

PSY9511: Seminar 8

Sequence modelling (with an emphasis on language)

Esten H. Leonardsen

24.11.25



UNIVERSITETET
I OSLO

Overview

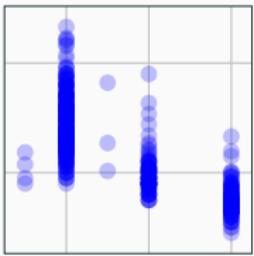
1. Language modelling

- 1.1 Introduction and motivation
- 1.2 Preprocessing
- 1.3 Bag of words
- 1.4 Vectorization
- 1.5 Recurrent neural networks
- 1.6 Transformers

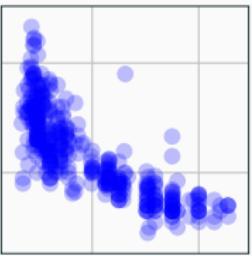


Exercise 5: Exploratory data analysis

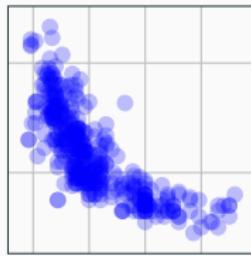
cylinders



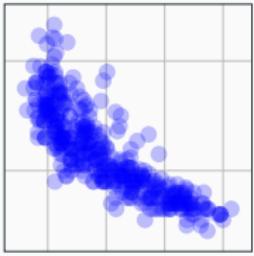
displacement



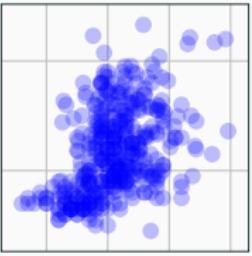
horsepower



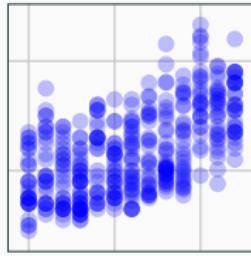
weight



acceleration



year



Exercise 6

[http://localhost:8888/notebooks/notebooks/Solution%
206%20\(Python\).ipynb](http://localhost:8888/notebooks/notebooks/Solution%206%20(Python).ipynb)



Language modelling



UNIVERSITETET
I OSLO

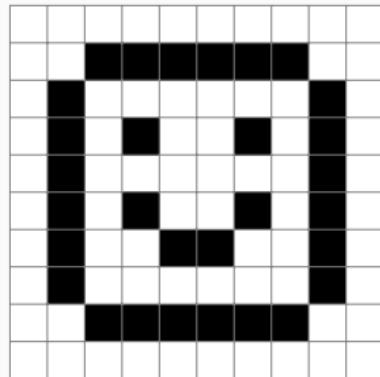
Introduction

Age	Sex	Education	Salary
25	Male	12	40,000
30	Female	16	65,000
35	Male	14	55,000
40	Female	18	80,000
45	Male	16	75,000



Introduction

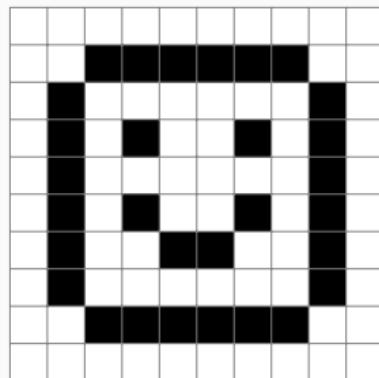
Age	Sex	Education	Salary
25	Male	12	40,000
30	Female	16	65,000
35	Male	14	55,000
40	Female	18	80,000
45	Male	16	75,000



Introduction

The movie was great, the actors were awesome.

Age	Sex	Education	Salary
25	Male	12	40,000
30	Female	16	65,000
35	Male	14	55,000
40	Female	18	80,000
45	Male	16	75,000



Introduction

The movie was great, the actors were awesome.



Introduction

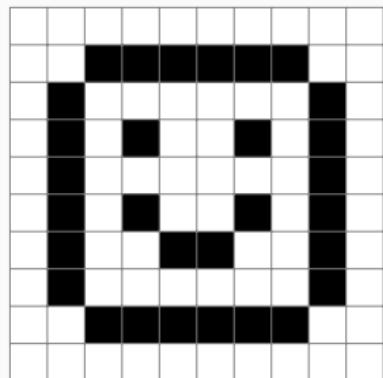
The movie was great, the actors were awesome.



Introduction

The movie was great, the actors were awesome.

Age	Sex	Education	Salary
25	Male	12	40,000
30	Female	16	65,000
35	Male	14	55,000
40	Female	18	80,000
45	Male	16	75,000



Introduction

Age	Sex	Education	Salary
25	Male	12	40,000
30	Female	16	65,000
35	Male	14	55,000
40	Female	18	80,000
45	Male	16	75,000



Introduction

Age

35



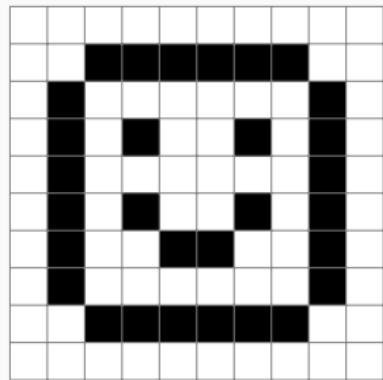
Introduction

Age Sex

35 Male



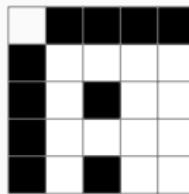
Introduction



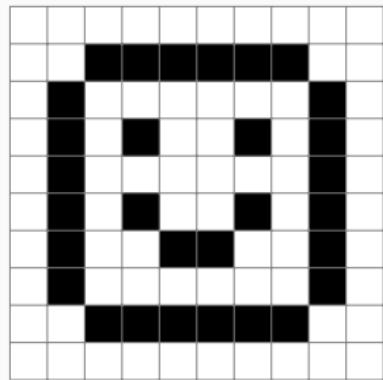
Introduction



Introduction



Introduction



Introduction

The movie was great, the actors were awesome.



Introduction

movie



Introduction

The movie was great, the actors were awesome.



Introduction

The movie was great, the actors were awesome.

Positive

Negative



Introduction

The movie was great, the actors were awesome.

Positive

Negative

The movie was awful, the actors were horrible.



Introduction

The movie was great, the actors were awesome.

La película fue genial, las actores fueron increíbles.



Introduction

The movie was great, the actors were awesome.

A diagram illustrating word-to-word alignment between two sentences. The top sentence is "The movie was great, the actors were awesome." and the bottom sentence is "La película fue genial, las actores fueron increíbles." Seven arrows point from the words in the first sentence to the corresponding words in the second sentence: 'The' points to 'La', 'movie' points to 'película', 'was' points to 'fue', 'great' points to 'genial', 'the' points to 'las', 'actors' points to 'actores', and 'awesome' points to 'increíbles'.

La película fue genial, las actores fueron increíbles.



Introduction

The movie was great, the actors were _____.



Introduction



Introduction

The movie was ____, the actors were _____.



Introduction

The movie was great, the actors were _____.



Introduction

The movie was great, the actors were awesome.



Introduction

The movie was great, the actors were awesome.



Introduction

The movie was great, we saw it at the new Cinema in the city center, the actors were awesome.



Introduction

The movie was great, we saw it at the new Cinema in the city center, right down by the restaurant where we went for my birthday that one year, the one where the clown was inside the cake, the actors were awesome.



Introduction

Language modelling: Using the innate structure in language to create better models

- Classification: Predict a class for a full sequence (sentiment analysis)
- Sequence-to-sequence: Predict a sequence from another sequence (translation)
- Generation: Predict the next token in a sequence of words



Preprocessing



UNIVERSITETET
I OSLO

Preprocessing

The movie was great, the actors were awesome.



Preprocessing

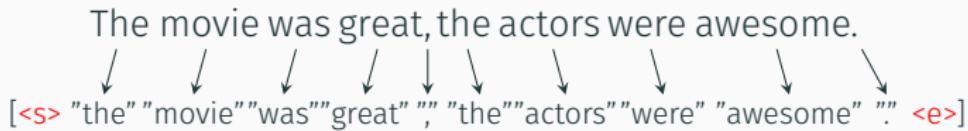
The movie was great, the actors were awesome.
↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓
["the" "movie" "was" "great" ",", "the" "actors" "were" "awesome" "."]

Tokenization



Preprocessing

The movie was great, the actors were awesome.
[<s> "the" "movie" "was" "great" ";" "the" "actors" "were" "awesome" ":" <e>]



The diagram illustrates the tokenization process. It shows the original sentence "The movie was great, the actors were awesome." above a list of tokens in brackets. Nine arrows point from each word in the sentence to its corresponding token in the list below. The tokens are: [<s> "the" "movie" "was" "great" ";" "the" "actors" "were" "awesome" ":" <e>]. The tokens are separated by double quotes, and punctuation marks like commas and a colon are included as part of the tokens.

Tokenization



Preprocessing

The movie was great, the actors were awesome.

[<s> "the" "movie" "was" "great" ",", "the" "actors" "were" "awesome" ":" <e>]

In[1]: `from nltk.tokenize import word_tokenize`

```
tokens = word_tokenize(s)
tokens = [token.lower() for token in tokens]
tokens = ['<s>'] + tokens + ['<e>']
print(tokens)
```

Out[1]: [<s>, 'the', 'movie', 'was', 'great', ',', 'the', 'actors',
'were', 'awesome', '.', '<e>']

Tokenization



Preprocessing

The movie was great, the actors were awesome.

```
[<s> "the" "movie" "was" "great" ",", "the" "actors" "were" "awesome" "." <e>]  
[<s> "The" "movie" "was" "great" ",", "the" "actor" "were" "awesome" "." <e>]
```

Stemming



Preprocessing

The movie was great, the actors were awesome.
[<s> "the" "movie" "was" "great" ",", "the" "actors" "were" "awesome" "." <e>]
[<s> "The" "movie" "was" "great" ",", "the" "actor" "were" "awesome" "." <e>]

```
In[1]: from nltk.stem.snowball import SnowballStemmer
```

```
stemmer = SnowballStemmer('english')
stemmed = [stemmer.stem(token) for token in tokens]
stemmed
```

```
Out[1]: [<s>, 'the', 'movi', 'was', 'great', ',', 'the', 'actor',
'were', 'awesom', '.', '<e>']
```

Stemming



Preprocessing

The movie was great, the actors were awesome.

The diagram illustrates the preprocessing of the sentence "The movie was great, the actors were awesome." It shows two parallel sequences of tokens, each starting with '<s>' and ending with '<e>'. The top sequence represents the original sentence tokens: "the", "movie", "was", "great", ",", "the", "actors", "were", "awesome", ".", The bottom sequence represents the processed tokens, where the word "actor" is highlighted in red: "The", "movie", "was", "great", ",", "the", "actor", "was", "great", ".". Arrows point from each token in the top sequence to its corresponding token in the bottom sequence, except for the punctuation marks which remain in their original positions.

[<s> "the" "movie" "was" "great" "," "the" "actors" "were" "awesome" "." <e>]
[<s> "The" "movie" "was" "great" "," "the" "actor" "was" "great" "." <e>]

Lemmatization



Preprocessing

The movie was great, the actors were awesome.

```
[<s> "the" "movie" "was" "great" ",", "the" "actors" "were" "awesome" "." <e>]  
[<s> "The" "movie" "was" "great" ",", "the" "actor" "was" "great" "." <e>]
```

```
In[1]: from nltk.stem import WordNetLemmatizer  
  
lemmatizer = WordNetLemmatizer()  
lemmatized = [lemmatizer.lemmatize(token) for token in tokens]  
print(lemmatized)
```



```
Out[1]: [<s>, 'the', 'movie', 'wa', 'great', ',', 'the', 'actor',  
'were', 'awesome', '.', <e>]
```

Lemmatization



Preprocessing

The movie was great, the actors were awesome.

[<s> "the" "movie" "was" "great" ",", "the" "actors" "were" "awesome" "." <e>]

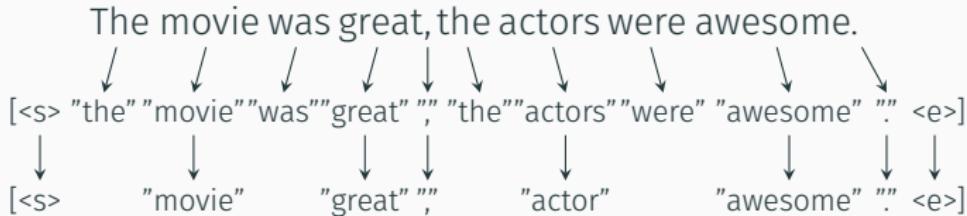
[<s> "movie" "great" "," "actor" "awesome" "<e>"]

```
graph TD; T1["[<s> "the" "movie" "was" "great" ", "the" "actors" "were" "awesome" "." <e>]"] --> T2["[<s> "movie" "great" "," "actor" "awesome" "<e>"]"]; T1 --> W1["<s>"]; T1 --> W2[""movie""]; T1 --> W3[""was""]; T1 --> W4[""great""]; T1 --> W5["", ""]; T1 --> W6[""the""]; T1 --> W7[""actors""]; T1 --> W8[""were""]; T1 --> W9[""awesome""]; T1 --> W10["".""]; T1 --> W11["<e>"]; T2 --> T1; T2 --> W12["<s>"]; T2 --> W13[""movie""]; T2 --> W14[""great""]; T2 --> W15["",""]; T2 --> W16[""actor""]; T2 --> W17[""awesome""]; T2 --> W18["<e>"];
```

Stopword removal



Preprocessing



```
In[1]: from nltk.corpus import stopwords
```

```
pruned = [token for token in tokens if not token in stopwords.  
          words('english')]  
print(pruned)
```

```
Out[1]: [<s>, 'movie', 'great', ',', 'actors', 'awesome', '.', <e>]
```

Stopword removal



Preprocessing

The movie was great, the actors were awesome.

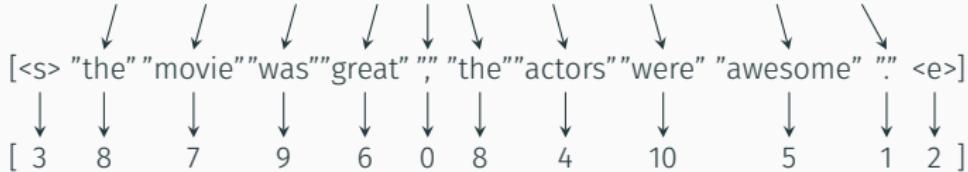
[<s> "the" "movie" "was" "great" ",", "the" "actors" "were" "awesome" "." <e>]

[", ".", <e> <s> "actors" "awesome" "great" "movie" "the" "was" "were"]
0 1 2 3 4 5 6 7 8 9 10



Preprocessing

The movie was great, the actors were awesome.

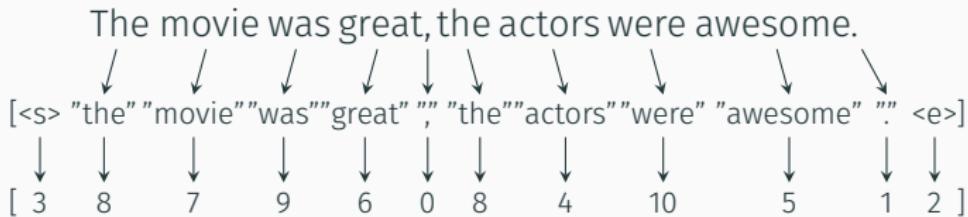


["", ".", <e>, <s>, "actors", "awesome", "great", "movie", "the", "was", "were"]
[0 1 2 3 4 5 6 7 8 9 10]

Integer encoding



Preprocessing



Integer encoding



Preprocessing

Language preprocessing: Highlighting important parts of a sentence while hiding redundancies

- Tokenization: Splitting text into tokens
- Stemming: Removing redundant suffixes
- Lemmatization: Mapping words to common lemmas
- Stopword removal: Removing non-informative words
- Integer encoding: Turning words into numbers
- Assumes we know what is important and what is redundant



Bag of words



UNIVERSITETET
I OSLO

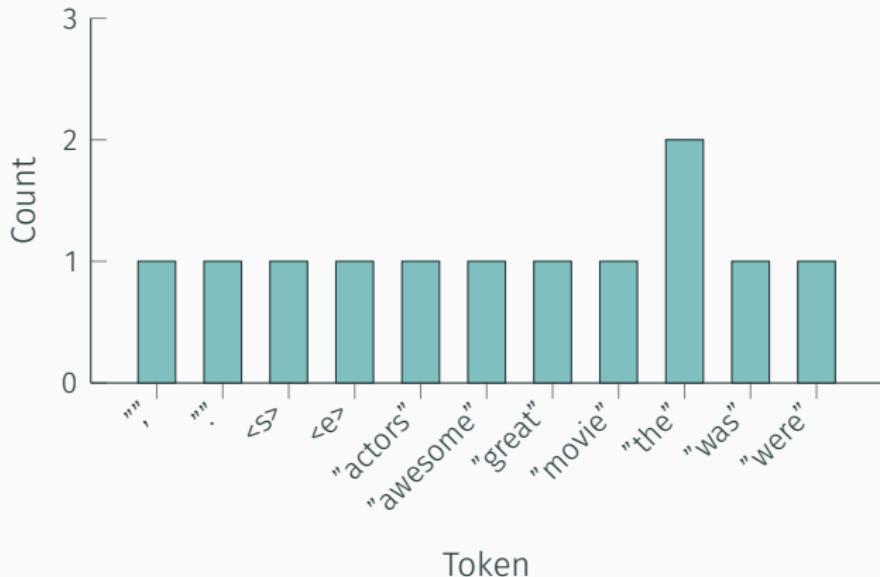
Bag of words

The movie was great, the actors were awesome.



Bag of words

The movie was great, the actors were awesome.



Bag of words

The movie was great, the actors were awesome.

,	.	<s>	<e>	actors	awesome	great	movie	the	was	were
1	1	1	1	1	1	1	1	2	1	1



Bag of words

The movie was great, the actors were awesome.

,	.	<s>	<e>	actors	awesome	awful	great	horrible	movie	the	was	were	sentiment
1	1	1	1	1	1	0	1	0	1	2	1	1	positive
1	1	1	1	1	0	1	0	1	1	2	1	1	negative

The movie was awful, the actors were horrible.



Bag of words

The movie was great, the actors were awesome.

,	.	<s>	<e>	actors	awesome	awful	great	horrible	movie	the	was	were	sentiment
1	1	1	1	1	1	0	1	0	1	2	1	1	positive
1	1	1	1	1	0	1	0	1	1	2	1	1	negative

The movie was awful, the actors were horrible.



Bag of words

,	.	<s>	<e>	actors	awesome	awful	great	horrible	movie	the	was	were	sentiment
1	1	1	1	1	1	0	1	0	1	2	1	1	positive
1	1	1	1	1	0	1	0	1	1	2	1	1	negative

$$y = \beta_0 + \sum_i \beta_i X_i$$



Bag of words

<http://localhost:8888/notebooks/notebooks/Bag%20of%20words.ipynb>



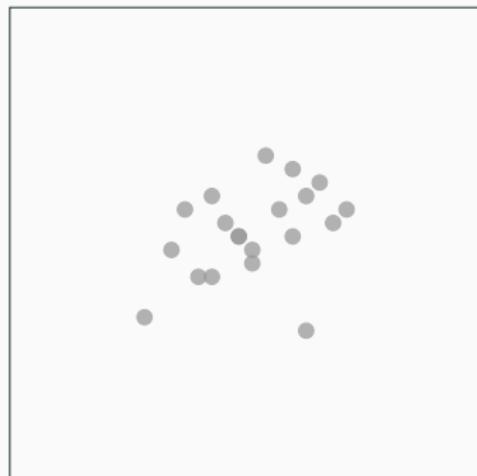
Bag of words

Bag of words: Model language by using word counts (or frequencies)

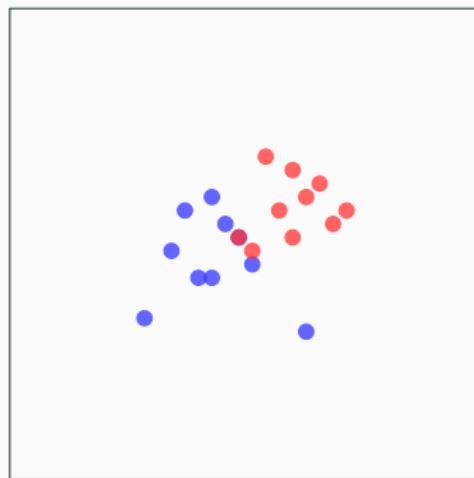
- Main advantage: Simple, useful when a few key words are sufficient to determine the correct prediction
- Main disadvantage: Does not understand word similarities



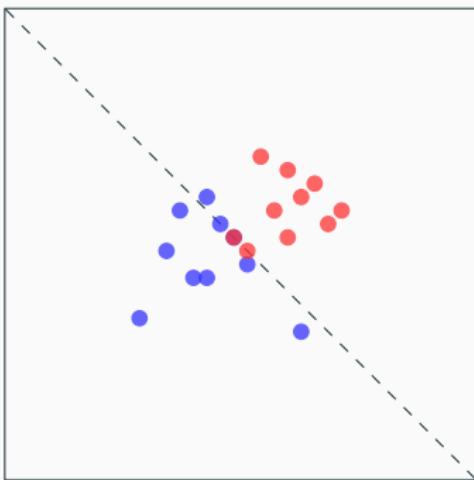
Bag of words: Disadvantages



Bag of words: Disadvantages



Bag of words: Disadvantages



Bag of words: Disadvantages

Dataset: ["This is awesome", "This is wonderful"]



Bag of words: Disadvantages

Dataset: ["This is awesome", "This is wonderful"]

Tokens: [["this" "is" "awesome"], ["this" "is" "wonderful"]]



Bag of words: Disadvantages

Dataset: ["This is awesome", "This is wonderful"]

Tokens: [["this" "is" "awesome"], ["this" "is" "wonderful"]]

Pruned: [["awesome"], ["wonderful"]]



Bag of words: Disadvantages

Dataset: ["This is awesome", "This is wonderful"]

Tokens: [["this" "is" "awesome"], ["this" "is" "wonderful"]]

Pruned: [["awesome"], ["wonderful"]]

Dictionary: ["awesome", "wonderful"]



Bag of words: Disadvantages

Dataset: ["This is awesome", "This is wonderful"]

Tokens: [["this" "is" "awesome"], ["this" "is" "wonderful"]]

Pruned: [["awesome"], ["wonderful"]]

Dictionary: ["awesome", "wonderful"]

Encoded:	awesome	wonderful
	1	0
	0	1



Bag of words: Disadvantages

Dataset: ["This is awesome", "This is wonderful"]

Tokens: [["this", "is", "awesome"], ["this", "is", "wonderful"]]

Pruned: [["awesome"], ["wonderful"]]

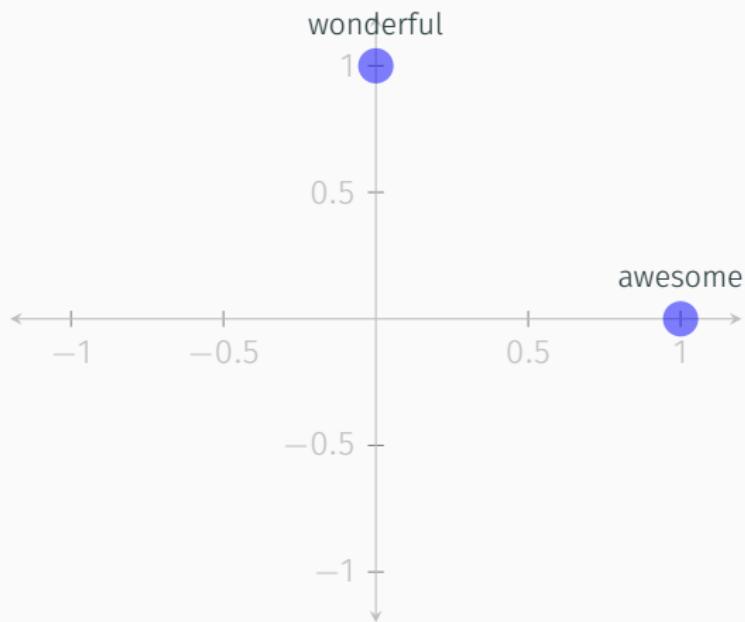
Dictionary: ["awesome", "wonderful"]

	awesome	wonderful
Encoded:	1	0
	0	1

Vectors: [[1, 0], [0, 1]]



Bag of words: Disadvantages



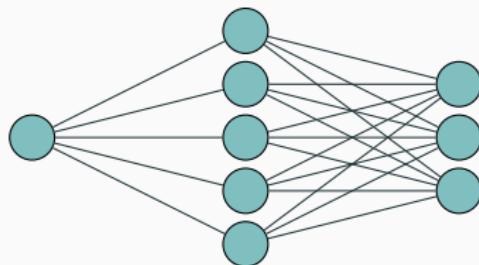
Semantic embedding



UNIVERSITETET
I OSLO

Word2vec

The movie was awesome.
The movie was wonderful.
The movie was fantastic.

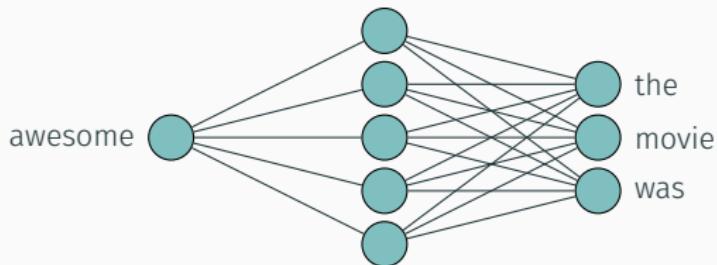


Word2vec

The movie was awesome.

The movie was wonderful.

The movie was fantastic.

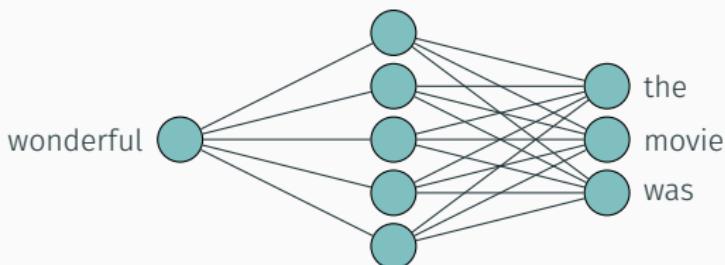


Word2vec

The movie was awesome.

The movie was wonderful.

The movie was fantastic.

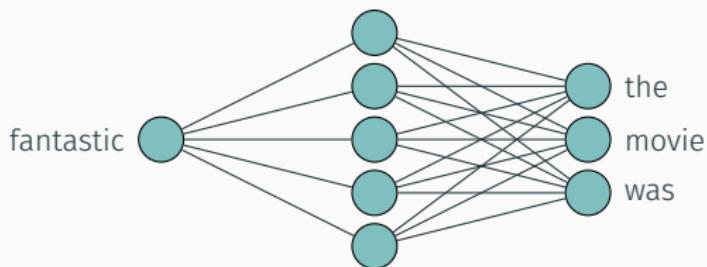


Word2vec

The movie was awesome.

The movie was wonderful.

The movie was fantastic.

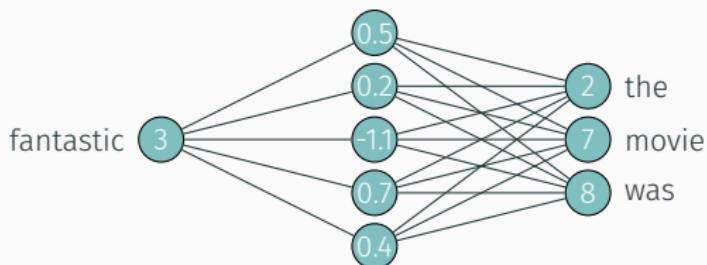


Word2vec

The movie was awesome.

The movie was wonderful.

The movie was fantastic.

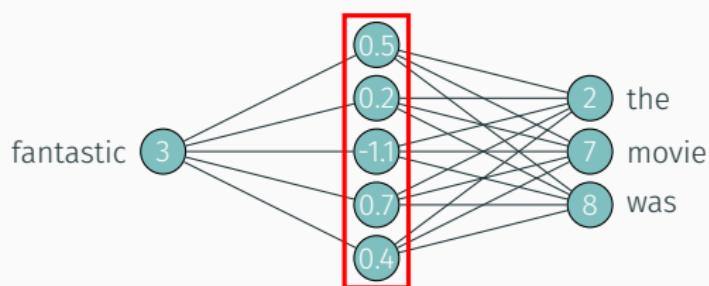


Word2vec

The movie was awesome.

The movie was wonderful.

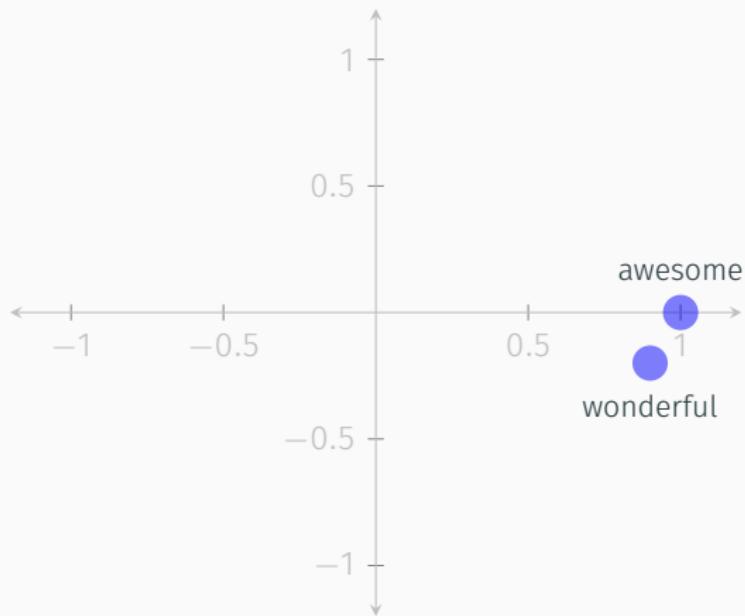
The movie was fantastic.



fantastic=[0.5, 0.2, -1.1, 0.7, 0.4]



Word2vec

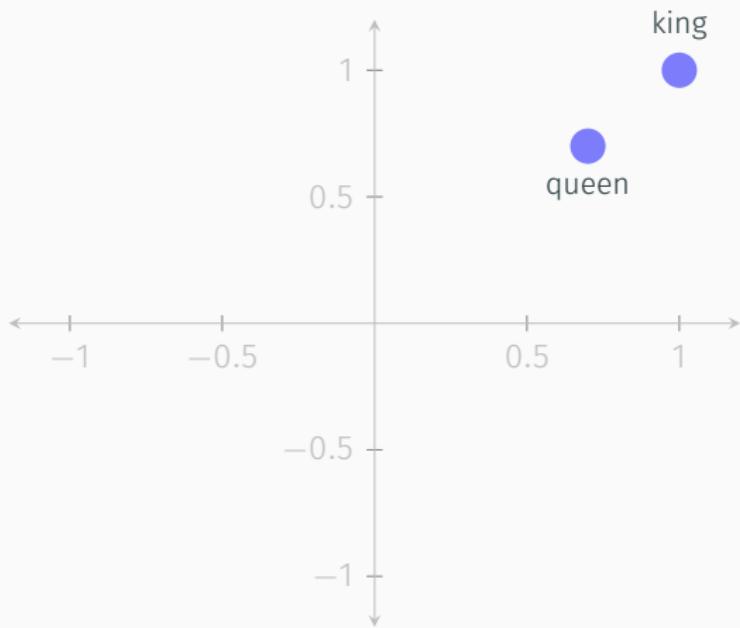


Word2vec

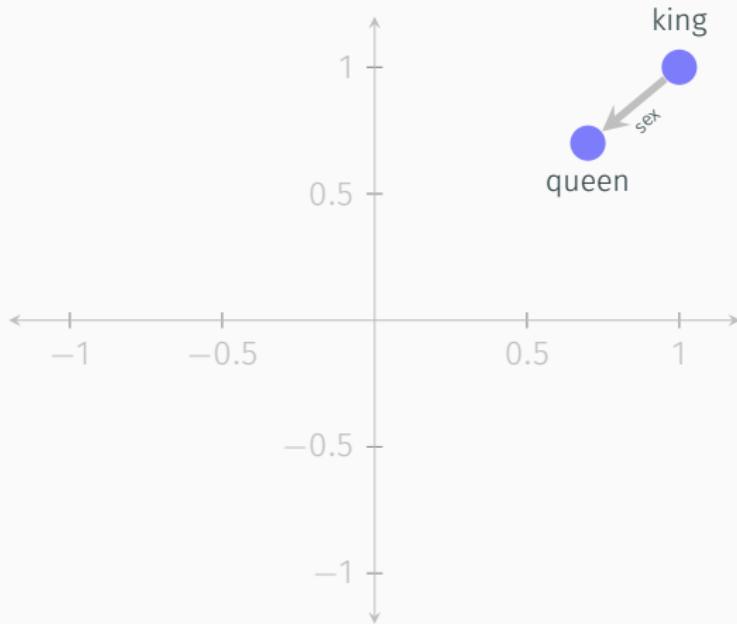
The movie was awesome.
The food was awesome.
The book was awesome.



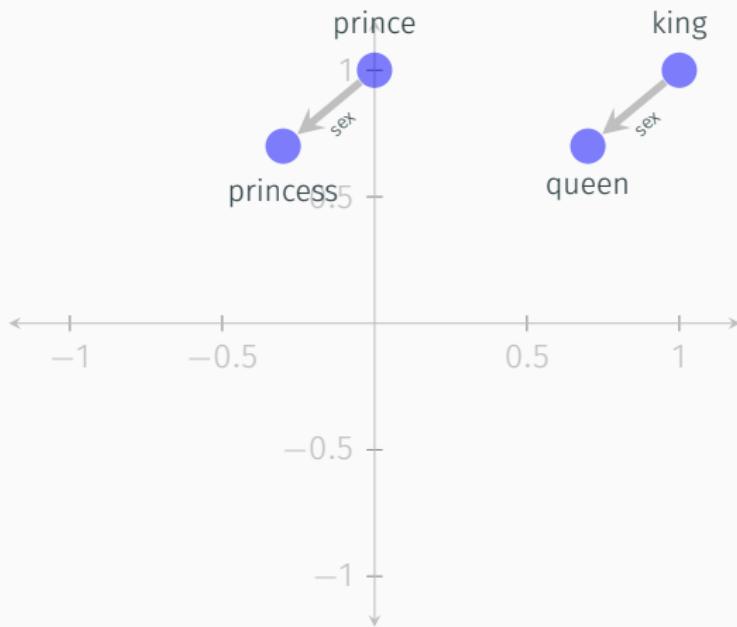
Word2vec



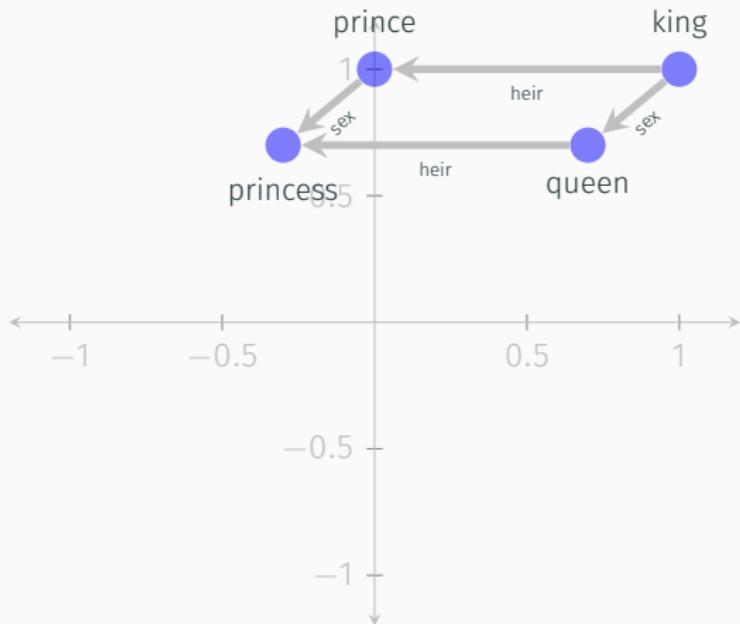
Word2vec



Word2vec



Word2vec

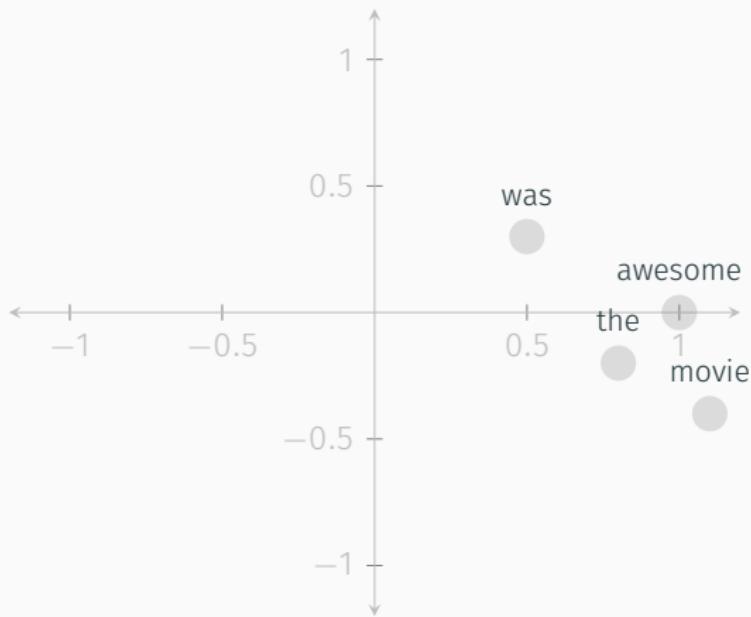


Word2vec

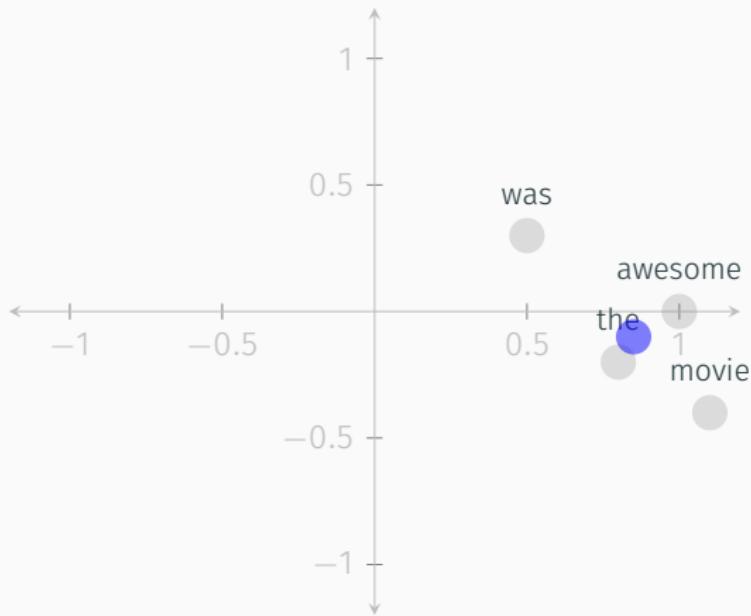
$\text{word2vec}(\text{queen}) = \text{word2vec}(\text{king}) - \text{word2vec}(\text{man}) + \text{word2vec}(\text{woman})$



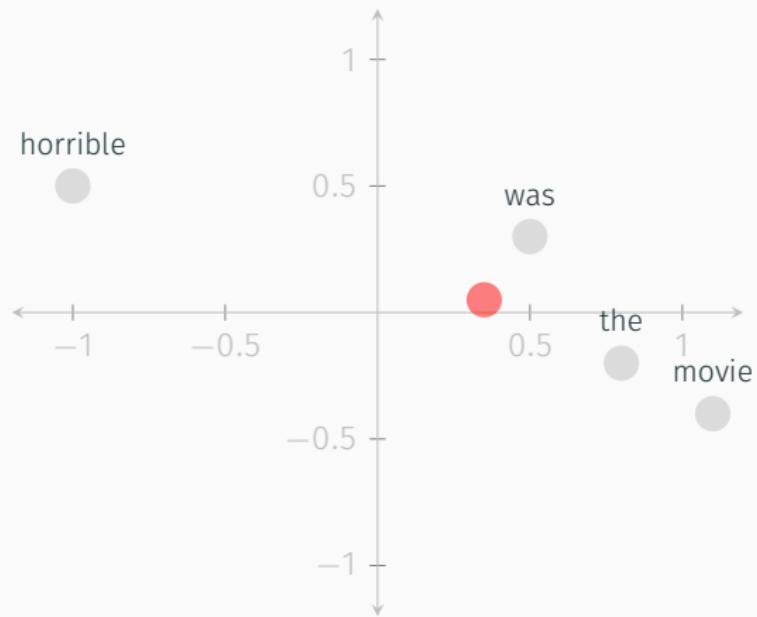
Word2vec



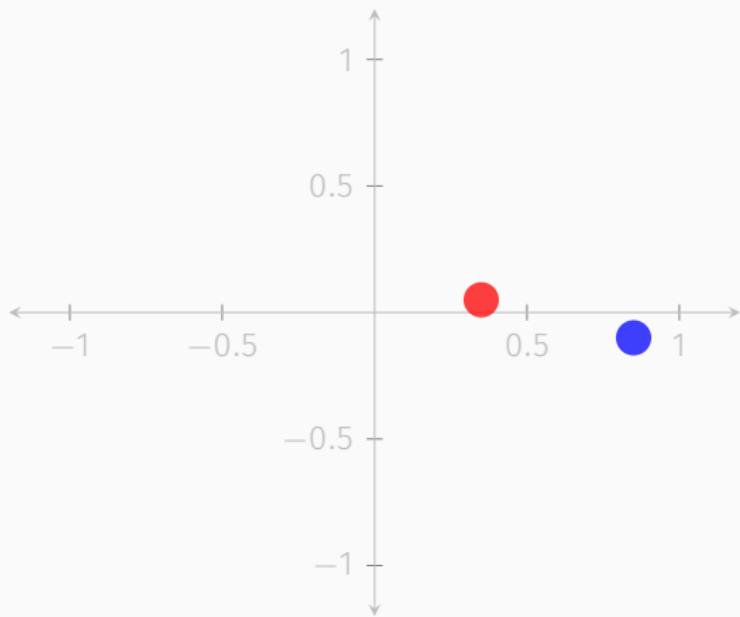
Word2vec



Word2vec



Word2vec



Word2vec

<http://localhost:8888/notebooks/notebooks/Word2vec.ipynb>



Word2vec: Disadvantages

I think the movie was really bad, but my friend said it was good.

=

I think the movie was really good, but my friend said it was bad.



Semantic embedding: Summary

Semantic embedding: Model words by vectors that encode their semantic content

- Main advantage: Models semantic meaning, allowing us to do mathematics with language
- Main disadvantage: Does not consider the structure innate to language



Recurrent neural networks



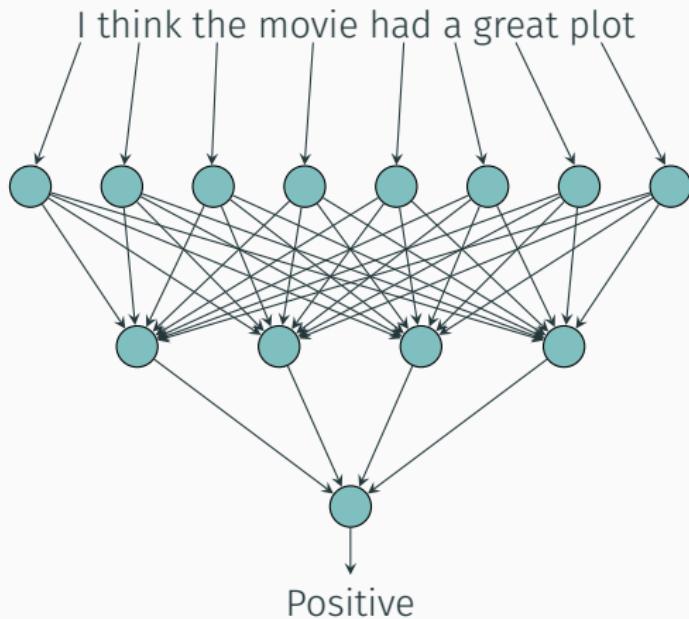
UNIVERSITETET
I OSLO

Recurrent neural networks

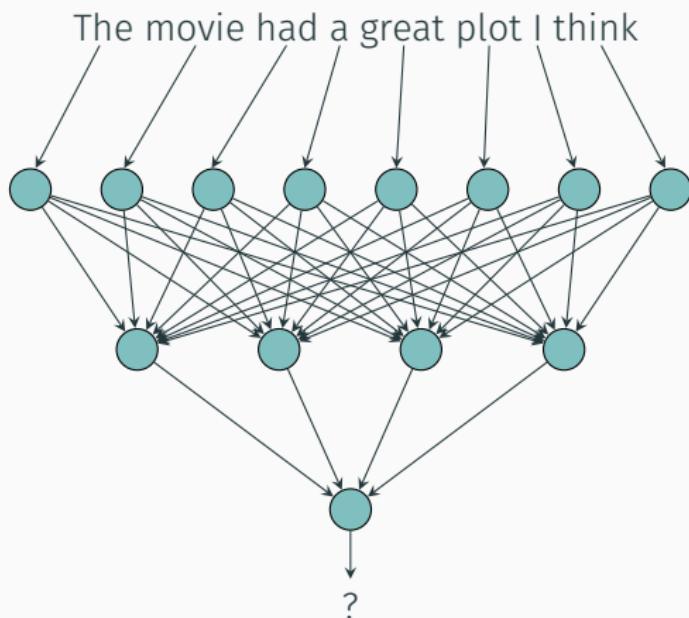
I think the movie had a great plot



Recurrent neural networks



Recurrent neural networks

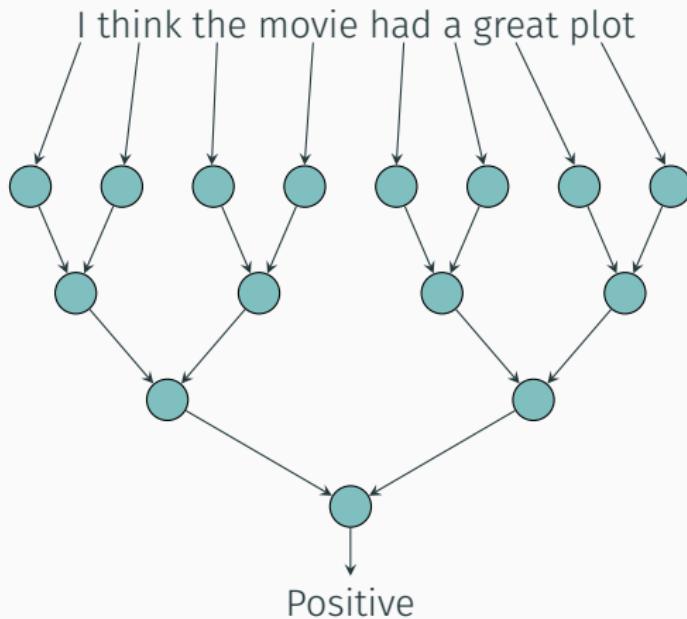


Recurrent neural networks

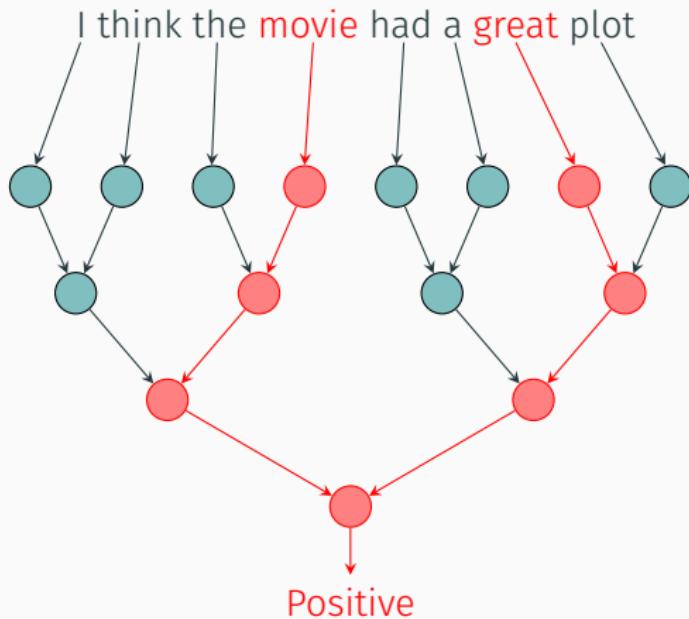
I think the movie had a great plot



Recurrent neural networks



Recurrent neural networks



Recurrent neural networks

I think the movie had a great plot

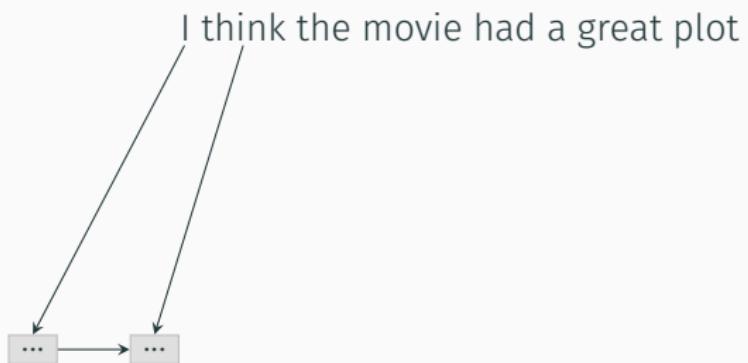


Recurrent neural networks

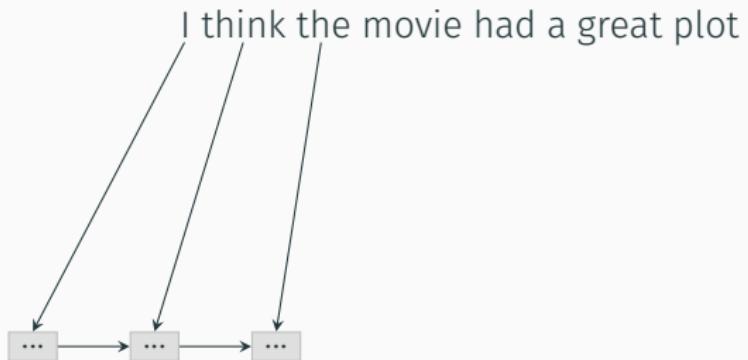
I think the movie had a great plot



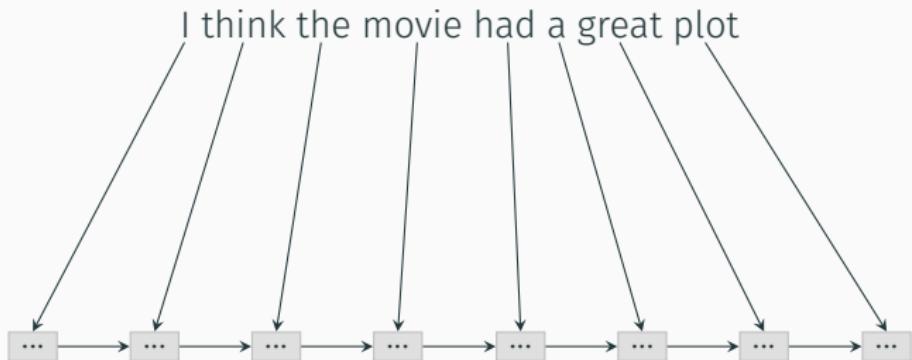
Recurrent neural networks



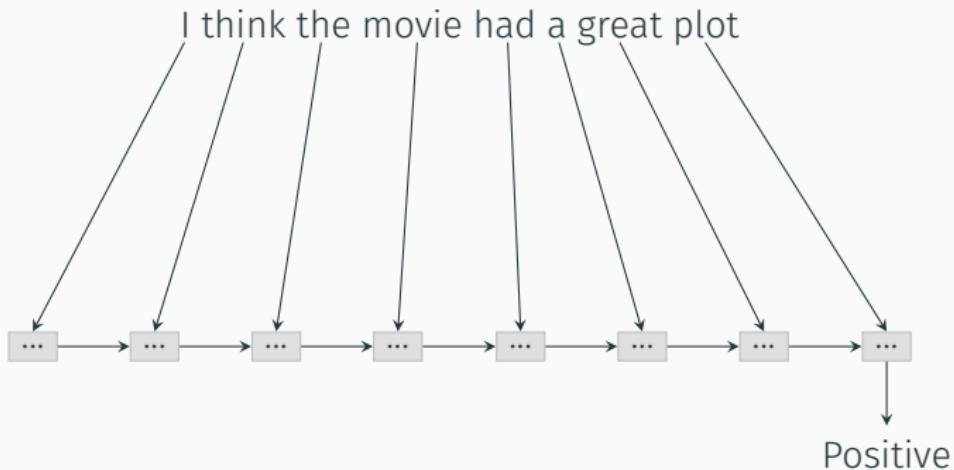
Recurrent neural networks



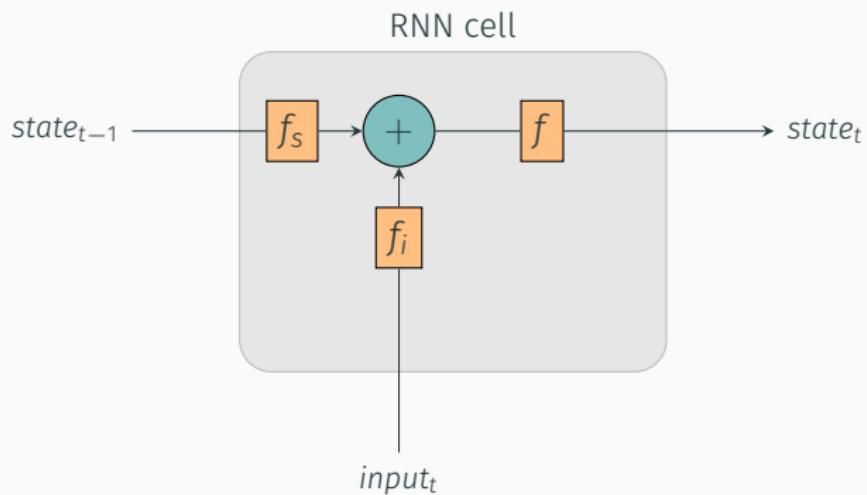
Recurrent neural networks



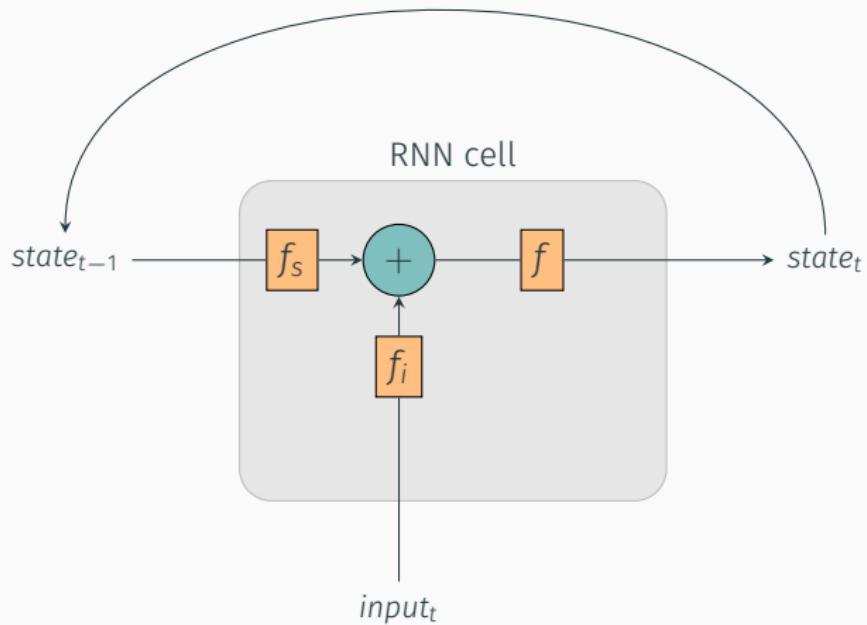
Recurrent neural networks



Recurrent neural networks



Recurrent neural networks

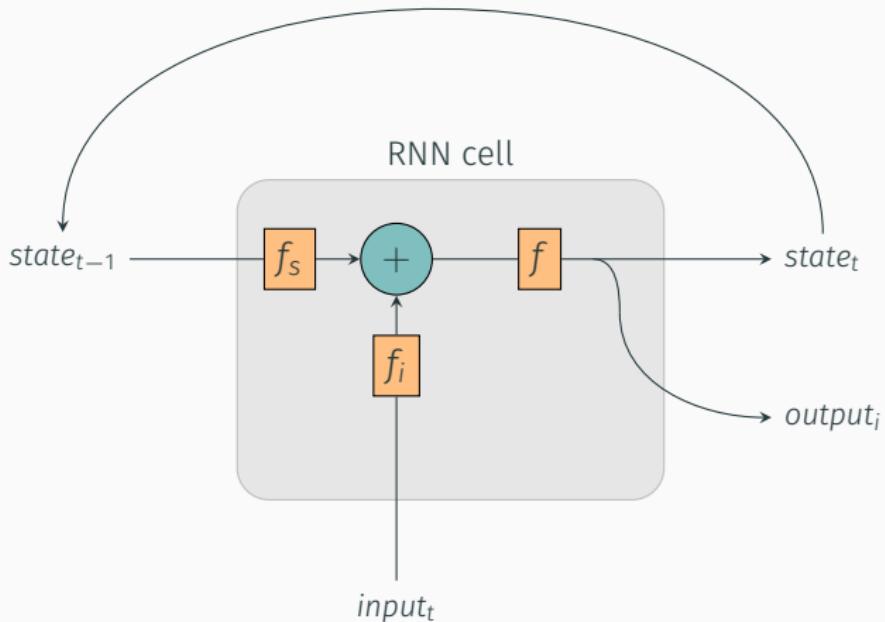


Recurrent neural networks

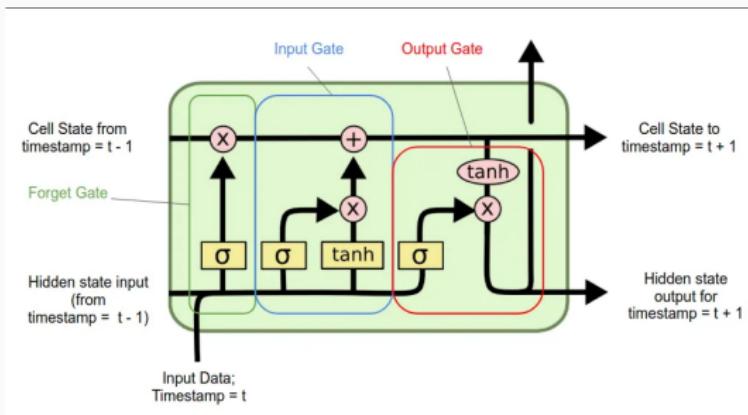
Blackboard demo!



Recurrent neural networks



Recurrent neural networks



Recurrent neural networks

https://www.tensorflow.org/guide/keras/working_with_rnns



Recurrent neural networks

RNNs: Models sequences by recursively considering what it has seen so far, and what the new input token is

- Main advantage: Is able to encompass both long- and short-term dependencies
- Main disadvantage: In practice it is hard to weigh long-term versus short-term

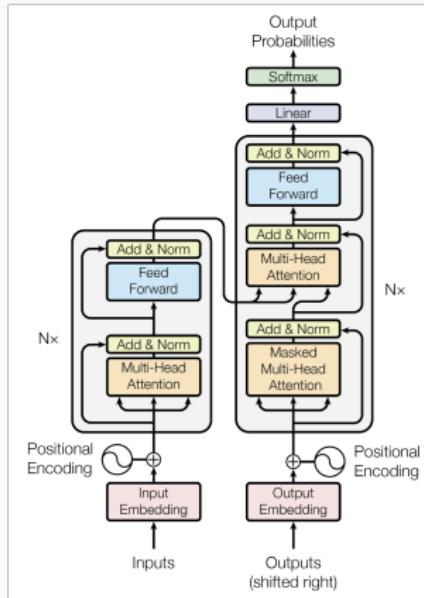


Transformers



UNIVERSITETET
I OSLO

Transformers



Transformers

Auto-regression: The model generates one token at a time, based on the tokens it has generated so far



Transformers: Attention

The movie was great, the actors were _____



Transformers: Attention

The movie was **great**, the actors were _____



Transformers: Attention

The movie was great, the actors were _____



Transformers: Attention

The movie was great, the actors were _____



Transformers: Attention

The movie was great, the actors were _____



Transformers: Attention

The movie was great, the actors were _____



Transformers: Attention

The movie was **great** the actors were _____



Transformers: Attention

The movie was **great**, the actors were _____



Transformers: Attention

The movie was great, the actors were _____



Transformers: Attention

The movie was great, we saw it at the new Cinema in the city center, right down by the restaurant where we went for my birthday that one year, the one where the clown was inside the cake, the actors were _____



Transformers: Attention

The movie was great, we saw it at the new Cinema in the city center, right down by the restaurant where we went for my birthday that one year, the one where the clown was inside the cake, the actors were _____



Transformers: Attention

The movie was great, we saw it at the new Cinema in the city center, right down by the restaurant where we went for my birthday that one year, the one where the clown was inside the cake, the actors were _____



Transformers: Attention

The movie was great, the actors were _____



Transformers: Attention

The movie was great, the actors were _____

[8 7 9 6 0 8 4 10] → ?



Transformers: Attention

The movie was great, the actors were _____

[8 7 9 6 0 8 4 10] → ?

[0 0 0 1 0 0 0 0]



Transformers: Attention

The movie was great, the actors were _____

$$\begin{array}{ccccccccc} [8 & 7 & 9 & 6 & 0 & 8 & 4 & 10] & \rightarrow & ? \\ & & \times & & & & & & \\ [0 & 0 & 0 & 1 & 0 & 0 & 0 & 0] & & = & & & & & & \\ [0 & 0 & 0 & 6 & 0 & 0 & 0 & 0] & & & & & & & & \end{array}$$



Transformers: Demo

https://huggingface.co/docs/transformers/model_doc/llama2



Transformers: Demo

<http://localhost:8888/notebooks/notebooks/GPT%20Embedding.ipynb>



Transformers

Transformers: Revolutionized language modelling by combining feed forward neural networks with, among other things, multihead attention (and infinite data and compute)

- Main advantage: Outperforms everything else for almost all language modelling tasks
- Main disadvantage: Can either be used locally, which is fidgety and requires a good computer, or via an API, which is costly and gives others access to your data

