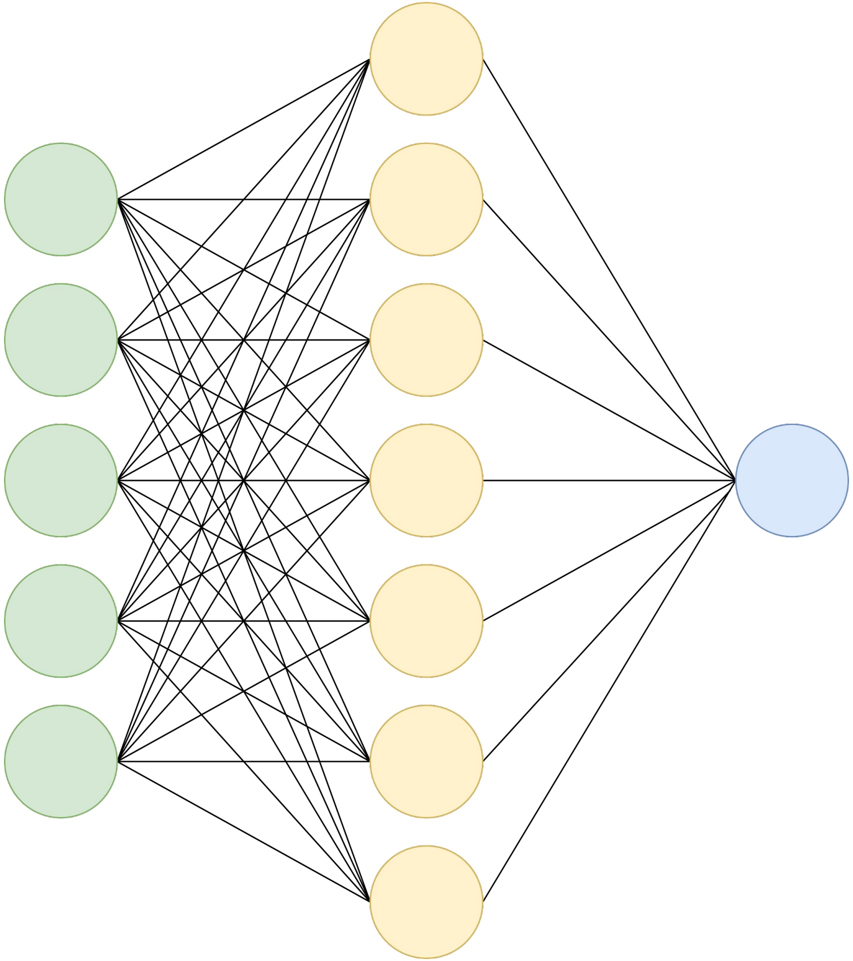
Deep Learning in NLP

From RNNs to Transformers and Beyond

Thomas Vakili - thomas.vakili@dsv.su.se

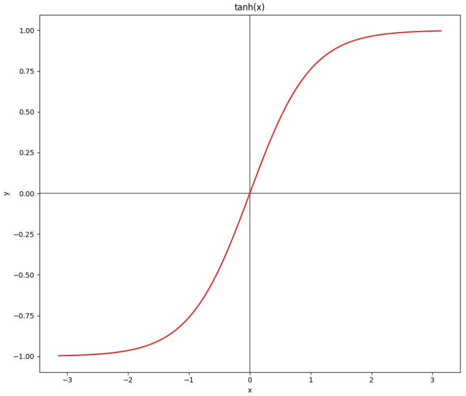


Neural Networks

Neural Networks

- Based mainly on two basic mathematical components:
    - Multiplying vectors and matrices
    - Non-linear activation functions

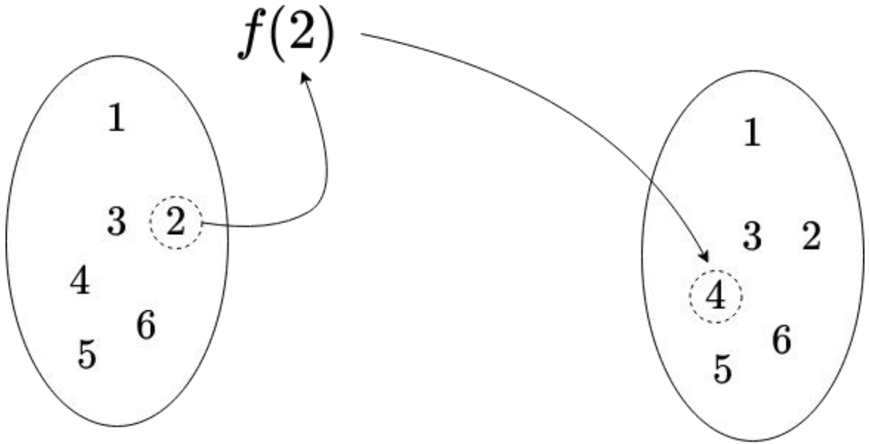
$$\begin{array}{cc} \vec{c}_1 & \vec{c}_2 \\ \multicolumn{2}{c}{\begin{bmatrix} w & x \\ y & z \end{bmatrix}} \end{array} \begin{array}{c} \vec{r}_1 \\ \begin{bmatrix} a & b \\ c & d \end{bmatrix} \\ \vec{r}_2 \end{array} = \begin{bmatrix} \vec{c}_1\vec{r}_1 & \vec{c}_2\vec{r}_1 \\ \vec{c}_1\vec{r}_2 & \vec{c}_2\vec{r}_2 \end{bmatrix}$$



Thomas Vakili - thomas.vakili@dsv.su.se

Universal Approximation Theorem

- A neural network with one hidden layer can approximate *any function**

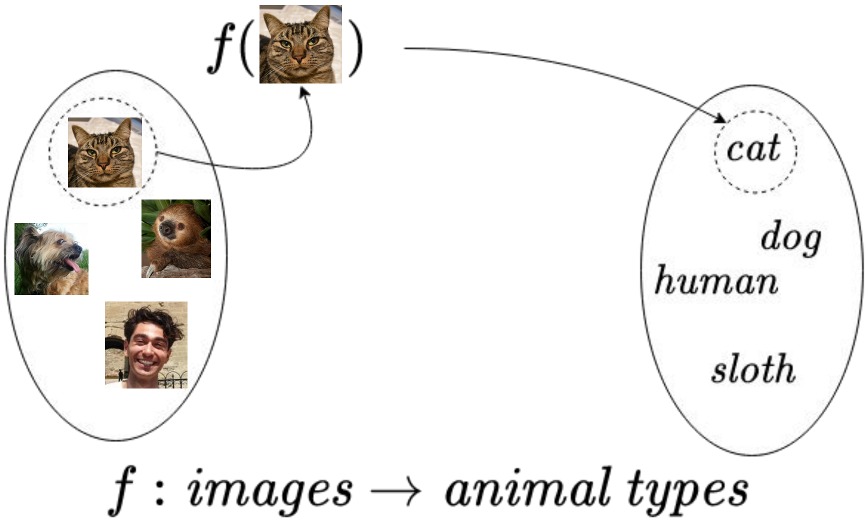


$$f : \mathbb{Z} \to \mathbb{Z}$$

* Given enough training data and parameters…

Universal Approximation Theorem

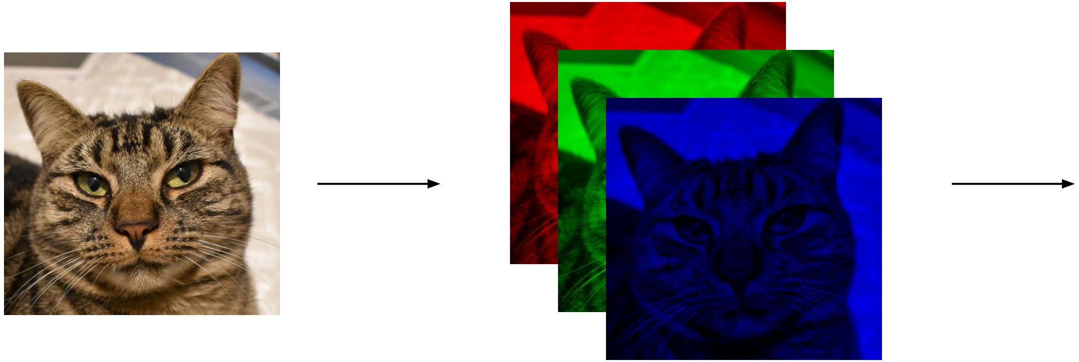
- A neural network with one hidden layer can approximate *any function**



* Given enough training data and parameters…

Vectorization

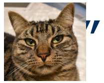
- Unfortunately, a computer can't process “  ”...
  We need a vector!

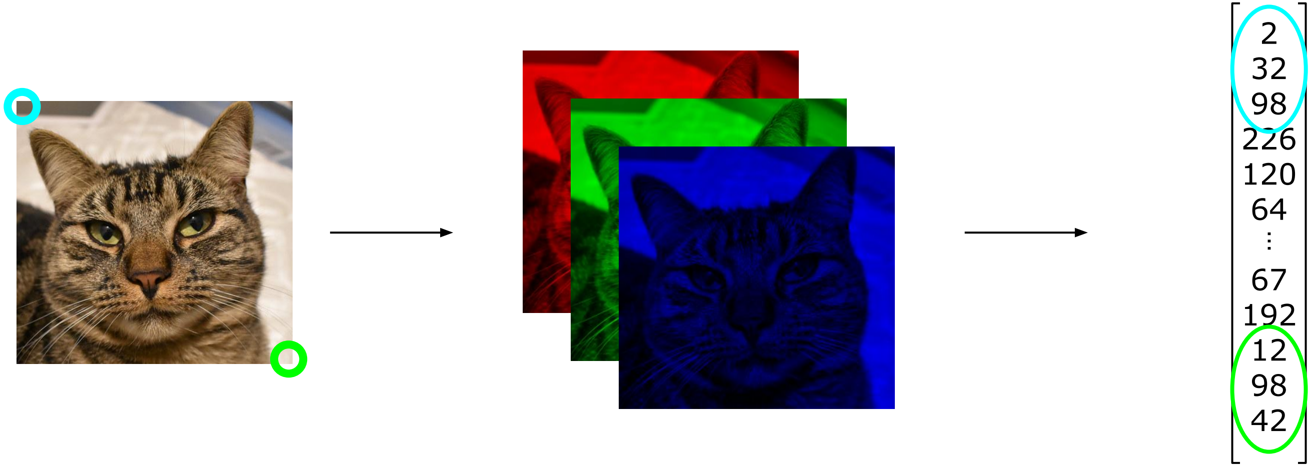


$$\begin{bmatrix} 2 \\ 32 \\ 98 \\ 226 \\ 120 \\ 64 \\ \vdots \\ 67 \\ 192 \\ 12 \\ 98 \\ 42 \end{bmatrix}$$

Thomas Vakili - thomas.vakili@dsv.su.se

Vectorization

- Unfortunately, a computer can't process "  "...
  We need a vector!



Thomas Vakili - thomas.vakili@dsv.su.se

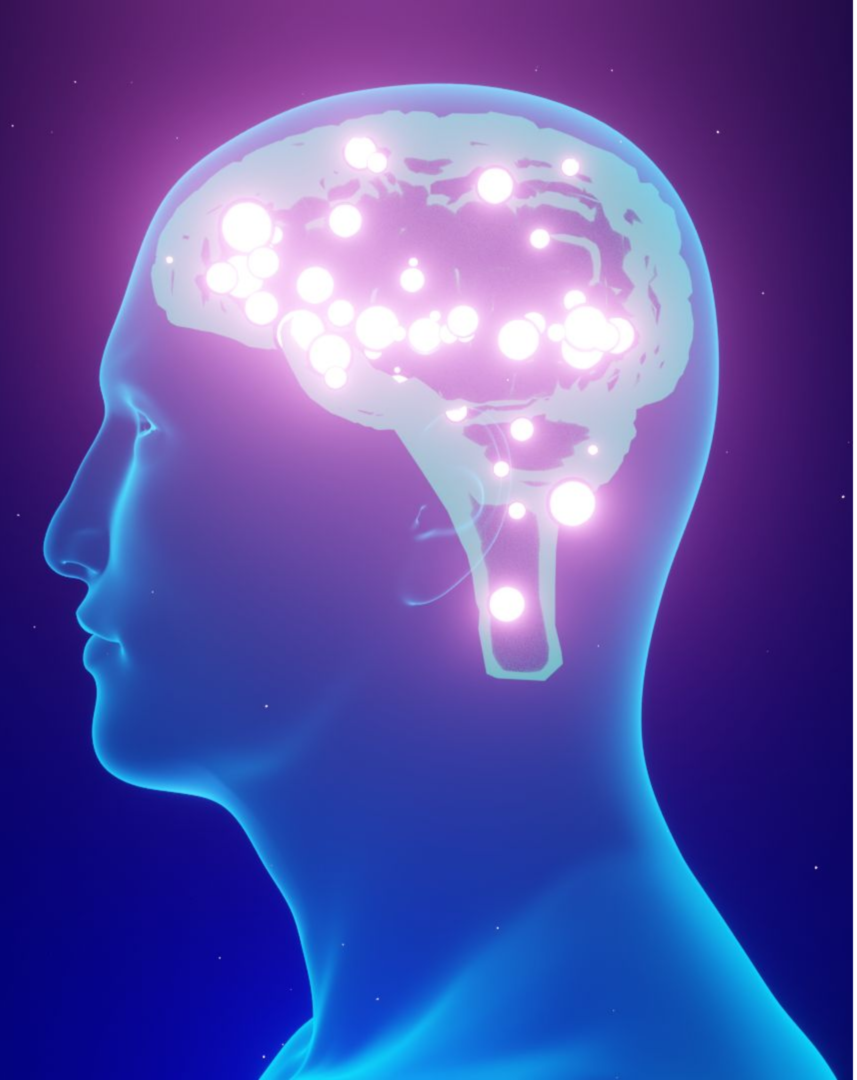
Word vectors lack order

- Words can also be vectorized, as Aron explained.
  But how do we handle *sequences?*

"life is work" ≠ "work is life"

BUT they result in the same set of vectors!

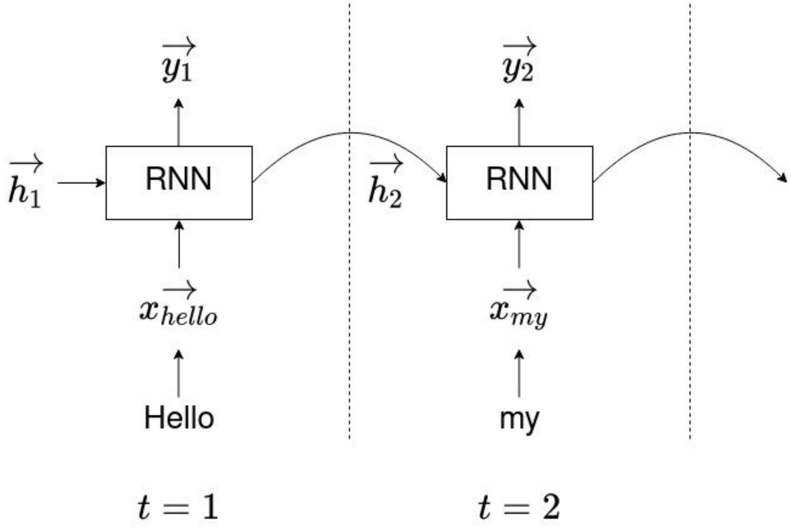
Thomas Vakili - thomas.vakili@dsv.su.se



Sequences with RNNs

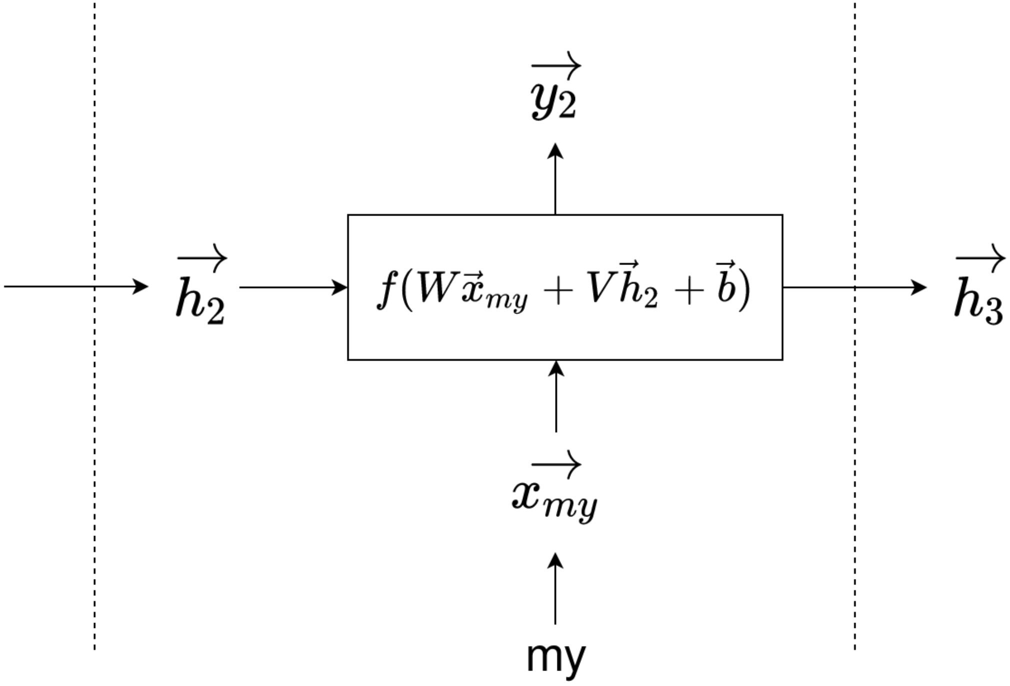
RNNs use a *hidden state*

- There's nothing hidden about the hidden state
- An RNN uses a vector $\vec{h}_t$ to **remember** the earlier states at time-step t+1



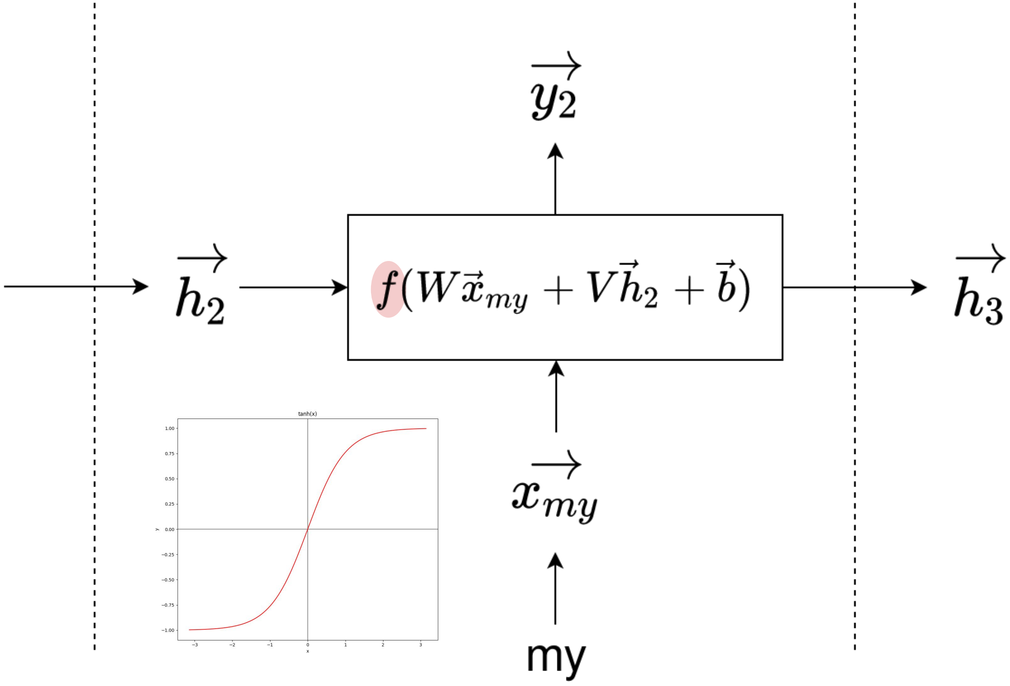
Thomas Vakili - thomas.vakili@dsv.su.se

How RNNs maintain their state



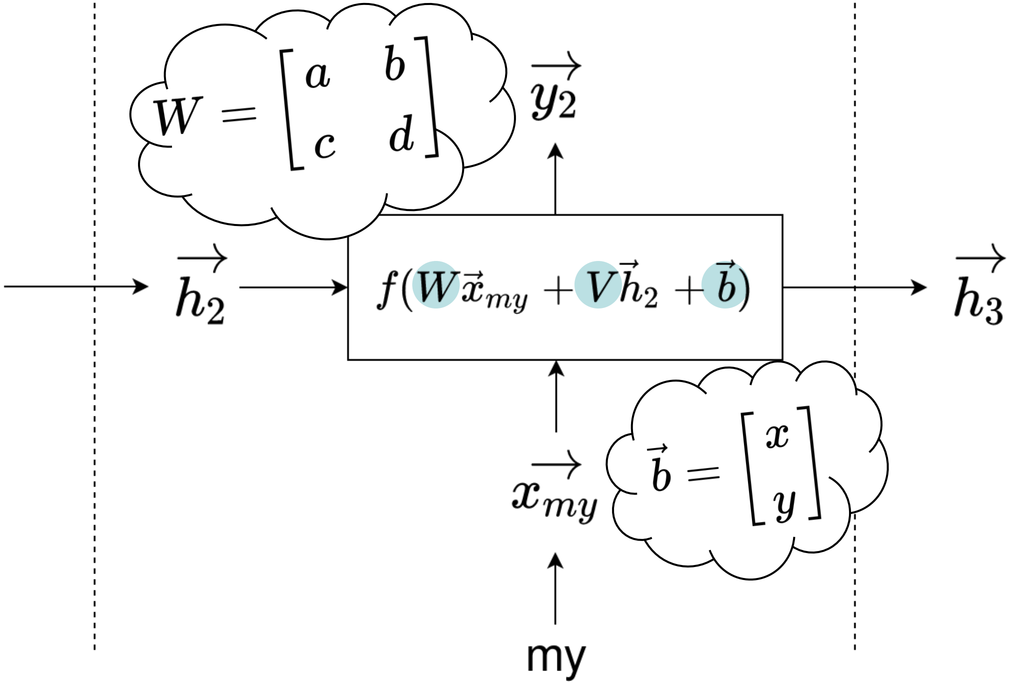
Thomas Vakili - thomas.vakili@dsv.su.se

How RNNs maintain their state



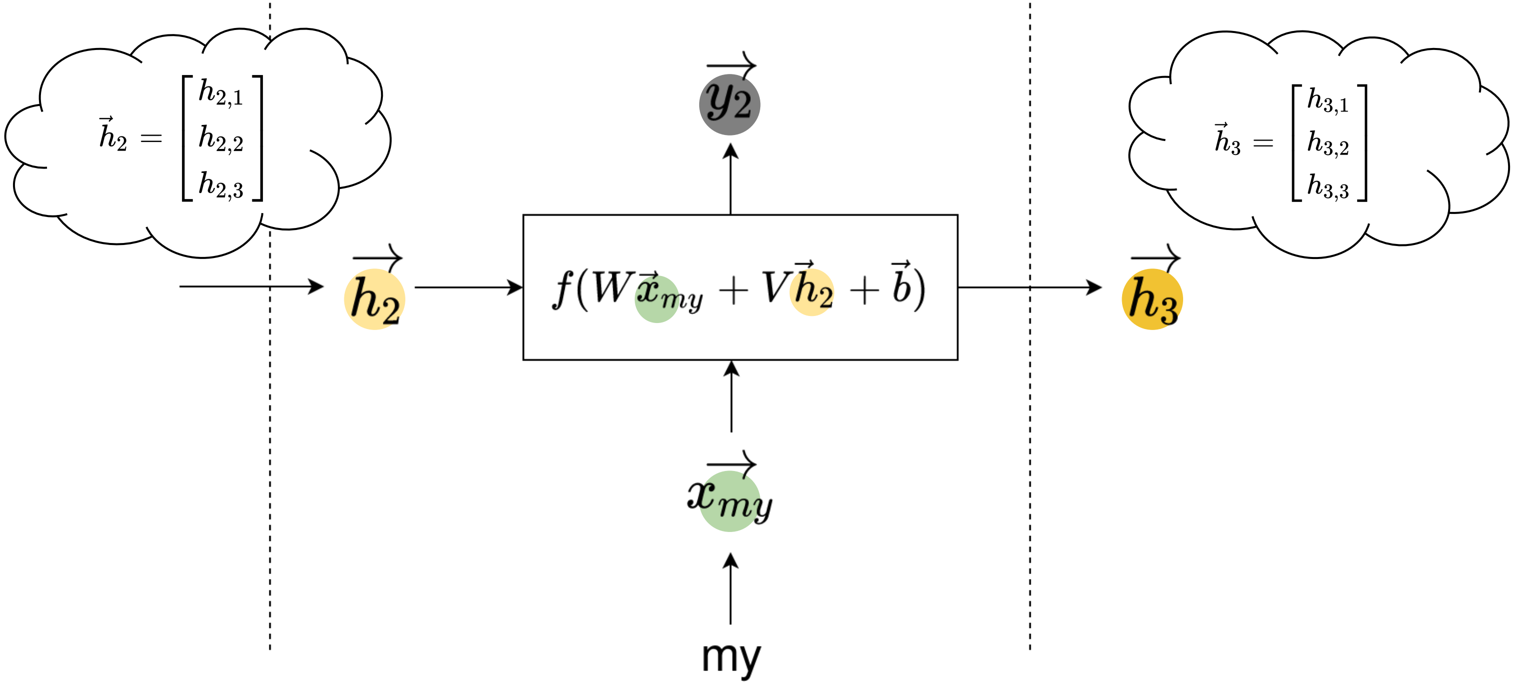
Thomas Vakili - thomas.vakili@dsv.su.se

How RNNs maintain their state



Thomas Vakili - thomas.vakili@dsv.su.se

How RNNs maintain their state



Thomas Vakili - thomas.vakili@dsv.su.se

Vanishing gradients

- RNNs are defined recursively: $\vec{h}_{t+1} = f(W\vec{x}_t + V\vec{h}_t + \vec{b})$
- Early states have a weaker impact on later states
- You can think of this as $\vec{h}_t$ getting crammed and leaking

$$h_4 = f(W\vec{x}_3 + V\underbrace{f(W\vec{x}_2 + V\overbrace{f(W\vec{x}_1 + V\vec{h}_1 + \vec{b})}^{\vec{h}_2} + \vec{b})}_{\vec{h}_3} + \vec{b})$$

$$h_{100} = \ldots$$

Thomas Vakili - thomas.vakili@dsv.su.se

Memory management with LSTMs

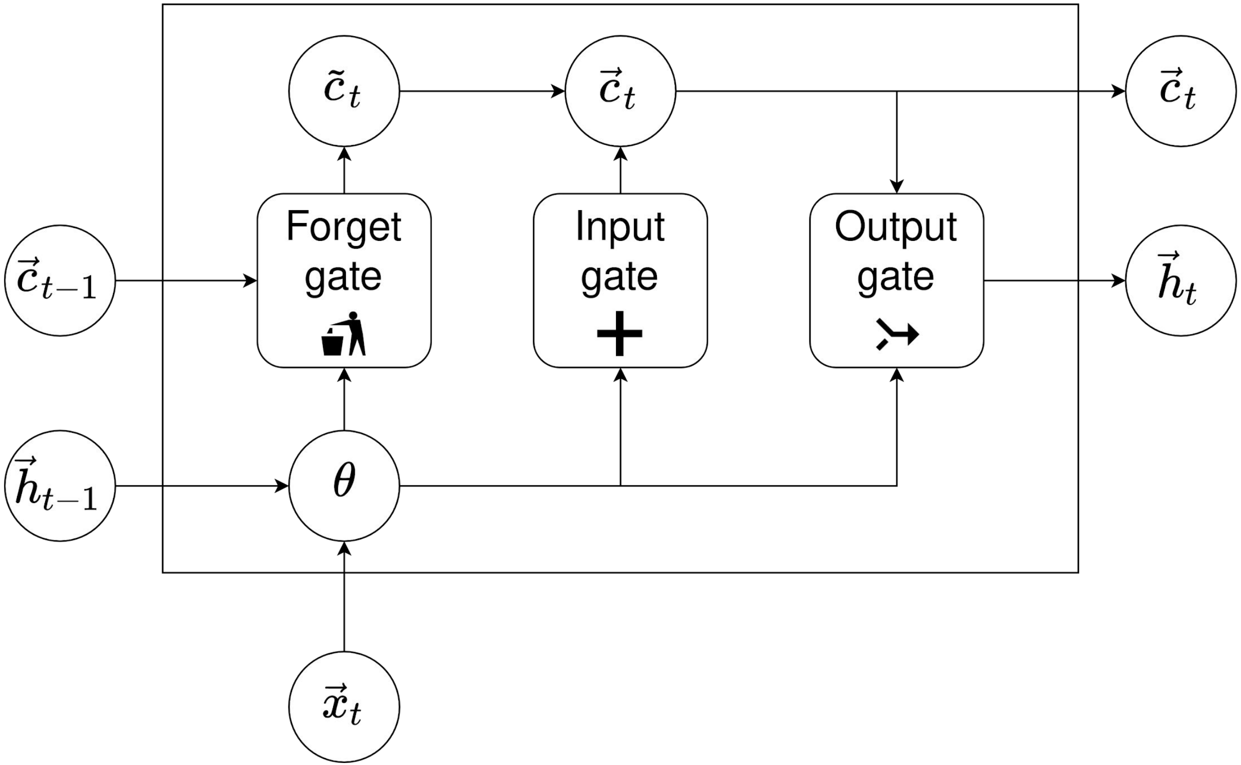
Learn to remember and *to forget*

- LSTM *cells* have three *gates* (i.e., new matrices)
  - Forget gate: what is now irrelevant?
  - Input gate: what new information do we keep?
  - Output gate: produce the next hidden state

- The maintain a *cell state*: $\vec{c}_t$

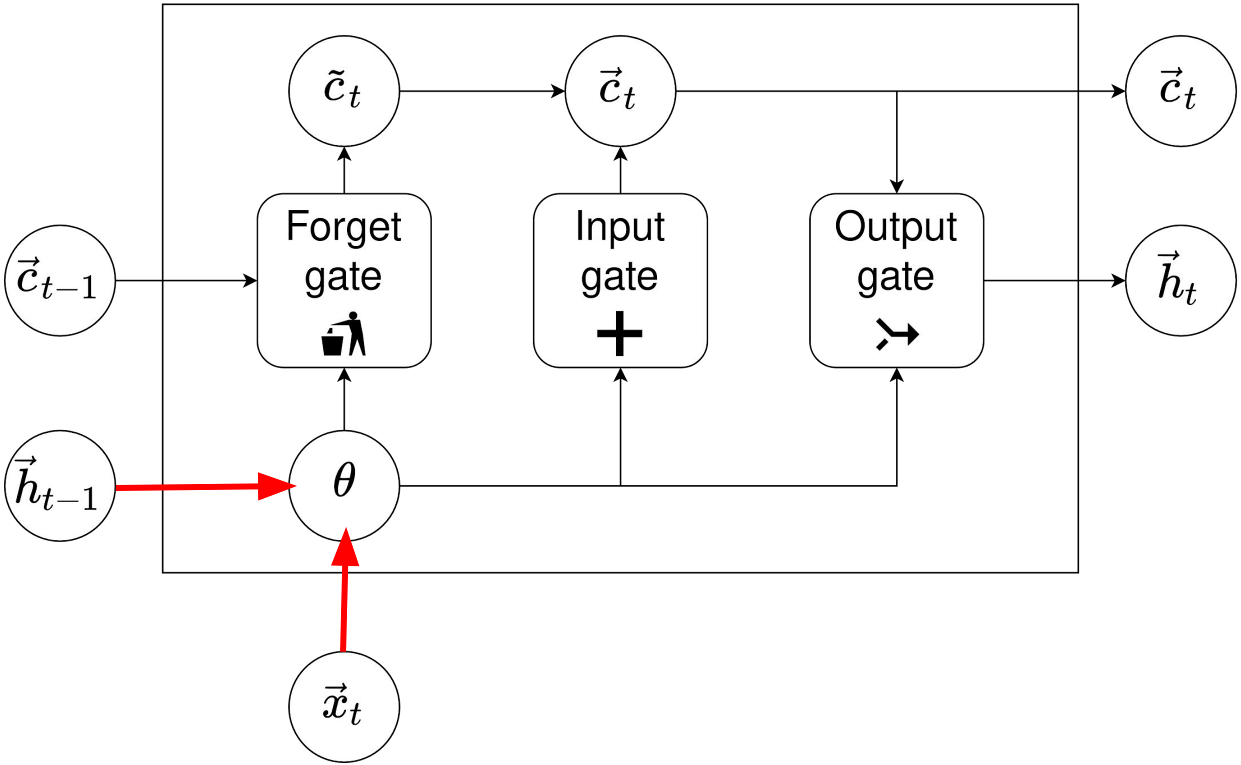
Thomas Vakili - thomas.vakili@dsv.su.se

Inside the LSTM



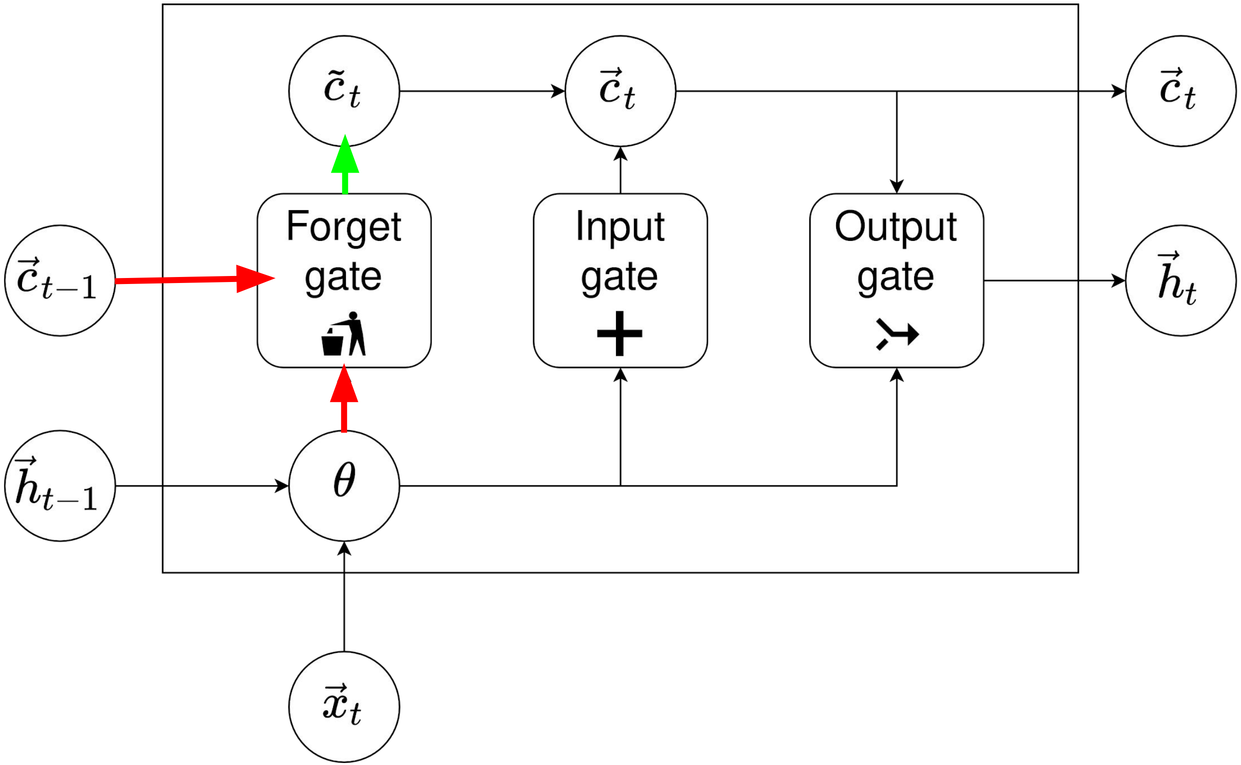
Thomas Vakili - thomas.vakili@dsv.su.se

Inside the LSTM



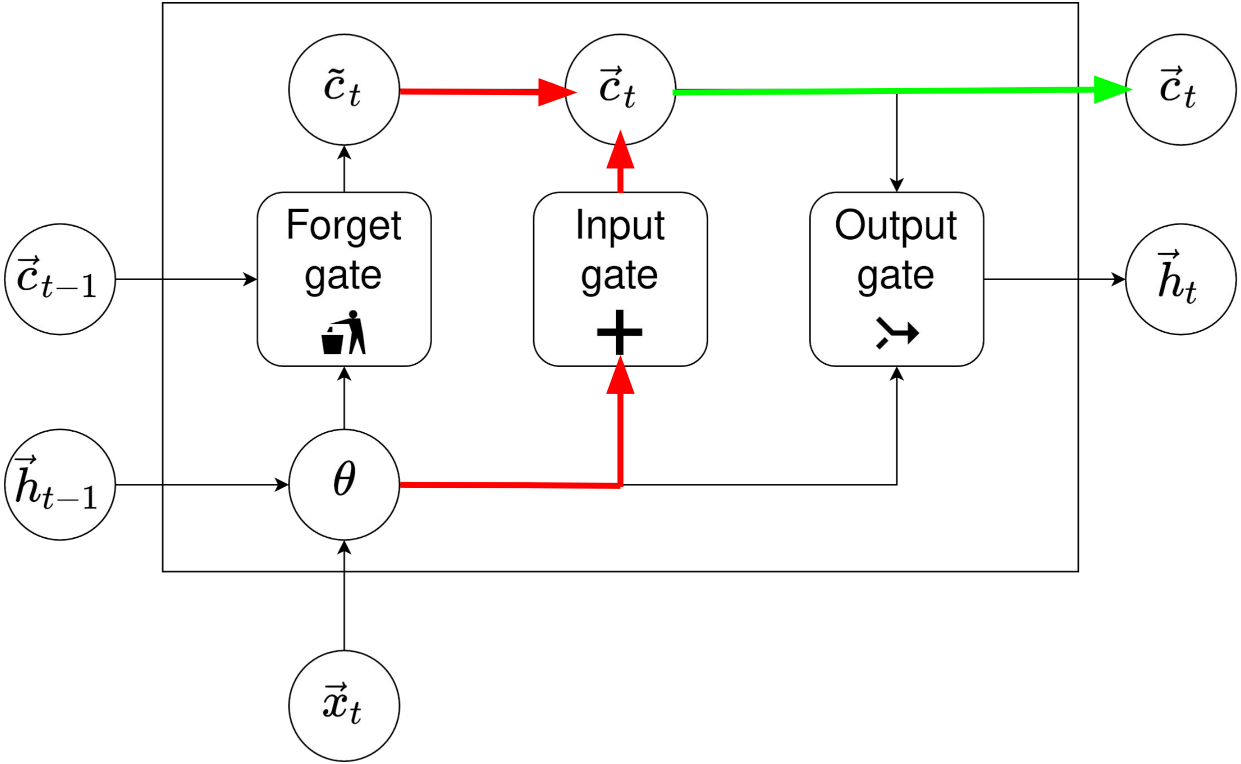
Thomas Vakili - thomas.vakili@dsv.su.se

Inside the LSTM

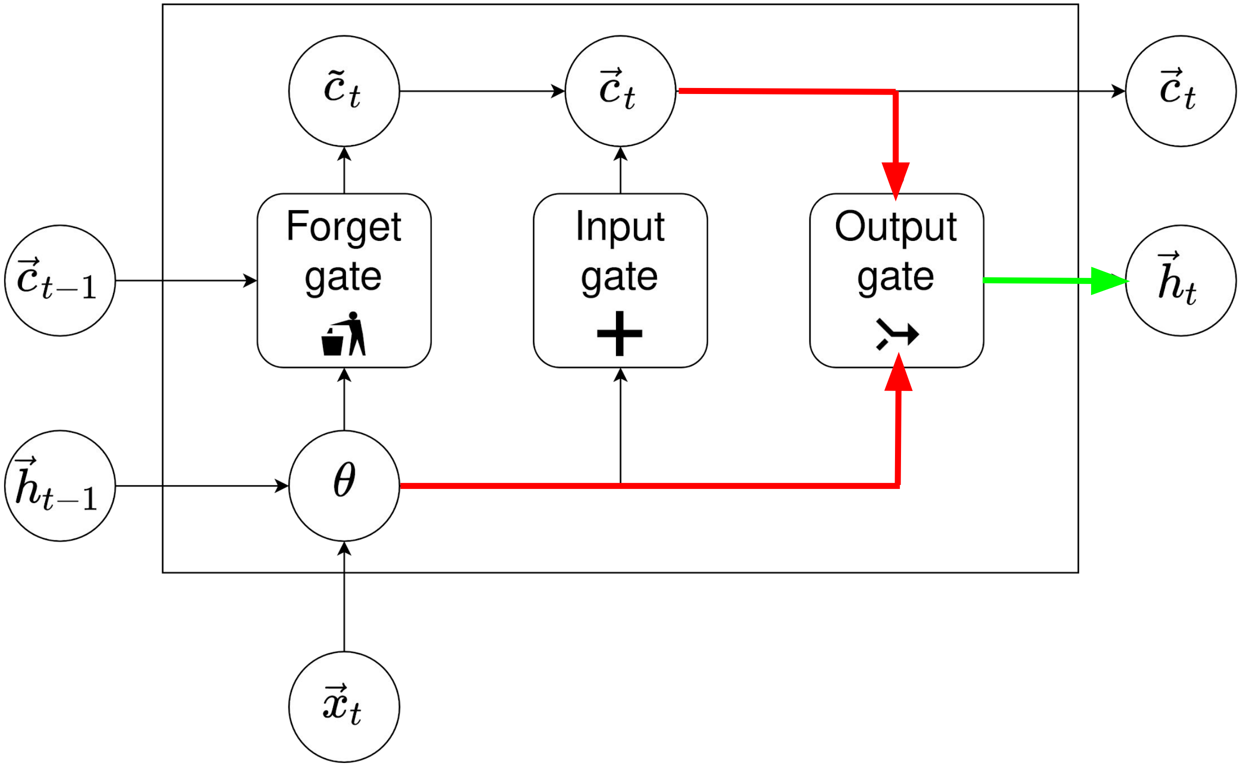


Thomas Vakili - thomas.vakili@dsv.su.se

Inside the LSTM



Inside the LSTM



Thomas Vakili - thomas.vakili@dsv.su.se

Some “small” mathematical tweaks

$$\vec{h}_{t+1} = f(W\vec{x}_t + V\vec{h}_t + \vec{b})$$

$$\Downarrow$$

Thomas Vakili - thomas.vakili@dsv.su.se

Some “small” mathematical tweaks

Stockholm University

$$\vec{h}_{t+1} = f(W\vec{x}_t + V\vec{h}_t + \vec{b})$$

$$\Downarrow$$

$$\begin{aligned} f_t &= \sigma(\vec{b}_f + W_f\theta) \\ g_t &= \sigma(\vec{b}_{g_1} + W_{g_1}\theta) \odot s(\vec{b}_{g_2} + W_{g_2}\theta) \\ q_t &= \sigma(\vec{b}_q + W_q\theta) \end{aligned}$$

$$\begin{aligned} \vec{c}_t &= f_t \odot \vec{c}_{t-1} + g_t \\ \vec{h}_t &= \tanh(\vec{c}_t) \odot q_t \end{aligned}$$

Thomas Vakili - thomas.vakili@dsv.su.se

Some “small” mathematical tweaks

$$\vec{h}_{t+1} = f(W\vec{x}_t + V\vec{h}_t + \vec{b})$$

$$\Downarrow$$

$$f_t = \sigma(\vec{b}_f + W_f\theta)$$

$$g_t = \sigma(\vec{b}_{g_1} + W_{g_1}\theta) \odot s(\vec{b}_{g_2} + W_{g_2}\theta)$$

$$q_t = \sigma(\vec{b}_q + W_q\theta)$$

$$\vec{c}_t = f_t \odot \vec{c}_{t-1} + g_t$$

$$\vec{h}_t = \tanh(\vec{c}_t) \odot q_t$$

Thomas Vakili - thomas.vakili@dsv.su.se

Pros and cons

LSTMs handle long-term dependencies quite well!

However:

- The latest input still unproportionally affects the state
- Data must be processed sequentially, word for word
- They handle transfer learning poorly

Time for a break!

Pre-trained Transformers

Attention is all you need

- Attention was first introduced to improve LSTMs
- Vaswani et al. (2017) realized that attention lets us do away with sequential processing!
- This resulted in a new deep learning architecture – the *Transformer* was born

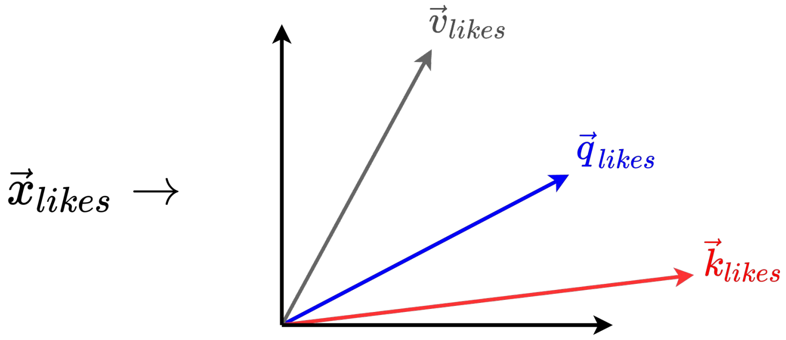
Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł. and Polosukhin, I., 2017. Attention is all you need. *Advances in neural information processing systems*, *30*.

Self-attention

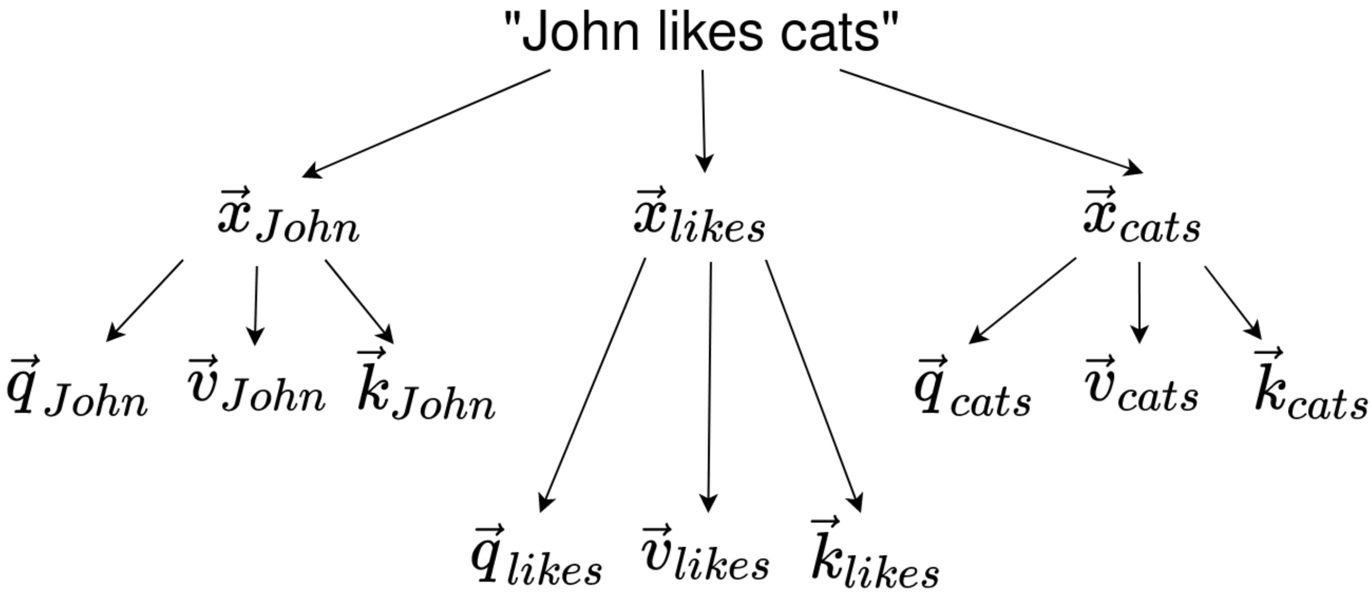
- Self-attention considers all combinations of words *directly* instead of using a hidden state

- This results in *contextualized representations*

- This is done by using *query*, *key* and *value vectors* created from all words

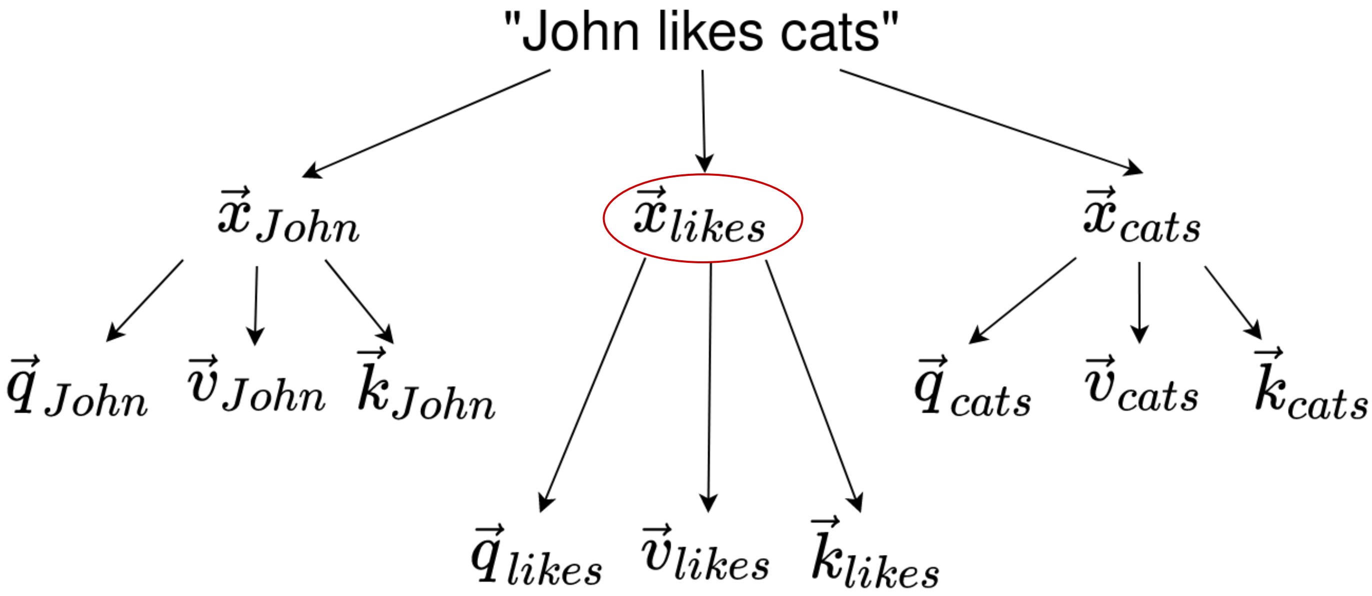


Self-attention



Thomas Vakili - thomas.vakili@dsv.su.se

Self-attention



Thomas Vakili - thomas.vakili@dsv.su.se

Self-attention

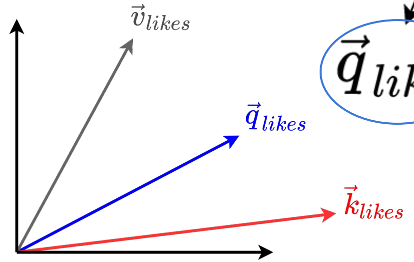
"John likes cats"

$\vec{x}_{John}$ $\vec{x}_{likes}$ $\vec{x}_{cats}$

$\vec{q}_{John}$ $\vec{v}_{John}$ $\vec{k}_{John}$ $\vec{q}_{cats}$ $\vec{v}_{cats}$ $\vec{k}_{cats}$

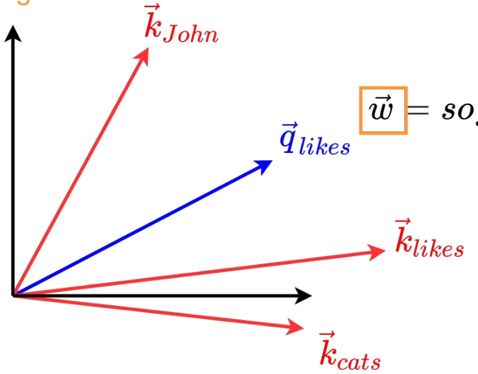
$\vec{q}_{likes}$ $\vec{v}_{likes}$ $\vec{k}_{likes}$

$\vec{x}_{likes} \rightarrow$



Creating the contextualized representation

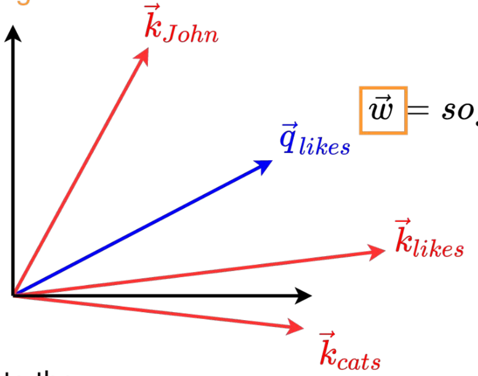
First calculate
relevancy weights



$$\boxed{\vec{w}} = softmax\left(\begin{bmatrix} \vec{q}_{likes}\vec{k}_{John} \\ \vec{q}_{likes}\vec{k}_{likes} \\ \vec{q}_{likes}\vec{k}_{cats} \end{bmatrix}\right)$$

Creating the contextualized representation

First calculate
relevancy weights

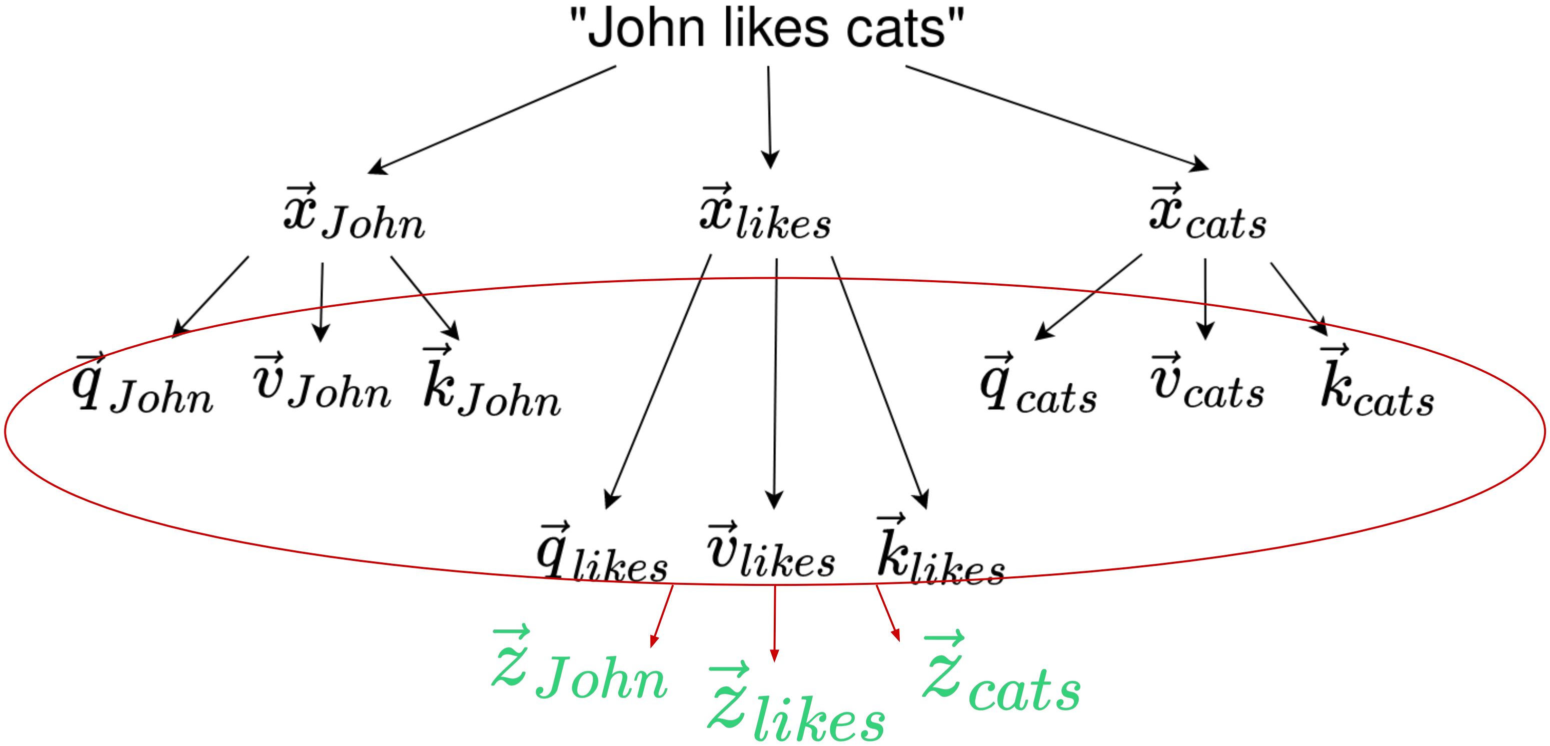


$$\boxed{\vec{w}} = softmax\left(\begin{bmatrix} \vec{q}_{likes}\vec{k}_{John} \\ \vec{q}_{likes}\vec{k}_{likes} \\ \vec{q}_{likes}\vec{k}_{cats} \end{bmatrix}\right)$$

Then calculate the
weighted representation

$$\vec{w} = \begin{bmatrix} 0.34 \\ 0.57 \\ 0.09 \end{bmatrix} \implies \boxed{\vec{z}_{likes}} = 0.34 \cdot \vec{v}_{John} + 0.57 \cdot \vec{v}_{likes} + 0.09 \cdot \vec{v}_{cats}$$

Self-attention



Thomas Vakili - thomas.vakili@dsv.su.se

Examples of self-attention

Jennifer and Stephanie met by the river bank for their weekly politics club. They had a lot to discuss.

Examples of self-attention

Jennifer and Stephanie met by the river bank for their weekly politics club. They had a lot to discuss.

$$\vec{q}_{bank} \cdot \vec{k}_{river} = 0.25$$

Thomas Vakili - thomas.vakili@dsv.su.se

Examples of self-attention

Jennifer and Stephanie met by the river bank for their weekly politics club.
They had a lot to discuss.

$$\vec{q}_{bank} \cdot \vec{k}_{river} = \boxed{0.25}$$

$$\vec{z}_{bank} = 0.6 \cdot \vec{v}_{bank} + 0.25 \cdot \vec{v}_{river} \ \ldots$$

Thomas Vakili - thomas.vakili@dsv.su.se

Examples of self-attention

Jennifer and Stephanie met by the river bank for their weekly politics club.
They had a lot to discuss.

$$\vec{z}_{They} = 0.4 \cdot \vec{v}_{Jennifer} + 0.4 \cdot \vec{v}_{Stephanie} \;\; \cdots$$

Examples of self-attention

Jennifer and Stephanie met by the river bank for their weekly politics club. They had a lot to discuss.

$$\vec{z}_{discuss} = 0.5 \cdot \vec{v}_{discuss} + 0.4 \cdot \vec{v}_{politics} \ldots$$

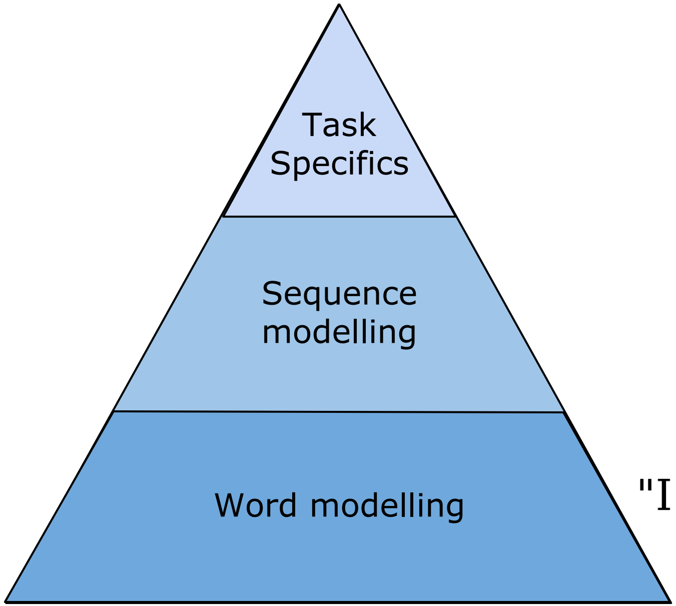
Multi-headed attention

Jennifer and Stephanie met by the river bank for their weekly politics club.
They had a lot to discuss.

Why Transformers are great

- Long term dependencies easy to learn because every word “sees” every other word
- Doing away with sequential processing allows for parallelization
- Most importantly: they are great at *transfer learning*

The NLP pyramid

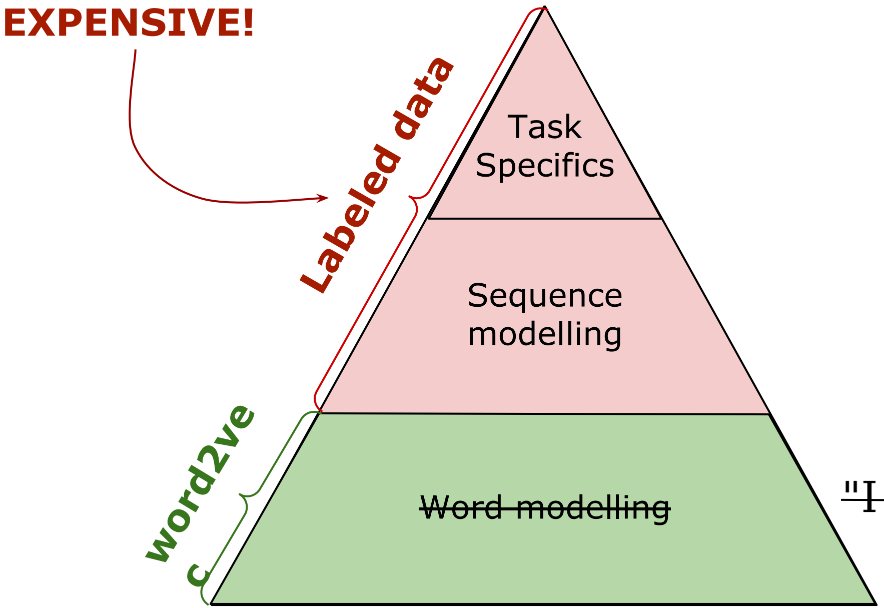


$$\vec{s}_{\text{"I am happy"}} \rightarrow positive$$

$$\{\vec{w}_I, \vec{w}_{am}, \vec{w}_{happy}\} \rightarrow \vec{s}_{\text{"I am happy"}}$$

$$\text{"I am happy"} \rightarrow \{\vec{w}_I, \vec{w}_{am}, \vec{w}_{happy}\}$$

LSTMs need more labeled data



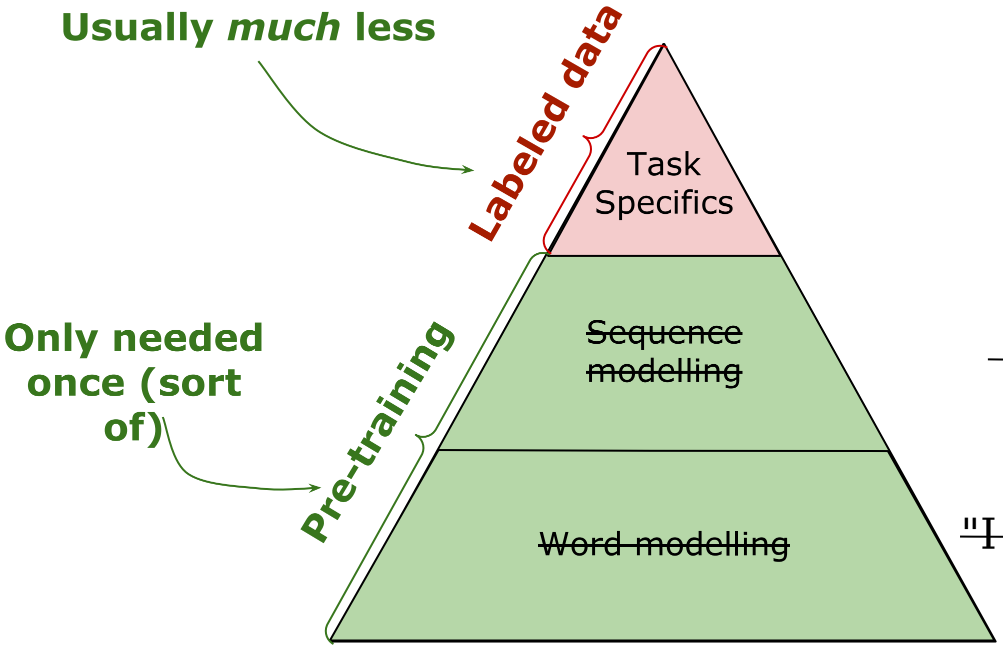
$$\vec{s}_{\text{"I am happy"}} \rightarrow \mathit{positive}$$

$$\{\vec{w}_I, \vec{w}_{am}, \vec{w}_{happy}\} \rightarrow \vec{s}_{\text{"I am happy"}}$$

$$\cancel{\text{"I am happy"} \rightarrow \{\vec{w}_I, \vec{w}_{am}, \vec{w}_{happy}\}}$$

Thomas Vakili - thomas.vakili@dsv.su.se

Pre-trained transformers need less



$$\vec{s}_{\text{"I am happy"}} \rightarrow \mathit{positive}$$

$$\cancel{\{\vec{w}_I, \vec{w}_{am}, \vec{w}_{happy}\} \rightarrow \vec{s}_{\text{"I am happy"}}}$$

$$\cancel{\text{"I am happy"} \rightarrow \{\vec{w}_I, \vec{w}_{am}, \vec{w}_{happy}\}}$$

Language Models

Stockholm University

- Language modelling is traditionally done "left-to-right"
- GPT-{1, 2, 3} do this
  - These are *autoregressive* language models

$$x_{i+1} = \underset{w \in V}{\operatorname{argmax}} P(w | x_1, x_2, ..., x_i)$$

BERT

- BERT (popular model) is a *masked* language model
- Trained by learning to “fill in the gaps”

Thomas Vakili is a PhD **[MASK]** at DSV

↓

Thomas Vakili is a PhD **student** at DSV

$$x_{mask} = \underset{w \in V}{\operatorname{argmax}} P(w | X \setminus x_{mask})$$

Traditional pipeline

Thomas Vakili is a PhD student at DSV

Word splitting ↓

["Thomas", "Vakili", "is", "a", "PhD", "student", "at", "DSV"]

Lemmatization ↓

*["Thomas", "Vakili", **"be"**, "a", "PhD", "student", "at", "DSV"]*

Traditional pipeline

Thomas Vakili is a PhD student at DSV

Word splitting ↓

[“Thomas”, “Vakili”, “is”, “a”, “PhD”, “student”, “at”, “DSV”]

Lemmatization ↓

*[“Thomas”, “Vakili”, **“be”**, “a”, “PhD”, “student”, “at”, “DSV”]*

Vocabulary lookup ↓

*[“Thomas”, **“[UNK]”**, “be”, “a”, “PhD”, “student”, “at”, **“[UNK]”**]*

Thomas Vakili - thomas.vakili@dsv.su.se

SentencePiece

- Sub-word tokenizers go beneath the word level
- SentencePiece[1] *learns* the vocabulary from data
- Vocabulary is not (only) words

temporality → *temporal* + *ity*

1: Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Tokenization

```
Python 3.10.6 (main, Nov 14 2022, 16:10:14) [GCC 11.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from transformers import AutoTokenizer
>>> t = AutoTokenizer.from_pretrained('bert-base-cased')
>>> t.tokenize('Thomas Vakili is a PhD student at DSV')
```

Tokenization

```
Python 3.10.6 (main, Nov 14 2022, 16:10:14) [GCC 11.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from transformers import AutoTokenizer
>>> t = AutoTokenizer.from_pretrained('bert-base-cased')
>>> t.tokenize('Thomas Vakili is a PhD student at DSV')
['Thomas', 'V', '##aki', '##li', 'is', 'a', 'PhD', 'student', 'at', 'DS', '##V']
```

Let’s take a few minutes to process that

Why Transformers are also *not* so great

Explosion in parameters

Thomas Vakili - thomas.vakili@dsv.su.se

Explosion in parameters

Thomas Vakili - thomas.vakili@dsv.su.se

Mini quiz: pre-training and CO_2

Strubell et al. [redacted] found that training this model resulted in CO_2 emissions equal to a trans-American flight:

A. GPT-2 (1.5 billion parameters)
B. T5-11B (11 billion parameters)
C. BERT-large (340 million parameters)

Strubell, E., Ganesh, A., & McCallum, A. [redacted]. Energy and Policy Considerations for Deep Learning in NLP. [redacted]

Mini quiz: pre-training and CO_2

Strubell et al. (2019) found that training this model resulted in CO_2 emissions equal to a trans-American flight:

A. ~~GPT-2 (1.5 billion parameters)~~
B. ~~T5-11B (11 billion parameters)~~
C. ~~BERT-large (340 million parameters)~~
D. **BERT-base (110 million parameters)**

Strubell, E., Ganesh, A., & McCallum, A. (2019). Energy and Policy Considerations for Deep Learning in NLP. *ArXiv:1906.02243 [Cs].* http://arxiv.org/abs/1906.02243

Cost of pre-training

Stockholm University

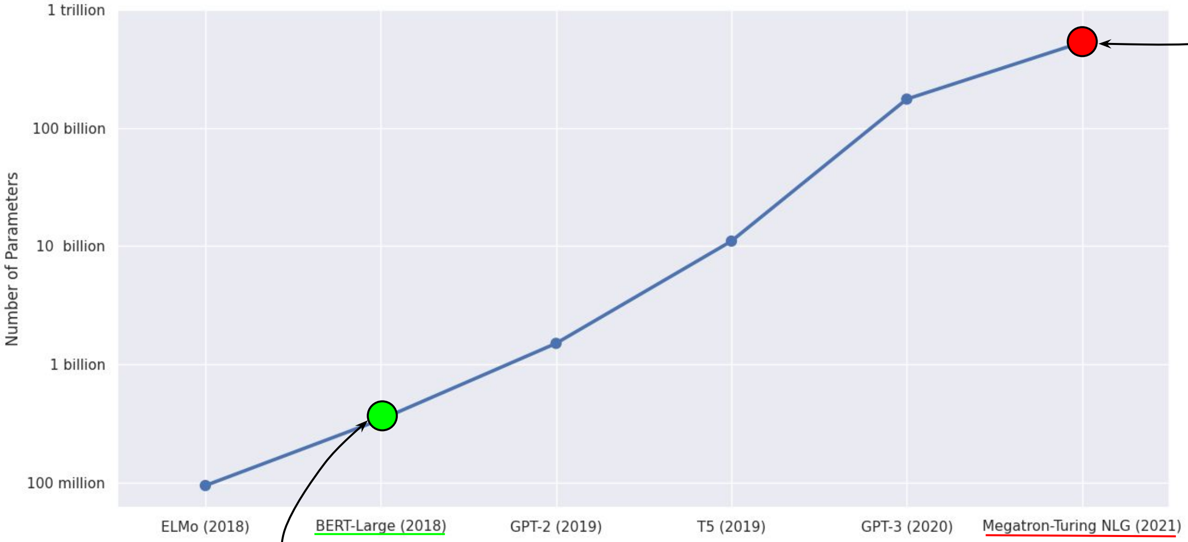
- Pre-training is *extremely* expensive, requires very powerful computers and consumes *a lot* of energy

- You only need to do it once, but…
  - Models are more or less monolingual (in English)
  - SOTA results require domain-adaptation
  - Reproducing results is basically impossible
  - It’s probably Google/OpenAI who trained it

Thomas Vakili - thomas.vakili@dsv.su.se

Explosion in data use

270 billion words



3.3 billion words

Thomas Vakili - thomas.vakili@dsv.su.se

Enormous data

- Labeled training data is expensive because actual human beings need to process it

- Learning from unlabeled data is cheap but…
  - It is impossible to guarantee its quality
  - How do we avoid learning stereotypes?
  - Who's language does the data represent?
  - Does the data contain personally identifiable info?

Whose language is being modeled?

- BERT uses Wikipedia and a corpus of books

- GPT-3 mainly uses a large data dump of the internet
  - Links collected from Reddit (22%)
  - Data from CommonCrawl (60%)
  - Rest is similar to BERT

- **Question:** what could be the problems with learning to model language based on this data?

Language is not static

Stockholm University

- A perfect model would still need to be re-trained
- Word use changes: nobody says “groovy” anymore
- The world changes!

The PM of Sweden is: ~~Magdalena Andersson~~ Ulf Kristersson

The Winter Olympics will be held in: ~~Pyeongchang~~ Beijing

Some things shouldn’t be learned

- Carlini et al. (2020) showed that GPT-2 memorizes
  - Entire Reddit threads
  - Social security numbers
  - Phone numbers

- They could *extract* this sensitive information by prompting it cleverly

(not really)

Thomas’ personal number is: 950208-1234

Stereotypes in language models

Stockholm University

- All language is cultural and reflects biases and prejudices that exist in society
- Language on the internet is even worse
- Language models have been shown to:
  - Associate African-American names with negative words
  - Stereotypically label professions by gender

```
ional as F
ons import Categorical
ed_bert import BertTokenizer, BertForMaskedLM, BertForMaskedLM



, tokenizer):
e sentence by tokenizing and converting to tensor. """
er.convert_tokens_to_ids(tokens)
h.tensor([tok_ids]) # pylint: disable=not-callable



d_sentence, tokenizer, masking, n_append_mask=0):
entence to decode from, possible with masks. """

masking):
 "none":
= []
== "random":
= []

= [int(d) for d in masking.split(',')]
ds

k
sk_ids(masking)

cting [MASK]
eed_sentence.replace(MASK, MASK_ATOM)
tokenize(seed_sentence)
merate(toks):
K_ATOM:
append(i)


sk_ids:
```

Demo time!