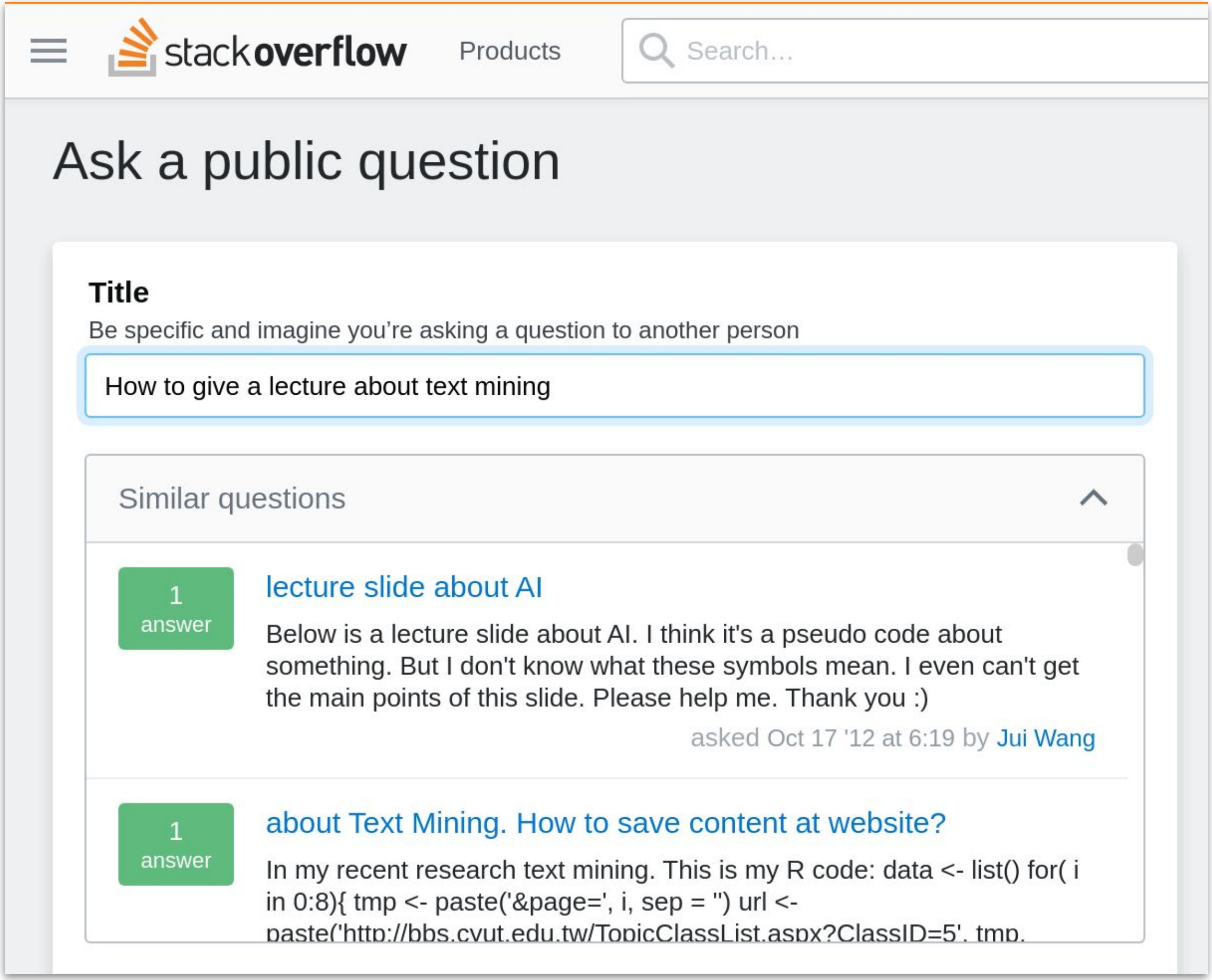
A Survey of Text Mining

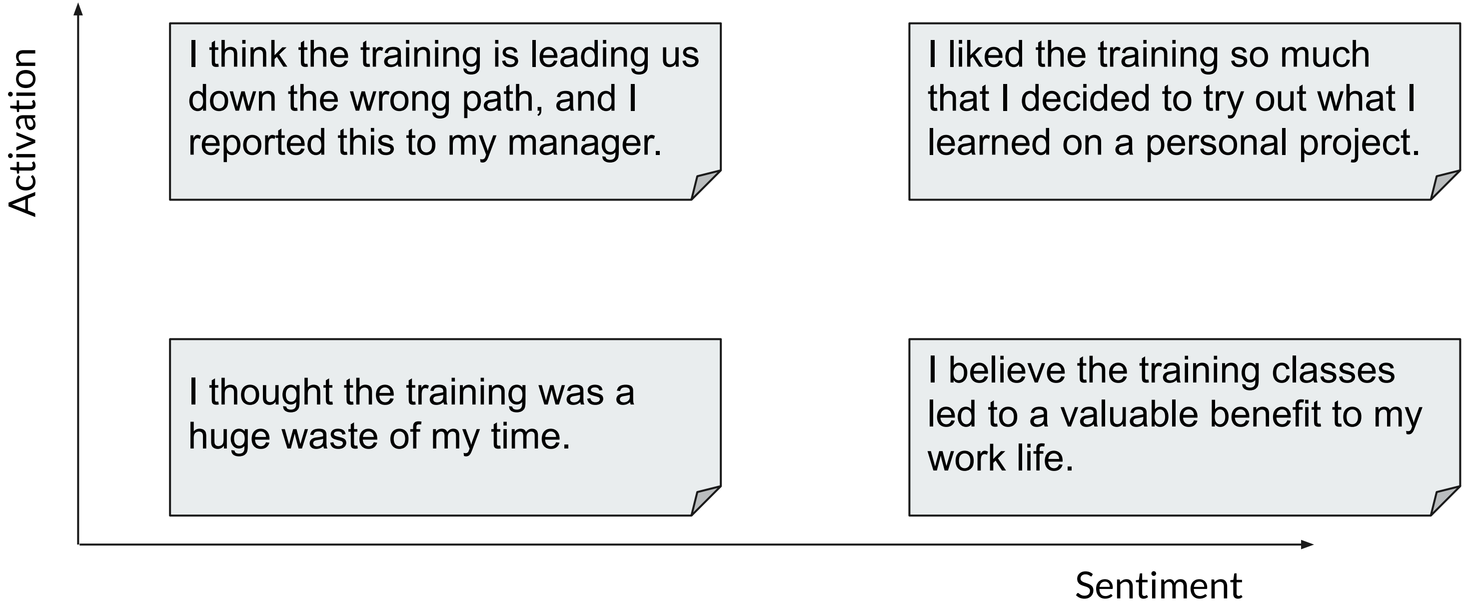
How to solve complex problems with text


reduce
No junk mail
ISLINGTON


stackoverflow
Products
Search…
Ask a public question
Title
Be specific and imagine you’re asking a question to another person
How to give a lecture about text mining
Similar questions
1
answer
lecture slide about AI
Below is a lecture slide about AI. I think it's a pseudo code about something. But I don't know what these symbols mean. I even can't get the main points of this slide. Please help me. Thank you :)
asked Oct 17 '12 at 6:19 by Jui Wang
1
answer
about Text Mining. How to save content at website?
In my recent research text mining. This is my R code: data <- list() for(i in 0:8){ tmp <- paste('&page=', i, sep = '') url <-

While not normally known for his musical talent, Elon Musk is releasing a debut album. The "Elon Musk" is a collection of eight new songs which are inspired by the founder's life. The music, which is available for pre-order on iTunes, was created by one-man-band and fellow Tesla Motors and SpaceX executive, Paul Kasmin, who's known for playing guitar at Tesla events. The album is a collaboration between Kasmin and Musk himself, although it's also being marketed under the Tesla brand.

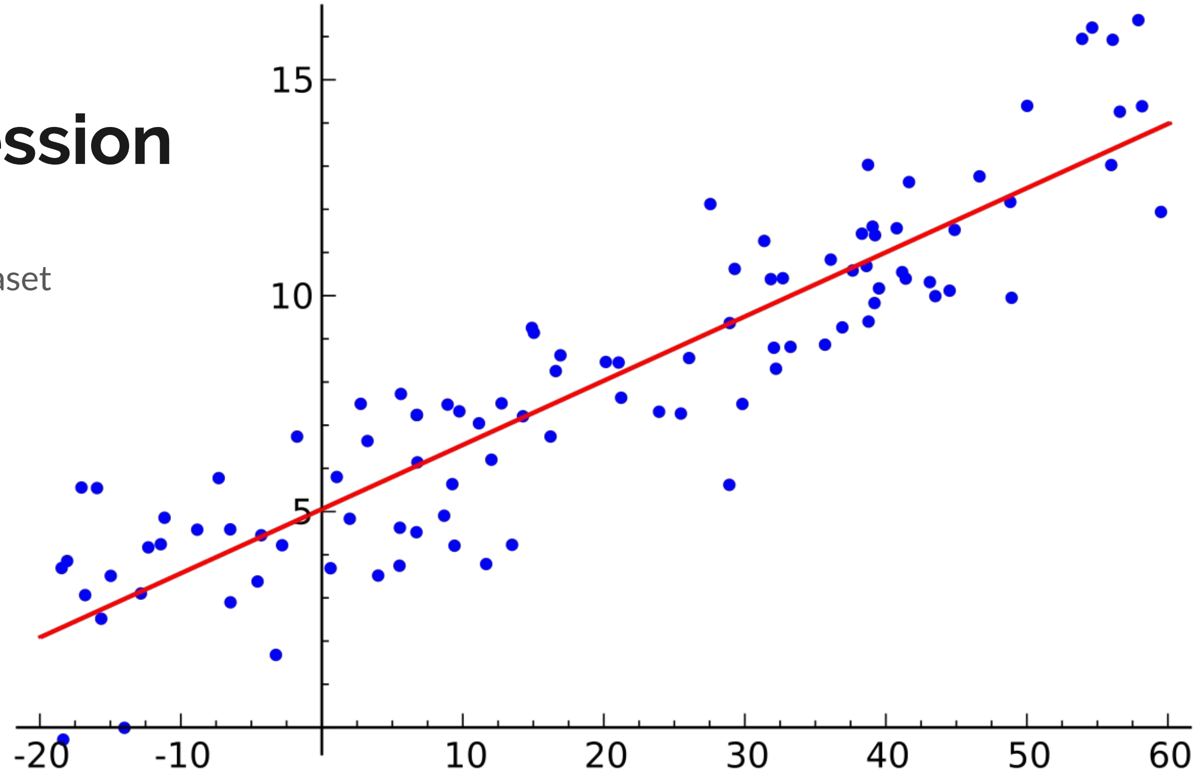
Example: Quantify Survey Results



Note: These are not real examples from our dataset.

Typical Regression

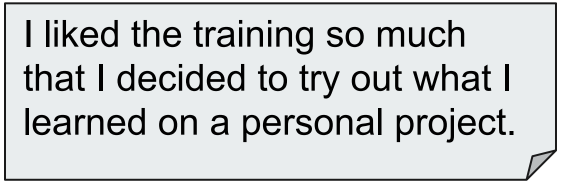
Example: Boston housing dataset

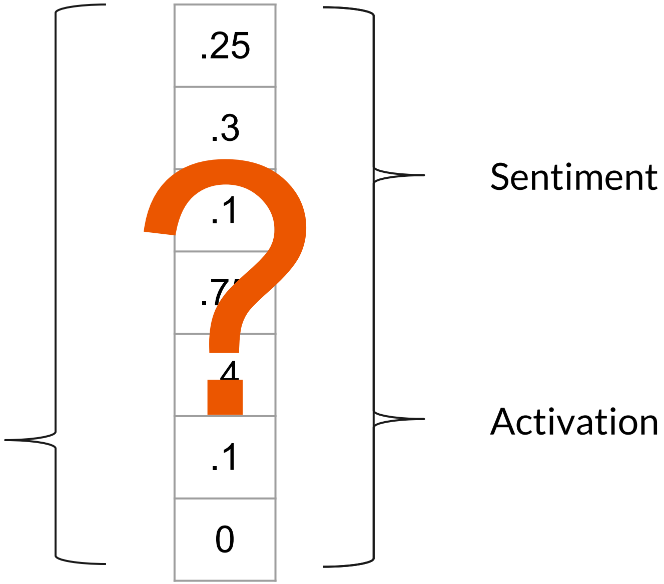


Challenge: Numeric Features from Text

Convert "plain text" written by humans

into vectors understandable by computers





Bag of Words (BOW)

Split words by spaces

Create a vector of word counts

I liked the training so much that I decided to try out what I learned on a personal project.

I	3
liked	1
the	1
training	1
so	1
much	1
that	1
decided	1
...	...

BOW Vector Properties

Vectors are sparse

Size of vocab = size of vector

I	3
liked	1
the	1
training	1
so	1
much	1
that	1
decided	1
...	...

Size of Vocabulary
Zeros Omitted

BOW Issues

Frequent words dominate the representation

Word order removed

Similar words become totally different features

I	3
liked	1
the	1
training	1
so	1
much	1
that	1
decided	1
...	...

Size of Vocabulary
Zeros Omitted

Working with BOW Vectors

Use cosine similarity to relate different texts

Documents have a similarity between 0 and 1

(if A and B are nonnegative)

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum\limits_{i=1}^{n} A_i B_i}{\sqrt{\sum\limits_{i=1}^{n} A_i^2} \sqrt{\sum\limits_{i=1}^{n} B_i^2}},$$

Cosine Similarity Example

I didn't like the training

A

I	1
didn't	1
like	1
the	1
training	1

I	1
liked	1
the	1
training	1
a	1
lot	1

I liked the training a lot

B

Text	A	B	Ai*Bi
I	1	1	1
like	1	0	0
liked	0	1	0
the	1	1	1
training	1	1	1
a	0	1	0
lot	0	1	0
didn't	1	0	0
	√5	√6	3
	$\sqrt{\sum_{i=1}^{n} A_i^2}$	$\sqrt{\sum_{i=1}^{n} B_i^2}$	$\sum_{i=1}^{n} A_i B_i$

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

similarity(A, B)

= 3 / (√5 * √6)

≈ 0.5477

Text	A	B	Ai*Bi
I	1	1	1
like	1	0	0
liked	0	1	0
the	1	1	1
training	1	1	1
a	0	1	0
lot	0	1	0
didn't	1	0	0
	√5	√6	3
	$\sqrt{\sum_{i=1}^{n} A_i^2}$	$\sqrt{\sum_{i=1}^{n} B_i^2}$	$\sum_{i=1}^{n} A_i B_i$

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}},$$

Issue:
Most similar words aren't relevant.

similarity(A, B)

= 3 / (√5 * √6)

≈ 0.5477

Stopwords

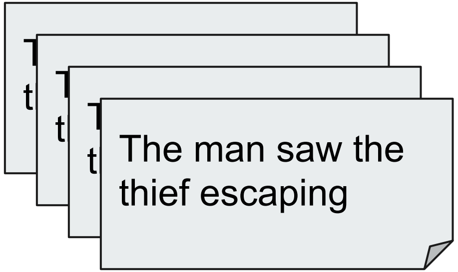
Words that we know aren't relevant

Often we can just remove these

Term Frequency Inverse Document Frequency (TF-IDF)

Prioritize rare words

Demote common words



Document Frequency:
docs containing the word

Term Frequency:
occurrences within a document

Let **t** be a term

The	

Let **t** be a term, **d** be a document

The man saw the thief escaping

The	2
man	1
saw	1
thief	1
escaping	1

d

Let **t** be a term, **d** be a document, and **C** be a corpus.

The man saw the thief escaping

The	2
man	1
saw	1
thief	1
escaping	1

d

The	100
man	75
saw	10
thief	5
escaping	3

150 total documents

C

Let **t** be a term, **d** be a document, and **C** be a corpus.

TF(t, d) = # times t occurs in d / size of d

The man saw the thief escaping (size = 6)

The	2/6
man	1/6
saw	1/6
thief	1/6
escaping	1/6

The	100
man	75
saw	10
thief	5
escaping	3

150 total documents

d

C

Let **t** be a term, **d** be a document, and **C** be a corpus.

TF(t, d) = # times t occurs in d / size of d

IDF(t, C) = log(size of C / number of documents in C containing t)

The man saw the thief escaping (size = 6)

d

The	2/6
man	1/6
saw	1/6
thief	1/6
escaping	1/6

The	log(150/100)
man	log(150/75)
saw	log(150/10)
thief	log(150/5)
escaping	log(150/3)

150 total documents

C

Let **t** be a term, **d** be a document, and **C** be a corpus.

TF(t, d) = # times t occurs in d / size of d

IDF(t, C) = log(size of C / number of documents in C containing t)

TFIDF(t, d, C) = TF(t, d) IDF(t, C)

The man saw the thief escaping (size = 6)

The	2/6
man	1/6
saw	1/6
thief	1/6
escaping	1/6

The	log(150/100)
man	log(150/75)
saw	log(150/10)
thief	log(150/5)
escaping	log(150/3)

150 total documents

d

C

Let **t** be a term, **d** be a document, and **C** be a corpus.

TF(t, d) = # times t occurs in d / size of d

IDF(t, C) = log(size of C / number of documents in C containing t)

TF-IDF(t, d, C) = TF(t, d) IDF(t, C)

The man saw the thief escaping (size = 6)

The	2/6
man	1/6
saw	1/6
thief	1/6
escaping	1/6

The	log(150/100)
man	log(150/75)
saw	log(150/10)
thief	log(150/5)
escaping	log(150/3)

150 total documents

TFIDF("the", d, C) = (2/6)(log(150/100) = **0.058**

TFIDF("thief", d, C) = (1/6)(log(150/5) = **0.246**

d

C

Text	A	B	Ai*Bi
I	1	1	1
like	1	0	0
liked	0	1	0
the	1	1	1
training	1	1	1
a	0	1	0
lot	0	1	0
didn't	1	0	0
	√5	√6	3
	$\sqrt{\sum_{i=1}^{n} A_i^2}$	$\sqrt{\sum_{i=1}^{n} B_i^2}$	$\sum_{i=1}^{n} A_i B_i$

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

Issue:
Most similar words aren't relevant.

similarity(A, B)

= 3 / (√5 * √6)

≈ 0.5477

Text	A	B	Ai*Bi
I	0.01	0.01	0.0001
like	0.25	0	0
liked	0	0.2	0
the	0.01	0.01	0.001
training	0.5	0.5	0.25
a	0	0.002	0
lot	0	0.03	0
didn't	0.02	0	0
	0.5596	0.5395	0.2511
	$\sqrt{\sum_{i=1}^{n} A_i^2}$	$\sqrt{\sum_{i=1}^{n} B_i^2}$	$\sum_{i=1}^{n} A_i B_i$

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

Issue:
~~Most similar words aren't relevant.~~
Solved!

similarity(A, B)

~~= 3 / (√5 * √6)~~

~~≈ 0.5477~~

= 0.2511 / (0.5596 * 0.5395)

≈ 0.8317

Text	A	B	Ai*Bi
I	0.01	0.01	0.0001
like	0.25	0	0
liked	0	0.2	0
the	0.01	0.01	0.001
training	0.5	0.5	0.25
a	0	0.002	0
lot	0	0.03	0
didn't	0.02	0	0
	0.5596	0.5395	0.2511
	$\sqrt{\sum_{i=1}^{n} A_i^2}$	$\sqrt{\sum_{i=1}^{n} B_i^2}$	$\sum_{i=1}^{n} A_i B_i$

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}},$$

Issue:
Many similar words are treated differently.

similarity(A, B)

~~= 3 / (√5 * √6)~~

~~≈ 0.5477~~

= 0.2511 / (0.5596 * 0.5395)

≈ 0.8317

Stemming and Lemmatization

Reduce words to their "root" or "lemma"

Stemming: find/replace word endings

Lemmatization: lookup "dictionary form" of word

Lemmatization requires part-of-speech tagging.

Original	Stemmed	Lemmatized
running	runn (-ing)	run
ran	ran	run
is	is	be
was	wa (-s)	be
studies	studi (-es)	study
studying	study (-ing)	study
better	bett (-er)	good
betting	bett (-ing)	bet

Part-of-Speech Tagging

Assign a "tag" to each word, such as:

- noun
- verb
- article
- adjective
- preposition
- pronoun
- adverb
- conjunction
- interjection.

Text	A	B	Ai*Bi
I	0.01	0.01	0.0001
like	0.25	0	0
liked	0	0.2	0
the	0.01	0.01	0.001
training	0.5	0.5	0.25
a	0	0.002	0
lot	0	0.03	0
didn't	0.02	0	0
	0.5596	0.5395	0.2511
	$\sqrt{\sum_{i=1}^{n} A_i^2}$	$\sqrt{\sum_{i=1}^{n} B_i^2}$	$\sum_{i=1}^{n} A_i B_i$

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}},$$

Issue:
Many similar words are treated differently.

similarity(A, B)

~~= 3 / (√5 * √6)~~

~~≈ 0.5477~~

= 0.2511 / (0.5596 * 0.5395)

≈ 0.8317

Text	A	B	Ai*Bi
I	0.01	0.01	0.0001
like	0.2	0.2	0.04
the	0.01	0.01	0.001
training	0.5	0.5	0.25
a	0	0.002	0
lot	0	0.03	0
didn't	0.02	0	0
	0.5596	0.5395	0.2911
	$\sqrt{\sum_{i=1}^{n} A_i^2}$	$\sqrt{\sum_{i=1}^{n} B_i^2}$	$\sum_{i=1}^{n} A_i B_i$

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

Issue:
~~Many similar words are treated differently.~~
Solved!

similarity(A, B)

= 0.2911 / (0.5596 * 0.5395)

≈ 0.964

Text	A	B	Ai*Bi
I	0.01	0.01	0.0001
like	0.2	0.2	0.04
the	0.01	0.01	0.001
training	0.5	0.5	0.25
a	0	0.002	0
lot	0	0.03	0
didn't	0.02	0	0
	0.5596	0.5395	0.2911
	$\sqrt{\sum_{i=1}^{n} A_i^2}$	$\sqrt{\sum_{i=1}^{n} B_i^2}$	$\sum_{i=1}^{n} A_i B_i$

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}},$$

similarity(A, B)

395)

≈ 0.964

Issue:
This measure doesn't incorporate *semantics*

Beyond Bag of Words

We would like to come up with a vector representation that captures meaning

Other wanted benefits:

- Smaller vectors
- Dense
- Reusable

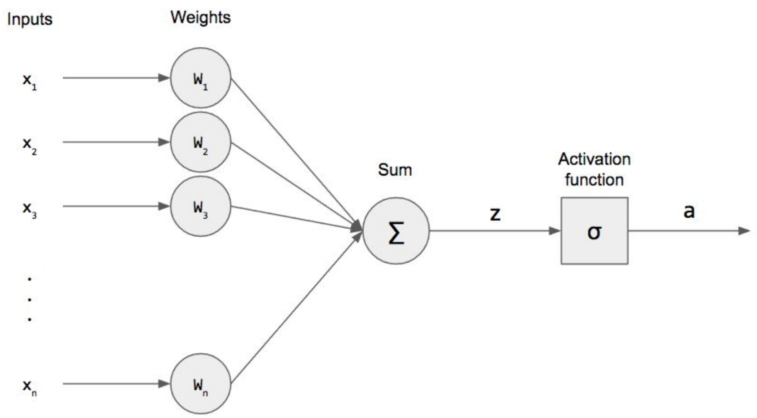
Super Basics of Neural Networks

Given input data, target outputs

Learn parameters to minimize loss

Training consists of feedforward and backpropagation

Basic building block: perceptron

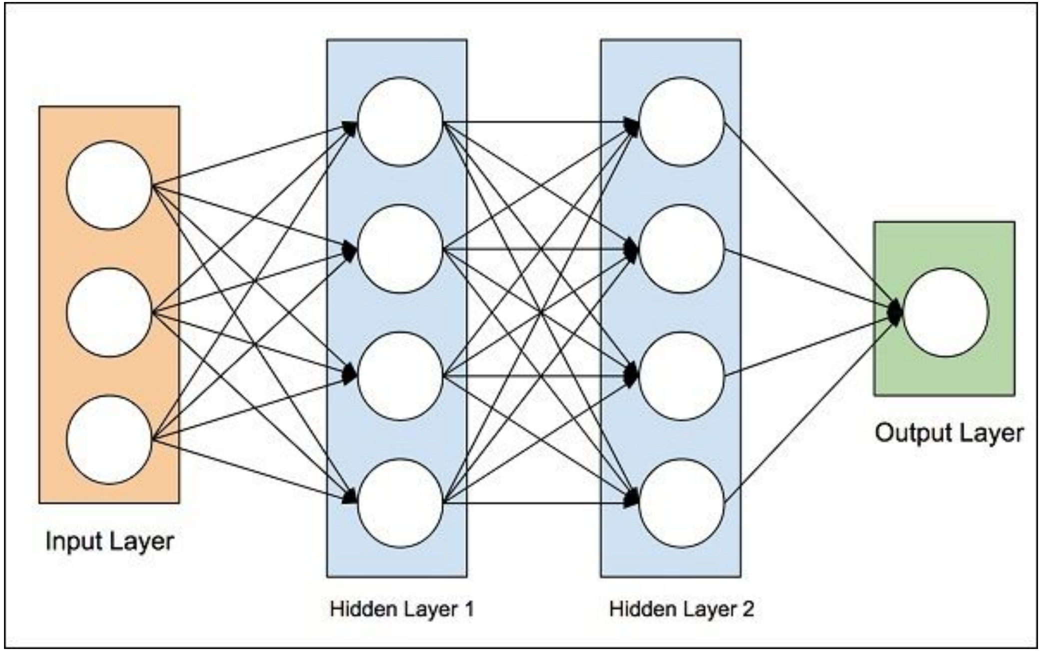


Super Basics of Neural Networks

Stack perceptrons to make network

Arrows indicate learnable weights

Circles sum all inputs and apply activation functions



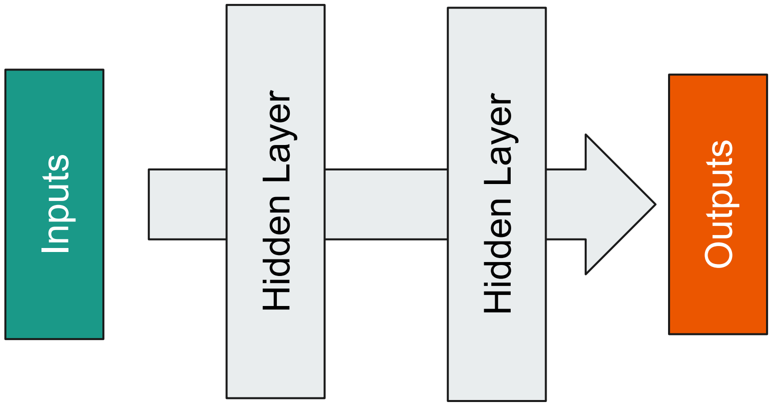
Super Basics of Neural Networks

Stack perceptrons to make network

Arrows indicate learnable weights

Circles sum all inputs and apply activation functions

People often really simplify these diagrams

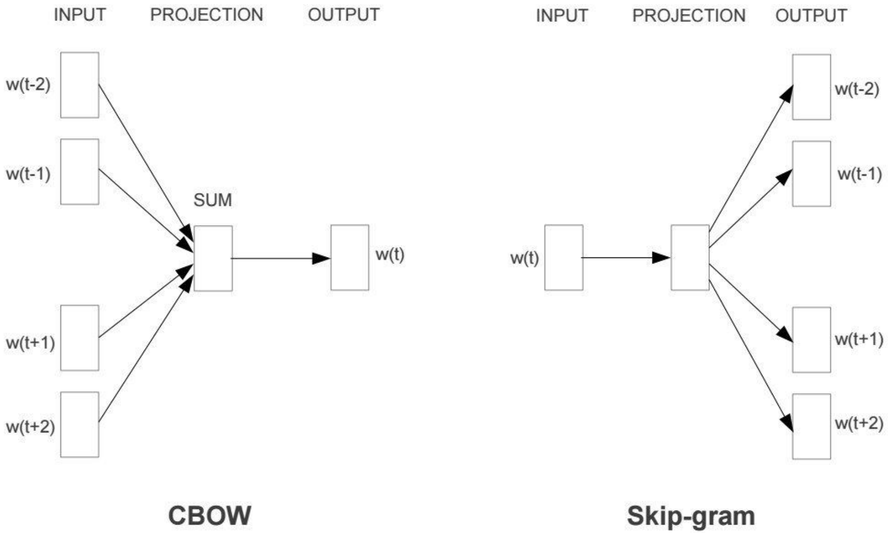


Word2Vec

Create a neural network to learn useful word representations

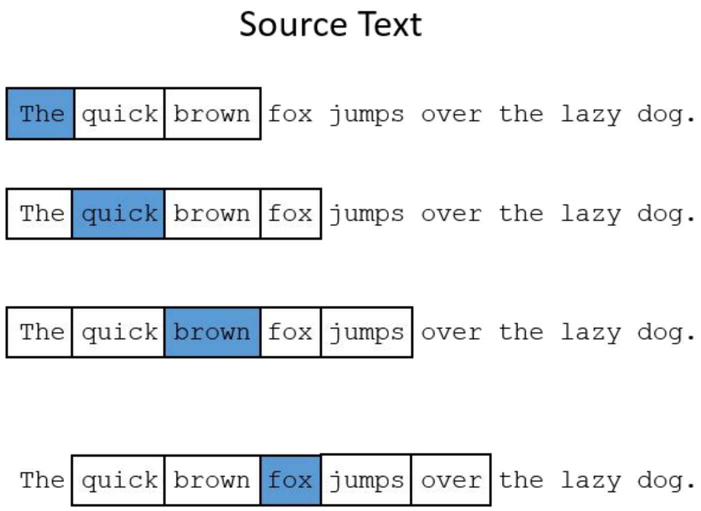
Words are "known by the company they keep"

Neural network learns to predict words by their co-occurrences



Sampling: Sliding Window

Record "center word" (in blue) and "context words"

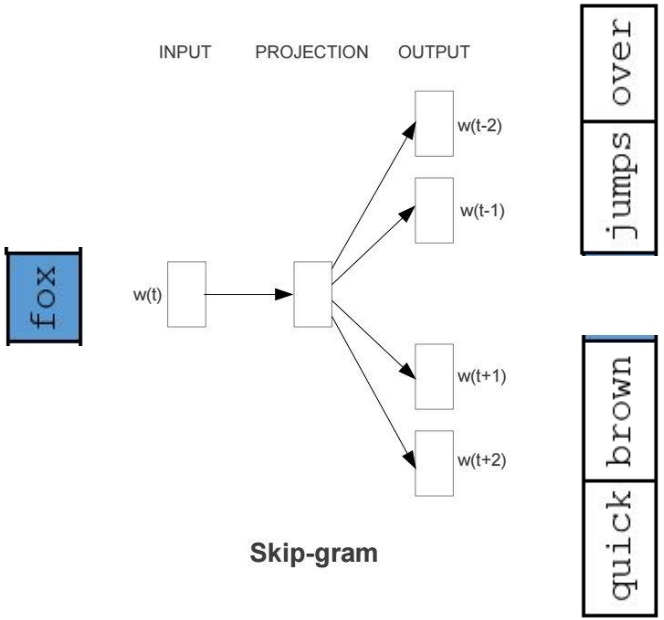


Training

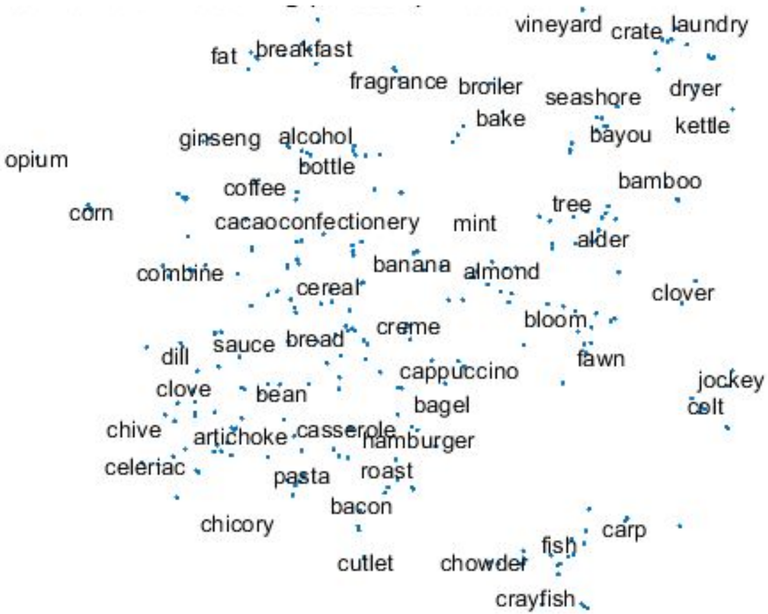
Lookup center word

Predict context words in order

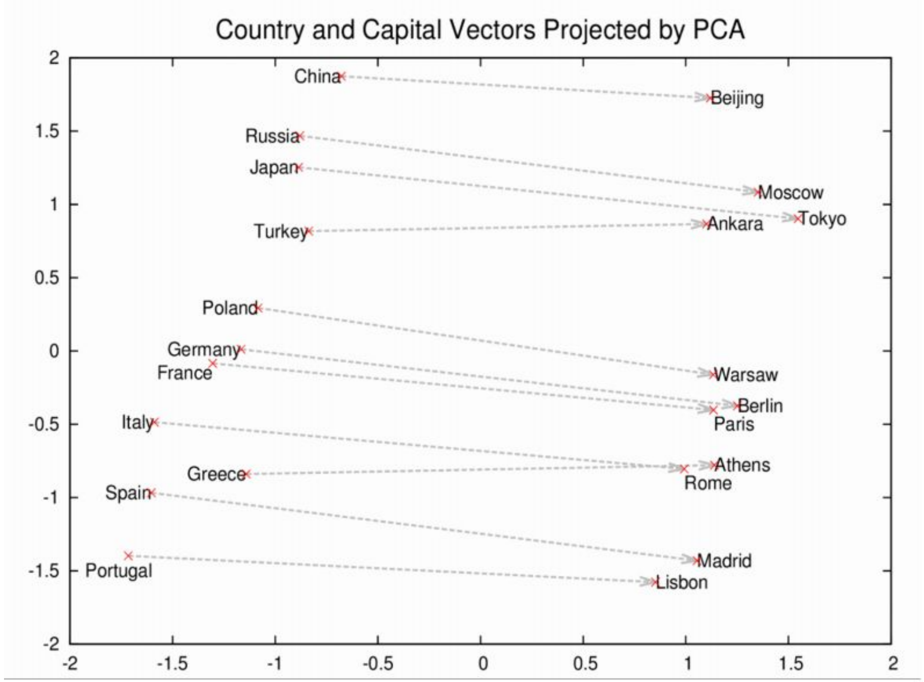
Extract embeddings from internal weights to the model



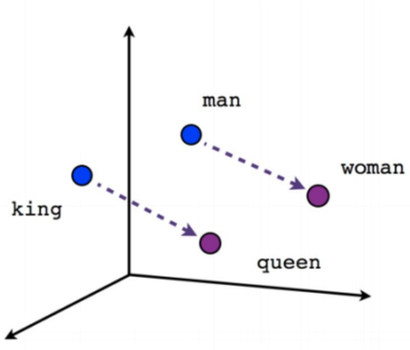
Embedding Visualized



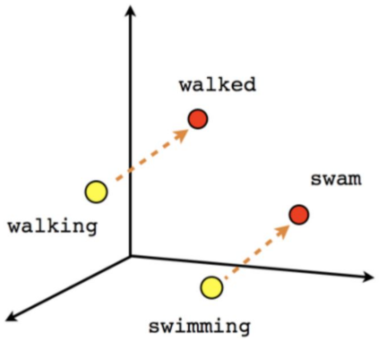
Properties of embeddings



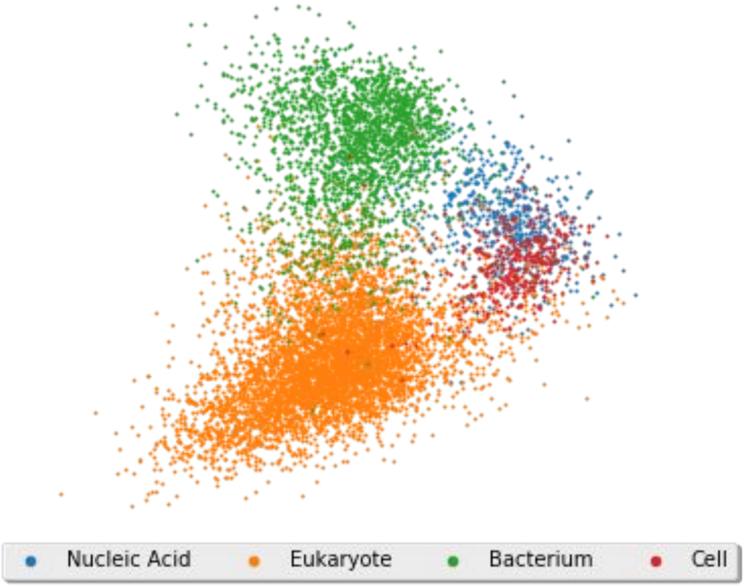
Additional Vector Properties



Male-Female



Verb tense



Improve performance

Preprocessing techniques, such as POS tagging, lemmatization, and stopword removal all improve performance of Word2Vec embeddings.

Corpus size: Google trained on Google News (3 billion words)

Embedding size: Between 100- and 500-dimensional embeddings

Text	A	B	Ai*Bi
I	0.01	0.01	0.0001
like	0.2	0.2	0.04
the	0.01	0.01	0.001
training	0.5	0.5	0.25
a	0	0.002	0
lot	0	0.03	0
didn't	0.02	0	0
	0.5596	0.5395	0.2911
	$\sqrt{\sum_{i=1}^{n} A_i^2}$	$\sqrt{\sum_{i=1}^{n} B_i^2}$	$\sum_{i=1}^{n} A_i B_i$

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}},$$

similarity(A, B)

395)

≈ 0.964

Issue:
This measure doesn't incorporate *semantics*

Cosine Similarity Embedding Example

Embedding Table			
	I	0.1	0
	didn't	0.1	-1
	like	0	0.5
	the	0	0.1
	train	-0.1	0.25
	a	0.1	0
	lot	-1	0

I didn't like the training

I liked the training a lot

I didn't like the training

I liked the training a lot

I didn't like the training

I	0.1	0
didn't	0.1	-1
like	0	0.2
the	0	0.1
train	-0.1	0.1

I liked the training a lot

I	0.1	0
like	0	0.2
the	0	0.1
train	-0.1	0.1
a	0.1	0
lot	-1	0

For short texts, average embeddings to get document representation.

I didn't like the training		
I	0.1	0
didn't	0.1	-1
like	0	0.2
the	0	0.1
train	-0.1	0.1
Average:	0.02	-0.12

I liked the training a lot		
I	0.1	0
like	0	0.2
the	0	0.1
train	-0.1	0.1
a	0.1	0
lot	-1	0
Average:	-0.15	0.06

For short texts, average embeddings to get document representation.

I didn't like the training		
I	0.1	0
didn't	0.1	-1
like	0	0.2
the	0	0.1
train	-0.1	0.1
Average:	0.02	-0.12

I liked the training a lot		
I	0.1	0
like	0	0.2
the	0	0.1
train	-0.1	0.1
a	0.1	0
lot	-1	0
Average:	-0.15	0.06

Replace BOW columns with embeddings

Doc A	Doc B	Ai*Bi
0.02	-0.15	−0.003
-0.12	0.06	−0.007
0.122	0.162	-0.01
$\sqrt{\sum_{i=1}^{n} A_i^2}$	$\sqrt{\sum_{i=1}^{n} B_i^2}$	$\sum_{i=1}^{n} A_i B_i$

Replace BOW columns with embeddings

For short texts, average embeddings to get document representation.

I didn't like the training		
I	0.1	0
didn't	0.1	-1
like	0	0.2
the	0	0.1
train	-0.1	0.1
Average:	0.02	-0.12

I liked the training a lot		
I	0.1	0
like	0	0.2
the	0	0.1
train	-0.1	0.1
a	0.1	0
lot	-1	0
Average:	-0.15	0.06

Doc A	Doc B	Ai*Bi
0.02	-0.15	−0.003
-0.12	0.06	−0.007
0.122	0.162	-0.01
$\sqrt{\sum_{i=1}^{n} A_i^2}$	$\sqrt{\sum_{i=1}^{n} B_i^2}$	$\sum_{i=1}^{n} A_i B_i$

$$\frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} = -0.01 / (0.122 * 0.162)$$

$$\approx -0.505$$

Smaller Issues with Word2Vec

1) Word2Vec cannot handle out-of-vocabulary words

Bad solution: add an "Unknown" embedding

2) Large vocabularies require very large embedding tables

Bad solution: remove any word that only occurs once

Bad solution: make "meta" tokens, such as "[NUMBER]" or "[NAME]"

Big Issue with Word2Vec

Embeddings are fixed after training

Homographs: same spelling, different word

BERT &
Transformer Models

First there was ElMO

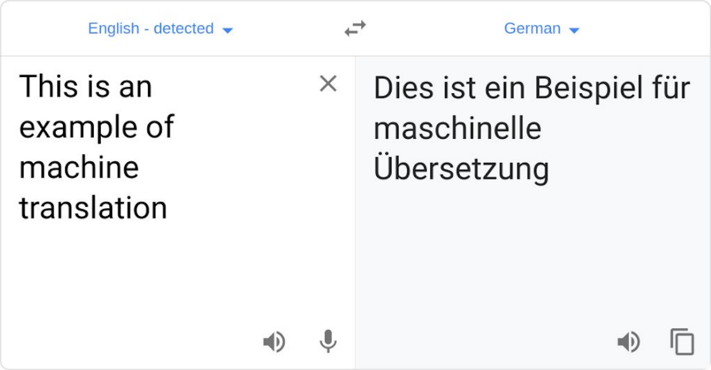
Then, there was BERT

Now, there's a GROVER, and ERNIE, and so many muppet names

Super Basics of Transformers

The model itself is very outside the scope of this talk

Designed around machine translation



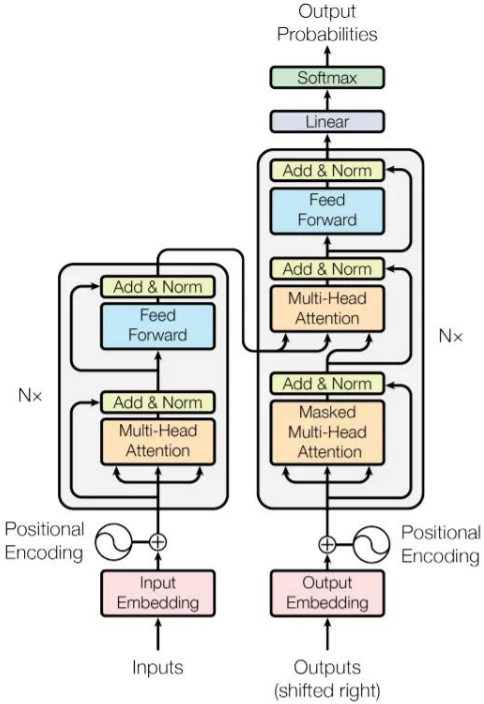
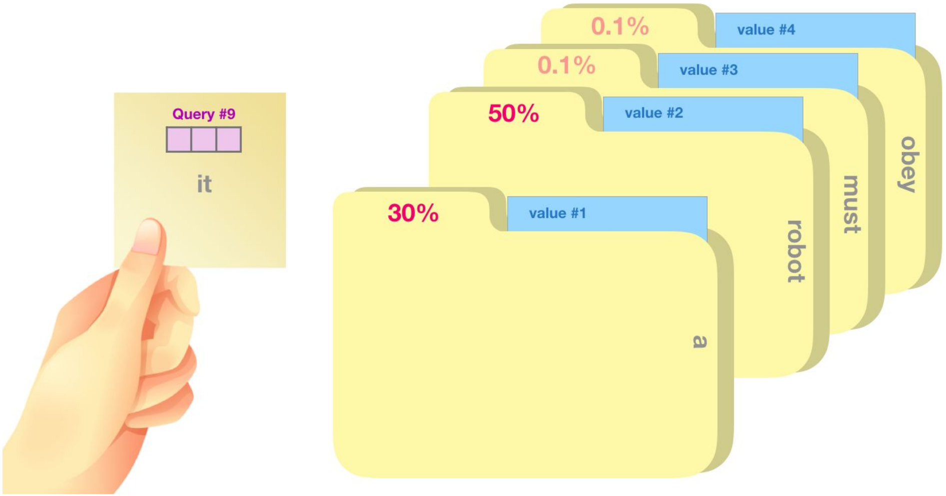


Figure 1: The Transformer - model architecture.

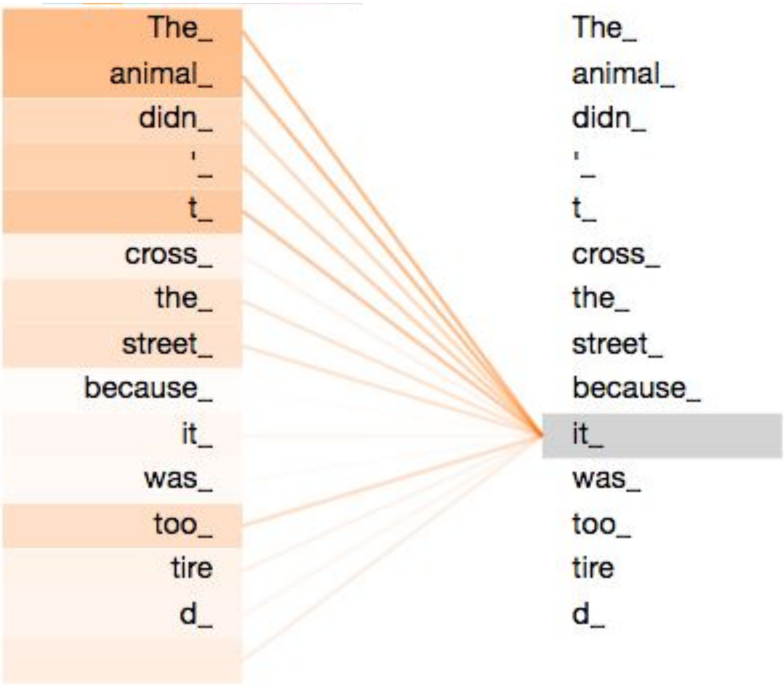
Super Basics of Transformers

Embeddings are learned weighted averages of other words in the same sentence



Super Basics of Transformers

Per-word weights are called "attentions" and are interpretable



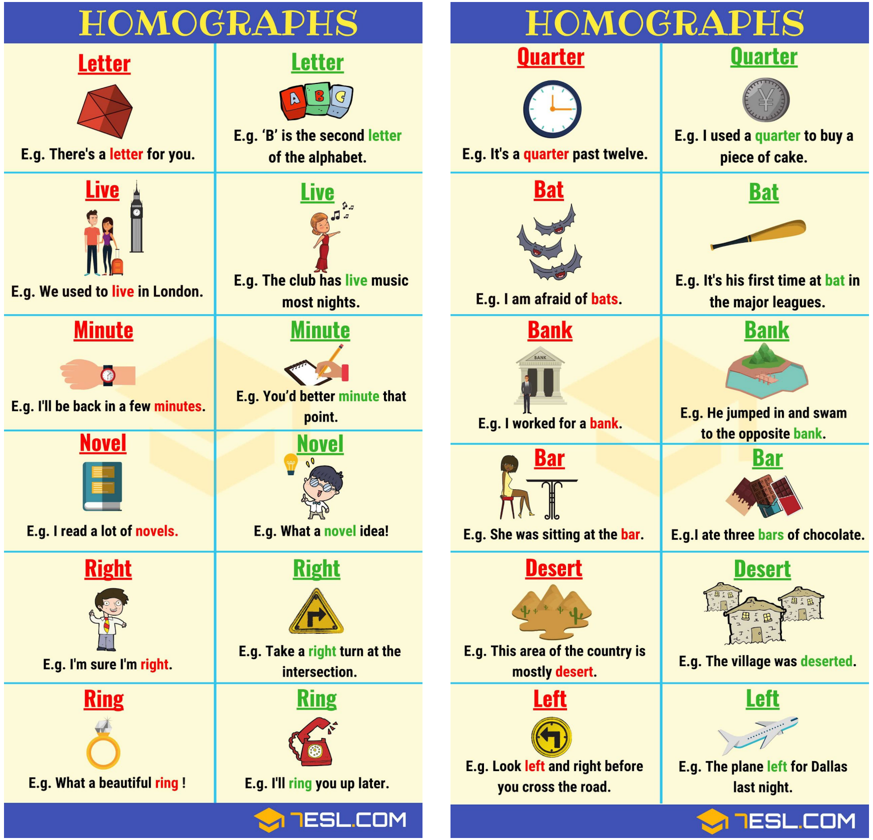
```python
import torch
from transformers import BertTokenizer, BertModel

text = "..."

# Load the model, downloads if nessesary
tokenizer = BertTokenizer.from_pretrained("bert-base-uncased")
model = BertModel.from_pretrained("bert-base-uncased")

# Convert text to int ids
tokens = torch.tensor([tokenizer.encode(text, add_special_tokens=True)])
# We're done!
embeddings = model(tokens)[0]
```

~~Big Issue with Word2Vec~~

~~Embeddings are fixed after training~~

~~Homographs: same spelling, different word~~

Solved!



Smaller Issues with Word2Vec

1) Word2Vec cannot handle out-of-vocabulary words

Bad solution: add an "Unknown" embedding

2) Large vocabularies require very large embedding tables

Bad solution: remove any word that only occurs once

Bad solution: make "meta" tokens, such as "[NUMBER]" or "[NAME]"

Beyond "Words"

BERT uses the "WordPiece" tokenizer

Rather than split text on spaces, learn useful character sequences

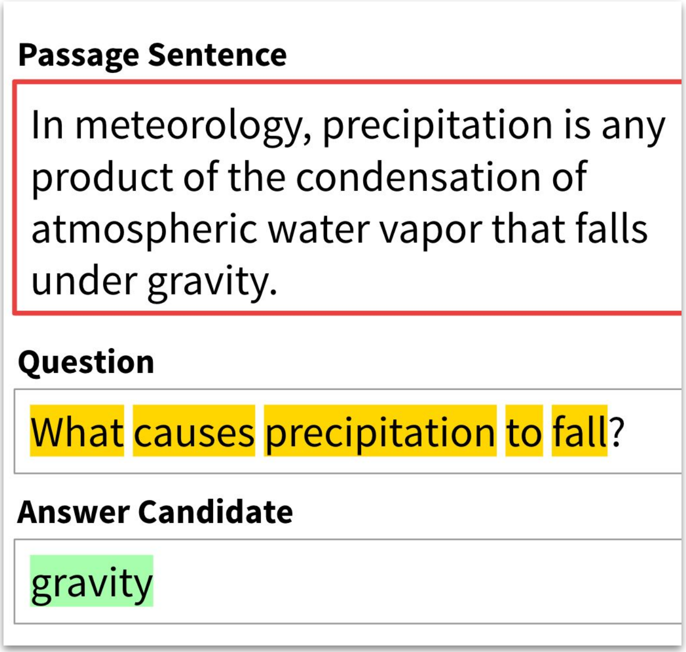
Helps with out-of-vocabulary words

Provides fixed vocab size

```
Input: "I saw a girl with a telescope."

Output: [I][_saw][_a][_girl][_with][_a][_][te][le][s][c][o][pe][.]
```

What can we do with BERT?

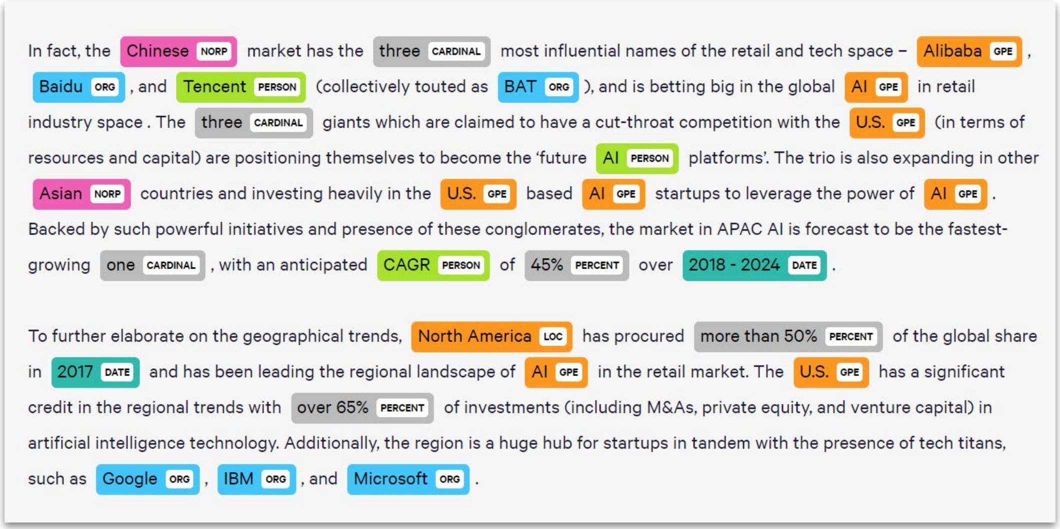
Question Answering



What can we do with BERT?

Question Answering

Entity Extraction



In fact, the Chinese NORP market has the three CARDINAL most influential names of the retail and tech space – Alibaba GPE , Baidu ORG , and Tencent PERSON (collectively touted as BAT ORG), and is betting big in the global AI GPE in retail industry space . The three CARDINAL giants which are claimed to have a cut-throat competition with the U.S. GPE (in terms of resources and capital) are positioning themselves to become the 'future AI PERSON platforms'. The trio is also expanding in other Asian NORP countries and investing heavily in the U.S. GPE based AI GPE startups to leverage the power of AI GPE . Backed by such powerful initiatives and presence of these conglomerates, the market in APAC AI is forecast to be the fastest-growing one CARDINAL , with an anticipated CAGR PERSON of 45% PERCENT over 2018 - 2024 DATE .

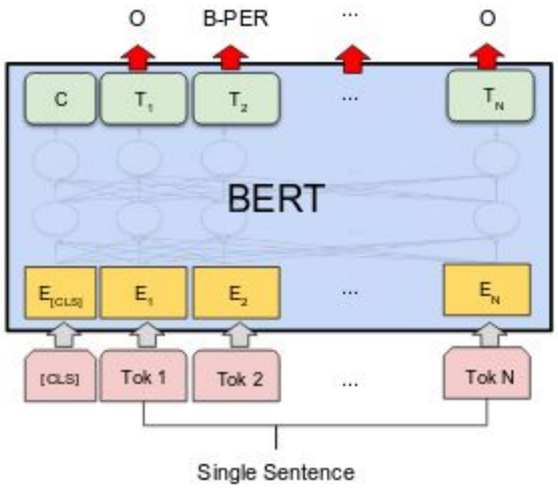
To further elaborate on the geographical trends, North America LOC has procured more than 50% PERCENT of the global share in 2017 DATE and has been leading the regional landscape of AI GPE in the retail market. The U.S. GPE has a significant credit in the regional trends with over 65% PERCENT of investments (including M&As, private equity, and venture capital) in artificial intelligence technology. Additionally, the region is a huge hub for startups in tandem with the presence of tech titans, such as Google ORG , IBM ORG , and Microsoft ORG .

What can we do with BERT?

Question Answering

Entity Extraction

Part of Speech Tagging



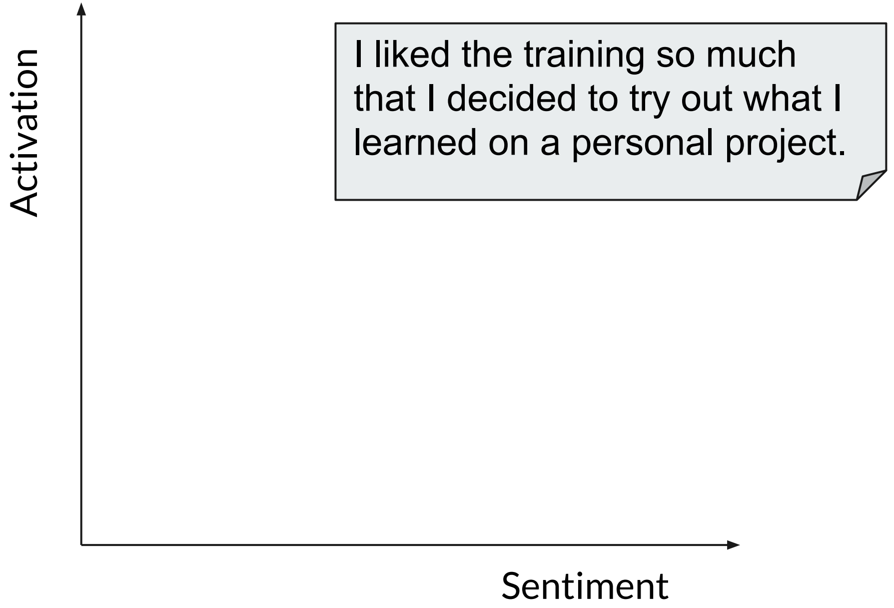
What can we do with BERT?

Question Answering

Entity Extraction

Part of Speech Tagging

Sentiment Analysis / Regression



Generating Text

Language model

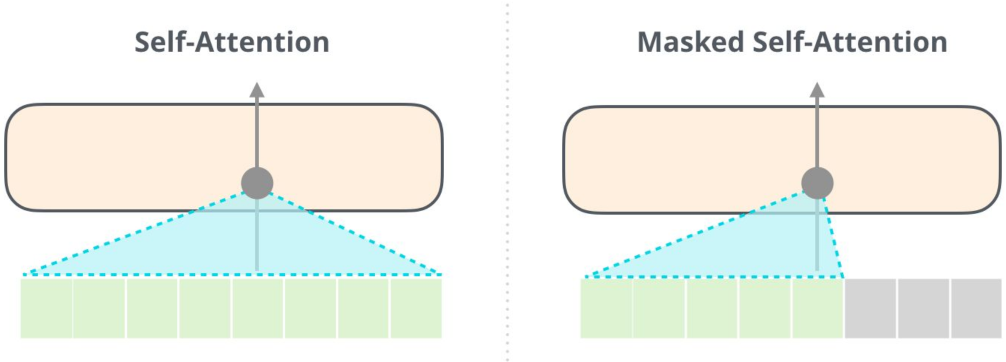
Given tokens t_0 -> t_{i-1} return the probability distribution of t_i

"If you're happy and you know it"

clap	0.7
your	0.2
head	0.01
hands	0.09
...	...

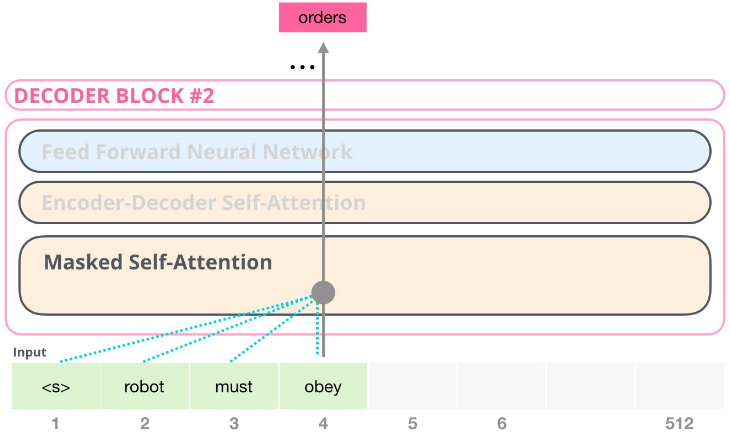
GPT-2

Changes the attention mechanism of BERT



GPT-2 as a Language Model

Masked attention allows us to predict every word given all PREVIOUS words.



Talk to Transformer: Play with generation!

https://talktotransformer.com/

While not normally known for his musical talent, Elon Musk is releasing a debut album. The "Elon Musk" is a collection of eight new songs which are inspired by the founder's life. The music, which is available for pre-order on iTunes, was created by one-man-band and fellow Tesla Motors and SpaceX executive, Paul Kasmin, who's known for playing guitar at Tesla events. The album is a collaboration between Kasmin and Musk himself, although it's also being marketed under the Tesla brand.

Summary

- BOW
    - Stopwords
    - TFIDF
    - Stemming & Lemmatization
    - Part-of-speech tagging
    - Cosine Similarity
- Word2Vec
    - CBOW + SkipGram
    - Learn words by company they keep
    - Embeddings capture semantic properties
- BERT
    - Change word embeddings based on company
    - Uses smaller wordpiece vocab