingDimension,numWords)
    lstmLayer(numHiddenUnits,'OutputMode','last')
    dropoutLayer(0.3)
    fullyConnectedLayer(numClasses)
    softmaxLayer
    classificationLayer];
```

Para cada elemento da sequência (vetorizado por `wordEmbeddingLayer`*):*

- *Produz um resultado de tamanho* `numHiddenUnits`
- *Atualiza o estado da* `lstmLayer`
- *Passa para o próximo elemento da sequência*

Ao final da sequência, reseta.

Essa camada aprende relações entre os elementos da sequência
A saída dessa camada é um "feature vector" para fazer a classificação (`fullyConnected`*,* `softmax`*)*

Classificação de sequências

```
inputSize = 1;
embeddingDimension = 64;
numWords = enc.NumWords;
numHiddenUnits = 32;
numClasses = numel(categories(labels));

layers=[sequenceInputLayer(inputSize)
    wordEmbeddingLayer(embeddingDimension,numWords)
    lstmLayer(numHiddenUnits,'OutputMode','last')
    dropoutLayer(0.3)
    fullyConnectedLayer(numClasses)
    softmaxLayer
    classificationLayer];
```

```
118    3    119    22    120 ...
0.8147
0.9058
0.1270
0.9134
0.6324
...
```

Para cada elemento da sequência (vetorizado por `wordEmbeddingLayer`*):*

- *Produz um resultado de tamanho* `numHiddenUnits`
- *Atualiza o estado da* `lstmLayer`
- *Passa para o próximo elemento da sequência*

Ao final da sequência, reseta.

Essa camada aprende relações entre os elementos da sequência
A saída dessa camada é um "feature vector" para fazer a classificação (`fullyConnected`*,* `softmax`*)*

Classificação de sequências

```
inputSize = 1;
embeddingDimension = 64;
numWords = enc.NumWords;
numHiddenUnits = 32;
numClasses = numel(categories(labels));

layers=[sequenceInputLayer(inputSize)
    wordEmbeddingLayer(embeddingDimension,numWords)
    lstmLayer(numHiddenUnits,'OutputMode','last')
    dropoutLayer(0.3)
    fullyConnectedLayer(numClasses)
    softmaxLayer
    classificationLayer];
```

```
118 → 3     119     22     120 ...

0.8147    0.0975
0.9058    0.2785
0.1270    0.5469
0.9134    0.9575
0.6324    0.9649
...       ...
```

Para cada elemento da sequência (vetorizado por `wordEmbeddingLayer`*):*

- *Produz um resultado de tamanho* `numHiddenUnits`
- *Atualiza o estado da* `lstmLayer`
- *Passa para o próximo elemento da sequência*

Ao final da sequência, reseta.

Essa camada aprende relações entre os elementos da sequência
A saída dessa camada é um "feature vector" para fazer a classificação (`fullyConnected`*,* `softmax`*)*

Classificação de sequências

```
inputSize = 1;
embeddingDimension = 64;
numWords = enc.NumWords;
numHiddenUnits = 32;
numClasses = numel(categories(labels));

layers=[sequenceInputLayer(inputSize)
    wordEmbeddingLayer(embeddingDimension,numWords)
    lstmLayer(numHiddenUnits,'OutputMode','last')
    dropoutLayer(0.3)
    fullyConnectedLayer(numClasses)
    softmaxLayer
    classificationLayer];
```

118 → 3 → 119   22   120 ...

0.8147	0.0975	0.7577
0.9058	0.2785	0.7431
0.1270	0.5469	0.3922
0.9134	0.9575	0.6555
0.6324	0.9649	0.1712
...	...	...

Para cada elemento da sequência (vetorizado por `wordEmbeddingLayer`*):*

- *Produz um resultado de tamanho* `numHiddenUnits`
- *Atualiza o estado da* `lstmLayer`
- *Passa para o próximo elemento da sequência*

Ao final da sequência, reseta.

Essa camada aprende relações entre os elementos da sequência
A saída dessa camada é um “feature vector” para fazer a classificação (`fullyConnected`*,* `softmax`*)*

Classificação de sequências

```
inputSize = 1;
embeddingDimension = 64;
numWords = enc.NumWords;
numHiddenUnits = 32;
numClasses = numel(categories(labels));

layers=[sequenceInputLayer(inputSize)
    wordEmbeddingLayer(embeddingDimension,numWords)
    lstmLayer(numHiddenUnits,'OutputMode','last')
    dropoutLayer(0.3)
    fullyConnectedLayer(numClasses)
    softmaxLayer
    classificationLayer];
```

118 → 3 → 119 → 22   120 ...

118	3	119	22
0.8147	0.0975	0.7577	0.6557
0.9058	0.2785	0.7431	0.0357
0.1270	0.5469	0.3922	0.8491
0.9134	0.9575	0.6555	0.9340
0.6324	0.9649	0.1712	0.6787
...	...	...	...

Para cada elemento da sequência (vetorizado por `wordEmbeddingLayer`*):*

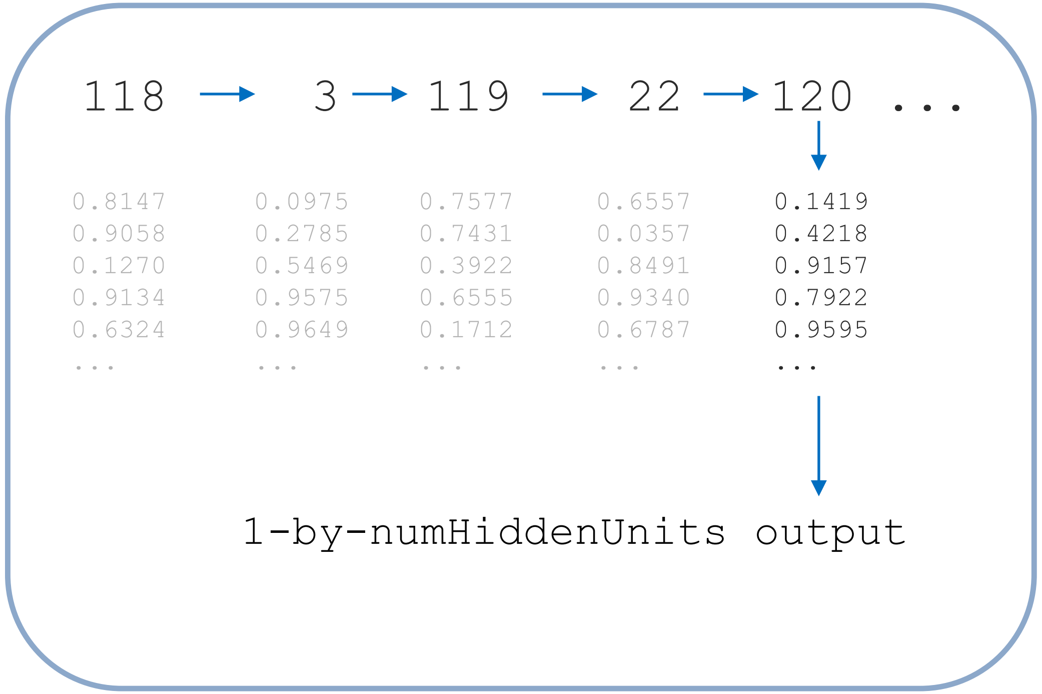
- *Produz um resultado de tamanho* `numHiddenUnits`
- *Atualiza o estado da* `lstmLayer`
- *Passa para o próximo elemento da sequência*

Ao final da sequência, reseta.

Essa camada aprende relações entre os elementos da sequência
A saída dessa camada é um "feature vector" para fazer a classificação (`fullyConnected`*,* `softmax`*)*

Classificação de sequências

```
inputSize = 1;
embeddingDimension = 64;
numWords = enc.NumWords;
numHiddenUnits = 32;
numClasses = numel(categories(labels));

layers=[sequenceInputLayer(inputSize)
     wordEmbeddingLayer(embeddingDimension,numWords)
     lstmLayer(numHiddenUnits,'OutputMode','last')
     dropoutLayer(0.3)
     fullyConnectedLayer(numClasses)
     softmaxLayer
     classificationLayer];
```



Para cada elemento da sequência (vetorizado por `wordEmbeddingLayer`*):*

- *Produz um resultado de tamanho* `numHiddenUnits`
- *Atualiza o estado da* `lstmLayer`
- *Passa para o próximo elemento da sequência*

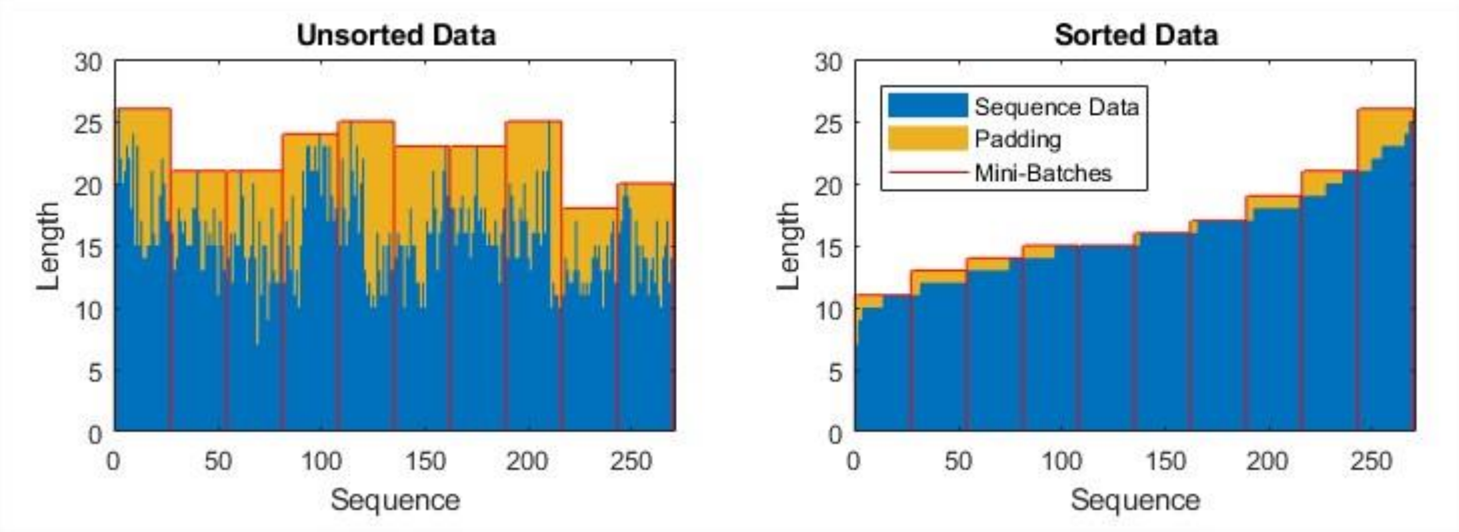
Ao final da sequência, reseta.

Essa camada aprende relações entre os elementos da sequência
A saída dessa camada é um “feature vector” para fazer a classificação (`fullyConnected`*,* `softmax`*)*

Treinamento

```
T, labels → txtTrain, labelsTrain
          → txtValid, labelsValid
          → txtTest, labelsTest
```

```
% ordenação
L=doclength(txtTrain);
[~,order]=sort(L,'ascend');
txtTrain=txtTrain(order);
labelsTrain=labelsTrain(order);
```

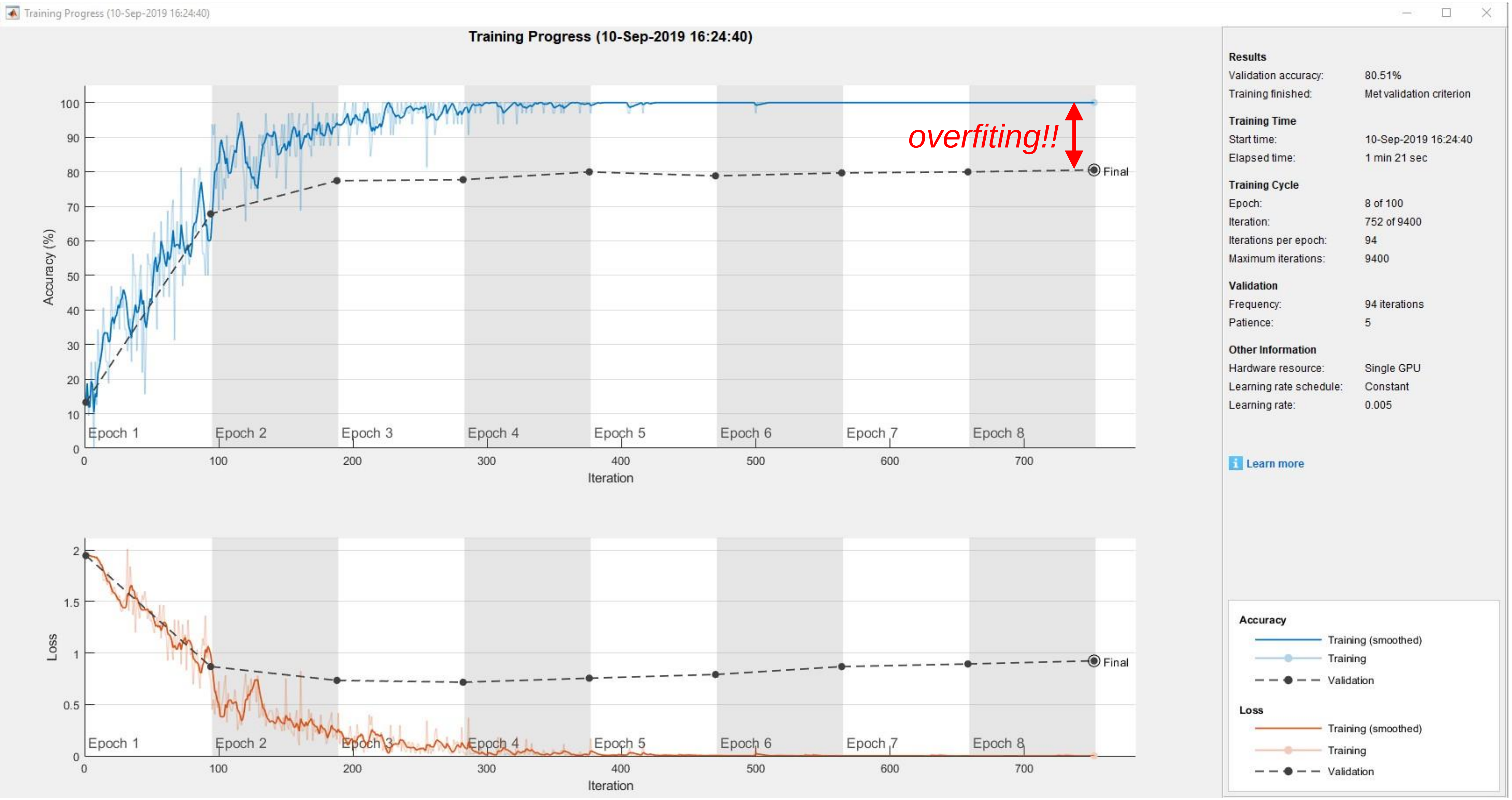


Ordenar textos por tamanho minimiza a quantidade de zero-padding necessária em cada minibatch e melhora o resultado

```
% truncamento da sequência
XTrain=doc2sequence(enc,txtTrain,'Length',median(L));
XValid=doc2sequence(enc,txtValid,'Length',median(L));


minibatch=32;
batchesPerEpoch=floor(numel(txtTrain)./minibatch);

opts=trainingOptions('adam','MaxEpochs',100,'MiniBatchSize',minibatch, ...
    'GradientThreshold',1,'InitialLearnRate',0.005, ...
    'ValidationData',{XValid,labelsValid},'ValidationPatience',5,'ValidationFrequency',batchesPerEpoch, ...
    'Plots','training-progress','Verbose',false,'Shuffle','never');


[net,trInfo]=trainNetwork(XTrain,labelsTrain,layers,opts);
```

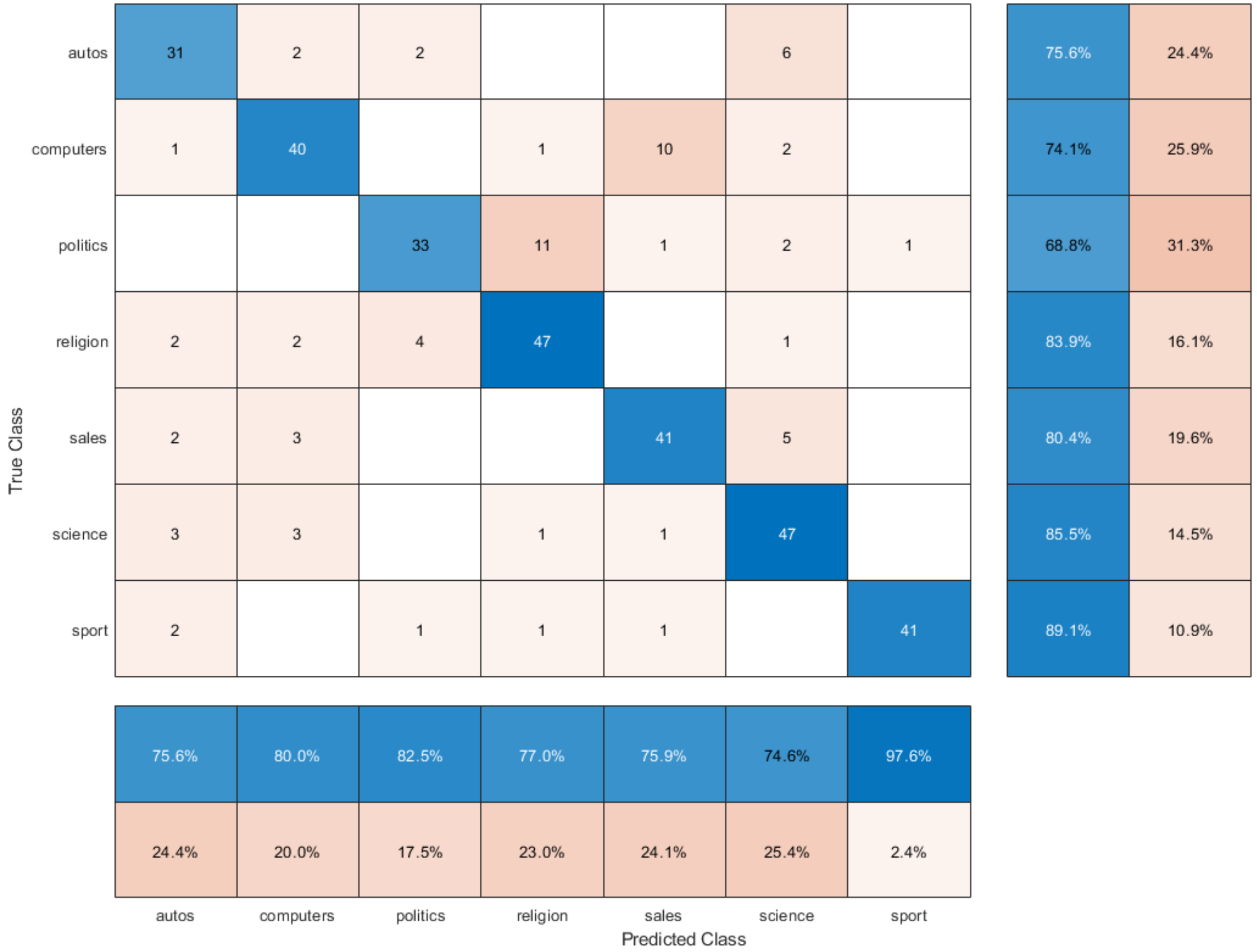
Se ordenou, não embaralhe...

Avaliação de performance



Avaliação de performance

```
Xtest=doc2sequence(enc,txtTest);
[classes,scores]=classify(net,Xtest);

figure, confusionchart(topicTest,classes,'RowSummary','row-normalized','ColumnSummary','column-normalized')
```



Reuters dataset: https://martin-thoma.com/nlp-reuters/

Características:

- *Notícias de economia e mercado financeiro*
- *10788 textos*
- *90 classes não balanceadas*
- *Cada texto pode ter várias classes (a maioria tem 1 ou 2 classes, mas pode ter até 15)*
- *Número médio de palavras por texto agrupados por classe: de 93 a 1263*
- *Vários Labels (assuntos) fortemente correlacionados*
- *Este dataset é muito usado para benchmarking de modelos de NLP (Natural Language Processing)*

Reuters dataset: https://martin-thoma.com/nlp-reuters/

Características:

- *Notícias de economia e mercado financeiro*
- *10788 textos*
- ~~*90 classes não balanceadas*~~
- ~~*Cada texto pode ter várias classes (a maioria tem 1 ou 2 classes, mas pode ter até 15)*~~
- *Número médio de palavras por texto agrupados por classe: de 93 a 1263*
- *Vários Labels (assuntos) fortemente correlacionados*
- *Este dataset é muito usado para benchmarking de modelos de NLP (Natural Language Processing)*

Não importa nesse caso,
usaremos aprendizado não-supervisionado

Preprocessamento

```
txt=tokenizedDocument(data.T);
txt=erasePunctuation(txt);
txt=removeLongWords(txt,20);
txt=removeShortWords(txt,4);
txt=removeStopWords(txt);

B=bagOfWords(txt);
B=removeInfrequentWords(B,50);
[~,ind]=maxk(sum(B.Counts~=0,1),25,2);
frequentWords=B.Words(ind);
B=removeWords(B,frequentWords);
B=removeEmptyDocuments(B);
```

Original

```
"Showers continued throughout the week in the Bahia
cocoa zone, alleviating the drought since early January
and improving prospects for the coming temporao,
although normal humidity levels have not been restored,
Comissaria Smith said in its weekly review. The dry
period means the temporao will ...”
```

Limpo

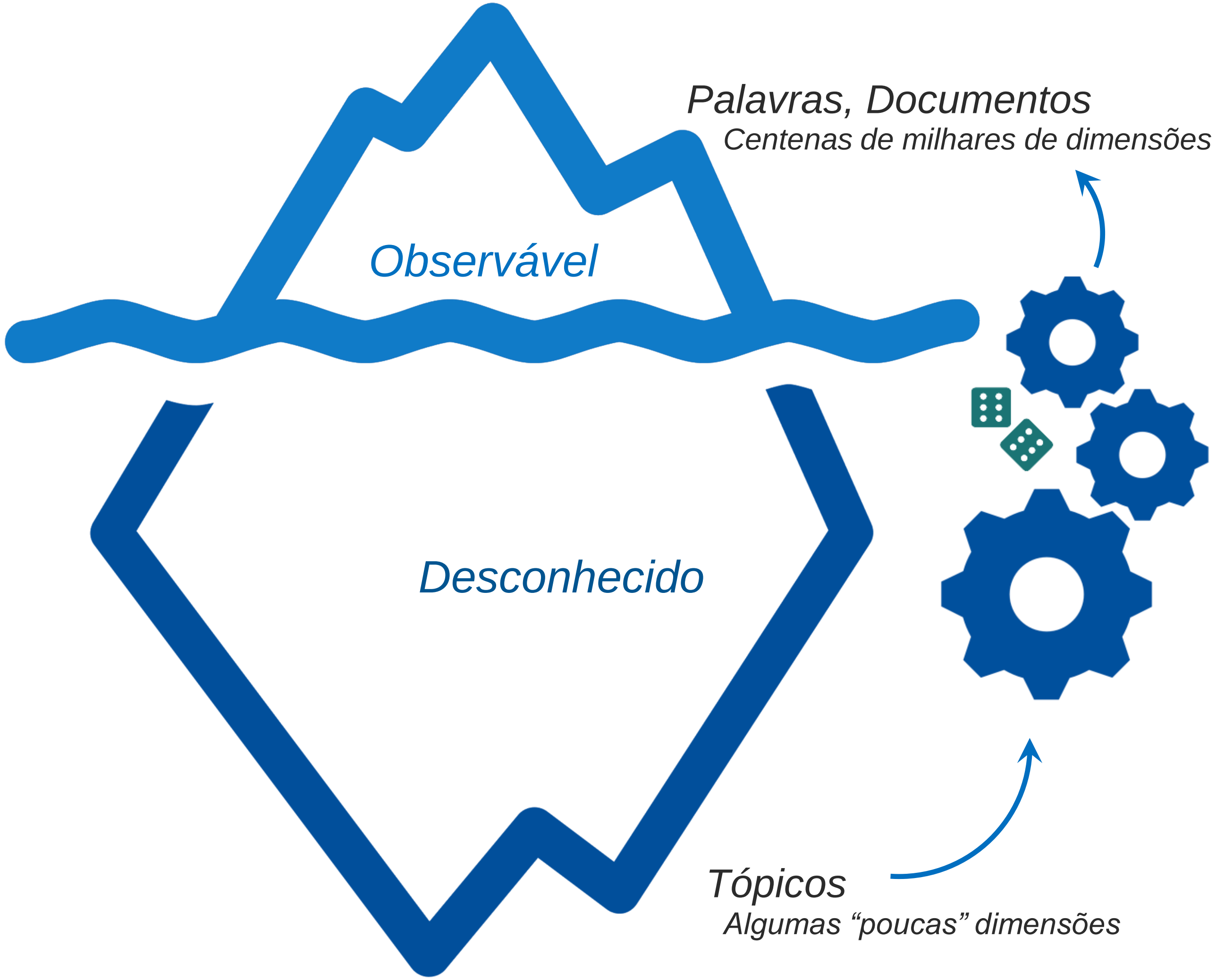
```
172 tokens: Showers continued throughout Bahia cocoa
alleviating drought early January improving prospects
coming temporao although normal humidity levels
restored Comissaria Smith weekly review period means
temporao ...
```

Conceito do o modelo LDA

Palavras, Documentos

Centenas de milhares de dimensões

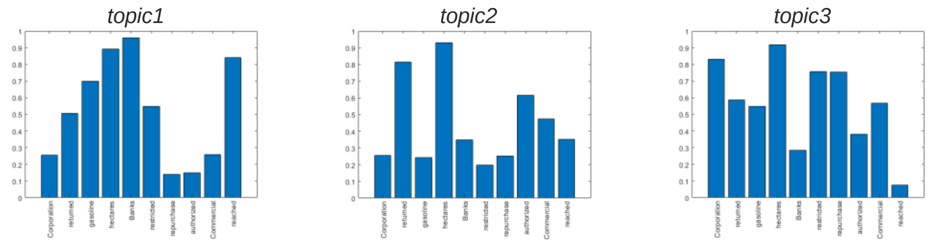
Conceito do o modelo LDA



Processo de Geração de Documentos

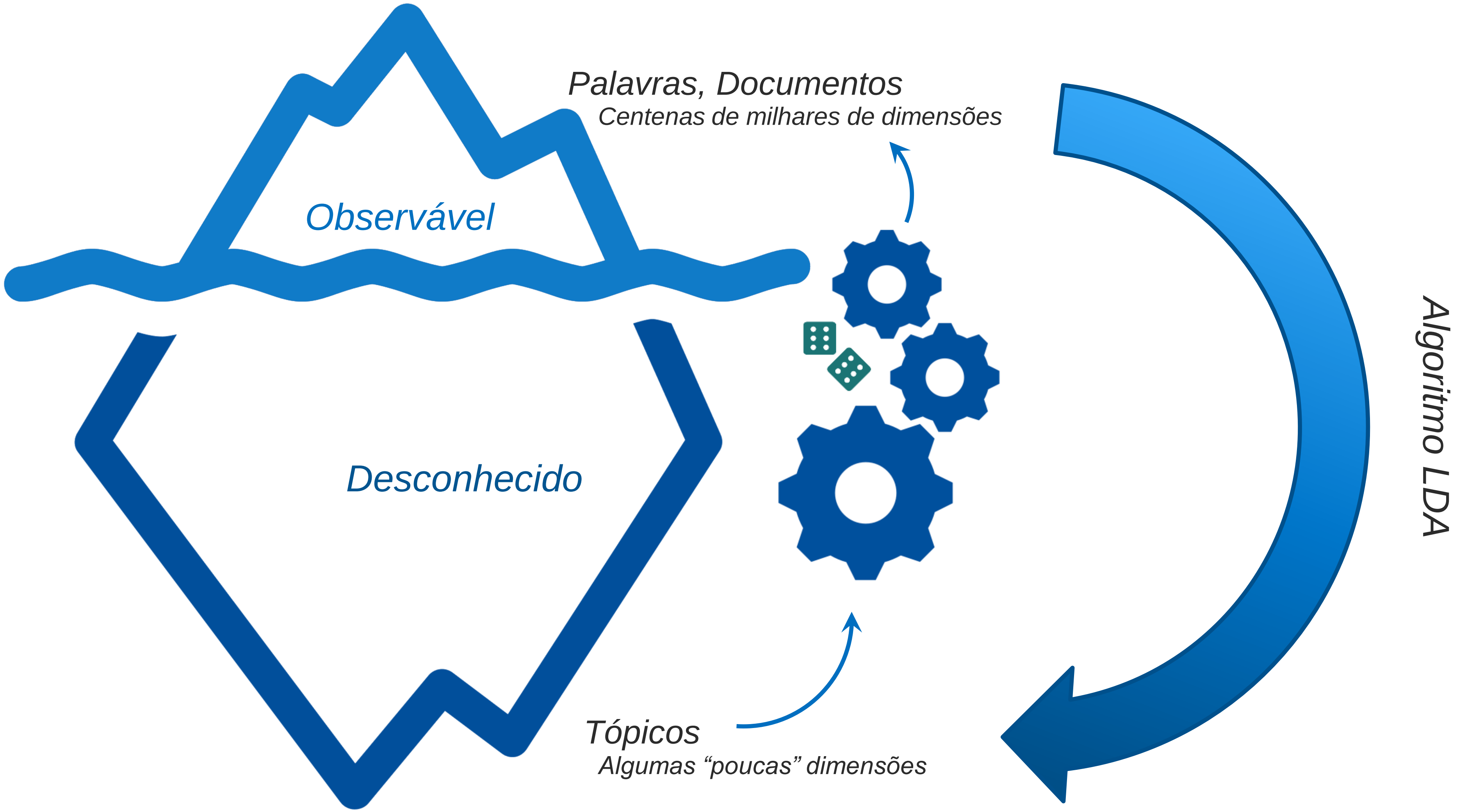
Assume-se:

- *Cada documento é uma mistura de tópicos*
- *Há um processo que gera os documentos amostrando sequencialmente de forma aleatória palavras de acordo com os tópicos do documento*



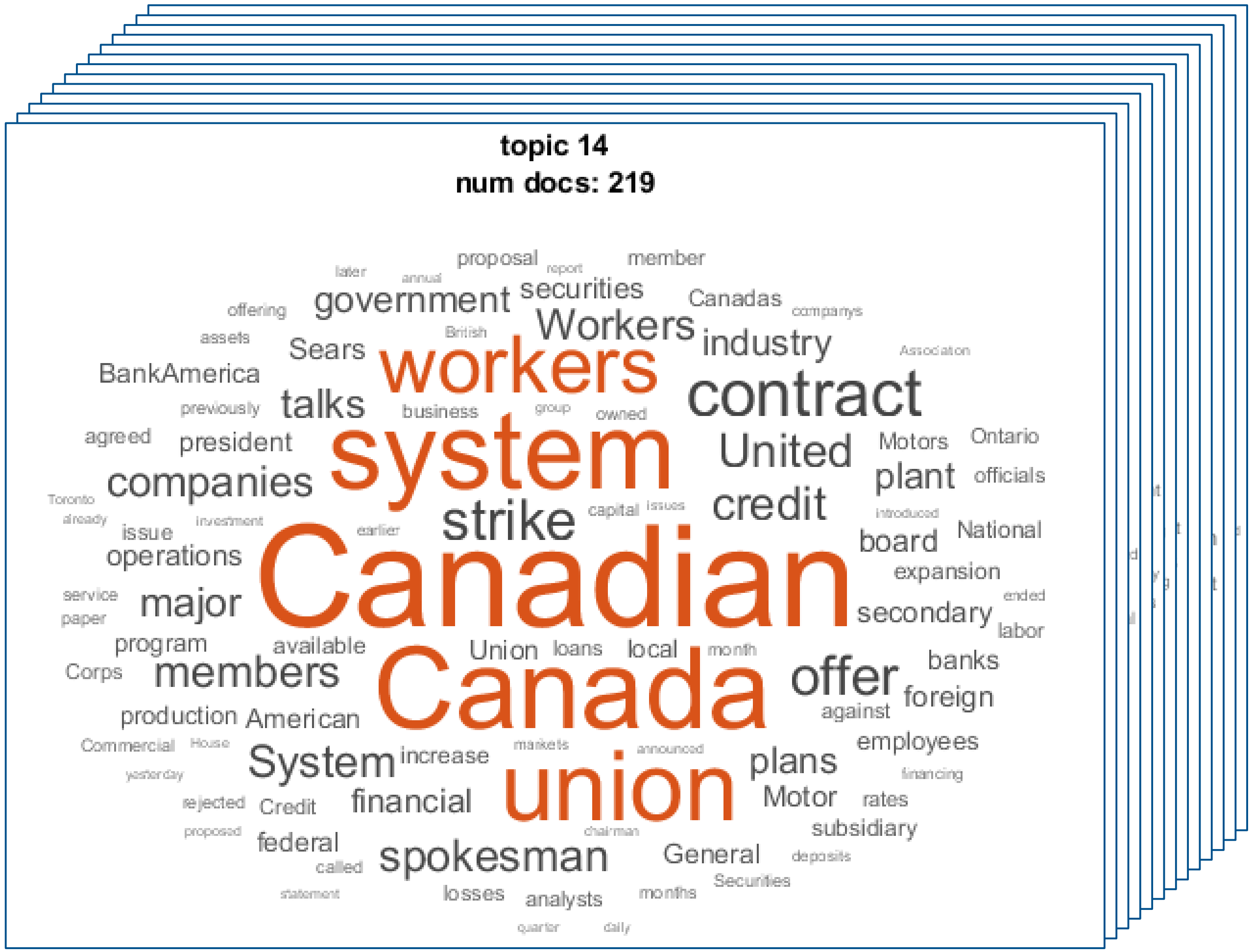
. . .

Conceito do o modelo LDA



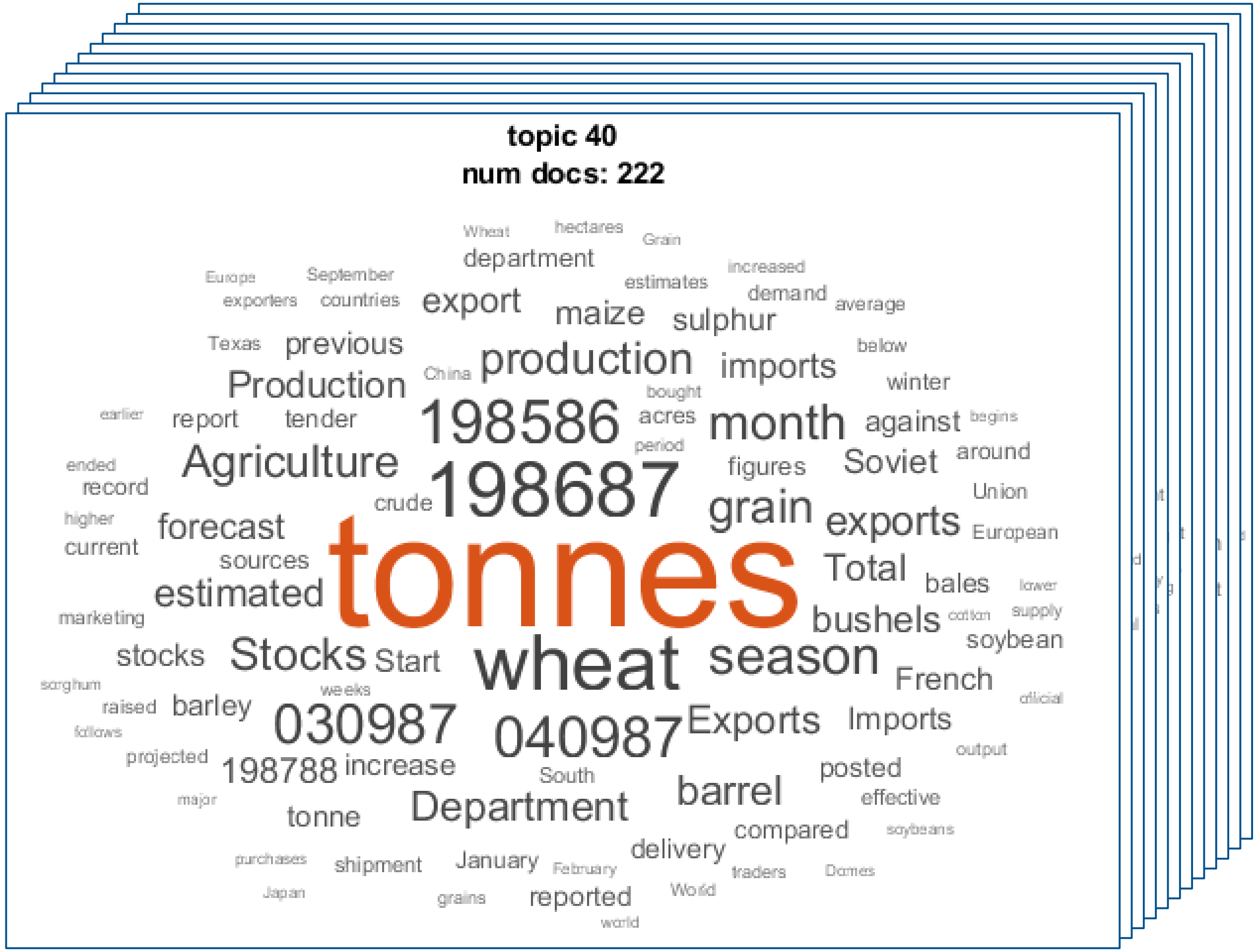
Inferência de Tópicos com Latent Dirichlet Allocation (LDA)

```
numTopics = 40;
mdl = fitlda(B,numTopics,...
    'Verbose',1,...
    'Solver','savb',...
    'FitTopicConcentration',true,...
    'DataPassLimit',10);

[~,topicInd] = max(...
mdl.DocumentTopicProbabilities,[],2);


for k = ...
    subB=removeDocument(B,find(topicInd~=k));
    figure, wordcloud(subB);
end
```



Inferência de Tópicos com Latent Dirichlet Allocation (LDA)

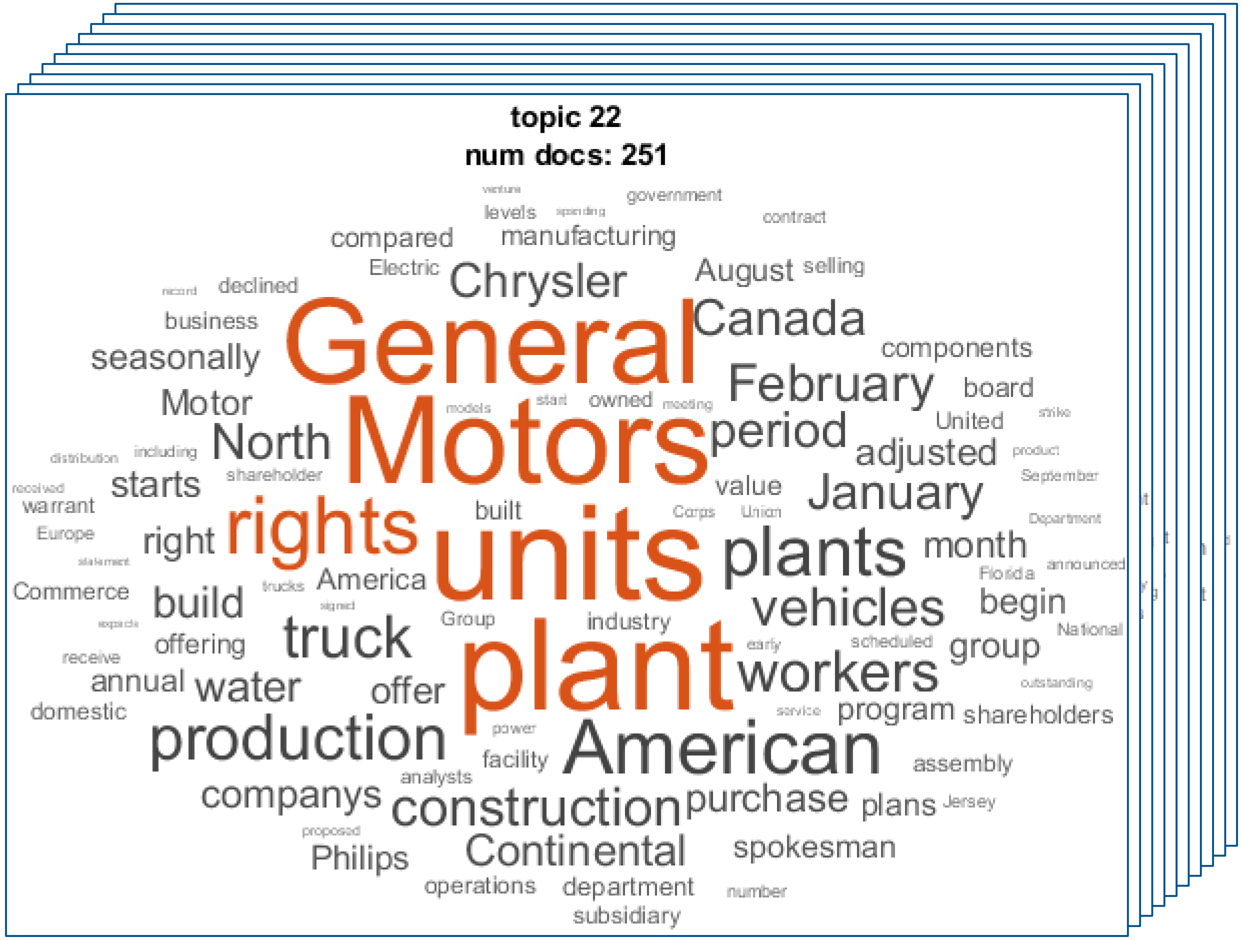
```
numTopics = 40;
mdl = fitlda(B,numTopics,...
    'Verbose',1,...
    'Solver','savb',...
    'FitTopicConcentration',true,...
    'DataPassLimit',10);

[~,topicInd] = max(...
mdl.DocumentTopicProbabilities,[],2);



for k = ...
    subB=removeDocument(B,find(topicInd~=k));
    figure, wordcloud(subB);
end
```



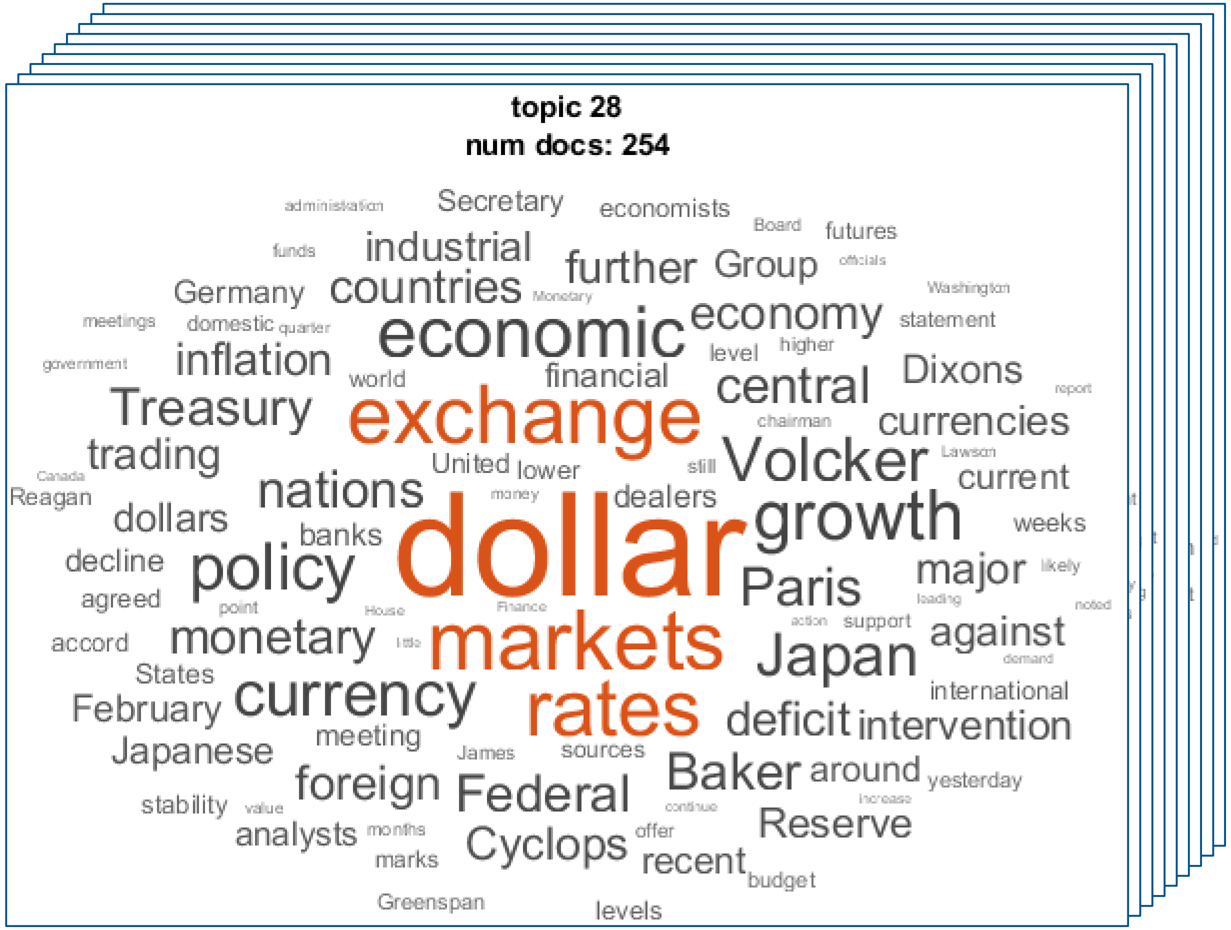
```
numTopics = 40;
mdl = fitlda(B,numTopics,...
    'Verbose',1,...
    'Solver','savb',...
    'FitTopicConcentration',true,...
    'DataPassLimit',10);

[~,topicInd] = max(...
mdl.DocumentTopicProbabilities,[],2);



for k = ...
    subB=removeDocument(B,find(topicInd~=k));
    figure, wordcloud(subB);
end
```

Inferência de Tópicos com Latent Dirichlet Allocation (LDA)

```
numTopics = 40;
mdl = fitlda(B,numTopics,...
    'Verbose',1,...
    'Solver','savb',...
    'FitTopicConcentration',true,...
    'DataPassLimit',10);

[~,topicInd] = max(...
mdl.DocumentTopicProbabilities,[],2);



for k = ...
    subB=removeDocument(B,find(topicInd~=k));
    figure, wordcloud(subB);
end
```

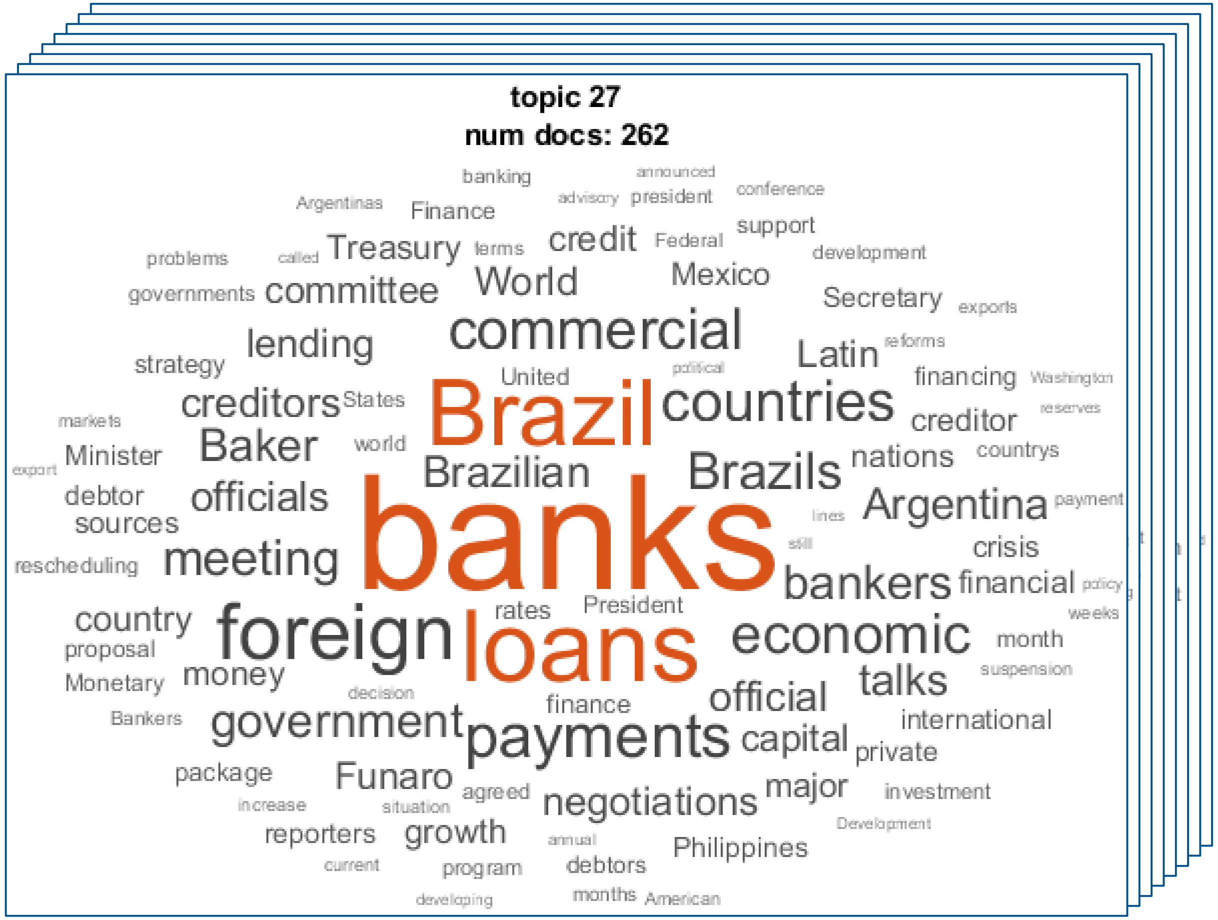


```
numTopics = 40;
mdl = fitlda(B,numTopics,...
    'Verbose',1,...
    'Solver','savb',...
    'FitTopicConcentration',true,...
    'DataPassLimit',10);

[~,topicInd] = max(...
mdl.DocumentTopicProbabilities,[],2);



for k = ...
    subB=removeDocument(B,find(topicInd~=k));
    figure, wordcloud(subB);
end
```



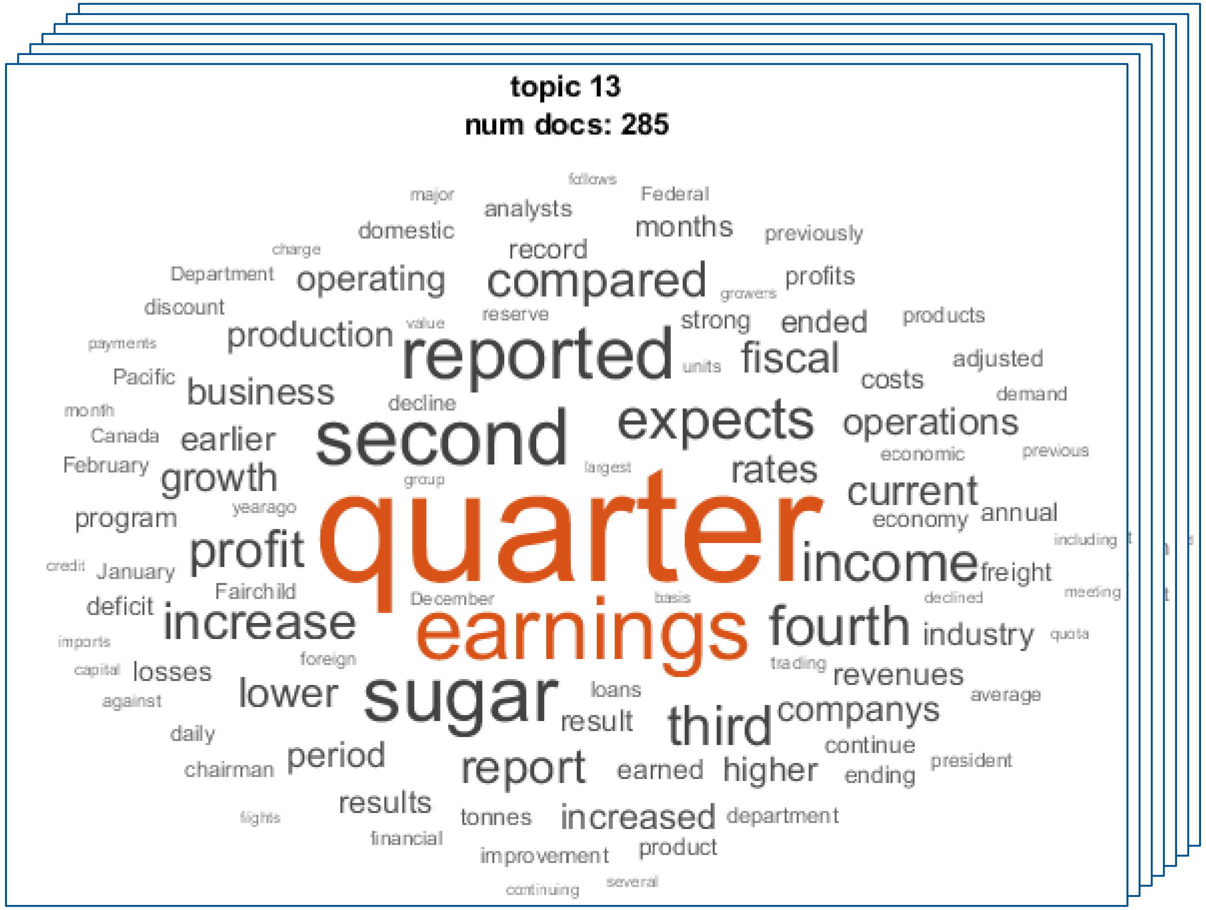
Inferência de Tópicos com Latent Dirichlet Allocation (LDA)

```
numTopics = 40;
mdl = fitlda(B,numTopics,...
     'Verbose',1,...
     'Solver','savb',...
     'FitTopicConcentration',true,...
     'DataPassLimit',10);

[~,topicInd] = max(...
mdl.DocumentTopicProbabilities,[],2);



for k = ...
     subB=removeDocument(B,find(topicInd~=k));
     figure, wordcloud(subB);
end
```



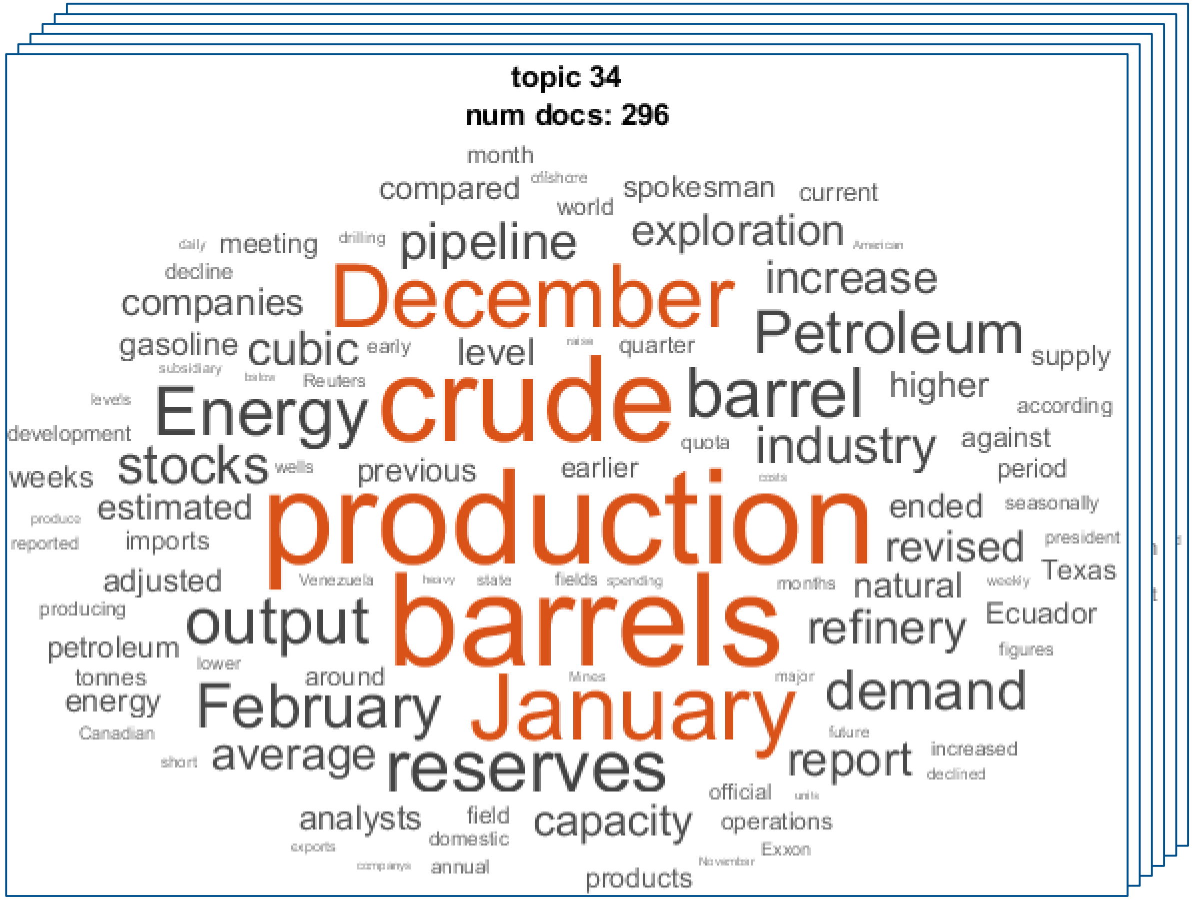
Inferência de Tópicos com Latent Dirichlet Allocation (LDA)

```
numTopics = 40;
mdl = fitlda(B,numTopics,...
    'Verbose',1,...
    'Solver','savb',...
    'FitTopicConcentration',true,...
    'DataPassLimit',10);

[~,topicInd] = max(...
mdl.DocumentTopicProbabilities,[],2);



for k = ...
    subB=removeDocument(B,find(topicInd~=k));
    figure, wordcloud(subB);
end
```



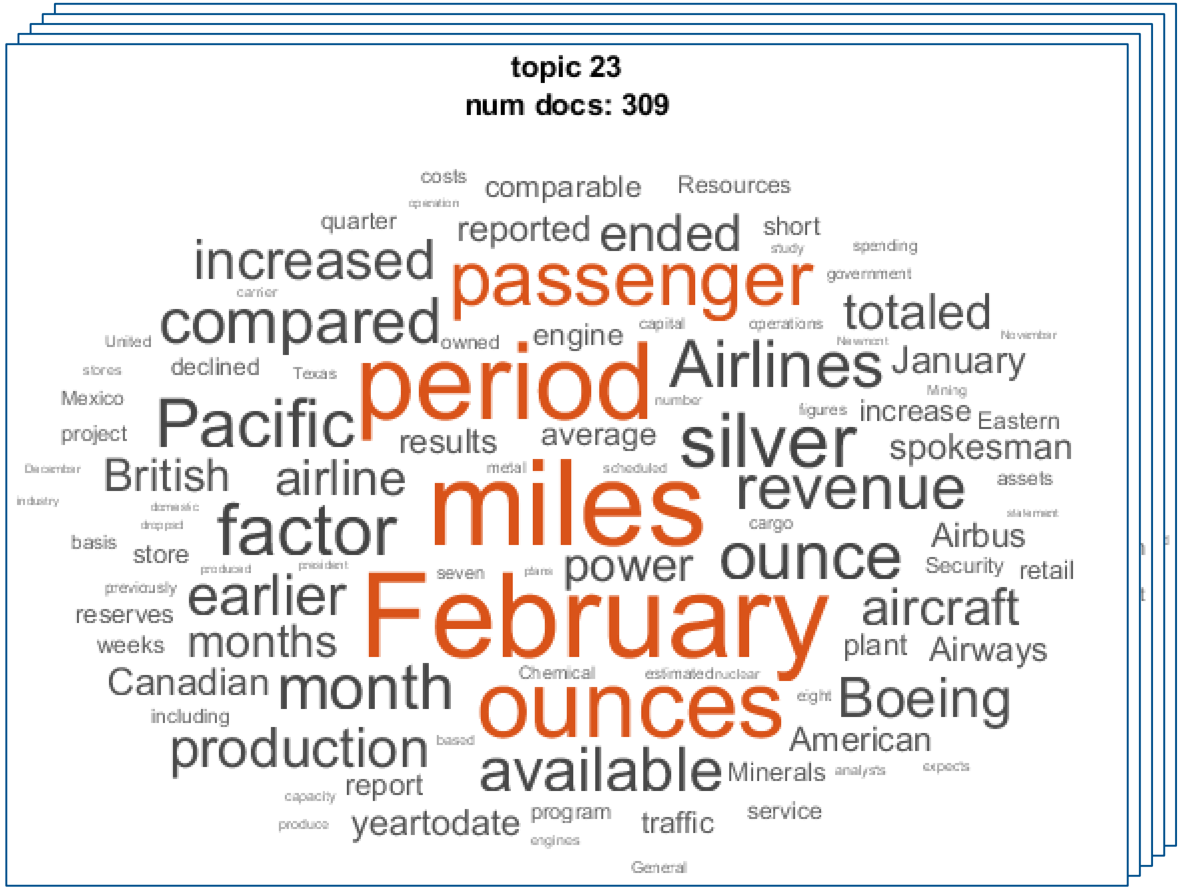
Inferência de Tópicos com Latent Dirichlet Allocation (LDA)

```
numTopics = 40;
mdl = fitlda(B,numTopics,...
    'Verbose',1,...
    'Solver','savb',...
    'FitTopicConcentration',true,...
    'DataPassLimit',10);

[~,topicInd] = max(...
mdl.DocumentTopicProbabilities,[],2);



for k = ...
    subB=removeDocument(B,find(topicInd~=k));
    figure, wordcloud(subB);
end
```

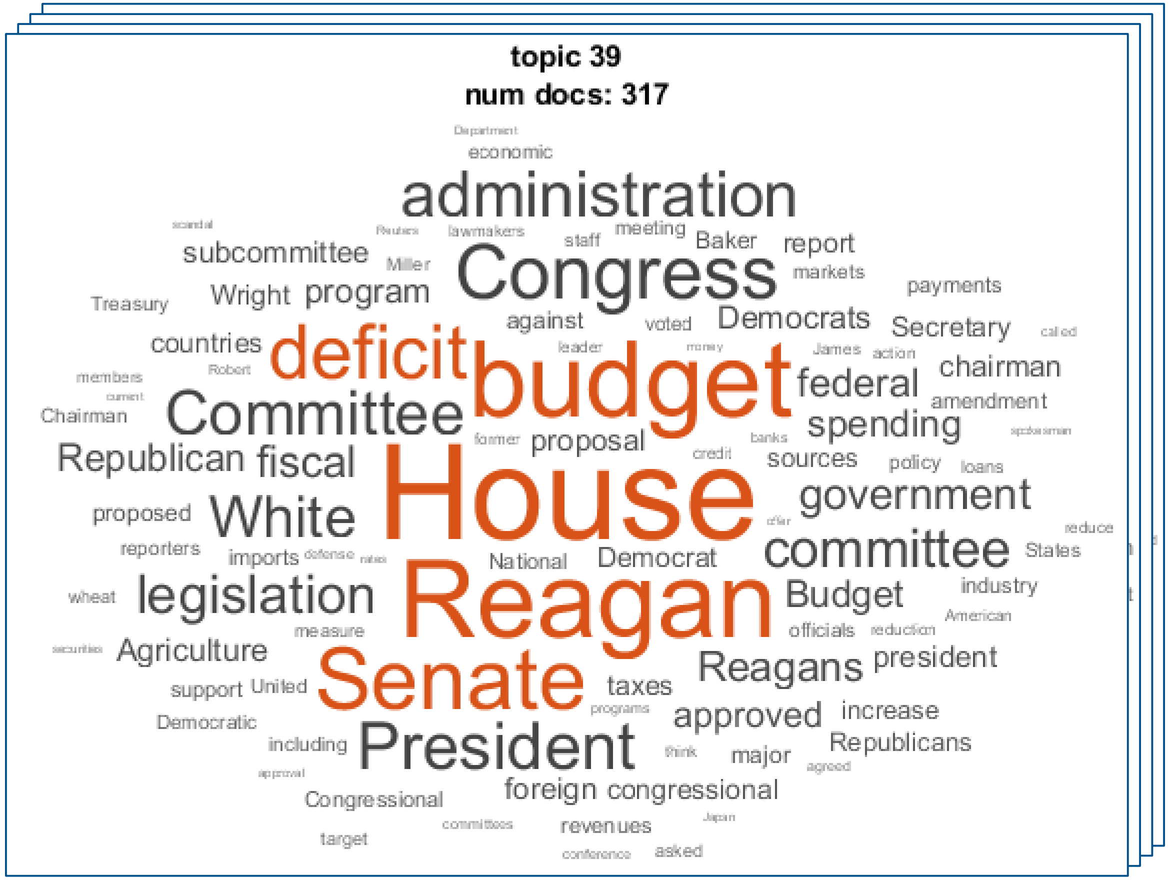


Inferência de Tópicos com Latent Dirichlet Allocation (LDA)

```
numTopics = 40;
mdl = fitlda(B,numTopics,...
    'Verbose',1,...
    'Solver','savb',...
    'FitTopicConcentration',true,...
    'DataPassLimit',10);

[~,topicInd] = max(...
mdl.DocumentTopicProbabilities,[],2);



for k = ...
    subB=removeDocument(B,find(topicInd~=k));
    figure, wordcloud(subB);
end
```



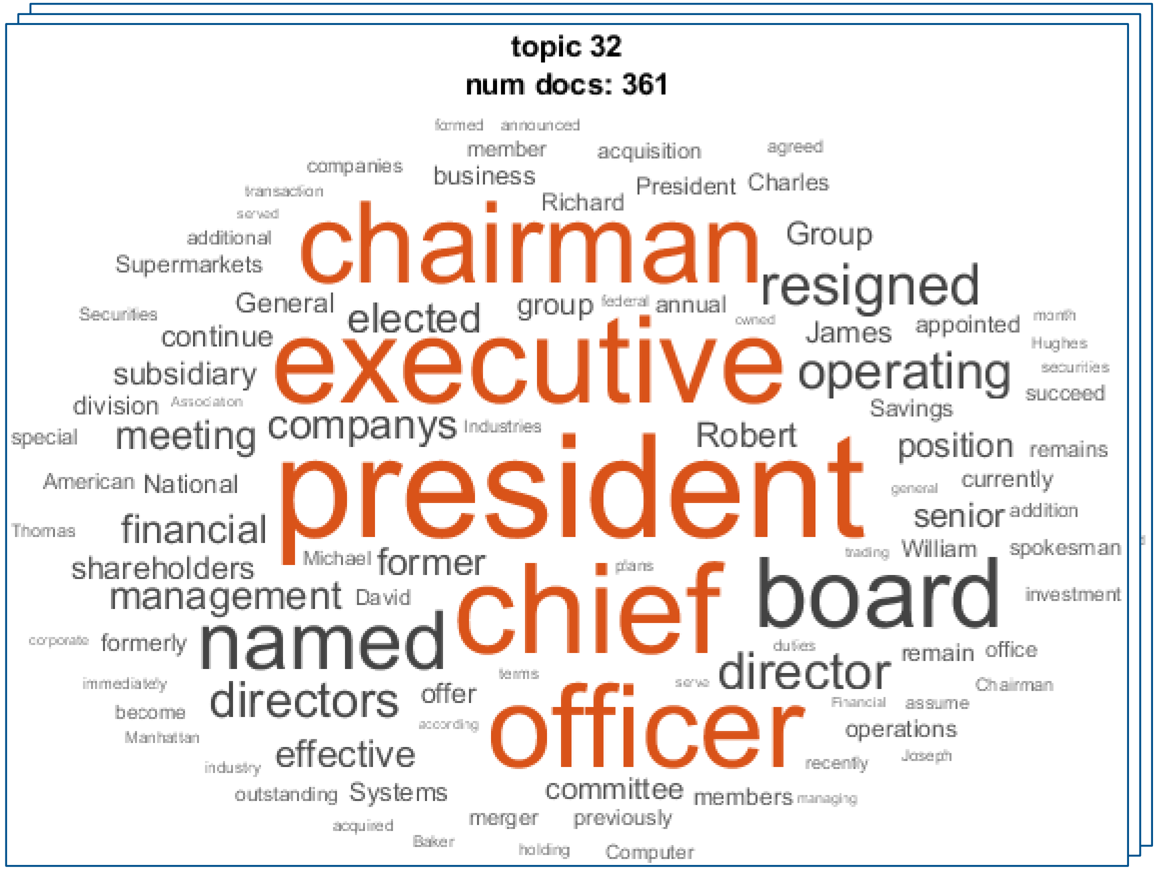
```
numTopics = 40;
mdl = fitlda(B,numTopics,...
    'Verbose',1,...
    'Solver','savb',...
    'FitTopicConcentration',true,...
    'DataPassLimit',10);

[~,topicInd] = max(...
mdl.DocumentTopicProbabilities,[],2);



for k = ...
    subB=removeDocument(B,find(topicInd~=k));
    figure, wordcloud(subB);
end
```



Inferência de Tópicos com Latent Dirichlet Allocation (LDA)

OPENCADD

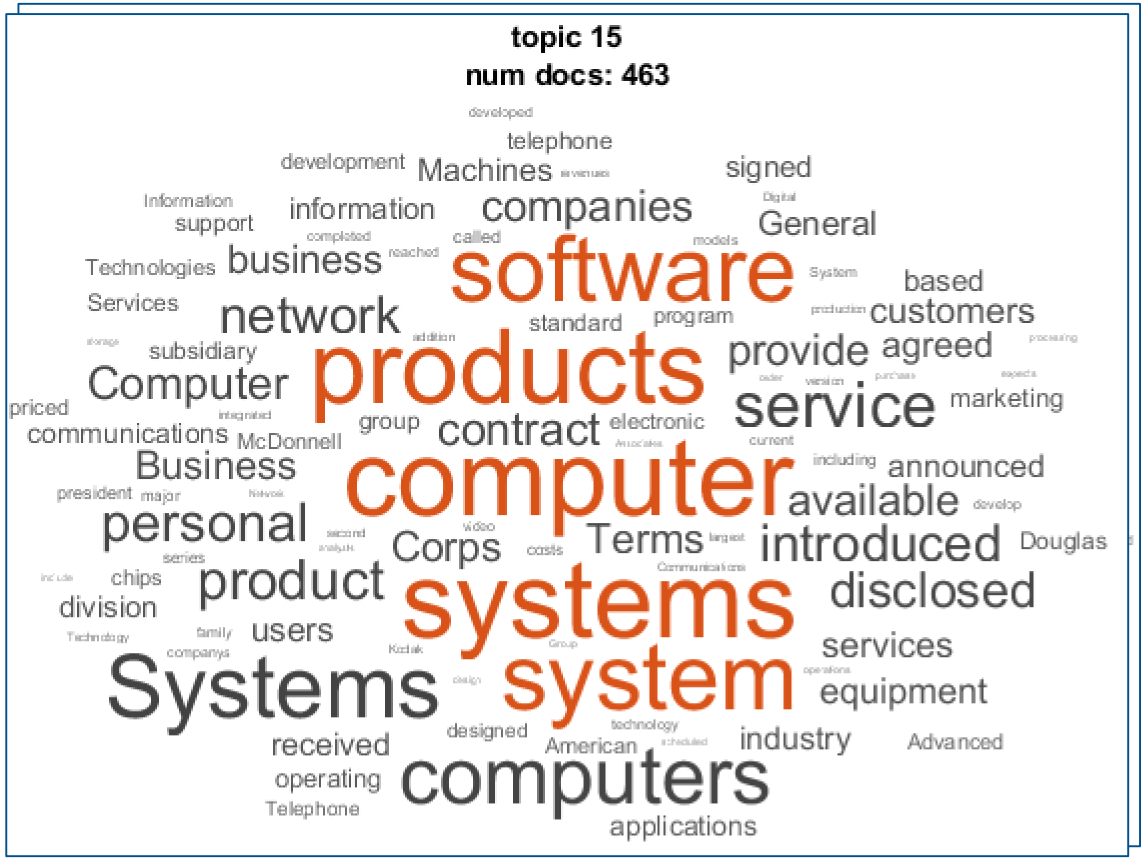
```
numTopics = 40;
mdl = fitlda(B,numTopics,...
     'Verbose',1,...
     'Solver','savb',...
     'FitTopicConcentration',true,...
     'DataPassLimit',10);

[~,topicInd] = max(...
mdl.DocumentTopicProbabilities,[],2);


for k = ...
     subB=removeDocument(B,find(topicInd~=k));
     figure, wordcloud(subB);
end
```



Inferência de Tópicos com Latent Dirichlet Allocation (LDA)

OPENCADD

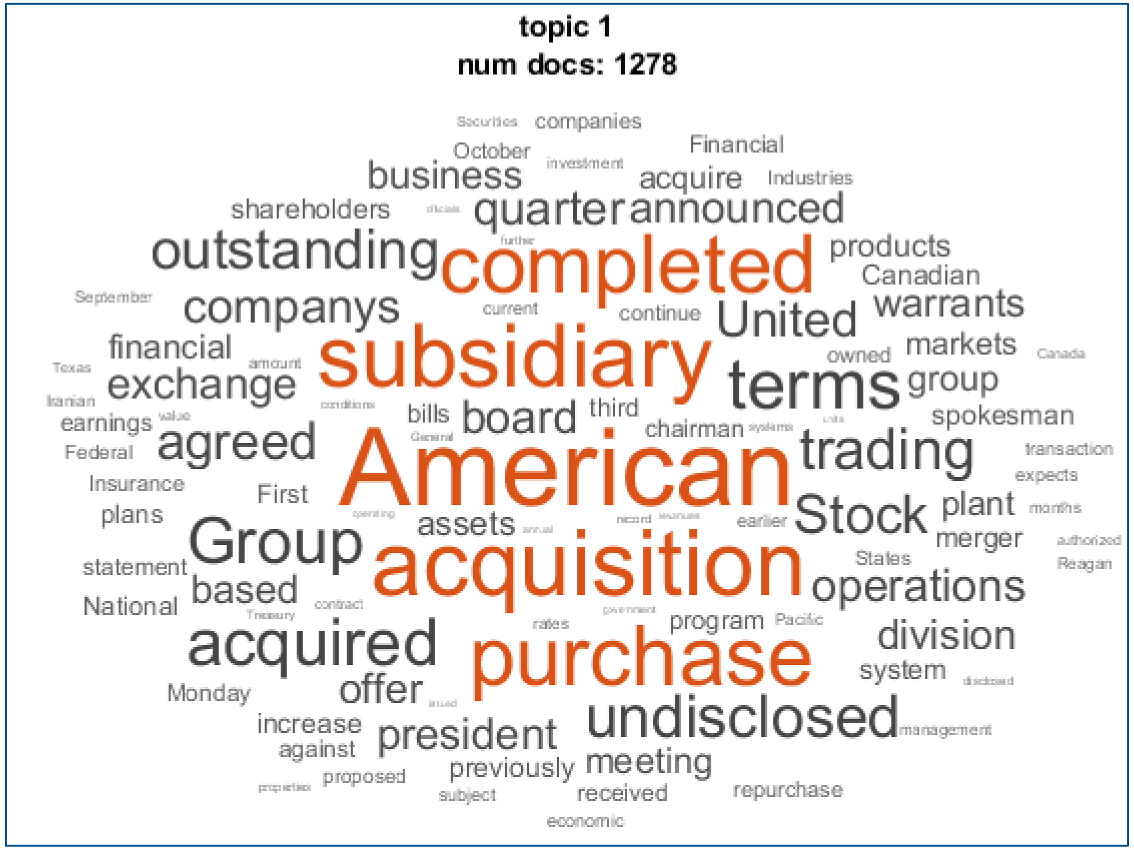
```
numTopics = 40;
mdl = fitlda(B,numTopics,...
     'Verbose',1,...
     'Solver','savb',...
     'FitTopicConcentration',true,...
     'DataPassLimit',10);

[~,topicInd] = max(...
mdl.DocumentTopicProbabilities,[],2);



for k = ...
    subB=removeDocument(B,find(topicInd~=k));
    figure, wordcloud(subB);
end
```



Inferência de Tópicos com Latent Dirichlet Allocation (LDA)

OPENCADD

```
numTopics = 40;
mdl = fitlda(B,numTopics,...
     'Verbose',1,...
     'Solver','savb',...
     'FitTopicConcentration',true,...
     'DataPassLimit',10);

[~,topicInd] = max(...
mdl.DocumentTopicProbabilities,[],2);



for k = ...
     subB=removeDocument(B,find(topicInd~=k));
     figure, wordcloud(subB);
end
```

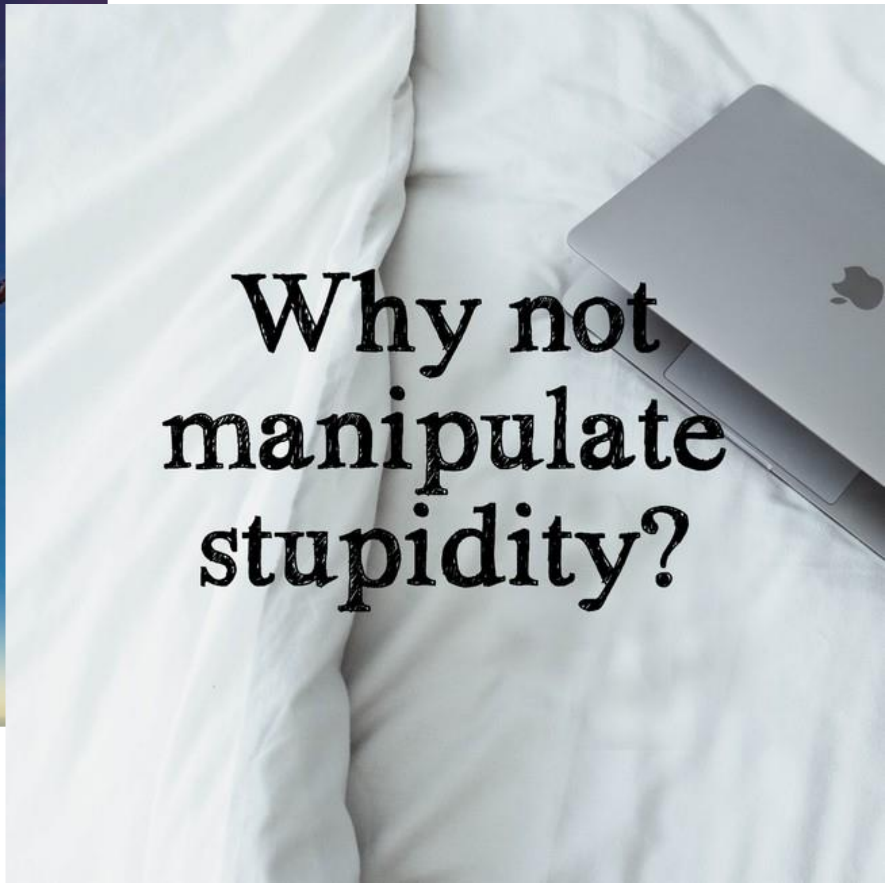


```
>> mdl

mdl =

  ldaModel with properties:

                     NumTopics: 40
             WordConcentration: 1
            TopicConcentration: 37.9175
      CorpusTopicProbabilities: [1×40 double]
    DocumentTopicProbabilities: [10546×40 double]
        TopicWordProbabilities: [2074×40 double]
                    Vocabulary: [1×2074 string]
                    TopicOrder: 'initial-fit-probability'
                       FitInfo: [1×1 struct]

>> mdl.FitInfo

ans =

  struct with fields:

          TerminationCode: 0
        TerminationStatus: "Iteration limit exceeded."
            NumIterations: 106
    NegativeLogLikelihood: 2.4190e+06
               Perplexity: 537.3574
                   Solver: "savb"
                  History: [1×1 struct]
           StochasticInfo: [1×1 struct]
```

Probabilidade inferida de cada documento pertencer a cada tópico

Probabilidade inferida de cada palavra aparecer em um documento de um certo tópico

Perplexidade: métrica de qualidade do ajuste dos tópicos inferidos (menor é melhor)



https://inspirobot.me/

A rede neural treinada para gerar frases motivacionais que deu muito errado

Fonte do dataset: https://www.gutenberg.org/browse/scores/top

Livros:

- *Grimm Fairy Tales (Grimm Brothers)*
- *The Dunwich Horror (H. P. Lovecraft)*
- *The Shunned House (H. P. Lovecraft)*

Cada parágrafo foi considerado um texto

Foram excluídos:

- *Parágrafos vazios*
- *Legendas de figura*
- *Títulos de capítulo*
- *Índice*
- *Informações sobre o livro*

Exemplos:

Grimm Fairy Tales
Hans took the stone, and went his way with a light heart: his eyes sparkled for joy, and he said to himself, 'Surely I must have been born in a lucky hour; everything I could want or wish for comes of itself. People are so kind; they seem really to think I do them a favour in letting them make me rich, and giving me good bargains.'

The Dunwich Horror
There was a hideous screaming which echoed above even the hill noises and the dogs' barking on the night Wilbur was born, but no known doctor or midwife presided at his coming. Neighbors knew nothing of him till a week afterward, when Old Whateley drove his sleigh through the snow into Dunwich Village and discoursed incoherently to the group of loungers at Osborn's general store. There seemed to be a change in the old man--an added element of furtiveness in the clouded brain which subtly transformed him from an object to a subject of fear--though he was not one to be perturbed by any common family event. Amidst it all he showed some trace of the pride later noticed in his daughter, and what he said of the child's paternity was remembered by many of his hearers years afterward.

Arquitetura da rede é a mesma que para classificação, mas o objetivo agora é predizer a próxima palavra da sequência:

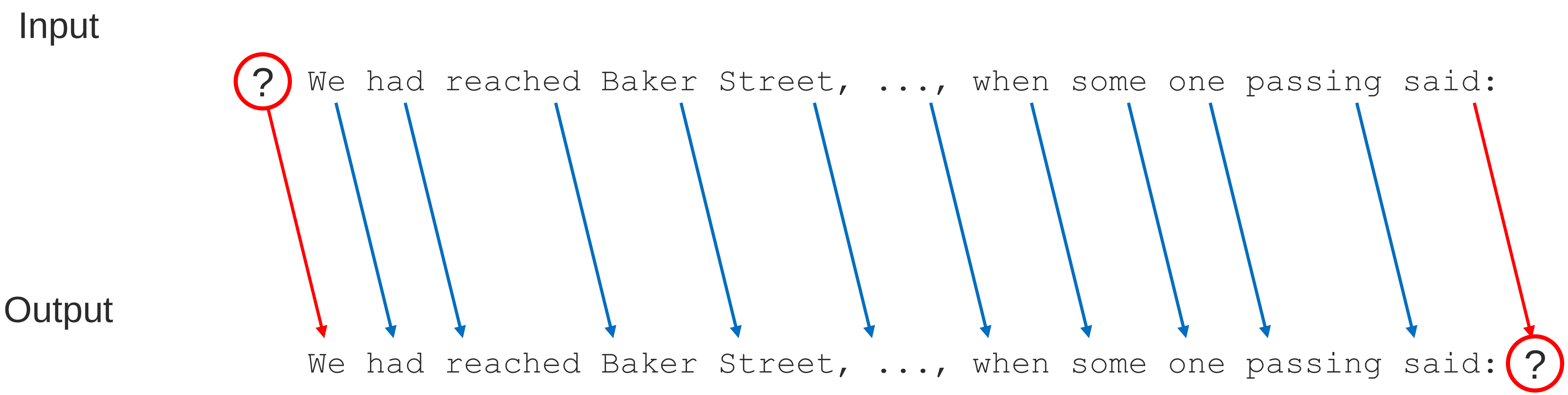
Input

`We had reached Baker Street, ..., when some one passing said:`

Output

`We had reached Baker Street, ..., when some one passing said:`

Arquitetura da rede é a mesma que para classificação, mas o objetivo agora é predizer a próxima palavra da sequência:



Geração de Texto com Deep Learning

Arquitetura da rede é a mesma que para classificação, mas o objetivo agora é predizer a próxima palavra da sequência:

Input

`"startOfText" We had reached Baker Street, ..., when some one passing said:`

Output

`We had reached Baker Street, ..., when some one passing said: "endOfText"`

OPENCADD

Arquitetura da rede é a mesma que para classificação, mas o objetivo agora é predizer a próxima palavra da sequência:

Input

"startOfText" We had reached Baker Street, ..., when some one passing said: *predictors*

Output

responses We had reached Baker Street, ..., when some one passing said: "endOfText"

OPENCADD

Datastore customizado

```
classdef documentGenerationDatastore < matlab.io.Datastore & matlab.io.datastore.MiniBatchable

    properties
        Documents
        Encoding
        MiniBatchSize
    end

    properties(SetAccess = protected)
        NumObservations
    end

    properties(Access = private)
        CurrentFileIndex
    end

    methods
        function ds = documentGenerationDatastore(documents,labels)

        function tf = hasdata(ds)

        function [data,info] = read(ds)

        function reset(ds)

    end

end
```

tokenizedDocument

wordEncoding

Propriedades necessárias em uma datastore

Método construtor

Métodos necessários em uma datastore

Este datastore foi criado modificando este exemplo pronto (execute o comando no MATLAB):

```
edit(fullfile(matlabroot,'examples','nnet','main','documentGenerationDatastore.m'))
```

Datastore customizado

```
function ds = documentGenerationDatastore(documents)
    % ds = documentGenerationDatastore(documents) creates a
    % document mini-batch datastore from an array of tokenized
    % documents.

    % Add startOfText token to documents
    startTokens = repmat(tokenizedDocument("startOfText"), size(documents));

    documents =  startTokens + documents;

    % Set Documents and MiniBatchSize properties.
    ds.Documents = documents;
    ds.MiniBatchSize = 128;

    % Create word encoding.
    ds.Encoding = wordEncoding(documents);

    % Datastore properties.
    numObservations = numel(documents);
    ds.NumObservations = numObservations;
    ds.CurrentFileIndex = 1;
end
```

"cola" o token `"startOfText"` *em todos os documentos*

OPENCADD

Datastore customizado

```
function ds = documentGenerationDatastore(documents,labels)
      % ds = documentGenerationDatastore(documents) creates a
      % document mini-batch datastore from an array of tokenized
      % documents.

      % Add startOfText token to documents
      startTokens = repmat(tokenizedDocument("startOfText"), size(documents));
      startTokens = tokenizedDocument(labels);
      documents =  startTokens + documents;

      % Set Documents and MiniBatchSize properties.
      ds.Documents = documents;
      ds.MiniBatchSize = 128;

      % Create word encoding.
      ds.Encoding = wordEncoding(documents);

      % Datastore properties.
      numObservations = numel(documents);
      ds.NumObservations = numObservations;
      ds.CurrentFileIndex = 1;
end
```

Dessa forma você pode gerar documentos de múltiplos assuntos com uma mesma rede, simplesmente especificando o assunto no token inicial

```
"Grimm_Brothers"
"H_P_Lovecraft"
```

OPENCADD

Datastore customizado

```
function [data,info] = read(ds)
    % [data,info] = read(ds) read one mini-batch of data.

    miniBatchSize = ds.MiniBatchSize;
    enc = ds.Encoding;
    info = struct;

    % Read batch of documents.
    startPos = ds.CurrentFileIndex;
    endPos = ds.CurrentFileIndex + miniBatchSize - 1;
    documents = ds.Documents(startPos:endPos);

    % Convert documents to sequences.
    numWords = enc.NumWords;
    predictors = doc2sequence(enc,documents, ...
        'PaddingValue',numWords+1);

    % Create categorical sequences of responses.
    classNames = [enc.Vocabulary "EndOfText"];
    for i = 1:miniBatchSize
        X = predictors{i};
        words = [ind2word(enc,X(2:end)) "EndOfText"];
        responses{i,1} = categorical(words,classNames);
    end

    % Update file index
    ds.CurrentFileIndex = ds.CurrentFileIndex + miniBatchSize;

    % Convert data to table.
    data = table(predictors,responses);
end
```

Seleciona 1 minibatch

Converte os documentos em sequencias usando o encoding criado

Saída = texto + `"EndOfText"`

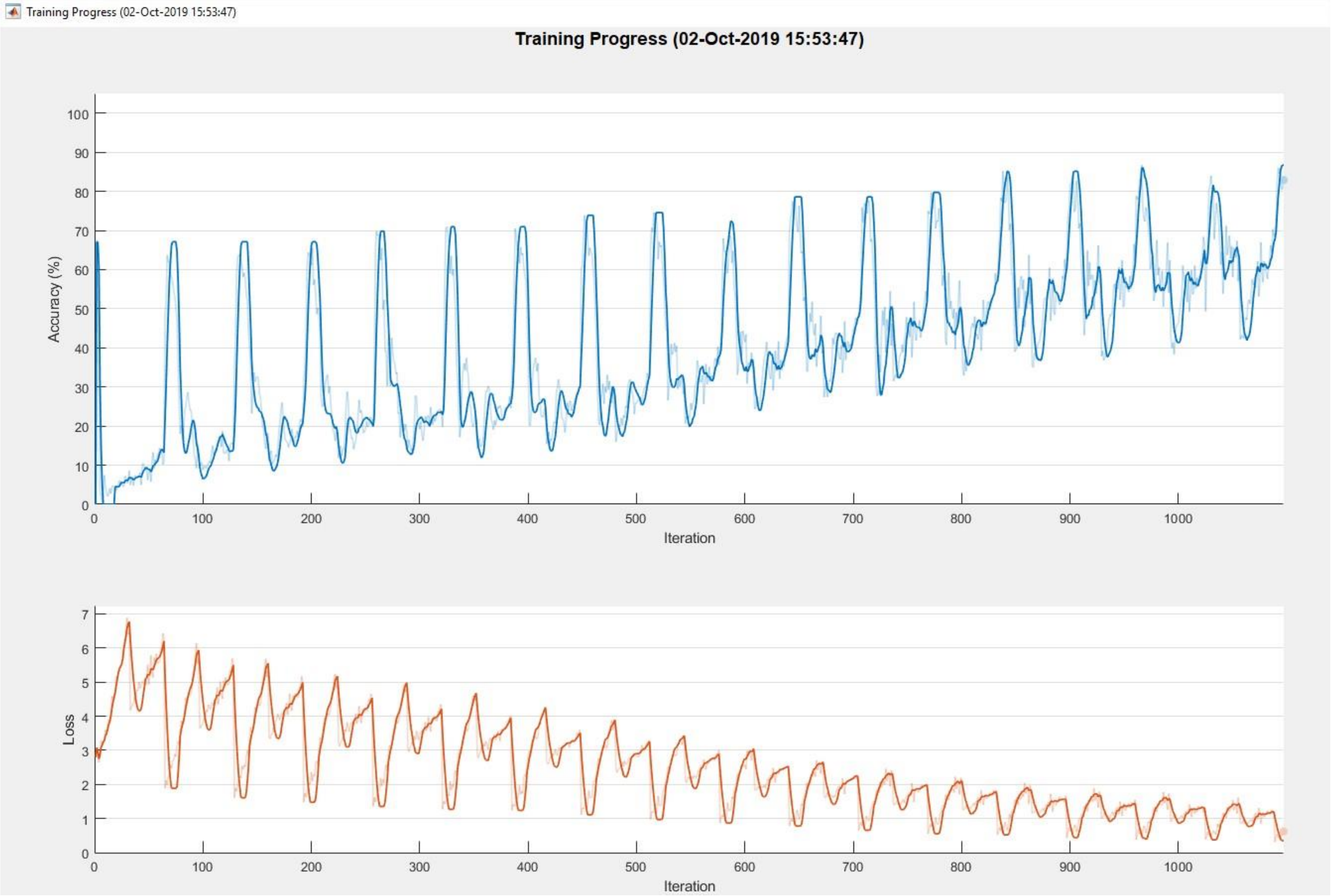
Treinamento

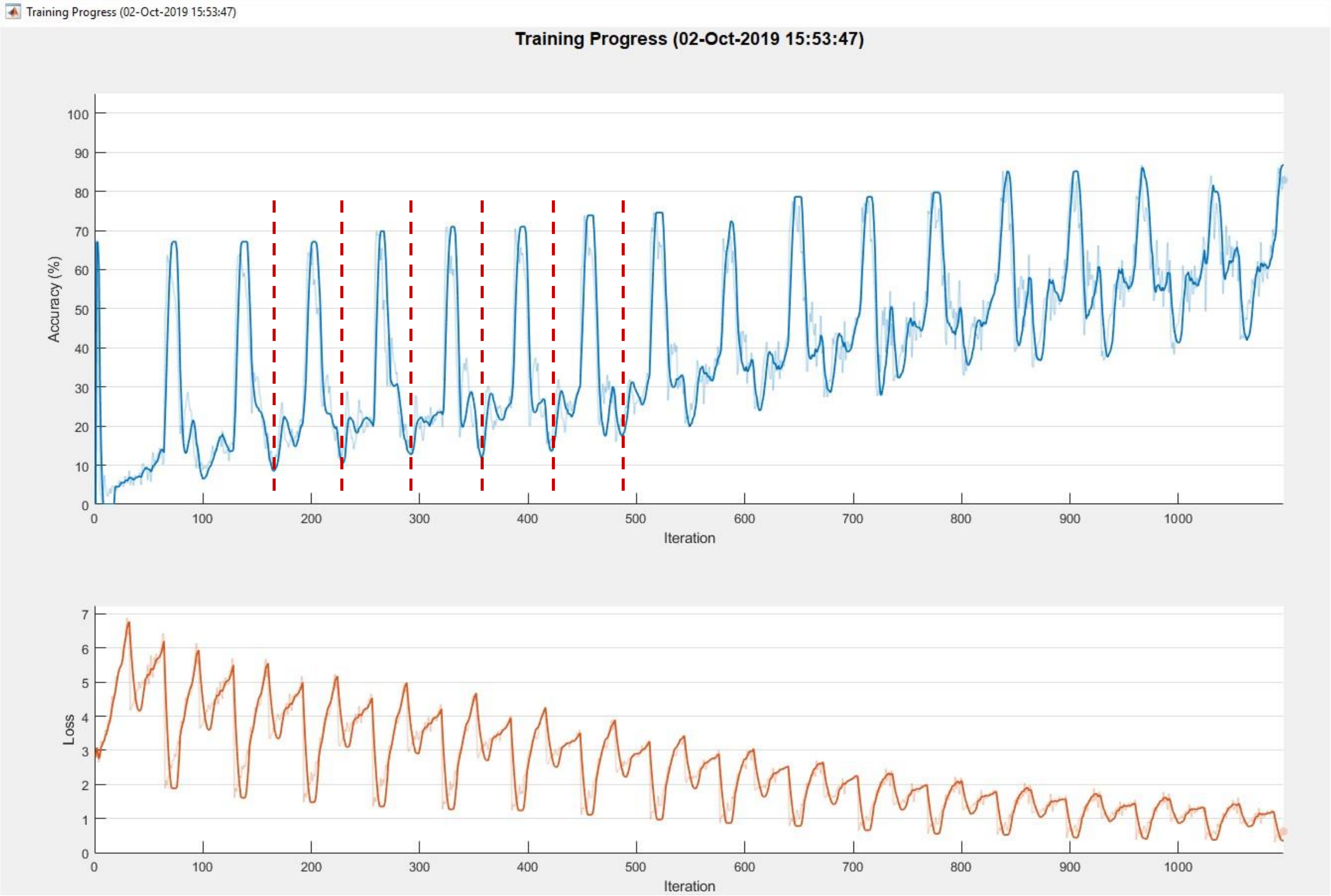
```
inputSize = 1;
embeddingDimension = 300;
numWords = numel(ds.Encoding.Vocabulary);
numClasses = numWords + 1;

layers = [
    sequenceInputLayer(inputSize)
    wordEmbeddingLayer(embeddingDimension,numWords)
    lstmLayer(300)
    dropoutLayer(0.2)
    fullyConnectedLayer(numClasses)
    softmaxLayer
    classificationLayer];

options = trainingOptions('adam', ...
    'MaxEpochs',300, ...
    'InitialLearnRate',0.01, ...
    'MiniBatchSize',4, ...
    'Shuffle','never', ...
    'Plots','training-progress', ...
    'Verbose',false);

net = trainNetwork(ds,layers,options);
```

Estrutura idêntica ao caso de classificação,
mas agora há 1 classe por palavra no vocabulário

Não tem sentido separar dados para validação e teste nesse caso,
pode usar tudo para treino.

Geração de Texto com Deep Learning




Training Progress (02-Oct-2019 15:53:47)
Training Progress (02-Oct-2019 15:53:47)
Accuracy (%)
100
90
80
70
60
50
40
30
20
10
0
0
100
200
300
400
500
600
700
800
900
1000
Iteration
Loss
7
6
5
4
3
2
1
0
0
100
200
300
400
500
600
700
800
900
1000
Iteration

OPENCADD

Gerando textos:

```
load('2authorsNet.mat')
load('2authorsDs.mat')

enc = ds.Encoding;
wordIndex = word2ind(enc,"H_P_Lovecraft"); % OR "Grimm_Brothers"
vocabulary = string(net.Layers(end).Classes);


generatedText = "";
maxLength = 5000;
while strlength(generatedText) < maxLength
    % Predict the next word scores.
    [net,wordScores] = predictAndUpdateState(net,wordIndex,'ExecutionEnvironment','cpu');

    % Sample the next word.
    newWord = datasample(vocabulary,1,'Weights',wordScores);
```

Usa a rede treinada para calcular as probabilidades de cada palavra, dada a palavra atual (começando de `"H_P_Lovecraft"`*) e o estado da rede até o momento, e atualiza o estado*

```
    % Stop predicting at the end of text.
    if newWord == "EndOfText"
        break
    end
```

Interrompe quando gerar `"endOfText"`

```
    % Add the word to the generated text.
    generatedText = generatedText + " " + newWord;

    % Find the word index for the next input.
    wordIndex = word2ind(enc,newWord);
end
```

Concatena palavra gerada com as palavras geradas anteriormente para gerar o texto
Passa para a próxima palavra

```
punctuationCharacters = ["." "," "'" ")" ":" "?" "!"];
generatedText = replace(generatedText," " + punctuationCharacters,punctuationCharacters);

punctuationCharacters = ["(" "`"];
generatedText = replace(generatedText,punctuationCharacters + " ",punctuationCharacters)
```

Remove espaços desnecessários gerados antes ou depois de sinais de pontuação

```
net = resetState(net);
```

Reseta para gerar outro texto

"Grimm_Brothers"

After some time, and then she began to grow unhappy. She could not see him he.' Then the willow-wren flew to the bear's hole and cried: 'Growler, you are growing child, and she was not one.' 'Well,' said he, 'I am quite willing to learn something - - indeed, if it could but be managed, I should like to learn how to shudder. I don't understand that at all yet.' The elder brother smiled when he heard that, and thought to himself: 'Goodness, what a blockhead that brother of mine is! He will never be good for anything as to go. 'Not for gold or silver, but for flesh and blood: let me again this evening speak with the bridegroom in his chamber, and I will give thee the whole brood.'

“H_P_Lovecraft"

Dr. Armitage, associating what he was reading with what he had heard of Dunwich and its brooding presences, and of Wilbur Whateley and his dim, hideous aura that stretched from a dubious birth to a cloud of probable matricide, felt a wave of fright as tangible as a draft of the tomb's cold clamminess. The bent, goatish giant before him seemed like a child of his first wife was a boy, who was as red as blood and as white as snow. The mother loved her daughter very much, and when she looked at her and then looked at the boy back in the morning light, and over the bridge to the business section where tall buildings seemed to guard me as modern material things guard the world from ancient tales of the common folk - - a notion likewise alluding to ghoulish, wolfish shapes taken by smoke from the great chimney, and queer contours assumed by certain information could an exhaustive research, and to form and in his inside coat pocket. He was perniciously, and a merry day, a very strange soldier's return, and moon that I had to handle, and had a chance of never eating again.’

Workflow para Sensitivity Anlysis

Tweets

```
ans = 508×1 string array
    "Walmart: "you wanna destroy Amazon?" Google: "bet" $WMT $GOOG
    "$WMT wants next level customer service w/highly personalized
    "Ironic prelude to $DIS buying $TWTR soon IMO $AAPL $GOOG $SPY
    "$AMZN the $WMT threat grows each and every day  https://t.co/
    "MU Investments Co. Ltd. Sells 30 Shares of Alphabet Inc. $GOO
    "Ad $ are going to $GOOG and $FB away from wppgy  #Advertising
    "Big bullish unusual option activity detected: $SPX, $GOOG, $G
    "REPORT: Apple to build data center in Iowa: https://t.co/jwH6
    "RT @theflynews: REPORT: Apple to build data center in Iowa: h
```

Algoritimo

```
scoreTweet(tweet,emb,svmModel)
```

Score

Lista de Palavras Positivas + Negativas

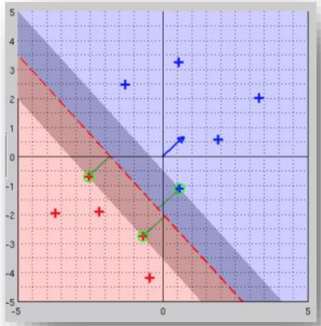
pos — 2006x1 string

	1
1	a+
2	abound
3	abounds
4	abundance
5	abundant
6	accessable
7	accessible

neg — 4783x1 string

	1
1	2-faced
2	2-faces
3	abnormal
4	abolish
5	abominable
6	abominably
7	abominate

Modelo de Machine Learning



Word Embedding

```
wordEmbedding with properties:

  Dimension: 100
 Vocabulary: [1×1193514 string]
```

Dúvidas???

OPENCADD

MODEL - BASED DESIGN DRIVEN COMPANY

Modeling for life!