

# PSY9511: Seminar 8

Sequence modelling (with an emphasis on language)

---

Esten H. Leonardsen

May 14, 2025



UNIVERSITETET  
I OSLO

1. Exercise 6
2. Language modelling
  - 2.1 Introduction and motivation
  - 2.2 Preprocessing
  - 2.3 Bag of words
  - 2.4 Vectorization
  - 2.5 Recurrent neural networks
  - 2.6 Transformers

## 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



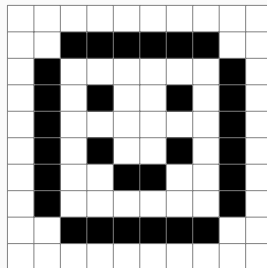
# Introduction

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

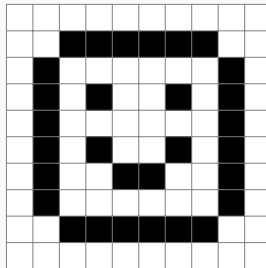
Age	Sex	Education	Salary
25	Male	12	40,000
30	Female	16	65,000
35	Male	14	55,000
40	Female	18	80,000
45	Male	16	75,000



# 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



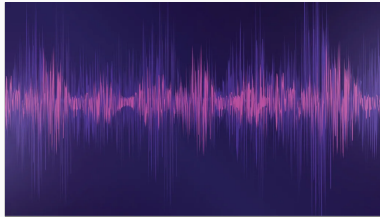


The movie was great, the actors were awesome.



# Introduction

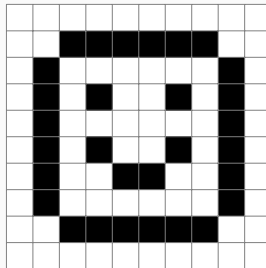
The movie was great, the actors were awesome.



# 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

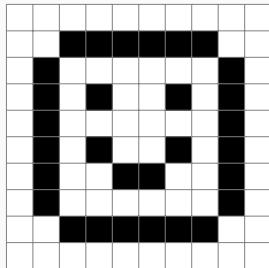
35



# Introduction

Age	Sex
35	Male

# Introduction

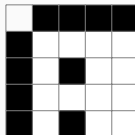


# Introduction

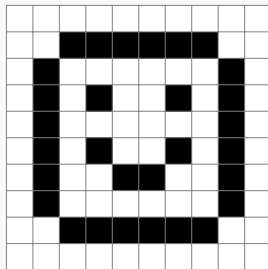




# Introduction



# Introduction



The movie was great, the actors were awesome.

movie

The movie was great, the actors were awesome.

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**.

The movie was great, the actors were awesome.

La película fue genial, los actores fueron increíbles.



The movie was great, the actors were awesome.



La película fue genial, los actores fueron increíbles.

The diagram illustrates word-to-word alignment between the English sentence "The movie was great, the actors were awesome." and the Spanish sentence "La película fue genial, los actores fueron increíbles." Eight arrows point from the English words to the Spanish words: "The" to "La", "movie" to "película", "was" to "fue", "great" to "genial", "the" to "los", "actors" to "actores", "were" to "fueron", and "awesome" to "increíbles".

The movie was great, the actors were \_\_\_\_\_ .

# Introduction

---



The movie was \_\_\_\_\_, the actors were \_\_\_\_\_.

The movie was great, the actors were \_\_\_\_\_.

The movie was great, the actors were awesome.

# Introduction

The movie was great, the actors were awesome.



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



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.

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)
- Generative: Predict the next token in a sequence of words

# Preprocessing

---



UNIVERSITETET  
I OSLO

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" "."]

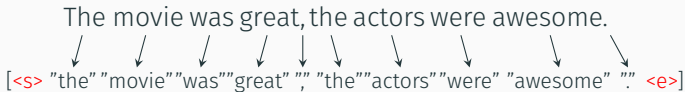
## *Tokenization*



# Preprocessing

The movie was great, the actors were awesome.

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



## *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.

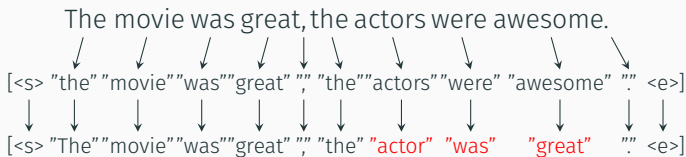
[<s> "the" "movie" "was" "great" "," "the" "actors" "were" "awesome" "." "<e>]

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

## *Lemmatization*



# Preprocessing



```
In[1]: from nltk.stem import WordNetLemmatizer

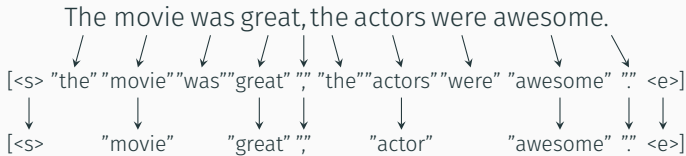
        lemmatizer = WordNetLemmatizer()
        lemmatized = [lemmatizer.lemmatize(token) for token in tokens]
        print(lemmatized)
```

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

## Lemmatization

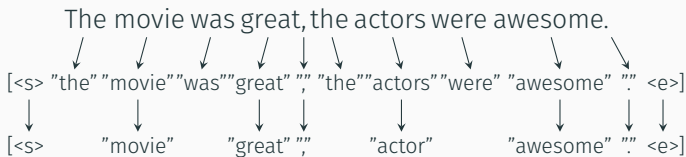


# Preprocessing



*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.

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

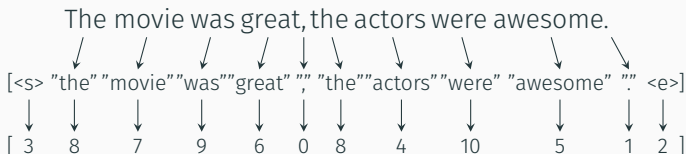
[ 3 8 7 9 6 0 8 4 10 5 1 2 ]

["", ".", <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

---

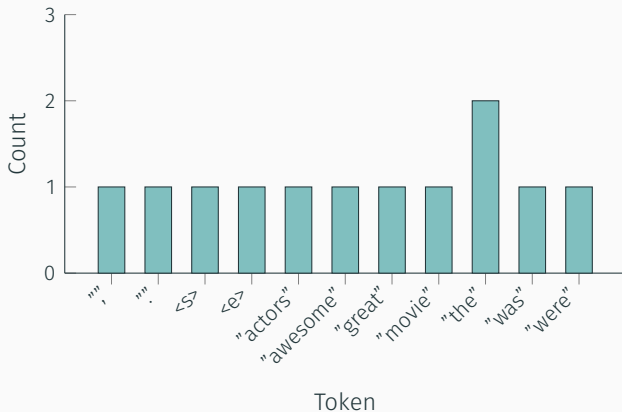


UNIVERSITETET  
I OSLO

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

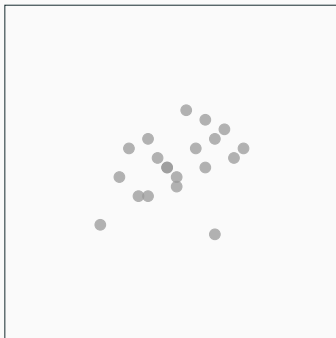
<http://localhost:8888/notebooks/notebooks/Bag%20of%20words.ipynb>



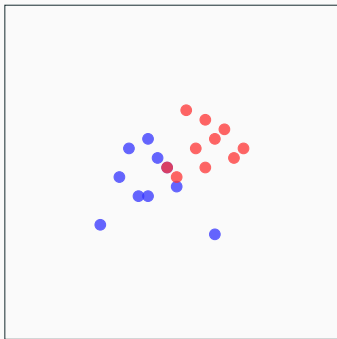
Bag of words: Model language by using word counts (or frequencies)

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

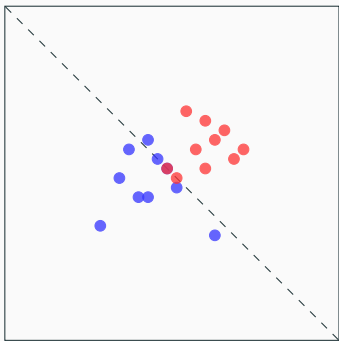
# Bag of words: Disadvantages



# 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	1	0
0	0	1

# Bag of words: Disadvantages

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

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

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

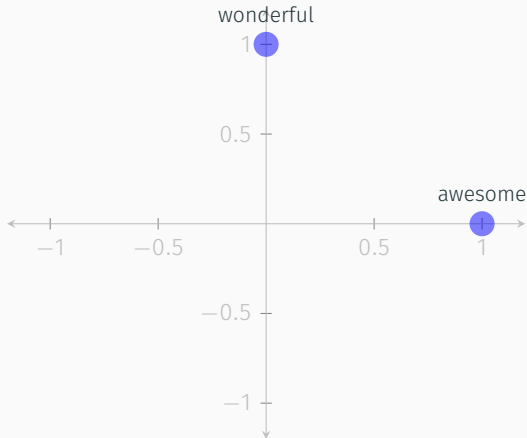
Dictionary: ["awesome", "wonderful"]

Encoded:

	awesome	wonderful
	1	0
	0	1

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

# Bag of words: Disadvantages



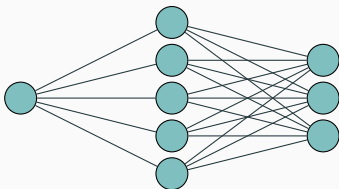
# Semantic embedding

---

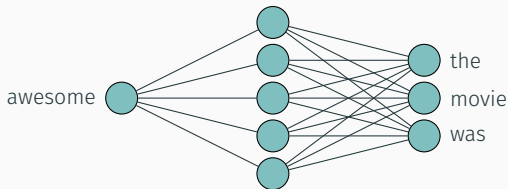


UNIVERSITETET  
I OSLO

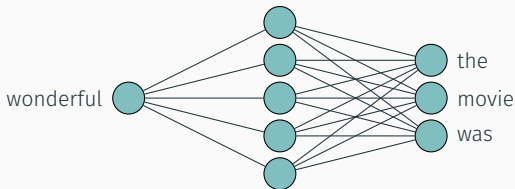
The movie was awesome.  
The movie was wonderful.  
The movie was fantastic.



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

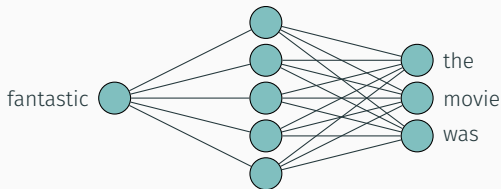


The movie was awesome.  
The movie was wonderful.  
The movie was fantastic.

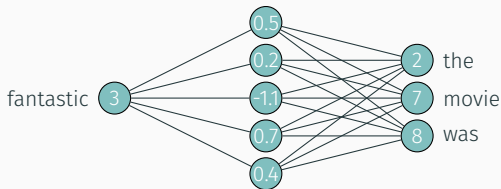




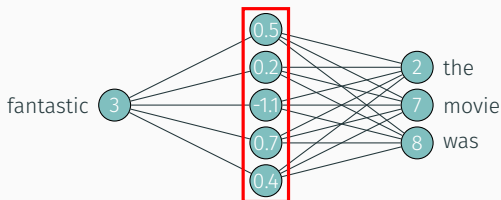
The movie was awesome.  
The movie was wonderful.  
The movie was fantastic.



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

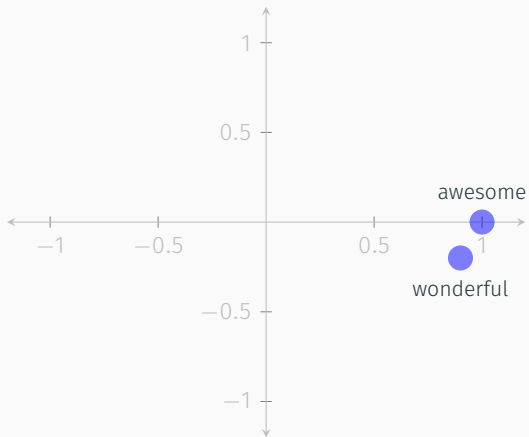


The movie was awesome.  
The movie was wonderful.  
The movie was fantastic.

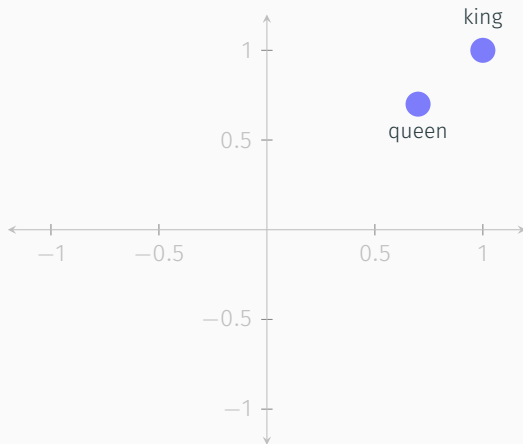


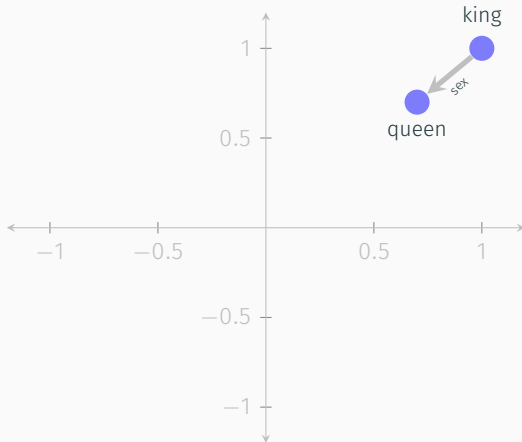
fantastic=[0.5, 0.2, -1.1, 0.7, 0.4]

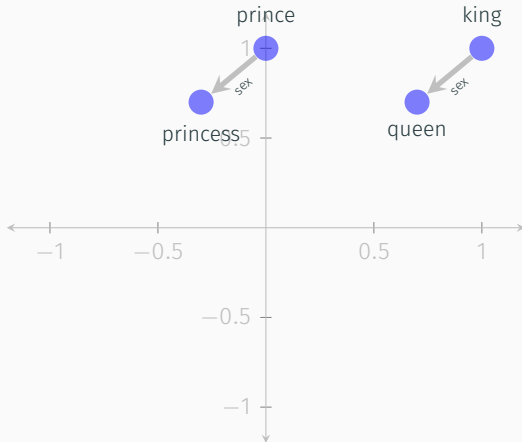
# Word2vec



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

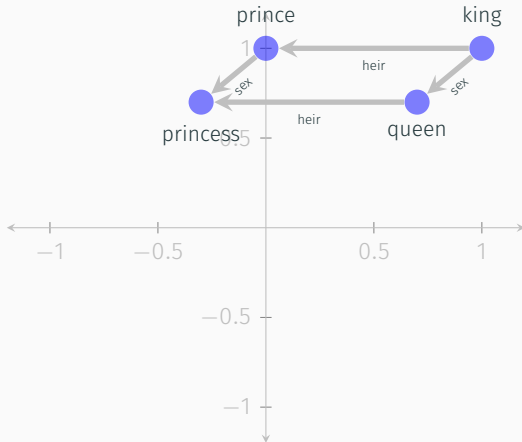






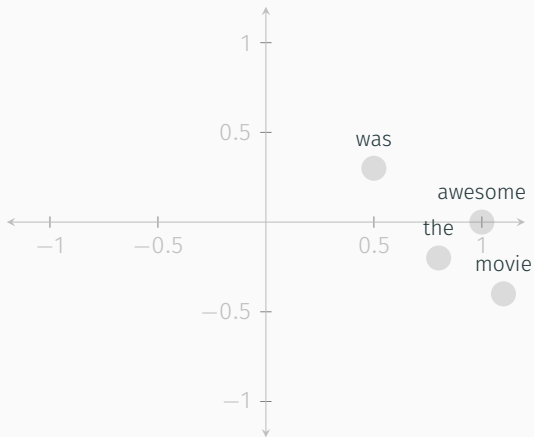


# Word2vec

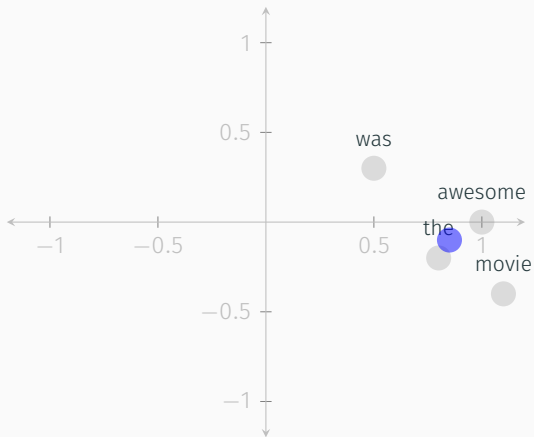


$$\text{word2vec}(\text{queen}) = \text{word2vec}(\text{king}) \\ - \text{word2vec}(\text{man}) \\ + \text{word2vec}(\text{woman})$$

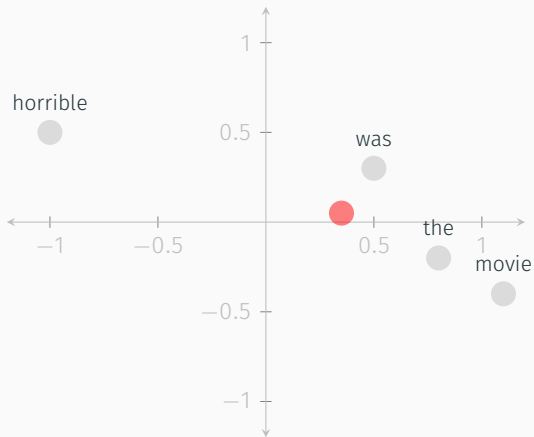
# Word2vec



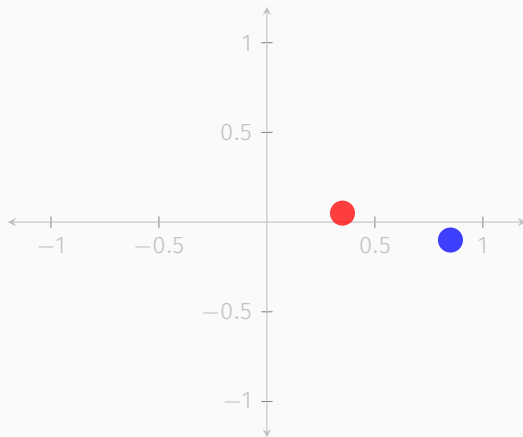
# Word2vec



# 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.



Word2vec: 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

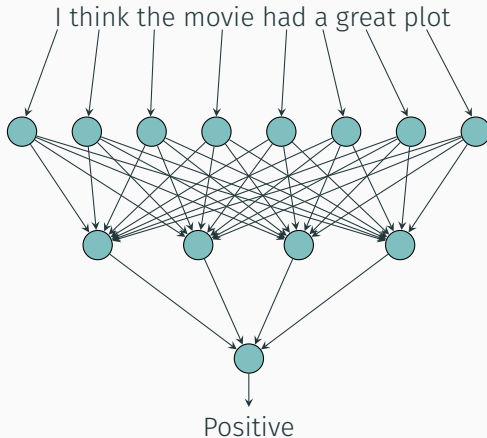
---



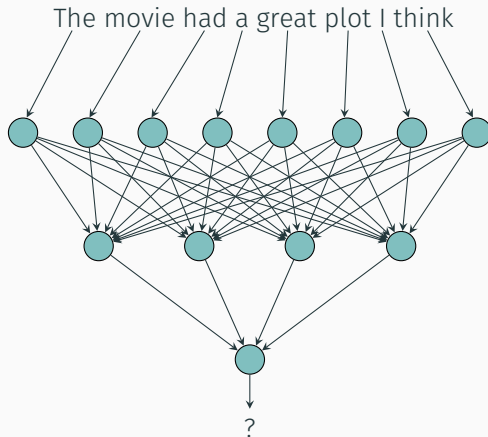
UNIVERSITETET  
I OSLO

I think the movie had a great plot

# Recurrent neural networks

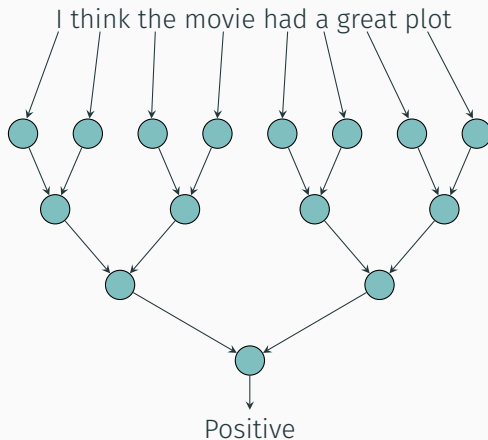


# Recurrent neural networks

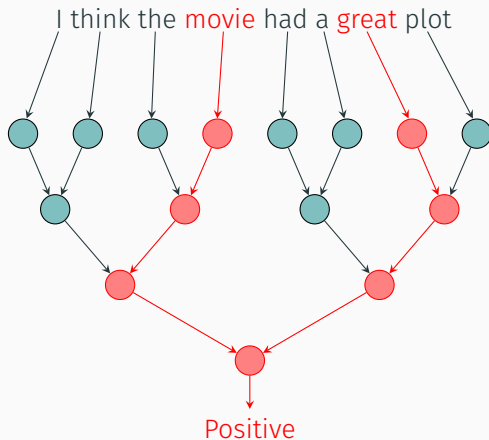


I think the movie had a great plot

# Recurrent neural networks



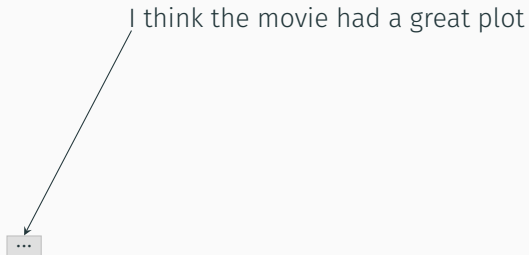
# Recurrent neural networks



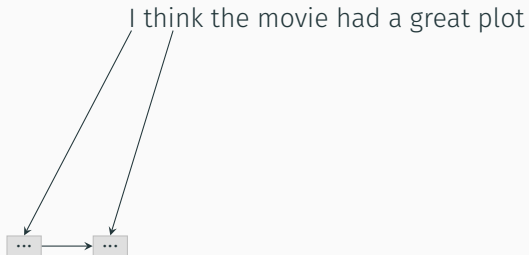


I think the movie had a great plot

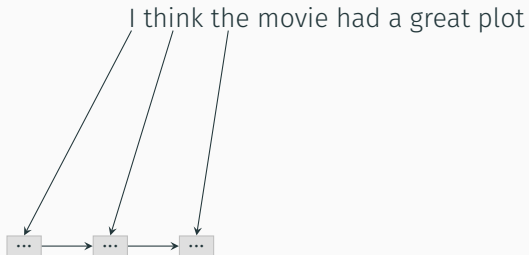
# Recurrent neural networks



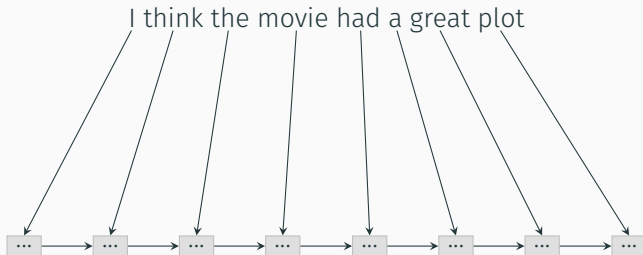
# Recurrent neural networks



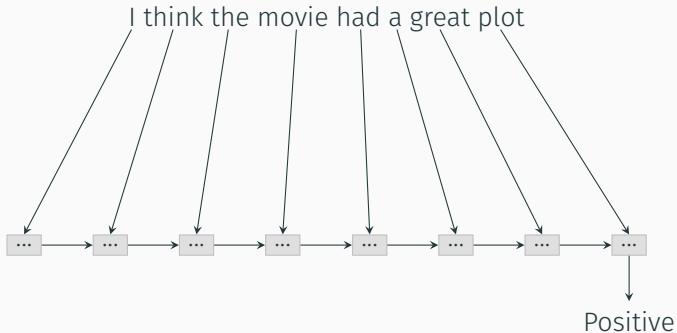
# Recurrent neural networks



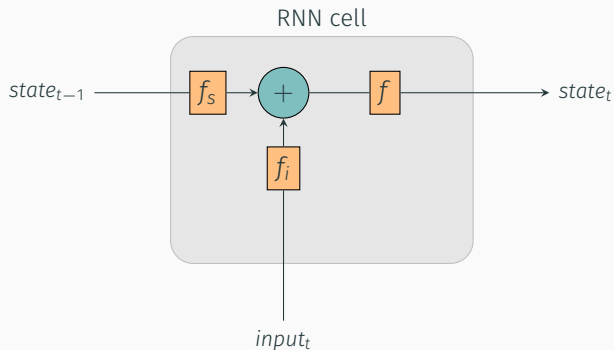
# Recurrent neural networks



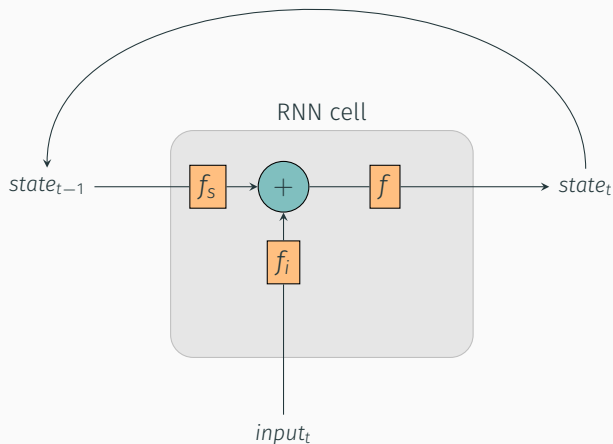
# Recurrent neural networks



# Recurrent neural networks



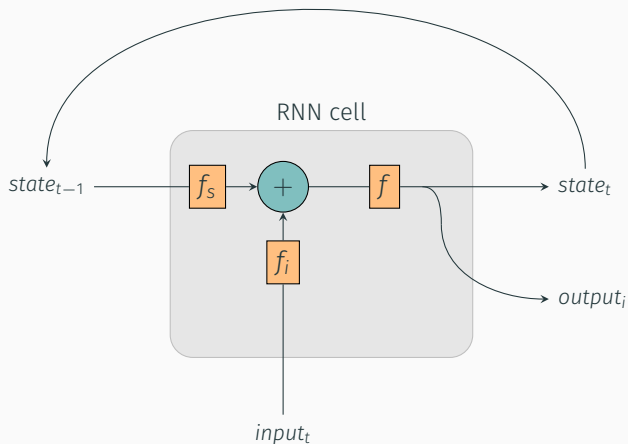
# Recurrent neural networks



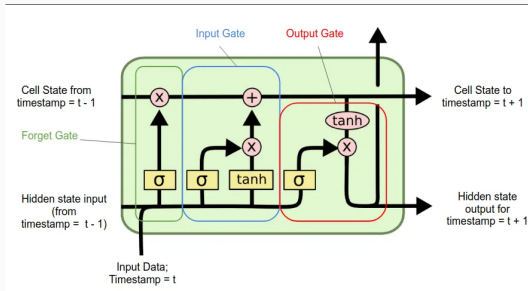


Blackboard demo!

# Recurrent neural networks



# Recurrent neural networks



# Recurrent neural networks

[https://www.tensorflow.org/guide/keras/working\\_with\\_rnn](https://www.tensorflow.org/guide/keras/working_with_rnn)



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

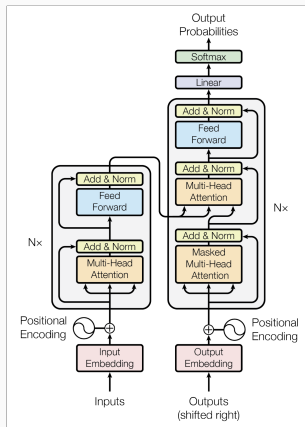
# Transformers

---



UNIVERSITETET  
I OSLO

# Transformers



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



The movie was great, the actors were\_\_\_\_\_

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 \_\_\_\_\_

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 \_\_\_\_\_

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 \_\_\_\_\_

[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]



[https://huggingface.co/docs/transformers/model\\_doc/llama2](https://huggingface.co/docs/transformers/model_doc/llama2)

# Transformers: Demo

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



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