KI-Campus
Die Lernplattform
für Künstliche Intelligenz

Deutsches Forschungszentrum für Künstliche Intelligenz GmbH

GEFÖRDERT VOM
Bundesministerium für Bildung und Forschung

Vector Representation

Salar Mohtaj | DFKI

Vector representation

- What is vectorization?
- Sparse vs. dense vectors
- Text to vector
- Vector similarity measures
- Text vectorization in Python

Vector representation

- What is vectorization?
- Sparse vs. dense vectors
- Text to vector
- Vector similarity measures
- Text vectorization in Python

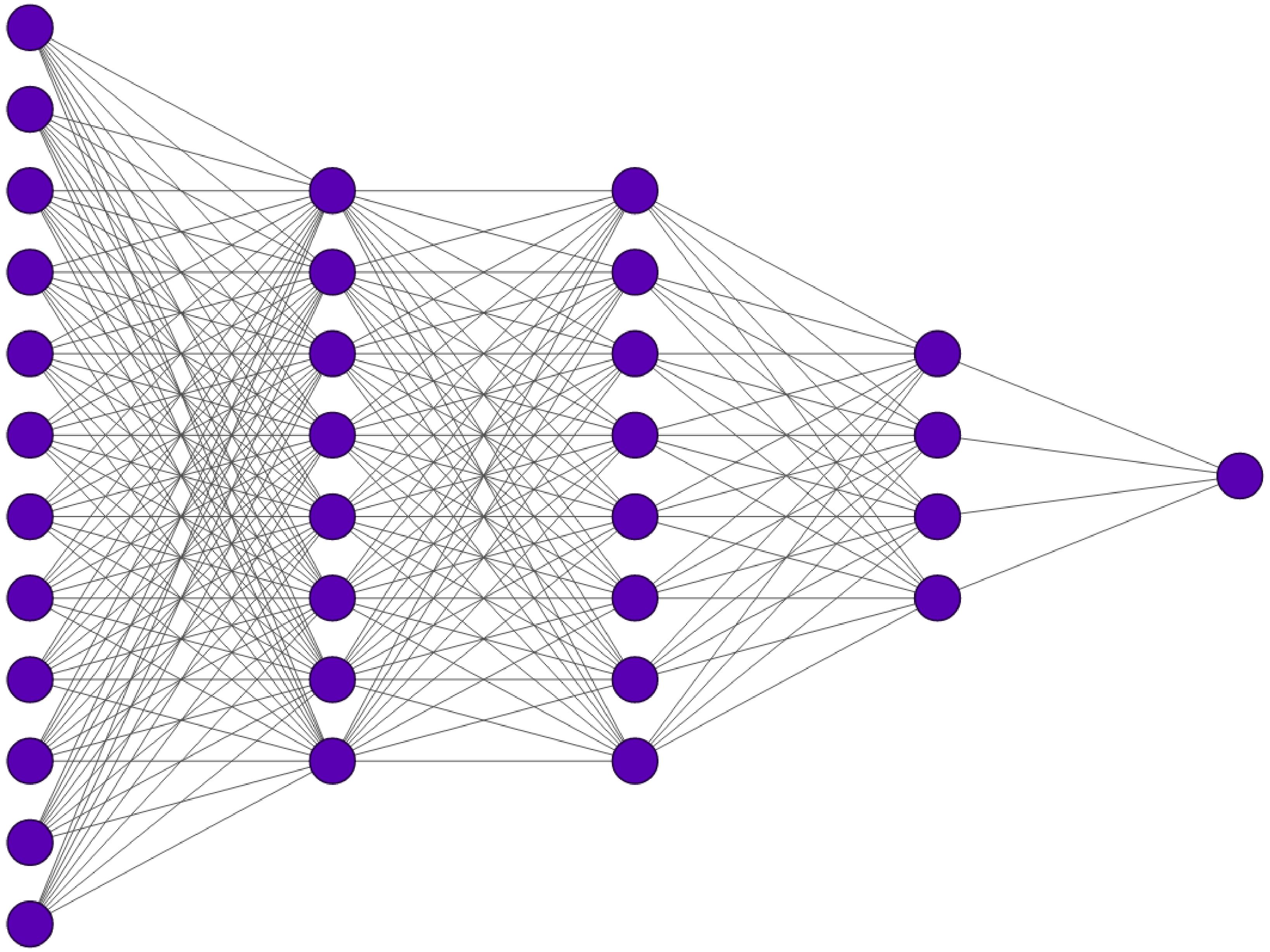
What is vectorization?

- Machine learning algorithms and deep learning architectures are incapable of processing ***strings*** or ***plain text*** in their raw form
- ***Vectorization*** is the process of converting ***string*** or ***plain text*** into a ***vector of numbers***
- ***Vectorization*** is one of the basic buildings blocks in NLP, especially for neural networks

What is vectorization?

language

1	0	0	…	-1



What is vectorization?

- Machine learning algorithms and deep learning architectures are incapable of processing ***strings*** or ***plain text*** in their raw form
- ***Vectorization*** is the process of converting ***string*** or ***plain text*** into a ***vector of numbers***
- ***Vectorization*** is one of the basic buildings blocks in NLP, especially for neural networks
- This process of converting ***text into vectors*** is called ***feature extraction*** or more simply, ***vectorization***

What is vectorization?

Word vectors

women	1	0	0	…	-1
history	-1	1	0	…	1
program	1	1	1	…	0
…	0	0	0	…	0

What is vectorization?

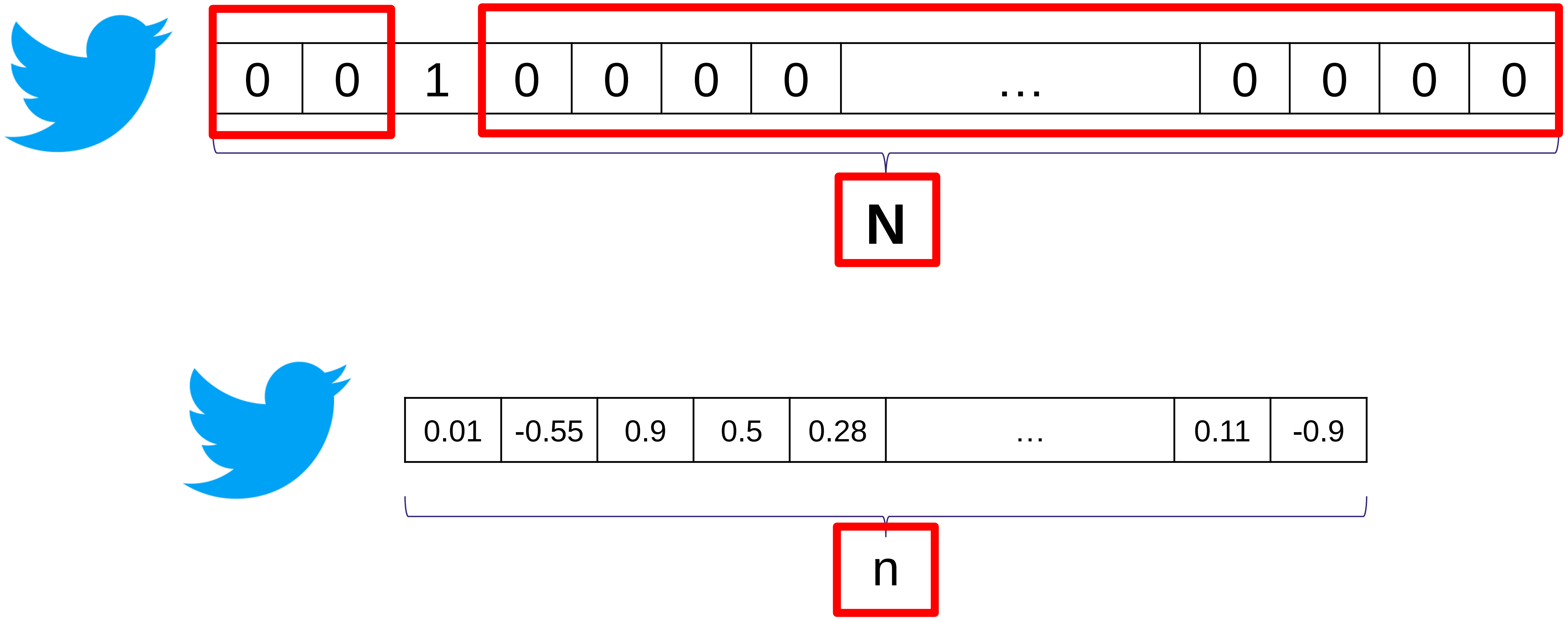
Text vectors

1	-1	0	1	0	0	…	-1

Vector representation

- What is vectorization?
- Sparse vs. dense vectors
- Text to vector
- Vector similarity measures
- Text vectorization in Python

Sparse vs. dense vectors

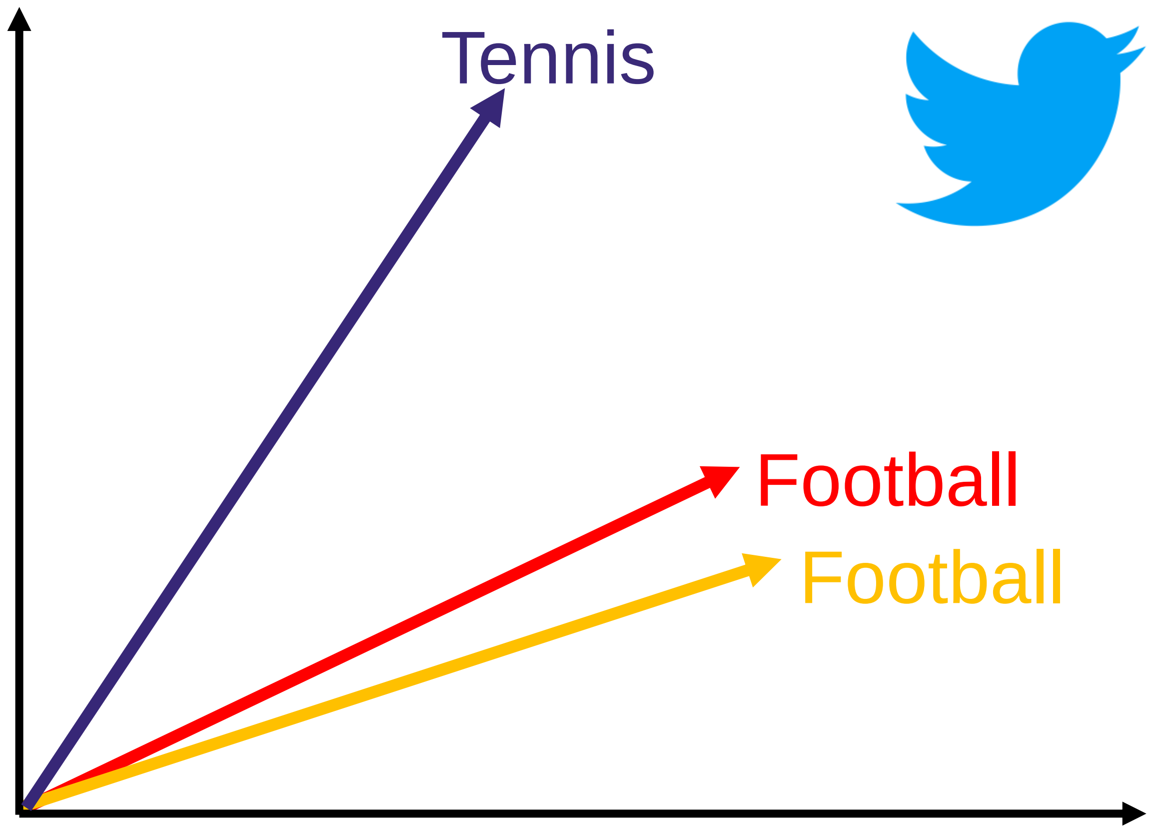


Vector representation

- What is vectorization?
- Sparse vs. dense vectors
- Text to vector
- Vector similarity measures
- Text vectorization in Python

Text vectorization

- Converting textual documents into machine readable vector of numbers
- Main objective: **similar text must result in closer vector**



Text vectorization

- D_1 = “Text is a complex human language representation.”
- D_2 = “Natural human language is complex and also is diverse.”
- D_3 = “Natural human body clock is complex.”
- D_4 = “Text representation differs from human to human.”

- ***Similar*** text must result in ***closer*** vector

Text vectorization

- Bag of Words (BoW)
- Bag of N-Gram
- TF-IDF

Bag of Words (BoW)

- The main idea is that similar documents contain similar terms
- It counts how many times a word appears in a document
- Why it is called **bag of words?**
  - Because any order of the words in the document is discarded

Bag of Words (BoW)

- D_1 = “Text is a complex human language representation.”
- D_2 = “Natural human language is complex and also is diverse.”
- D_3 = “Natural human body clock is complex.”
- D_4 = “Text representation differs from human to human.”

clock
is
human
language
by
natural
a
diverse
to
text
differs
representation
of
complex
body
and
also

- Preprocessing
  - Lowercasing
  - Punctuation removal
  - Word tokenization
  - Stop word removal

Bag of Words (BoW)

- D_1 = “Text is a complex human language representation.”
- D_2 = “Natural human language is complex and also is diverse.”
- D_3 = “Natural human body clock is complex.”
- D_4 = “Text representation differs from human to human.”

clock
is
human
language
natural
diverse
text
differs
representation
complex
body

clock
is
human
language
by
natural
a
diverse
to
text
differs
representation
of
complex
body
and
also

	clock	is	human	language	natural	diverse	text	differs	represen tation	complex	body
D_1	0	1	1	1	0	0	1	0	1	1	0
D_2	0	2	1	1	1	1	0	0	0	1	0
D_3	1	1	1	0	1	0	0	0	0	1	1
D_4	0	0	2	0	0	0	1	1	1	0	0

Bag of Words (BoW)

- Pros
  - Simple and easy to understand
- Cons
  - Ignores the location information of words

Forrest Gump is better than Shawshank redemption
Shawshank redemption is better than Forrest Gump

Bag of Words (BoW)

- Pros
  - Simple and easy to understand
- Cons
  - Ignores the location information of words
  - The intuition that high frequency words are more important fails in many cases

	clock	is	human	language	natural	diverse	text	differs	represen tation	complex	body
D_1	0	1	1	1	0	0	1	0	1	1	0
D_2	0	2	1	1	1	1	0	0	0	1	0
D_3	1	1	1	0	1	0	0	0	0	1	1
D_4	0	0	2	0	0	0	1	1	1	0	0

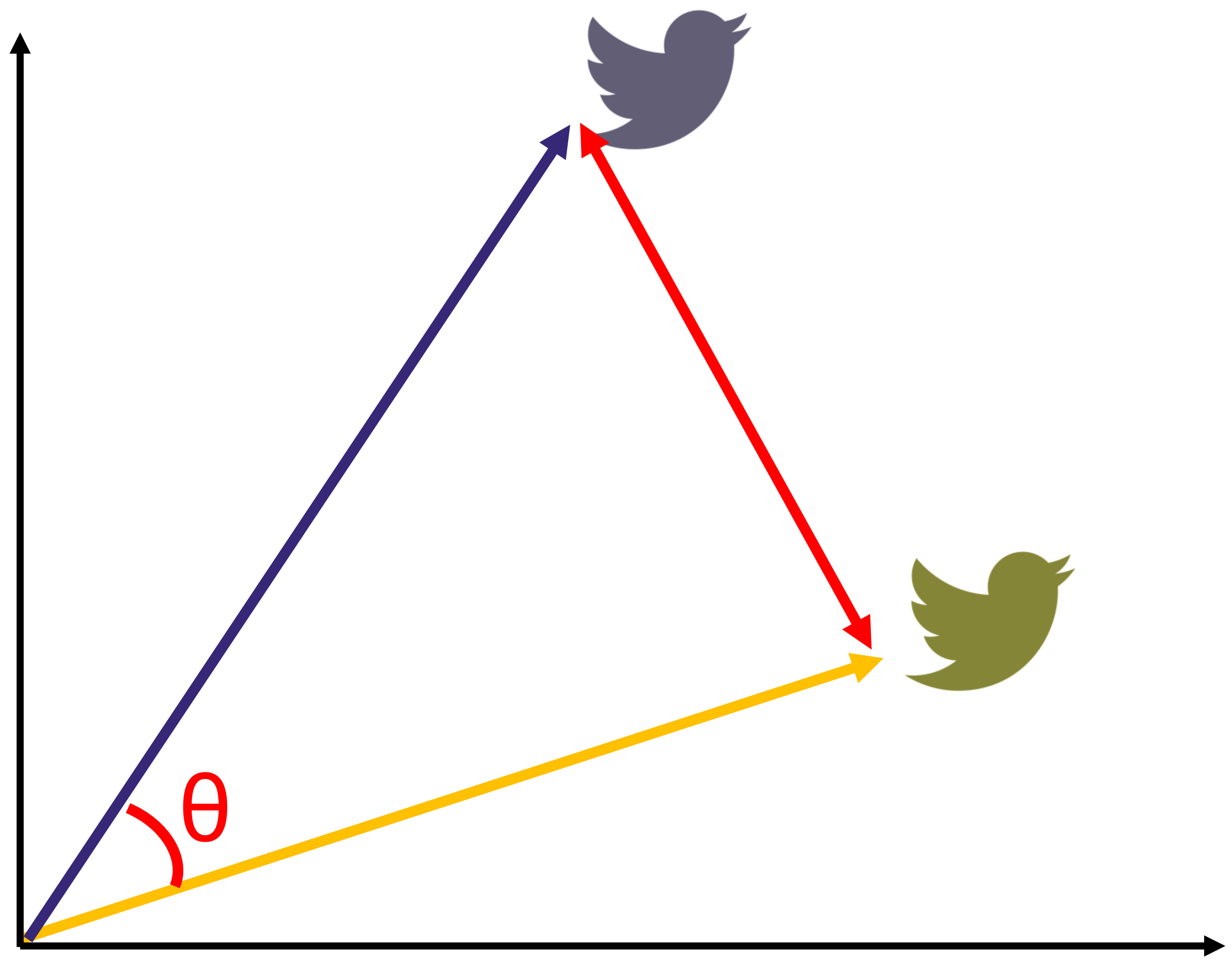
Vector representation

- What is vectorization?
- Sparse vs. dense vectors
- Text to vector
- Vector similarity measures
- Text vectorization in Python

Vector similarity measures

- The objective is to measure and quantify the similarity between two or more vectors (documents)
- Main similarity metrics
  - Cosine similarity
  - Euclidean distance

Distance != Similarity



Cosine similarity

- It is measured by the cosine of the angle between two vectors and determines whether two vectors are pointing in roughly the same direction
- It is often used to measure document similarity in text analysis
- It ranges from 0 to 1
  - 1 means two vectors are in exactly the same direction
  - 0 means two vectors are orthogonal

Cosine similarity

$$\cos(\theta) = \frac{A.B}{\|A\|\|B\|} = \frac{\sum_1^n A_i B_i}{\sqrt{\sum_{1=1}^n A_i^2}\sqrt{\sum_{1=1}^n B_i^2}}$$

- D_1 = “natural language processing”
- D_2 = “natural language understanding”

	natural	language	processing	understanding
D_1	1	1	1	0
D_2	1	1	0	1

$D_1 = [1,1,1,0]$

$D_2 = [1,1,0,1]$

$$\cos(\theta) = \frac{(1*1)+(1*1)+(1*0)+(0*1)}{\sqrt{1+1+1}\sqrt{1+1+1}} = \frac{2}{3}$$

Euclidean distance

- The Euclidean distance metric allows to identify how far two points or two vectors are apart from each other
- It's the length of a line segment between the two points
- It ranges from 0 to N
  - 0 means two vectors are exactly the same
  - N is the distance and can be any positive number

Euclidean distance

$$d(A,B) = \sqrt{\sum_{1=1}^{n} (A_i - B_i)^2}$$

- D_1 = "natural language processing"
- D_2 = "natural language understanding"

	natural	language	processing	understanding
D_1	1	1	1	0
D_2	1	1	0	1

D_1 = [1,1,1,0]

D_2 = [1,1,0,1]

$$\mathrm{d}(D_1, D_2) = \sqrt{(1-1)^2 + (1-1)^2 + (1-0)^2 + (0-1)^2} = \sqrt{2}$$

Backing to our example

- D_1 = “Text is a complex human language representation.”
- D_2 = “Natural human language is complex and also is diverse.”
- D_3 = “Natural human body clock is complex.”
- D_4 = “Text representation differs from human to human.”

	clock	is	human	language	natural	diverse	text	differs	represen tation	complex	body
D_1	0	1	1	1	0	0	1	0	1	1	0
D_2	0	2	1	1	1	1	0	0	0	1	0
D_3	1	1	1	0	1	0	0	0	0	1	1
D_4	0	0	2	0	0	0	1	1	1	0	0

Backing to our example

D_1 = [0,1,1,1,0,0,1,0,1,1,0]
D_2 = [0,2,1,1,1,1,0,0,0,1,0]
D_3 = [1,1,1,0,1,0,0,0,0,1,1]
D_4 = [0,0,2,0,0,0,1,1,1,0,0]

Cosine Similarity				
	D_1	D_2	D_3	D_4
D_1	1			
D_2	0.68	1		
D_3	0.5	0.68	1	
D_4	0.61	0.25	0.30	1

Euclidean Distance				
	D_1	D_2	D_3	D_4
D_1	0			
D_2	2.23	0		
D_3	2.44	2.23	0	
D_4	2.23	3.46	3.0	0

Cosine vs. Euclidean

- Both metrics are **symmetric**
- ***Euclidean distance*** strongly relies on length
- ***Cosine similarity*** is generally used when the magnitude of the vectors does not matter
  - Documents of uneven lengths (e.g., Wikipedia articles)

Text vectorization

- Bag of Words (BoW)
- Bag of N-Gram
- TF-IDF

Bag of N-Gram

- An ***n-gram*** is a contiguous sequence of n items (words) from a given sample of text

natural language processing

- 1-gram (unigram)
- 2-gram (bigram)
- 3-gram (trigram)
- …

natural | language | processing

natural language | language processing

natural language processing

Bag of N-Gram

- It allows the bag-of-words to capture a little bit more meaning from the document
- Comparing to simple bag of word model, it helps to keep the order of words

good not bad

bad not good

Bag of N-Gram

- D_1 = “Text is a complex human language representation.”
- D_2 = “Natural human language is complex and also is diverse.”

- Preprocessing
  - Lowercasing
  - Punctuation removal
  - Word tokenization
- Generating 2-grams

text is
is a
a complex
complex human
human language
language representation
natural human
language is
is complex
complex and
and also
also is
is diverse

Bag of N-Gram

- D_1 = "Text is a complex human language representation."
- D_2 = "Natural human language is complex and also is diverse."

	text is	is a	a complex	complex human	human language	language representation	natural human	language is	is complex	complex and	and also	also is	is diverse
D_1	1	1	1	1	1	1	0	0	0	0	0	0	0
D_2	0	0	0	0	1	0	1	1	1	1	1	1	1

text is
is a
a complex
complex human
human language
language representation
natural human
language is
is complex
complex and
and also
also is
is diverse

Backing to our example

- D_1 = “Text is a complex human language representation.”
- D_2 = “Natural human language is complex and also is diverse.”
- D_3 = “Natural human body clock is complex.”
- D_4 = “Text representation differs from human to human.”

Cosine Similarity				
	D_1	D_2	D_3	D_4
D_1	1			
D_2	0.14	1		
D_3	0.00	0.31	1	
D_4	0.00	0.00	0.00	1

Text vectorization

- Bag of Words (BoW)
- Bag of N-Gram
- TF-IDF

TF-IDF

- A problem with scoring word frequency (e.g., bag of word) is that highly frequent words start to dominate in the document (e.g. larger score)
- But may not contain as much ***informational content*** to the model as rarer but perhaps domain specific words

Natural human language is complex and also is diverse.

- TF-IDF rescale the frequency of words by counting how often they appear in all documents
  - Scores for frequent words like “is” that are also frequent across all documents are penalized

TF-IDF

- ***Term Frequency*** is a scoring of the frequency of the word in the current document
- ***Inverse Document Frequency*** is a scoring of how rare the word is across documents

$$tf = \frac{Number\ of\ repetitions\ of\ word\ in\ a\ document}{Total\ number\ of\ words\ in\ document}$$

$$idf = \log\left(\frac{Total\ number\ of\ documents}{Number\ of\ documents\ containing\ the\ word}\right)$$

$$tfidf = tf\ *idf$$

TF-IDF

- D_1 = "Text is a complex human language representation."
- D_2 = "Natural human language is complex and also is diverse."
- D_3 = "Natural human body clock is complex."
- D_4 = "Text representation differs from human to human."

clock
is
human
language
by
natural
a
diverse
to
text
differs
representation
of
complex
body
and
also

- Preprocessing
  - Lowercasing
  - Punctuation removal
  - Word tokenization
  - Stop word removal

TF-IDF

$$tf = \frac{Number\ of\ repetitions\ of\ word\ in\ a\ document}{Total\ number\ of\ words\ in\ document}$$

- D_1 = “Text is a complex human language representation.”
- D_2 = “Natural human language is complex and also is diverse.”
- D_3 = “Natural human body clock is complex.”
- D_4 = “Text representation differs from human to human.”

TF	is	complex	human
D_1	1/7	0	1/7
D_2	2/9	1/9	1/9
D_3	1/6	1/6	1/6
D_4	0	0	2/7

clock
is
human
language
by
natural
a
diverse
to
text
differs
representation
of
complex
body
and
also

clock
is
human
language
natural
diverse
text
differs
representation
complex
body

TF-IDF

$$idf = \log\left(\frac{Total\ number\ of\ documents}{Number\ of\ documents\ containing\ the\ word}\right)$$

- D_1 = “Text is a complex human language representation.”
- D_2 = “Natural human language is complex and also is diverse.”
- D_3 = “Natural human body clock is complex.”
- D_4 = “Text representation differs from human to human.”

	is	complex	human
IDF	$\log(4/3)$	$\log(4/2)$	$\log(4/4)$
	0.12	0.30	0.0

clock
is
human
language
natural
diverse
text
differs
representation
complex
body

TF-IDF

- D_1 = "Text is a complex human language representation."
- D_2 = "Natural human language is complex and also is diverse."
- D_3 = "Natural human body clock is complex."
- D_4 = "Text representation differs from human to human."

TF-IDF	is	complex	human
D_1	0.14 * 0.12 = 0.01	0	0
D_2	0.02	0.03	0
D_3	0.02	0.05	0
D_4	0	0	0

clock
is
human
language
natural
diverse
text
differs
representation
complex
body

TF-IDF

- D_1 = "Text is a complex human language representation."
- D_2 = "Natural human language is complex and also is diverse."
- D_3 = "Natural human body clock is complex."
- D_4 = "Text representation differs from human to human."

	clock	is	human	language	natural	diverse	text	differs	represen tation	complex	body
D_1	0	0.01	0	0.45	0	0	0.45	0	0.45	0	0
D_2	0	0.02	0	0.30	0.30	0.39	0	0	0	0.03	0
D_3	0.51	0.02	0	0	0.40	0	0	0	0	0.05	0.51
D_4	0	0	0	0	0	0	0.34	0.43	0.34	0	0

Backing to our example

- D_1 = "Text is a complex human language representation."
- D_2 = "Natural human language is complex and also is diverse."
- D_3 = "Natural human body clock is complex."
- D_4 = "Text representation differs from human to human."

Cosine Similarity				
	D_1	D_2	D_3	D_4
D_1	1			
D_2	0.48	1		
D_3	0.32	0.42	1	
D_4	0.45	0.09	0.12	1

TF-IDF

- Different ***TF*** variations

$$\boldsymbol{binary} = 0{,}1$$

$$\boldsymbol{raw\ count} = Number\ of\ repetitions\ of\ word\ in\ a\ document$$

$$\boldsymbol{term\ frequency} = \frac{Number\ of\ repetitions\ of\ word\ in\ a\ document}{Total\ number\ of\ words\ in\ document}$$

$$\boldsymbol{log\ normalization} = \log(1 + Number\ of\ repetitions\ of\ word\ in\ a\ document)$$

TF-IDF

- Different ***IDF*** variations

$$\boldsymbol{unary} = 1$$

$$\boldsymbol{inverse\ document\ frequency} = \log\left(\frac{Total\ number\ of\ documents}{Number\ of\ documents\ containing\ the\ word}\right)$$

$$\boldsymbol{inverse\ document\ frequency\ smooth} = \log\left(\frac{Total\ number\ of\ documents}{1 + Number\ of\ documents\ containing\ the\ word}\right) + 1$$

From TF-IDF to BoW

- Different ***TF*** and ***IDF*** variations

$$\boldsymbol{raw\ count} = Number\ of\ repetitions\ of\ word\ in\ a\ document$$

$$\boldsymbol{unary} = 1$$

$$tfidf = Bag\ of\ Word$$

IDF and Stop-words

- Low ***IDF*** score can hint to ***Stop-word***
  - ***“and”, “a”, “the”, “that”*** and …
  - High representation in almost all documents means very low ***IDF*** score
  - Specific domains (e.g., medical texts)

Vector representation

- What is vectorization?
- Sparse vs. dense vectors
- Text to vector
- Vector similarity measures
- Text vectorization in Python

Text vectorization in Python

- scikit-learn

scikit-learn

- CountVectorizer (Bag of Word)

```
>>> from sklearn.feature_extraction.text import CountVectorizer
>>> corpus = [“sample sentence one”,
                  “sample sentence two”]
>>> BoF_vectorizer = CountVectorizer()
>>> X = BoF_vectorizer.fit_transform(corpus)
>>> print(BoF_vectorizer.get_feature_names())
['one', 'sample', 'sentence', 'two']
>>> print(X.toarray())
[[1, 1, 1, 0],
[0, 1, 1, 1]]
```

scikit-learn

- CountVectorizer (Bag of Word)

```
>>> from sklearn.feature_extraction.text import CountVectorizer
>>> corpus = [“sample sentence one”,
                   “sample sentence two”]
>>> BoF_vectorizer = CountVectorizer()
>>> X = BoF_vectorizer.fit_transform(corpus)
>>> print(BoF_vectorizer.get_feature_names())
['one', 'sample', 'sentence', 'two']
>>> print(X.toarray())
[[1, 1, 1, 0],
[0, 1, 1, 1]]
```

scikit-learn

- CountVectorizer (Bag of N-Gram)

```
>>> from sklearn.feature_extraction.text import CountVectorizer
>>> corpus = [“sample sentence one”,
               “sample sentence two”]
>>> bigram_vectorizer = CountVectorizer(ngram_range=(2, 2))
>>> X = bigram_vectorizer.fit_transform(corpus)
>>> print(bigram_vectorizer.get_feature_names())
['sample sentence', 'sentence one', 'sentence two’]
>>> print(X.toarray())
[[1, 1, 0],
[1, 0, 1]]
```

scikit-learn

- CountVectorizer

```
>>> from sklearn.feature_extraction.text import CountVectorizer
>>> corpus = [“sample sentence one”,
                  “sample sentence two”]
>>> vectorizer = CountVectorizer(ngram_range=(1, 2))
>>> X = vectorizer.fit_transform(corpus)
>>> print(vectorizer.get_feature_names())
['one', 'sample', 'sample sentence', 'sentence', 'sentence one',
'sentence two', 'two']
>>> print(X.toarray())
[1, 1, 1, 1, 1, 0, 0],
[0, 1, 1, 1, 0, 1, 1]]
```

scikit-learn

- TfidfVectorizer

```
>>> from sklearn.feature_extraction.text import TfidfVectorizer
>>> corpus = [“sample sentence one”,
                  “sample sentence two”]
>>> bigram_vectorizer = TfidfVectorizer(ngram_range=(2, 2))
>>> X = bigram_vectorizer.fit_transform(corpus)
>>> print(bigram_vectorizer.get_feature_names())
['sample sentence', 'sentence one', 'sentence two’]
>>> print(X.toarray())
[[0.57973867, 0.81480247, 0.],
[0.57973867, 0.  , 0.81480247]]
```

Summary

0	0	1	0	0	0	0	…	0	0	0	0

N

0.01	-0.55	0.9	0.5	0.28	…	0.11	-0.9

n

- D_1 = “Text is a complex human language representation.”
- D_2 = “Natural human language is complex and also is diverse.”
- D_3 = “Natural human body clock is complex.”
- D_4 = “Text representation differs from human to human.”

Summary

- Bag of Words (BoW)
- Bag of N-Gram
- TF-IDF