Deep Learning for Natural Language Processing

Jan Buys
Department of Computer Science
University of Cape Town
SACAIR 2020 Tutorial

UNIVERSITY OF CAPE TOWN
IYUNIVESITHI YASEKAPA • UNIVERSITEIT VAN KAAPSTAD

Natural Language Processing (NLP)

- Systems that process language (text or speech) and enable human-computer interaction through language

Deep Learning

Machine Learning with Neural Networks

- Large datasets, large models, high computational cost
- Representation learning: learn the features

Why Deep Learning for NLP?

- Language is hard!
- Data sparsity
- Long sequences

Deep learning enables learning reusable representations

NLP Applications

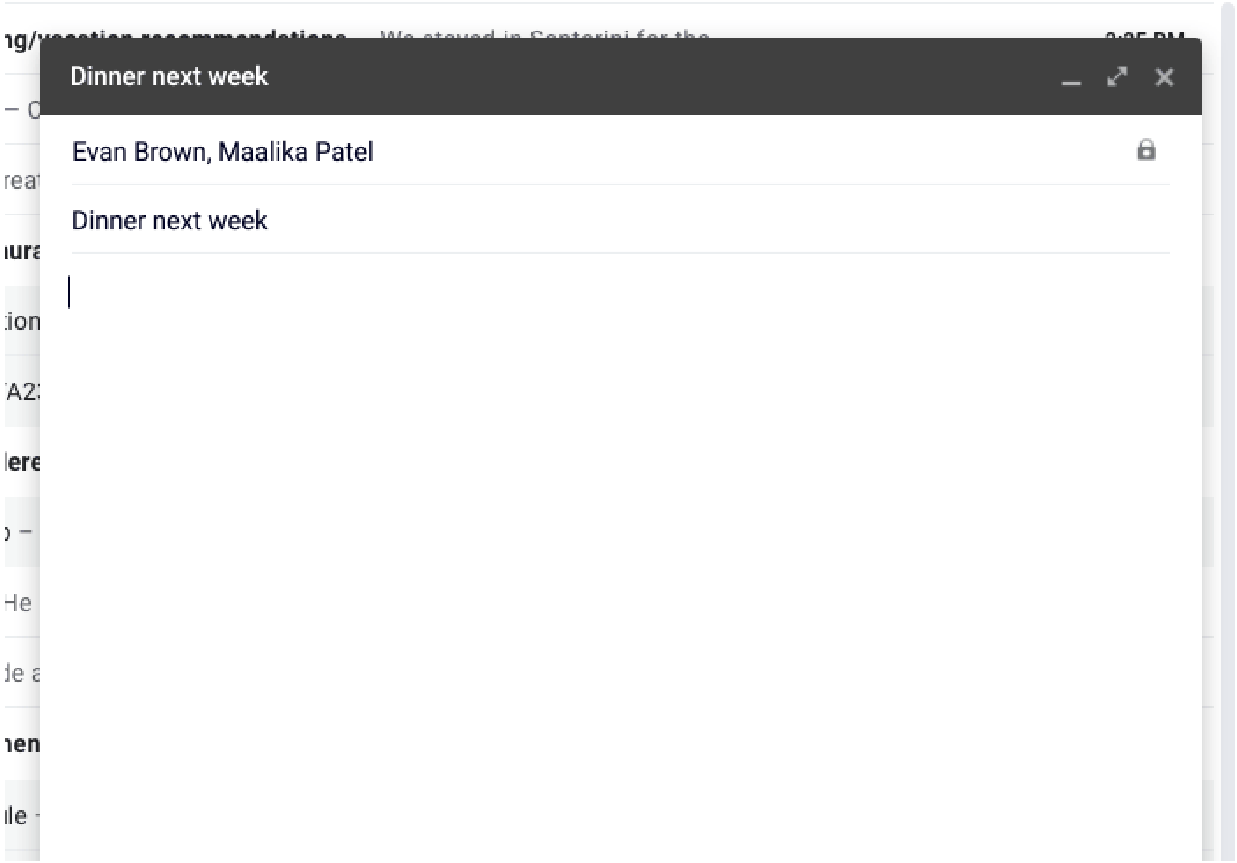
NLP Applications

- Spam Detection

CONGRATULATION!!!
With reference to the 1,377th EuroMillions draw which took place on Tuesday 1st December 2020 at 21:00 CEST (20:00 BST) and the winning numbers drawn were: Lucky numbers 14-20-29-47-49 Star Number 4-12 Millionaire Maker: MNHF52876 serial number ZWWD49193 Prize credited to file EURO/86169/2021
An official letter was sent to your address. Your email address has been awarded the sum of 2,713,908.40 GB pounds. Kindly, confirm receipt of this notification by contacting your claims officer Mr. Kennith William for more details. visit the link https://www.euro-millions.com/results/01-12-2020 to view your winning details as published on the Euro-Millions site.
Euro-Millions prizes must be claimed within 180 days of the draw date. This is a confidential mail sent to ONLY winners of this draws.
If you have any questions, please contact our customer support.

NLP Applications

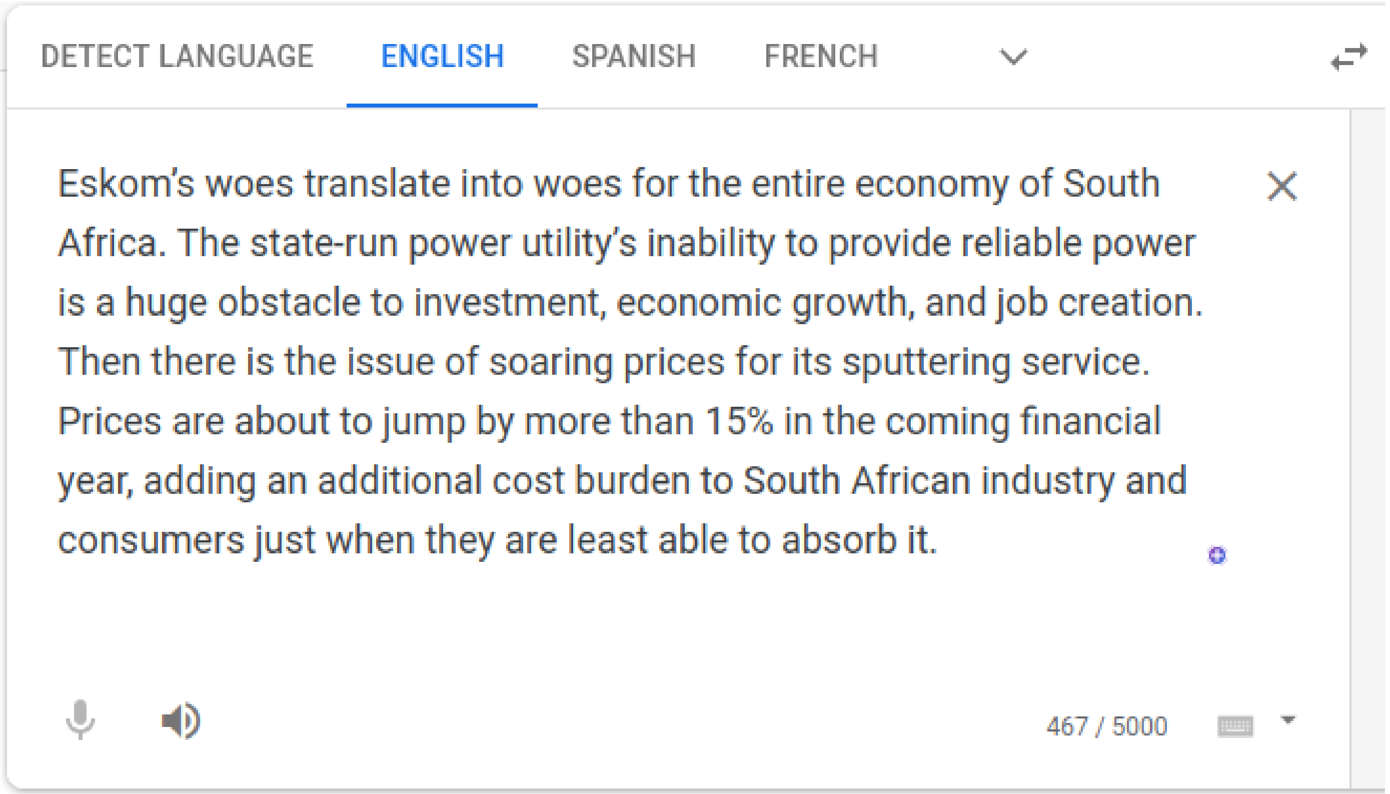
- Smart Compose



https://ai.googleblog.com/2018/05/smart-compose-using-neural-networks-to.html

NLP Applications

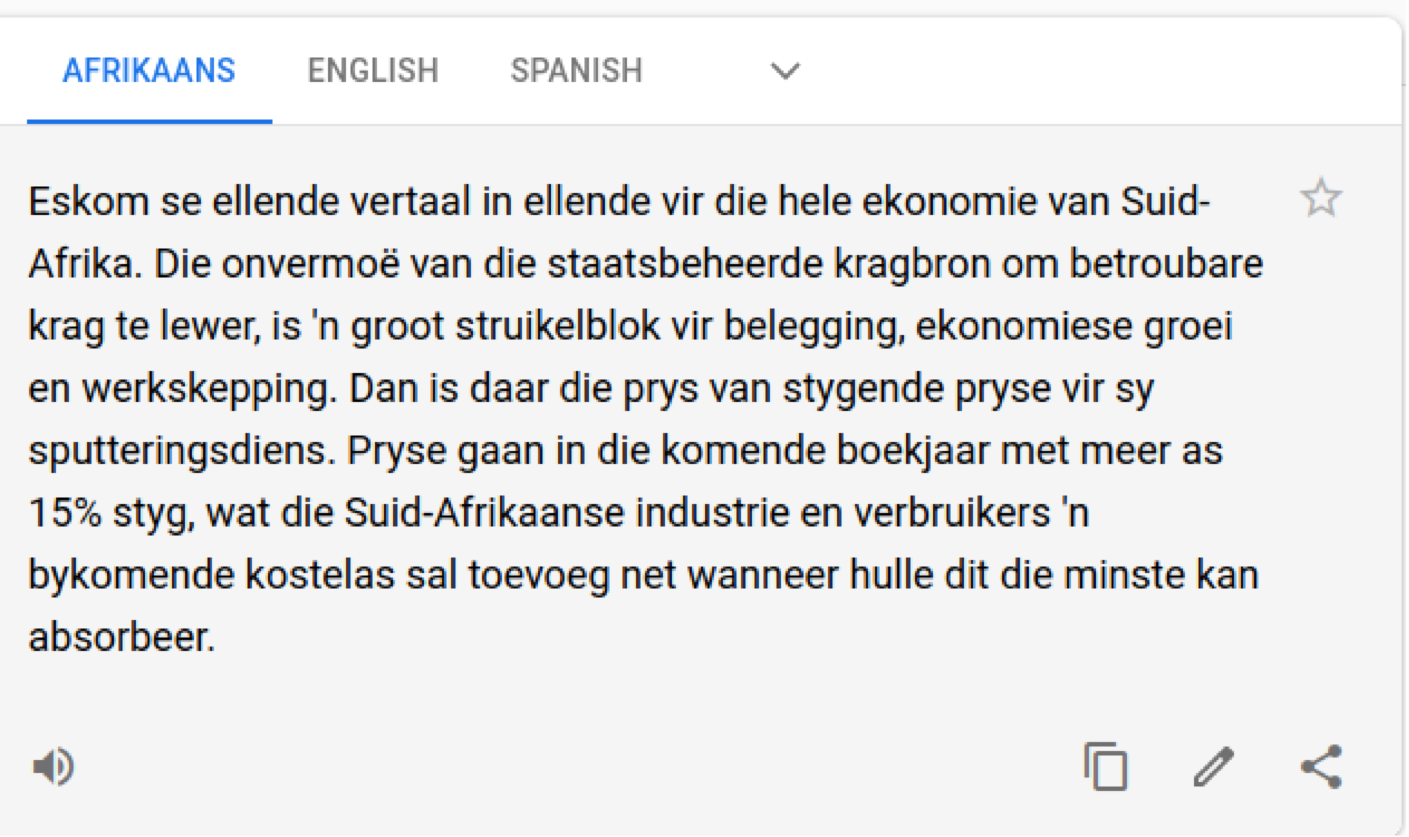
- Machine Translation



translate.google.com
Text: www.dailymaverick.co.za

NLP Applications

- Machine Translation



translate.google.com
Text: www.dailymaverick.co.za

NLP Applications

- Virtual Assistants

"Alexa, wake me up at 7 in the morning."

"Alexa, what's my Flash Briefing?"

"Alexa, what's on my calendar today?"

"Alexa, what's the weather in London?"

"Alexa, play Katy Perry from Prime Music."

"Alexa, how's my commute?"

Outline

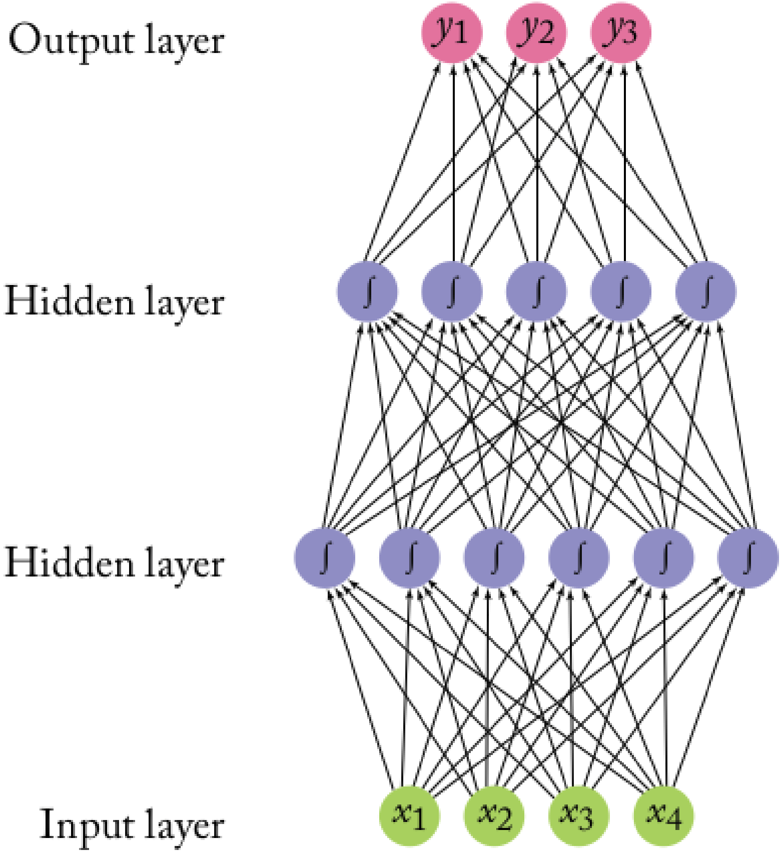
- 1. Word representations
- 2. Sequence processing
- 3. Transformers and contextualized representations

We won't cover:

- Machine Learning basics
- Model Optimization
- Implementation details

1. Word Representations

Feed-forward Neural Networks



Goldberg (2017) Ch. 4

Feed-forward Neural Networks (FFNNs)

1 Hidden layer:

$$\mathrm{NN}_{\mathrm{MLP1}}(\boldsymbol{x}) = g(\boldsymbol{x}\boldsymbol{W}^{\mathbf{1}} + \boldsymbol{b}^{\mathbf{1}})\boldsymbol{W}^{\mathbf{2}} + \boldsymbol{b}^{\mathbf{2}}$$

$$\boldsymbol{x} \in \mathbb{R}^{d_{in}}, \quad \boldsymbol{W}^{\mathbf{1}} \in \mathbb{R}^{d_{in} \times d_1}, \quad \boldsymbol{b}^{\mathbf{1}} \in \mathbb{R}^{d_1}, \quad \boldsymbol{W}^{\mathbf{2}} \in \mathbb{R}^{d_1 \times d_2}, \quad \boldsymbol{b}^{\mathbf{2}} \in \mathbb{R}^{d_2}.$$

2 Hidden layers:

$$\begin{aligned}
\mathrm{NN}_{\mathrm{MLP2}}(\boldsymbol{x}) &= \boldsymbol{y} \\
\boldsymbol{h}^{\mathbf{1}} &= g^{1}(\boldsymbol{x}\boldsymbol{W}^{\mathbf{1}} + \boldsymbol{b}^{\mathbf{1}}) \\
\boldsymbol{h}^{\mathbf{2}} &= g^{2}(\boldsymbol{h}^{\mathbf{1}}\boldsymbol{W}^{\mathbf{2}} + \boldsymbol{b}^{\mathbf{2}}) \\
\boldsymbol{y} &= \boldsymbol{h}^{\mathbf{2}}\boldsymbol{W}^{\mathbf{3}}.
\end{aligned}$$

Goldberg (2017) Ch. 4

Words and tokens

- Original Text:

“A number of comparable countries also experienced ‘bounce-backs’ in employment (to varying degrees), including Brazil, Colombia, Ghana, India, Mexico and Uruguay,” says principal investigator Nic Spaull in a consolidated report.

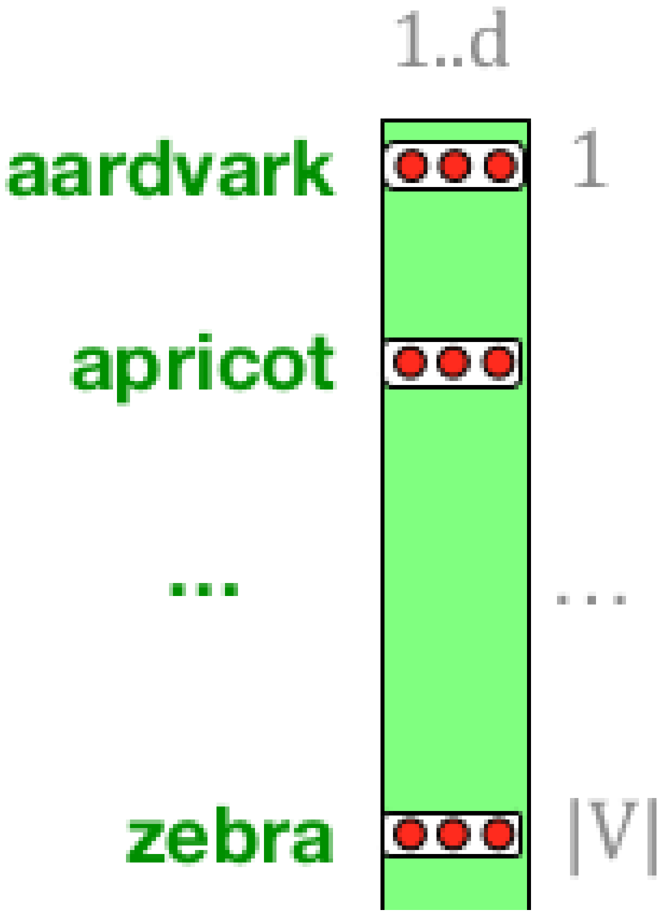
- Tokenized Text:

" A number of comparable countries also experienced ' bounce – backs ' in employment (to varying degrees) , including Brazil , Colombia , Ghana , India , Mexico and Uruguay , " says principal investigator Nic Spaull in a consolidated report .

Source: Daily Maverick

Defining a vocabulary

- Set vocabulary size
- Handle Out of Vocabulary words
- Learn a vector for each word in vocabulary



Jurafsky and Martin (2020) Ch. 6

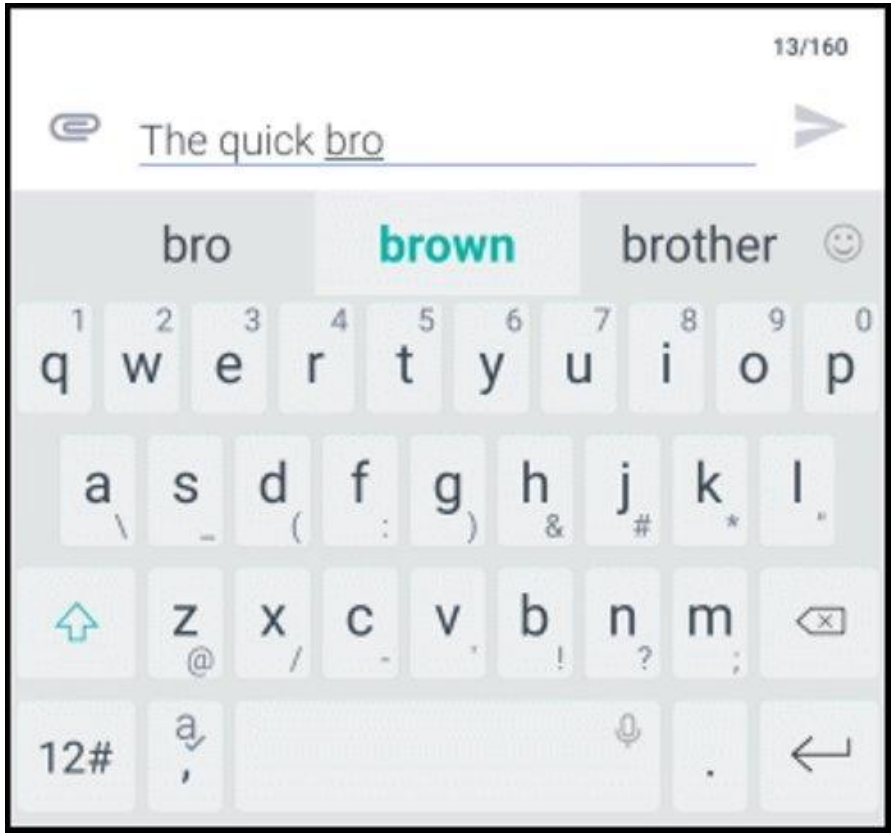
Word vectors (embeddings)

- Similar words should have similar vectors (based on cosine similarity)

Two-dimensional (t-SNE) projection of embeddings

Language Modelling

- Autocomplete



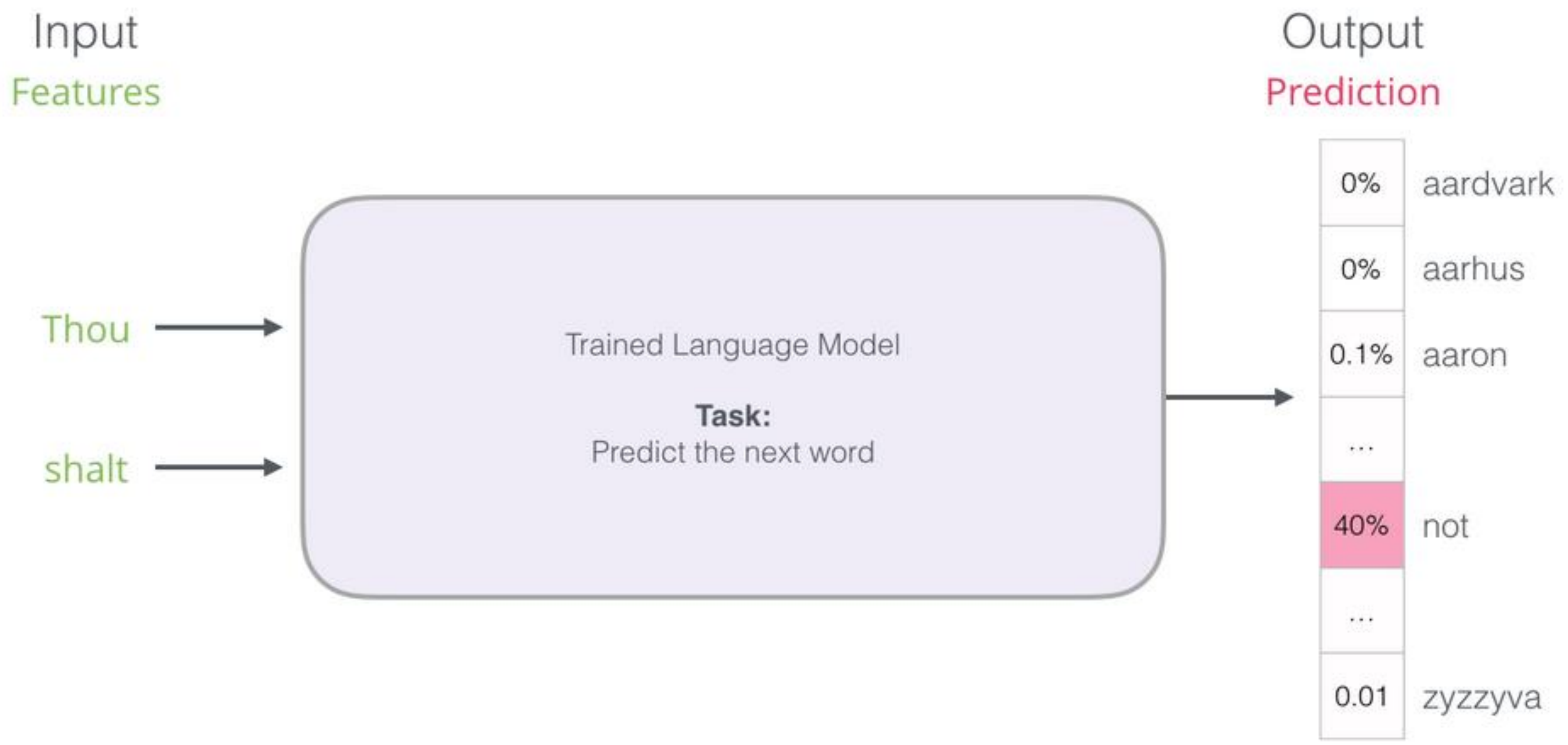
Language Modelling

https://jalammar.github.io/illustrated-word2vec/

Language Modelling

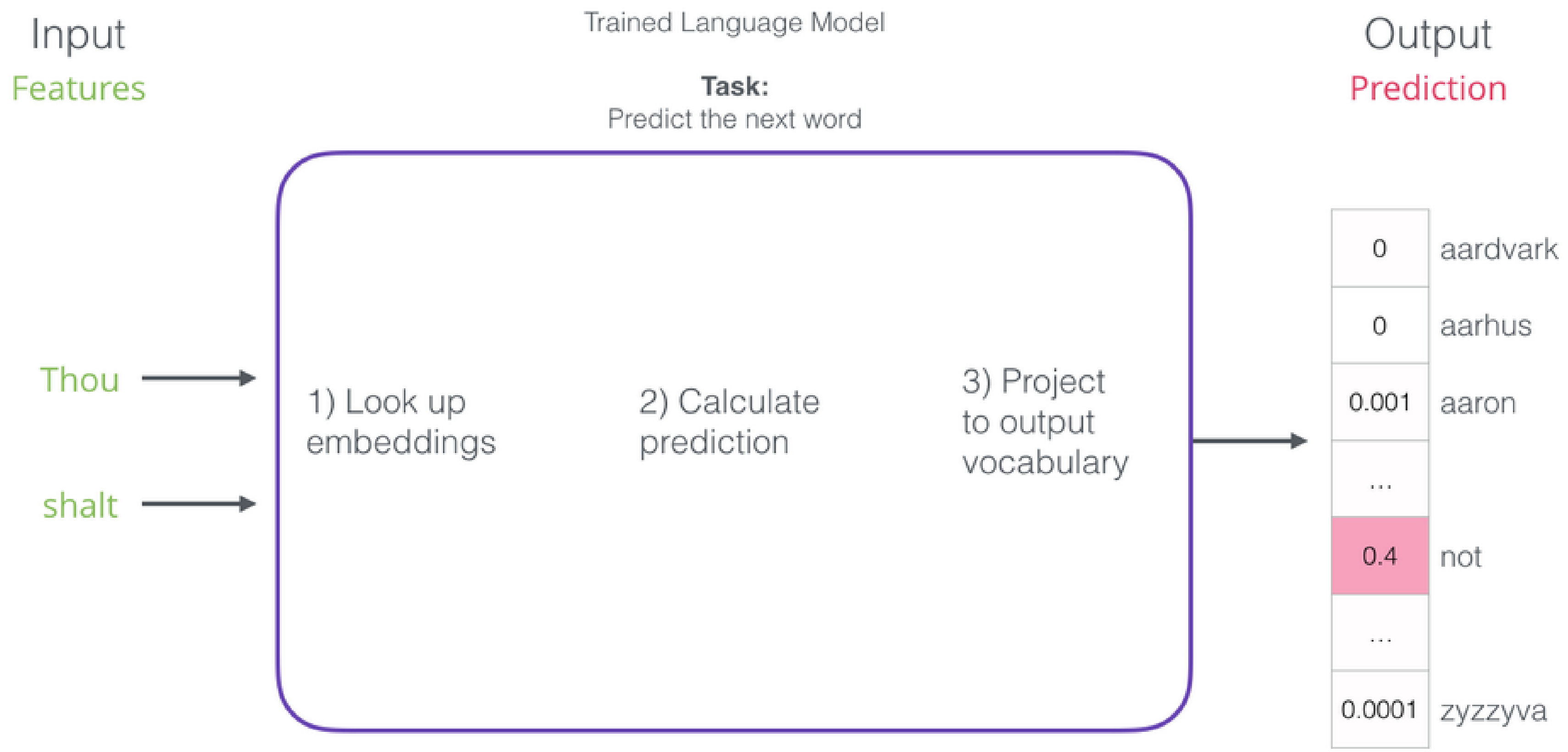
https://jalammar.github.io/illustrated-word2vec/

Language Modelling



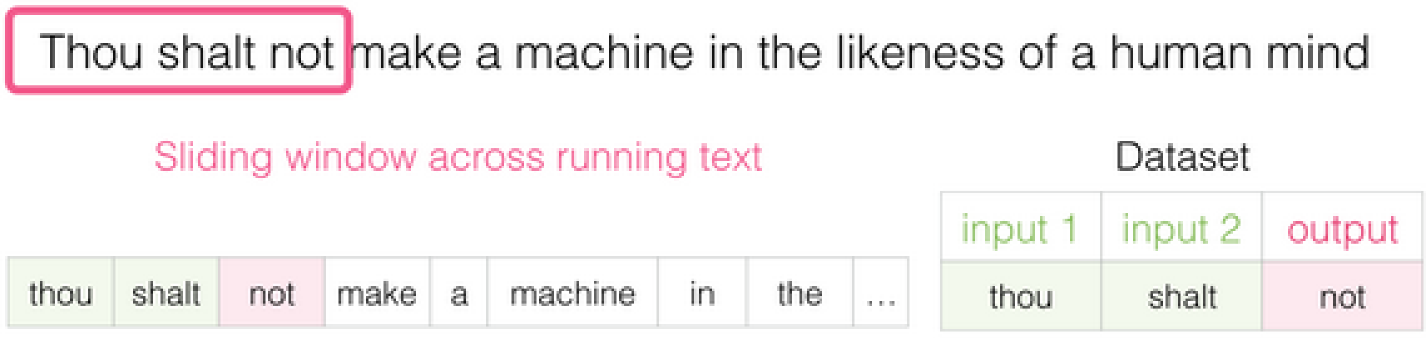
https://jalammar.github.io/illustrated-word2vec/

Language Modelling



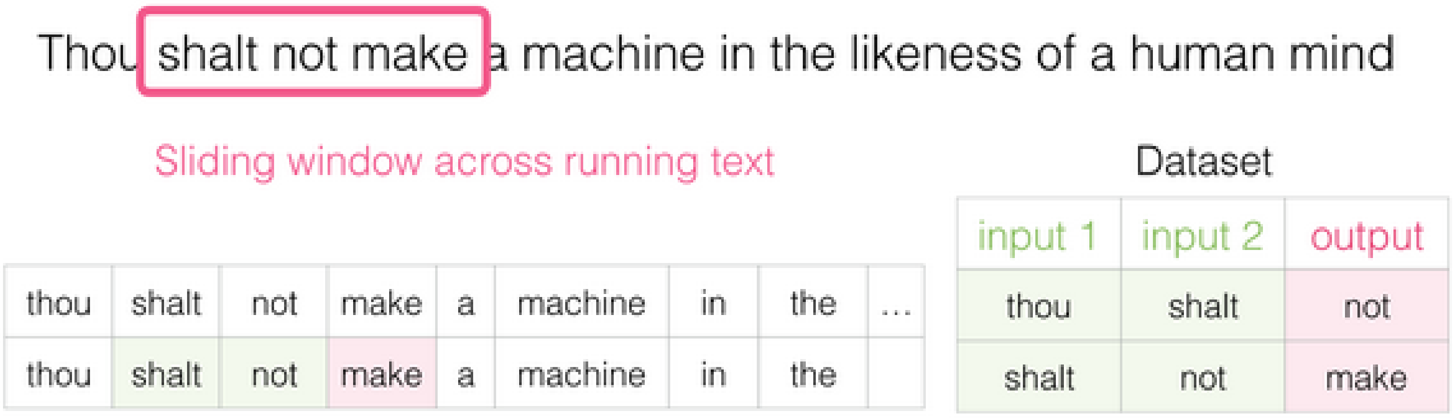
https://jalammar.github.io/illustrated-word2vec/

FFNN Language Modelling



https://jalammar.github.io
/illustrated-word2vec/

FFNN Language Modelling



https://jalammar.github.io/illustrated-word2vec/

FFNN Language Modelling

Thou shalt not make a machine in the likeness of a human mind

Sliding window across running text

thou	shalt	not	make	a	machine	in	the	…
thou	shalt	not	make	a	machine	in	the	
thou	shalt	not	make	a	machine	in	the	
thou	shalt	not	make	a	machine	in	the	
thou	shalt	not	make	a	machine	in	the	

Dataset

input 1	input 2	output
thou	shalt	not
shalt	not	make
not	make	a
make	a	machine
a	machine	in

https://jalammar.github.io/illustrated-word2vec/

FFNN Language Modelling

- Language Modelling objective

$$P(w_{1:n}) = \prod_{i=1}^{n} P(w_i | w_{<i})$$

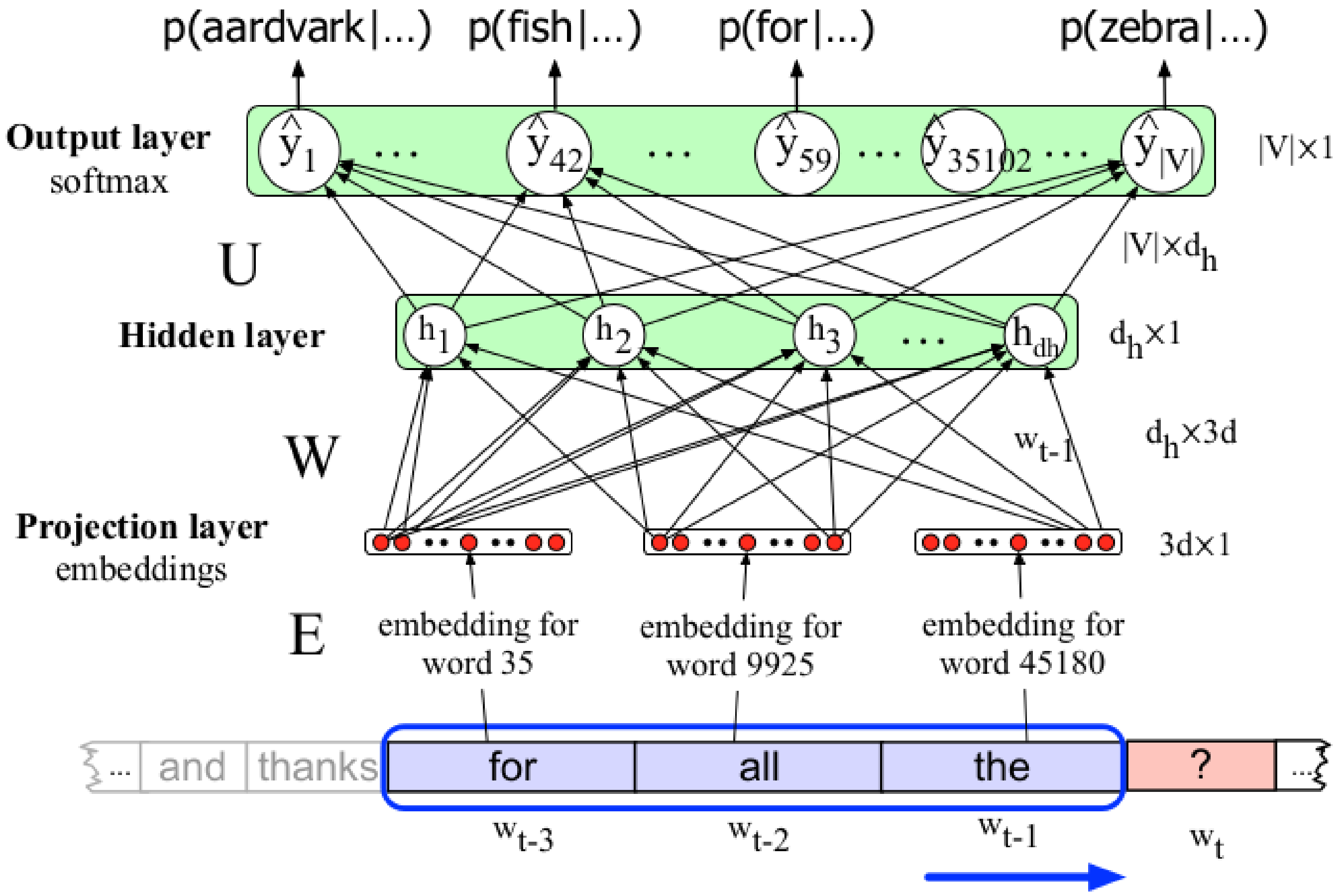
- FFNN Language Models estimate

$$P(w_n | w_{1:n-1}) \approx P(w_n | w_{(n-N+1):(n-1)})$$

- Can be used to train word embeddings

Jurafsky and Martin (2020) Ch. 9

FFNN Language Modelling



Jurafsky and Martin (2020) Ch. 9

Learning word embeddings: word2vec

- Use context from both directions

https://jalammar.github.io/illustrated-word2vec/

Learning word embeddings: word2vec

- Skip-gram: Predict neighbouring words from current word

Jay was hit by a red bus in...

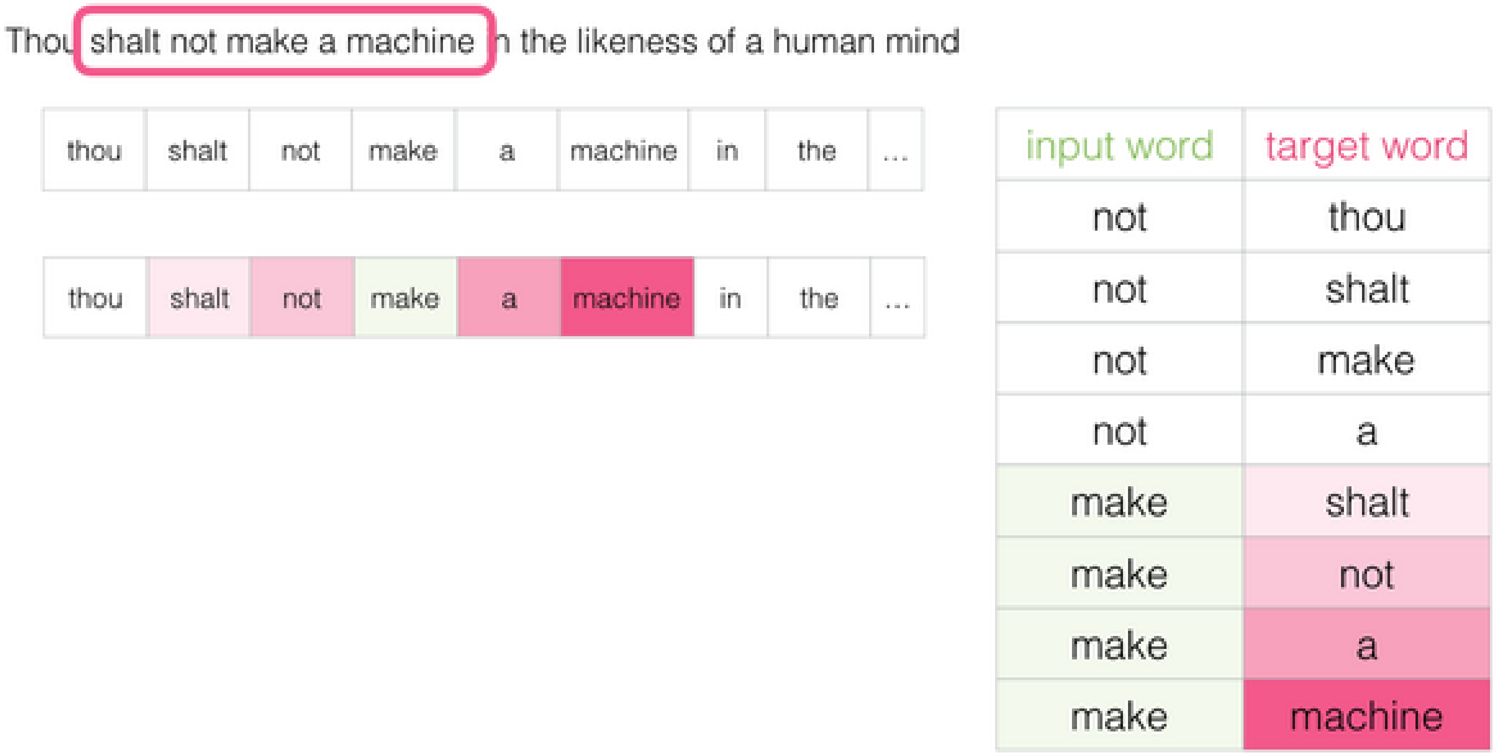
by	a	red	bus	in

input	output
red	by
red	a
red	bus
red	in

https://jalammar.github.io/illustrated-word2vec/

Learning word embeddings: word2vec

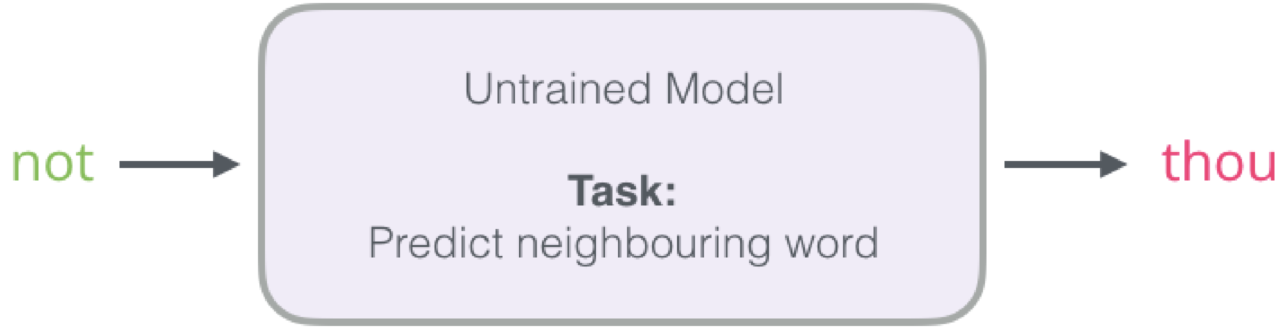
- Skip-gram: Predict neighbouring words from current word



https://jalammar.github.io/illustrated-word2vec/

Learning word embeddings: word2vec

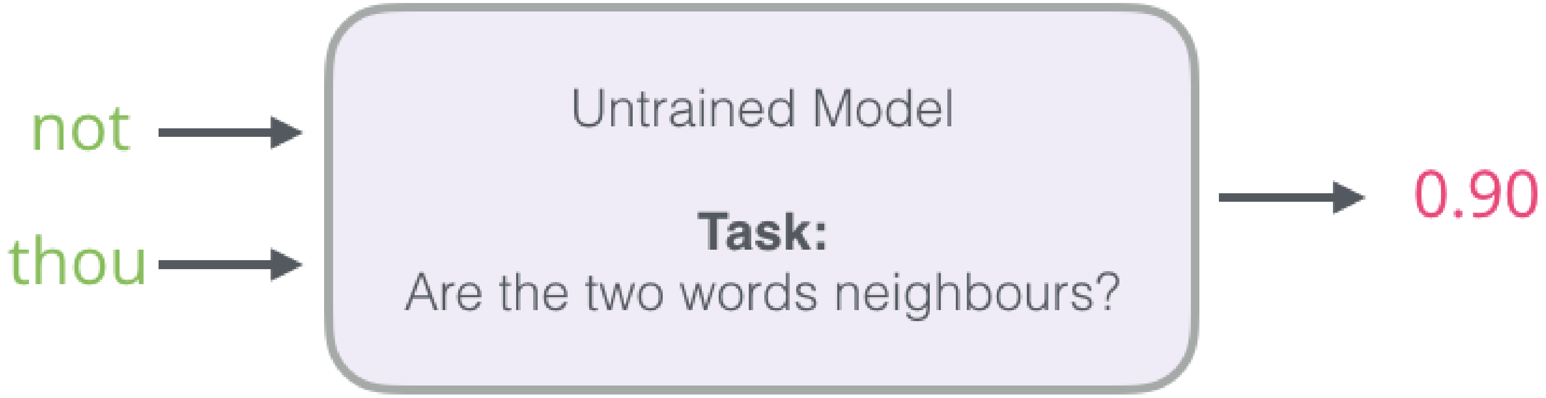
- Skip-gram with negative sampling (Mikolov et al., 2013)
- Change the word prediction task:



https://jalammar.github.io/illustrated-word2vec/

Learning word embeddings: word2vec

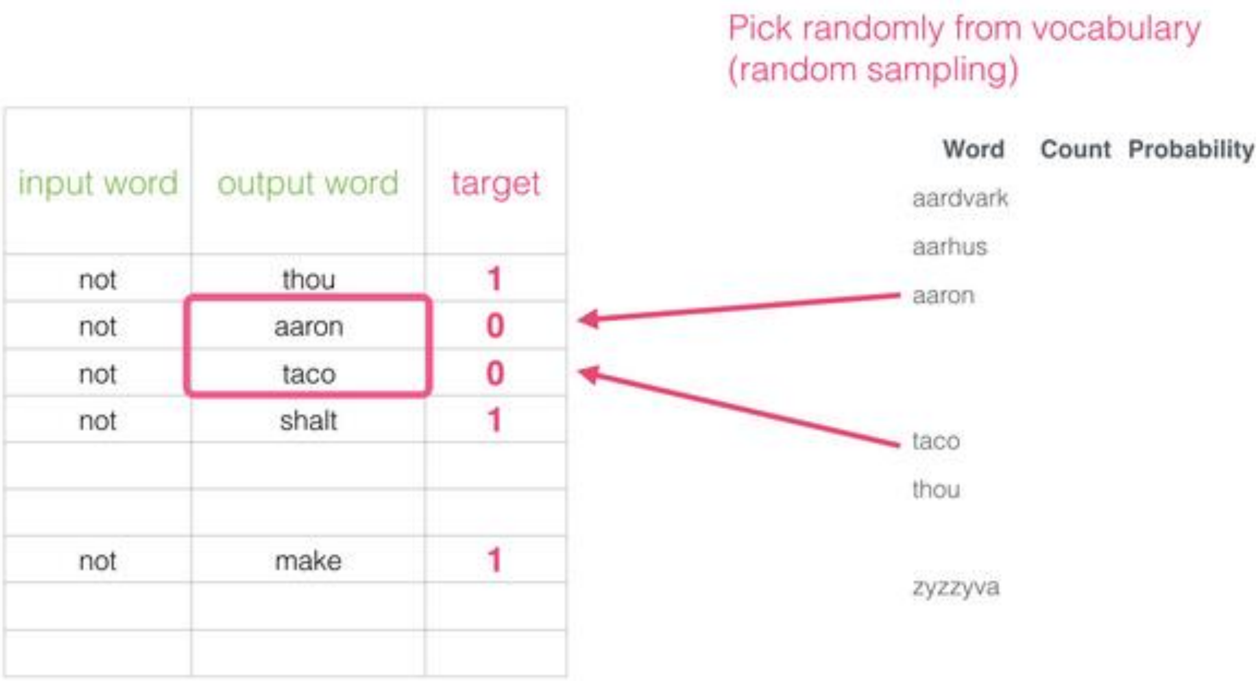
- To a binary classification task:



https://jalammar.github.io/illustrated-word2vec/

Learning word embeddings: word2vec

- The model is trained using neighbouring context words as positive examples and sampled non-neighbour words as negative examples



https://jalammar.github.io/illustrated-word2vec/

Learning word embeddings: word2vec

- Binary classification task: does word *w* co-occur with context word *c*

$$P(+|w,c) = \sigma(c \cdot w) = \frac{1}{1+\exp(-c \cdot w)}$$

$$\begin{aligned} P(-|w,c) &= 1 - P(+|w,c) \\ &= \sigma(-c \cdot w) = \frac{1}{1+\exp(c \cdot w)} \end{aligned}$$

Jurafsky and Martin (2020) Ch. 6

Learning word embeddings: word2vec

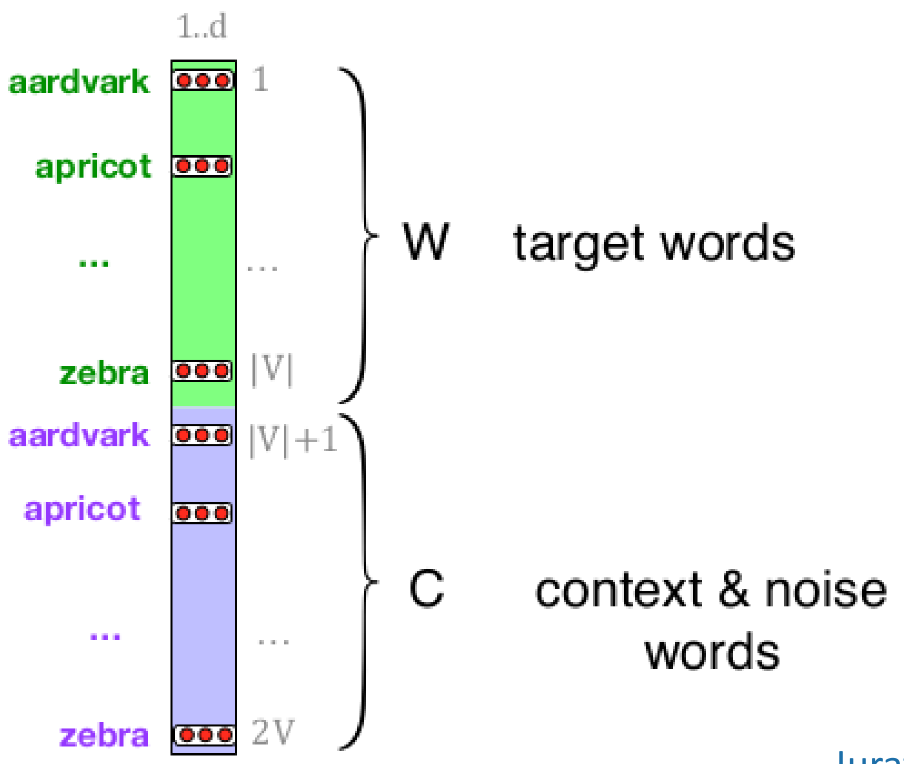
- Loss function with one positive and *k* negative examples:

$$-\left[\log \sigma(c_{pos} \cdot w) + \sum_{i=1}^{k} \log \sigma(-c_{neg_i} \cdot w)\right]$$

Jurafsky and Martin (2020) Ch. 6

Learning word embeddings: word2vec

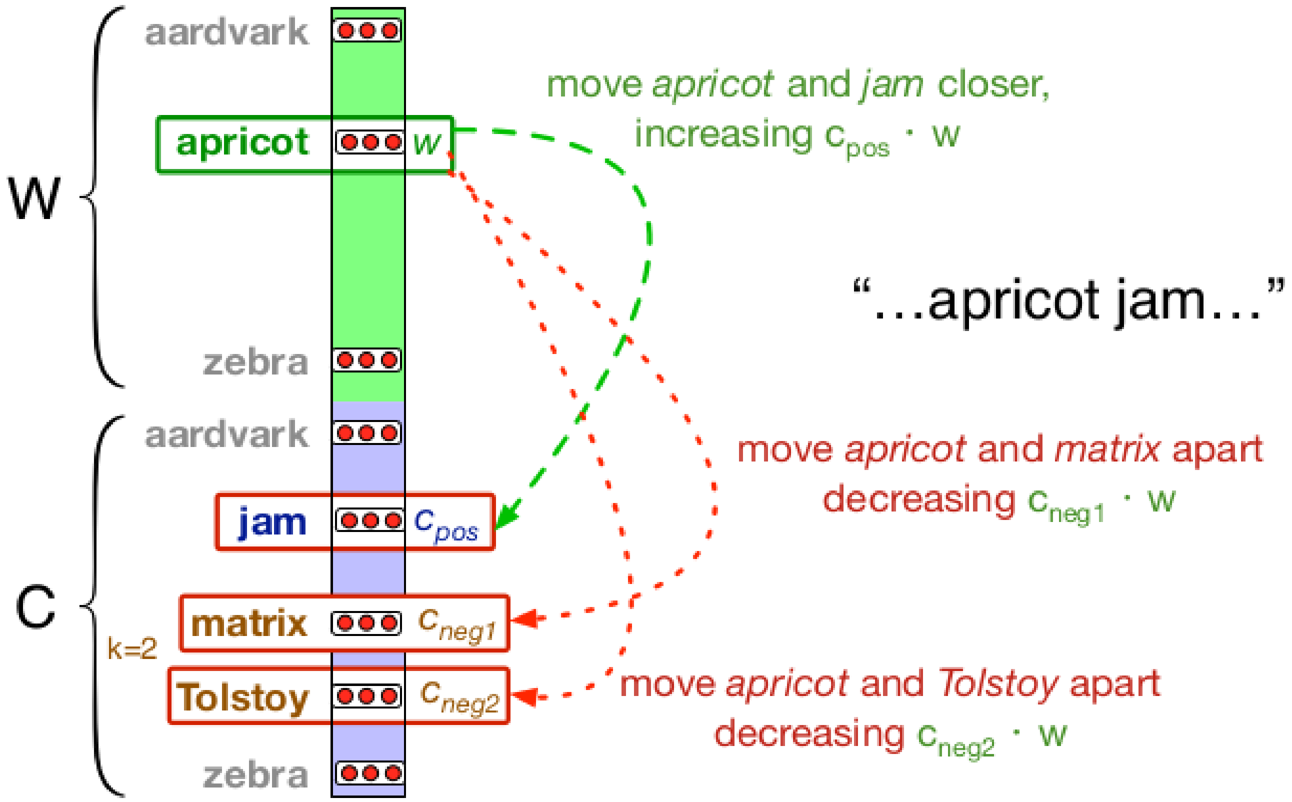
- Learns 2 vectors for each word:



Jurafsky and Martin (2020) Ch. 6

Learning word embeddings: word2vec

- Training intuition:



Jurafsky and Martin (2020) Ch. 6

Word Embedding Inference

king – man + woman ~= queen

Czech + currency	Vietnam + capital	German + airlines	Russian + river
koruna	Hanoi	airline Lufthansa	Moscow
Check crown	Ho Chi Minh City	carrier Lufthansa	Volga River
Polish zolty	Viet Nam	flag carrier Lufthansa	upriver
CTK	Vietnamese	Lufthansa	Russia

Mikolov et al. (2013)

Word-based Classification

- Window-based features: the word embeddings corresponding to relative position in the text are fed as input to the FFNN
- Continuous bag of words: all the embeddings *v(f)* in a variable-length context are averaged:

$$\mathrm{CBOW}(f_1, \ldots, f_k) = \frac{1}{k} \sum_{i=1}^{k} v(f_i).$$

- This can be extended to a weighted average with weights *a*:

$$\mathrm{WCBOW}(f_1, \ldots, f_k) = \frac{1}{\sum_{i=1}^{k} a_i} \sum_{i=1}^{k} a_i v(f_i).$$

Goldberg (2017) Ch. 8

Sentiment Analysis

https://stanford-cs221.github.io/autumn2019/assignments/sentiment/index.html

Sentiment Analysis

- Extract content words
- Map to word vectors
- Average (CBoW)
- Feed through a feed-forward layer
- Binary classification (positive or negative) with sigmoid output layer

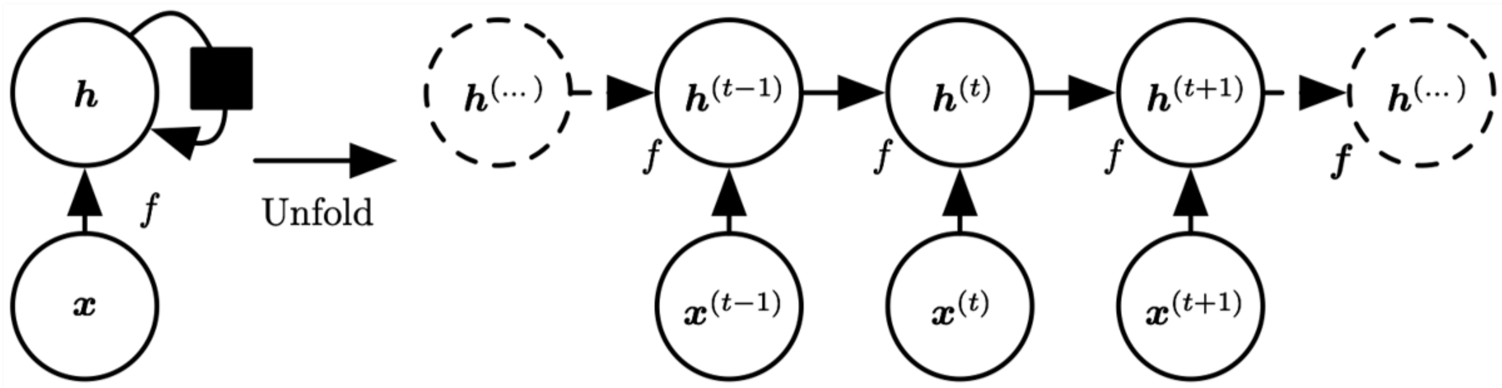
funny
whimsical
delightful
Fault
is
movies
engages
minds
ages

hard
generate
sense
warmth
plot
points
feel
coldly
calculated
doesn't
help
musical
numbers
pedestrian

2. Sequence Processing

Recurrent Neural Networks (RNNs)

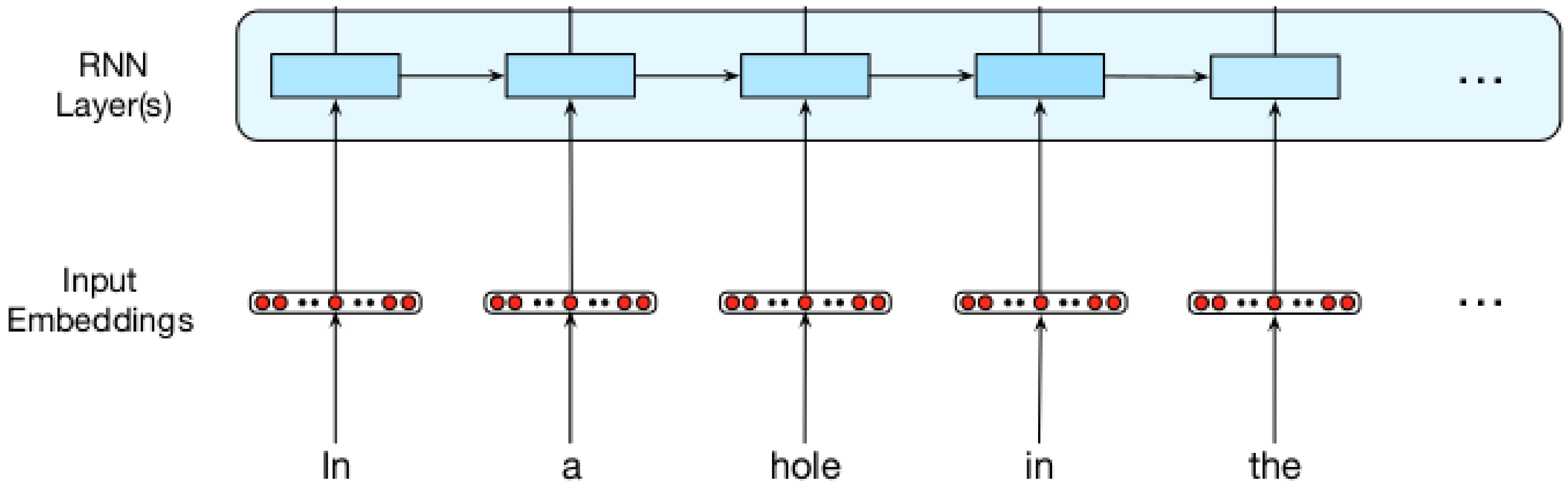
- Neural network that processes a sequence one time step at a time, perform same computation at each timestep (recurrently)
- Represent with unfolding computation graphs



Goodfellow (2016) Ch. 10

Recurrent Neural Networks (RNNs)

- An RNN can read a sequence of words:



Jurafsky and Martin (2020) Ch. 9

Recurrent Neural Networks (RNNs)

- RNN Computation at each time step:

$$e_t = E^T x_t$$ Embedding layer

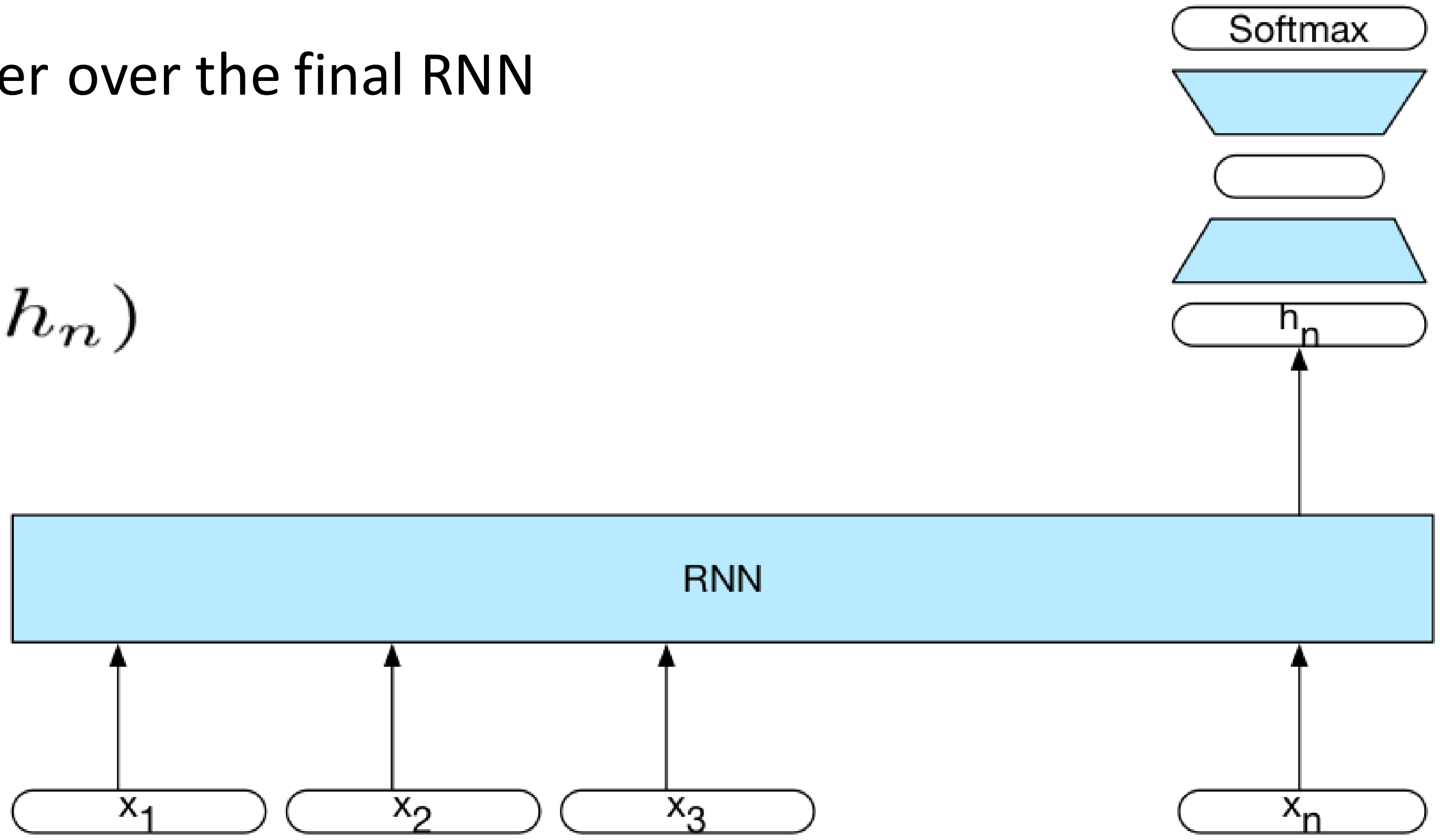
$$h_t = g(Uh_{t-1} + We_t)$$ Recurrent layer

- Each RNN step computes a new hidden state using the previous state and a new input
- Parameters are shared (tied) across all time steps
- *g* is a non-linearity (usually tanh)

RNN Applications: 1. Sequence to label

- Place a classification layer over the final RNN hidden state

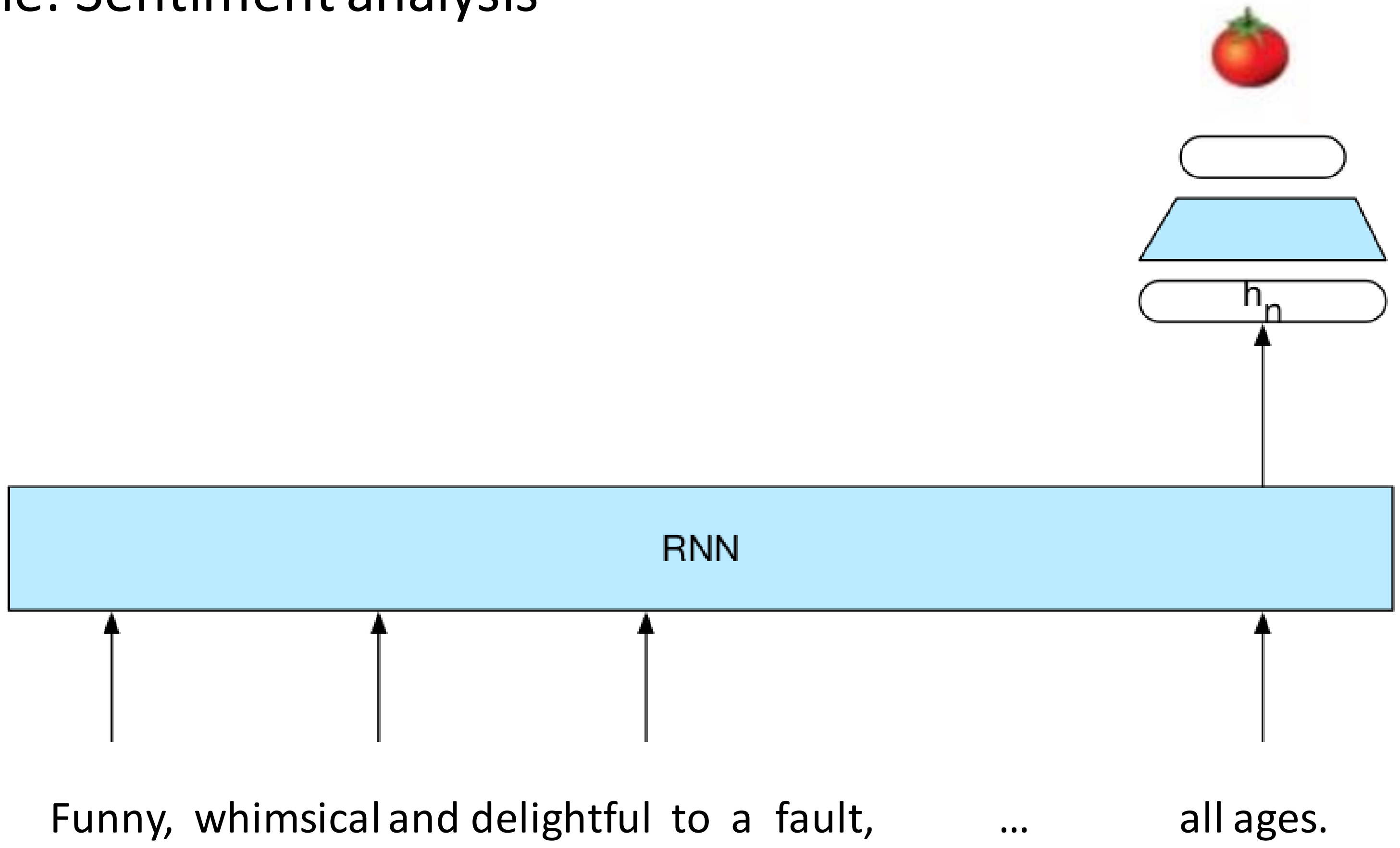
$$y = \mathrm{softmax}(Vh_n)$$



Jurafsky and Martin (2020) Ch. 9

RNN Applications: 1. Sequence to label

- Example: Sentiment analysis



RNN Applications: 2. Language Modelling

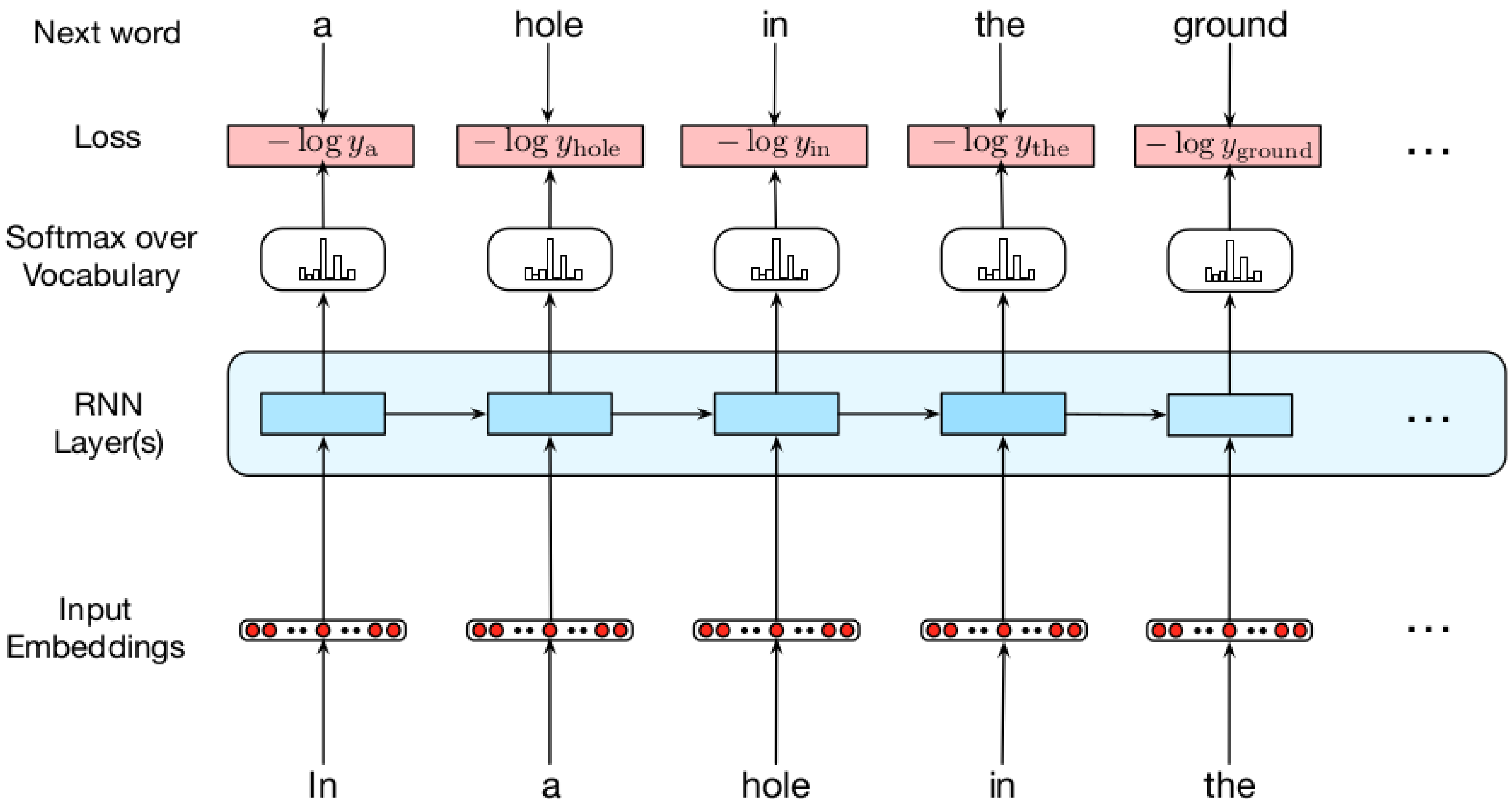
- Assign probability to each word in a sequence: Condition on all preceding words

$$P(w_{1:n}) = \prod_{i=1}^{n} P(w_i | w_{1:i-1})$$
$$= \prod_{i=1}^{n} y^i_{w_i}$$

$$y_t = \text{softmax}(V h_t)$$

Jurafsky and Martin (2020) Ch. 9

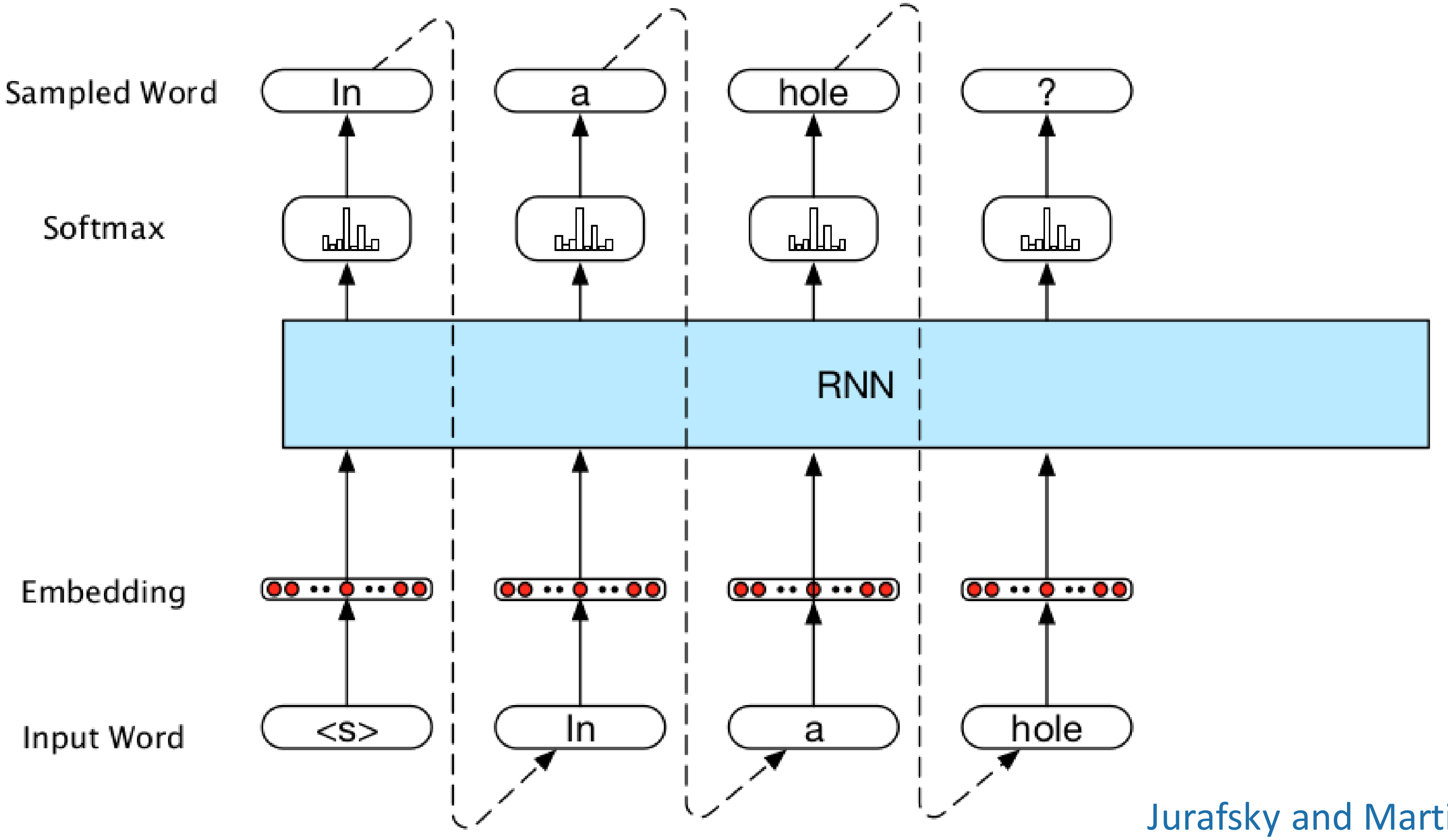
RNN Applications: 2. Language Modelling



Jurafsky and Martin (2020) Ch. 9

RNN Applications: 2. Language Modelling

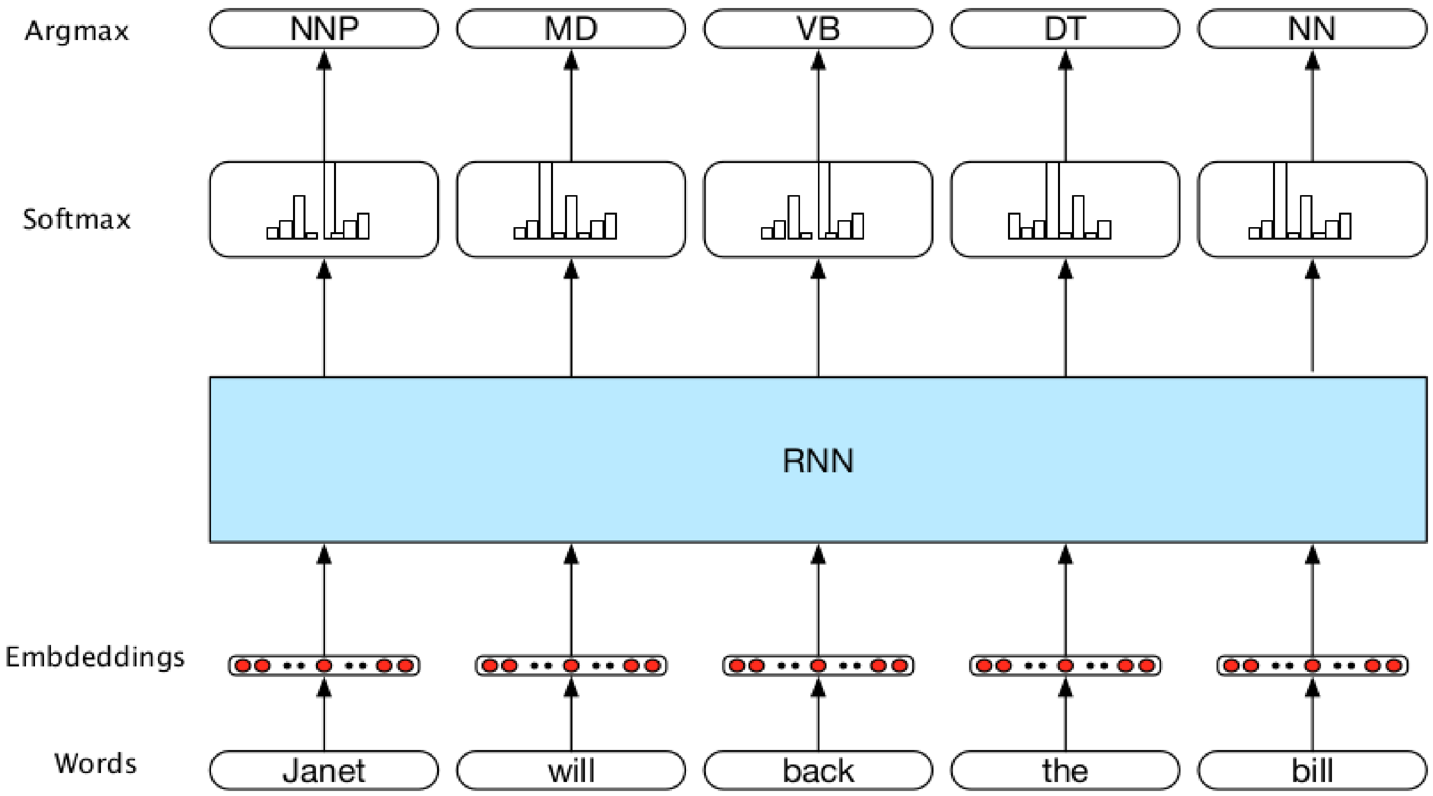
- Generating text from a language model:



Jurafsky and Martin (2020) Ch. 9

RNN Applications: 3. Sequence Labelling

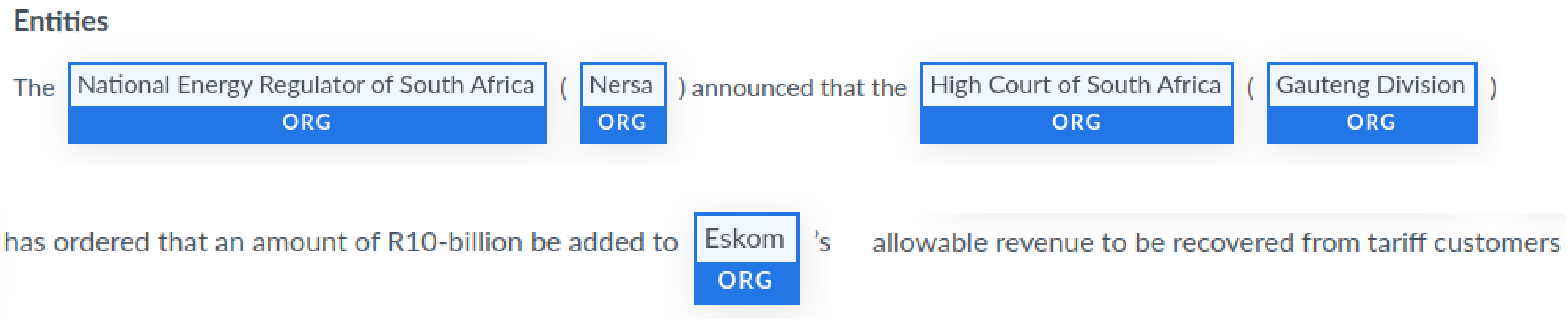
- Sequence to sequence with the same length
- Example: Parts-of-Speech Tagging



Jurafsky and Martin (2020) Ch. 9

RNN Applications: 3. Sequence Labelling

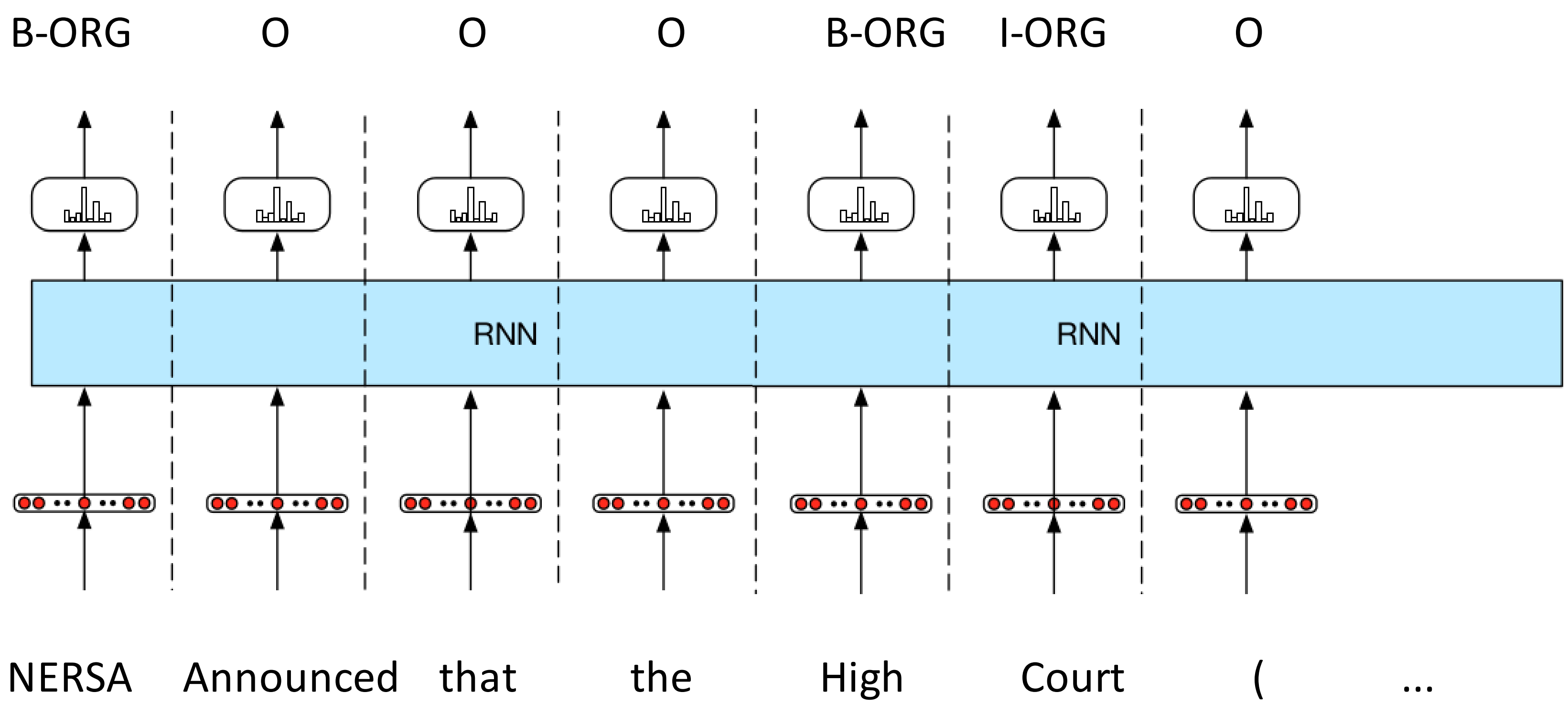
- Example: Named Entity Recognition



https://demo.allennlp.org/named-entity-recognition/named-entity-recognition

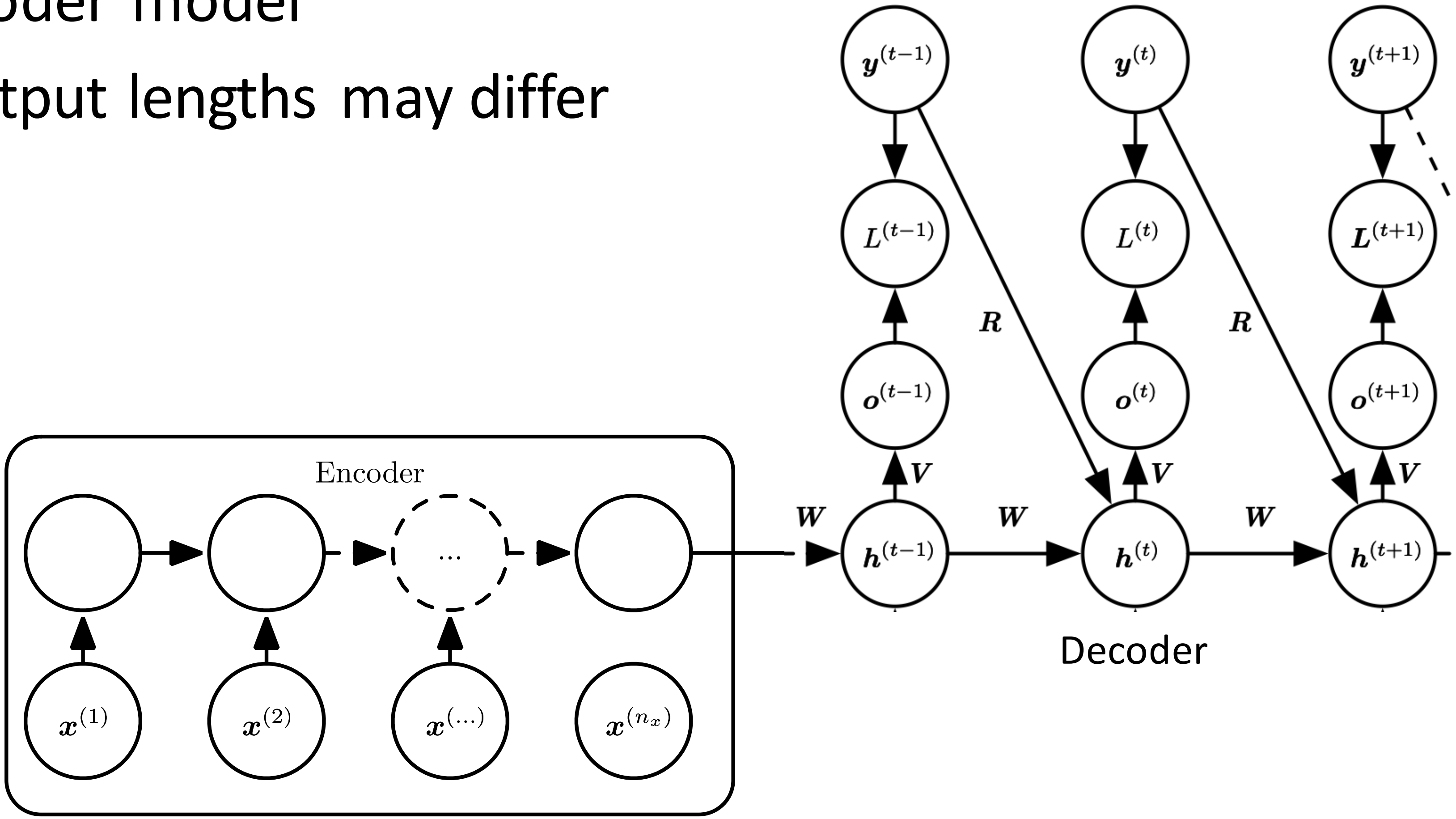
RNN Applications: 3. Sequence Labelling

- Example: Named Entity Recognition



RNN Applications: 4. Sequence to Sequence

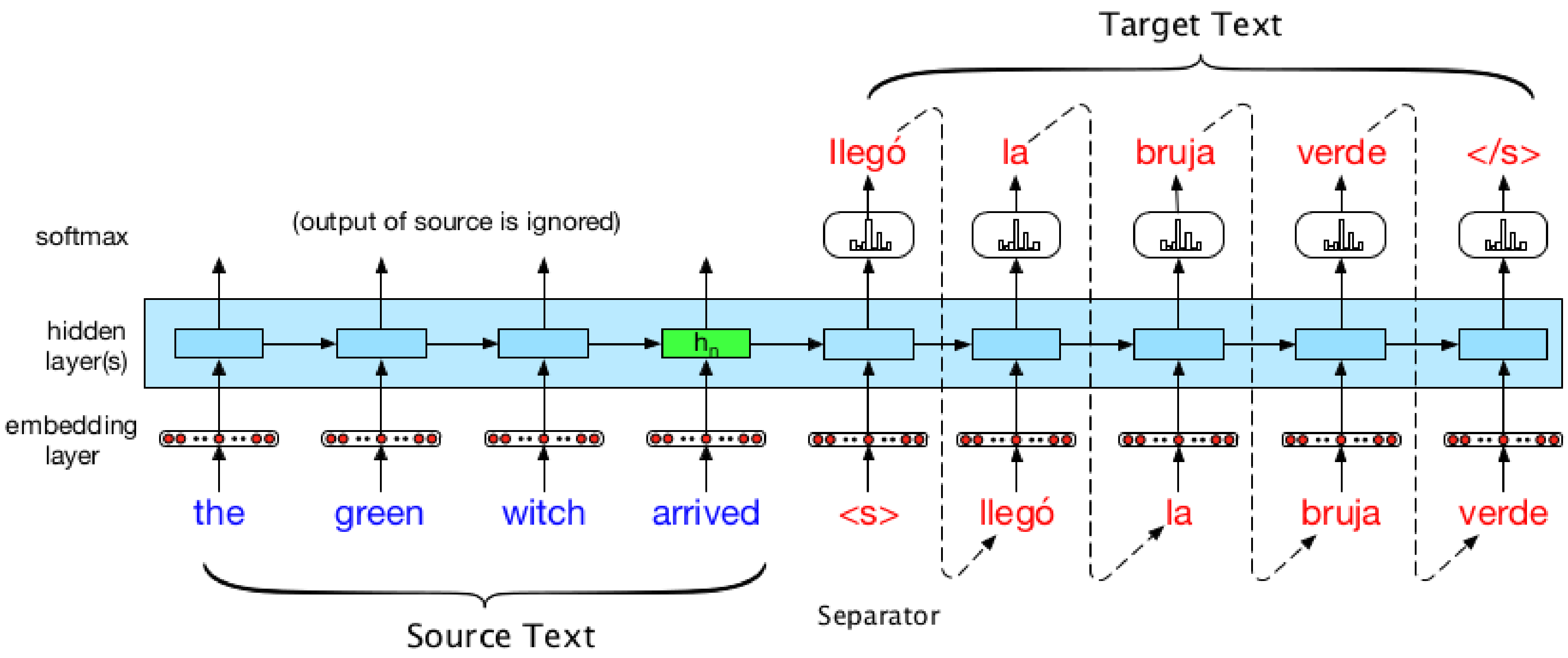
- Encoder-decoder model
- Input and output lengths may differ



Goodfellow (2016) Ch. 10

RNN Applications: 4. Sequence to Sequence

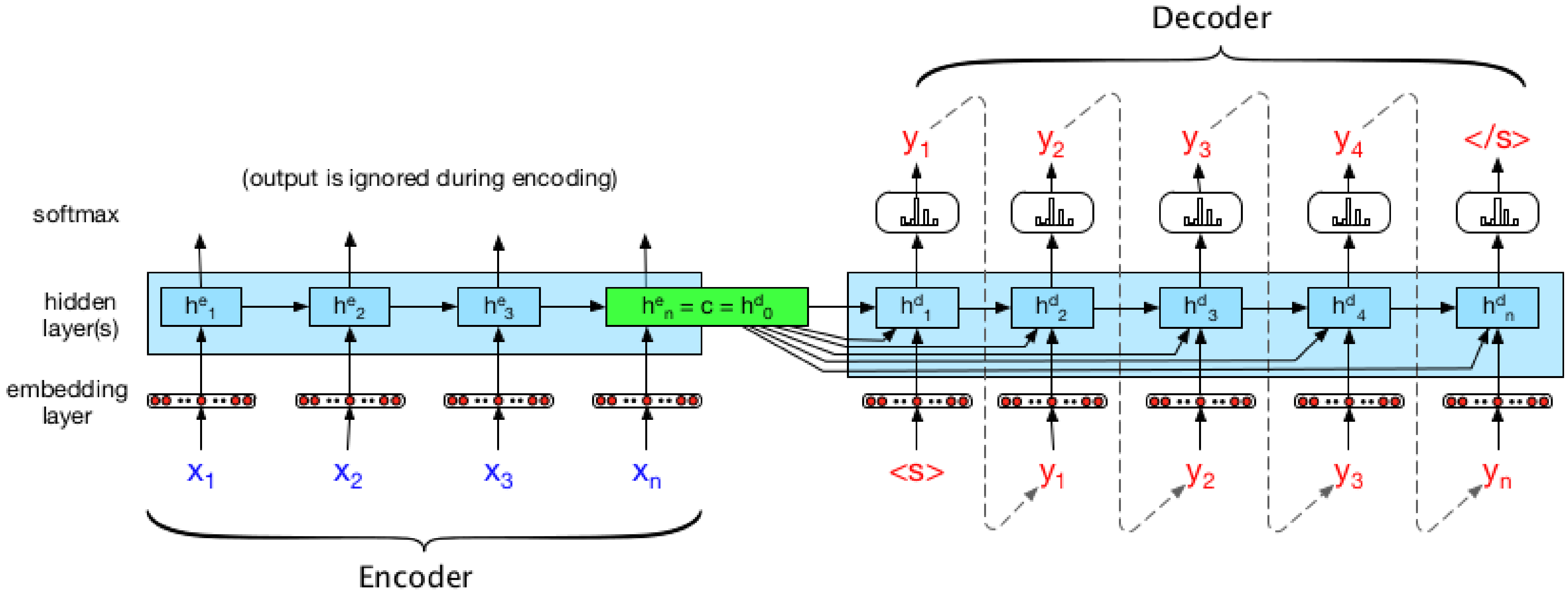
- Example: Machine Translation



Jurafsky and Martin (2020) Ch. 11

RNN Applications: 4. Sequence to Sequence

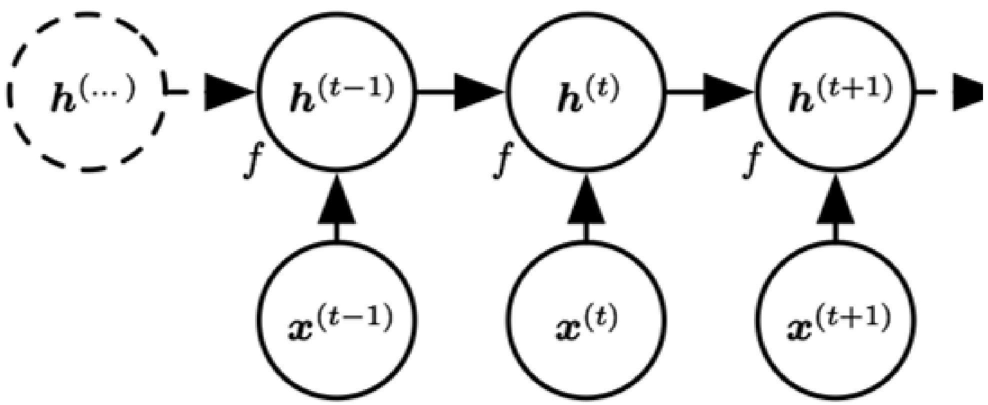
- Re-use the context vector from the encoder at each step in the decoder
- Train with teacher forcing



Jurafsky and Martin (2020) Ch. 11

RNN Training: Backpropagation through time

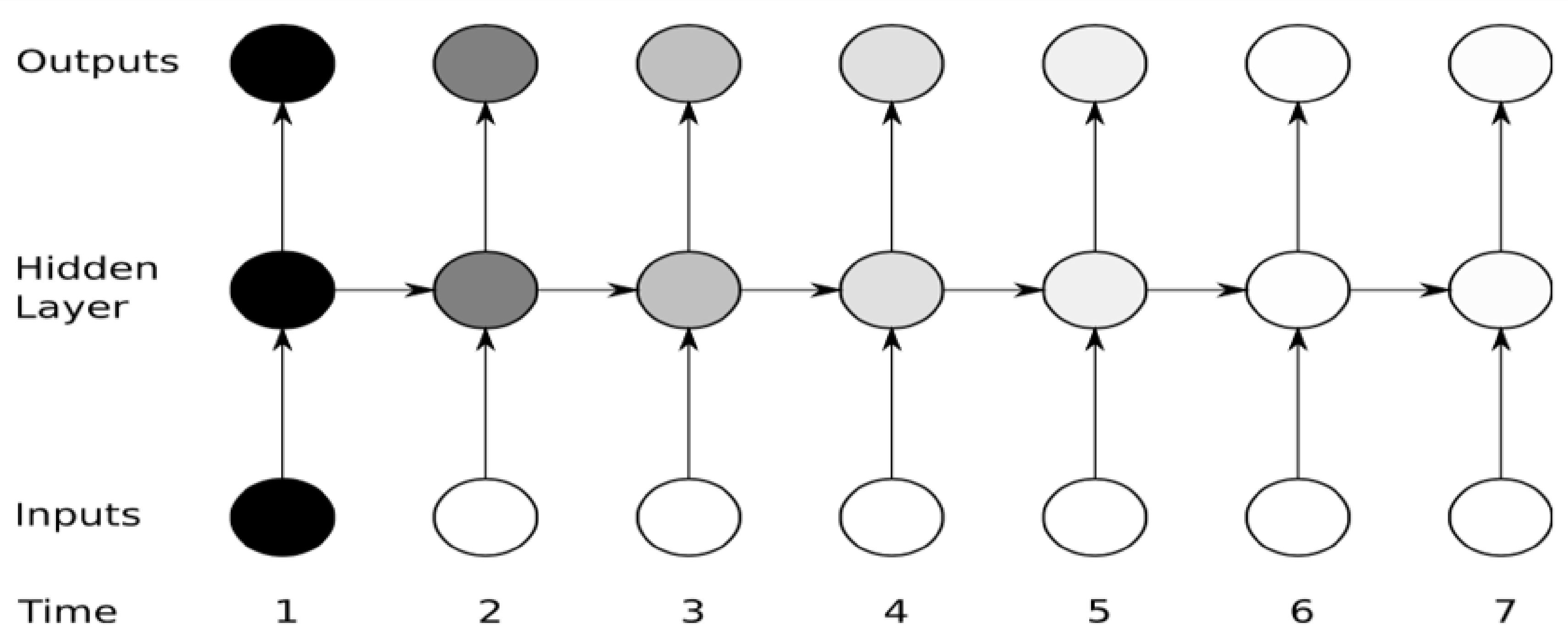
- Unroll the computation graph so that the units at all time steps repeat exactly the same parameters
- Backpropagate the parameters at each unit as if they were different parameters
- Update the parameters by averaging the gradients over all the units in the chain



Goodfellow (2016) Ch. 10

RNN Training: Vanishing gradients problem

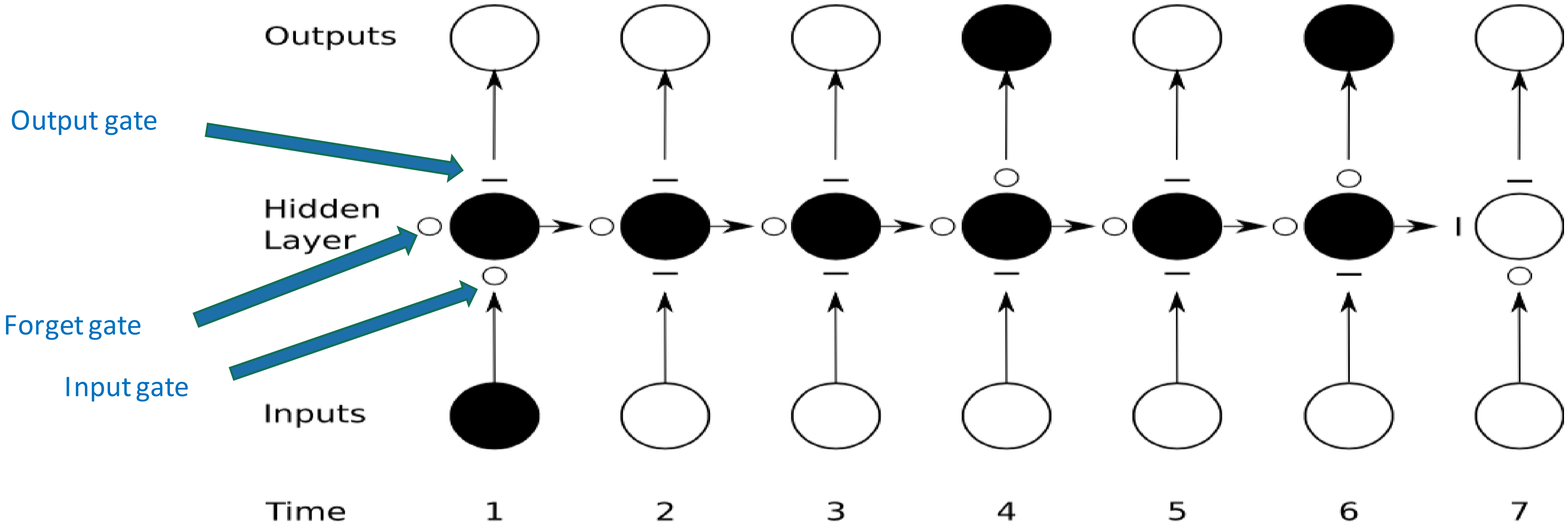
- The sensitivity of nodes to an input at one time step decreases over time
- As new inputs overwrite the activations of the hidden layer the network "forgets" earlier inputs



Example from Graves (2012)

RNN Training: Vanishing gradients problem

- Solution: Use gates to control the flow of information across time steps and between the input, hidden, and output layers



Slide from Yejin Choi

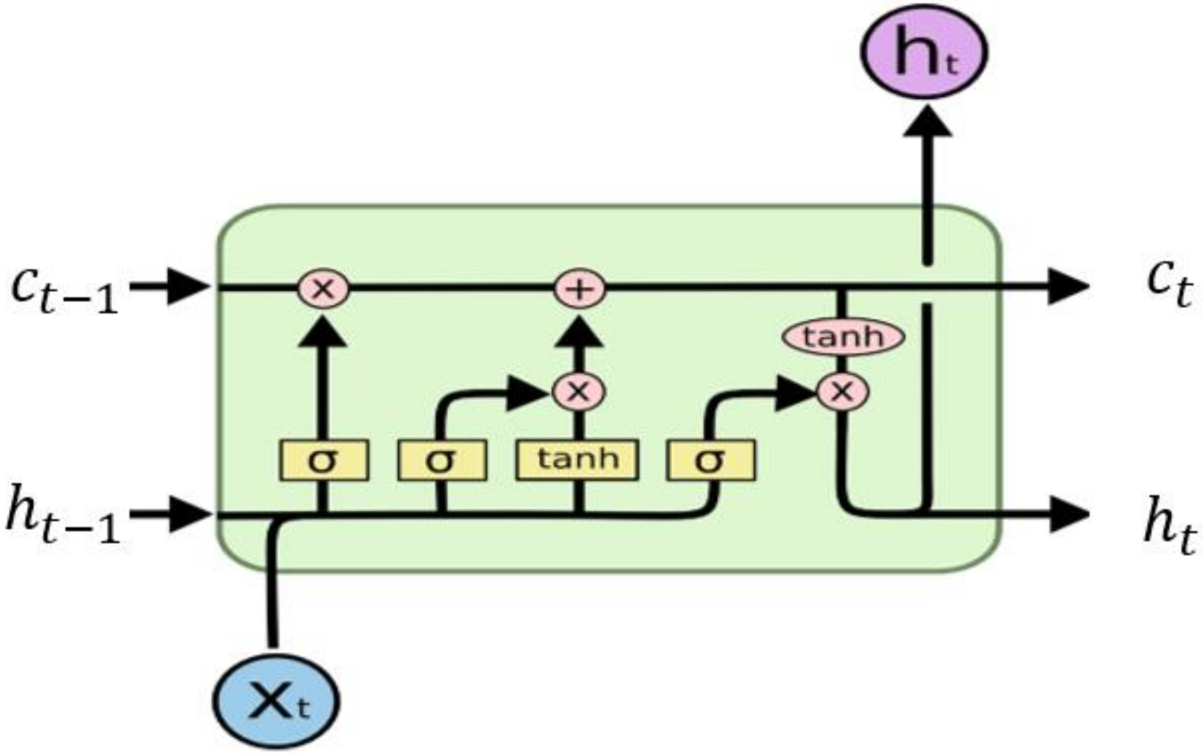
Long Short-Term Memory Networks (LSTMs)

- Replaces the RNN cell with a more complex calculation:

Input gate	$i_t = \sigma(U^{(i)} x_t + W^{(i)} h_{t-1} + b^{(i)})$
Forget gate	$f_t = \sigma(U^{(f)} x_t + W^{(f)} h_{t-1} + b^{(f)})$
Output gate	$o_t = \sigma(U^{(o)} x_t + W^{(o)} h_{t-1} + b^{(o)})$
	$\tilde{c}_t = \tanh(U^{(c)} x_t + W^{(c)} h_{t-1} + b^{(c)})$
Memory state	$c_t = f_t \circ c_{t-1} + i_t \circ \tilde{c}_t$
Hidden state	$h_t = o_t \circ \tanh(c_t)$

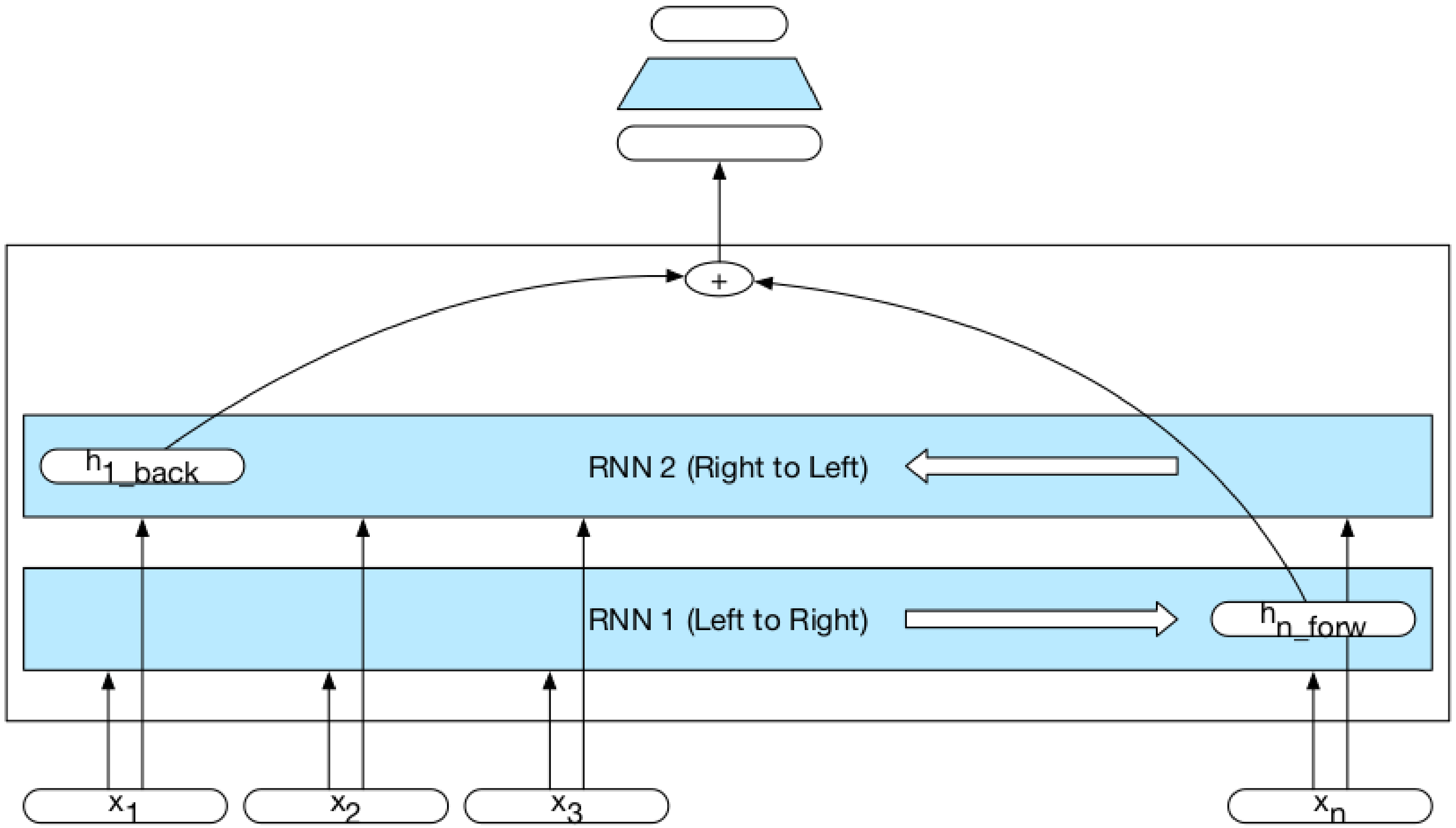
Slide from Yejin Choi

Long Short-Term Memory Networks (LSTMs)



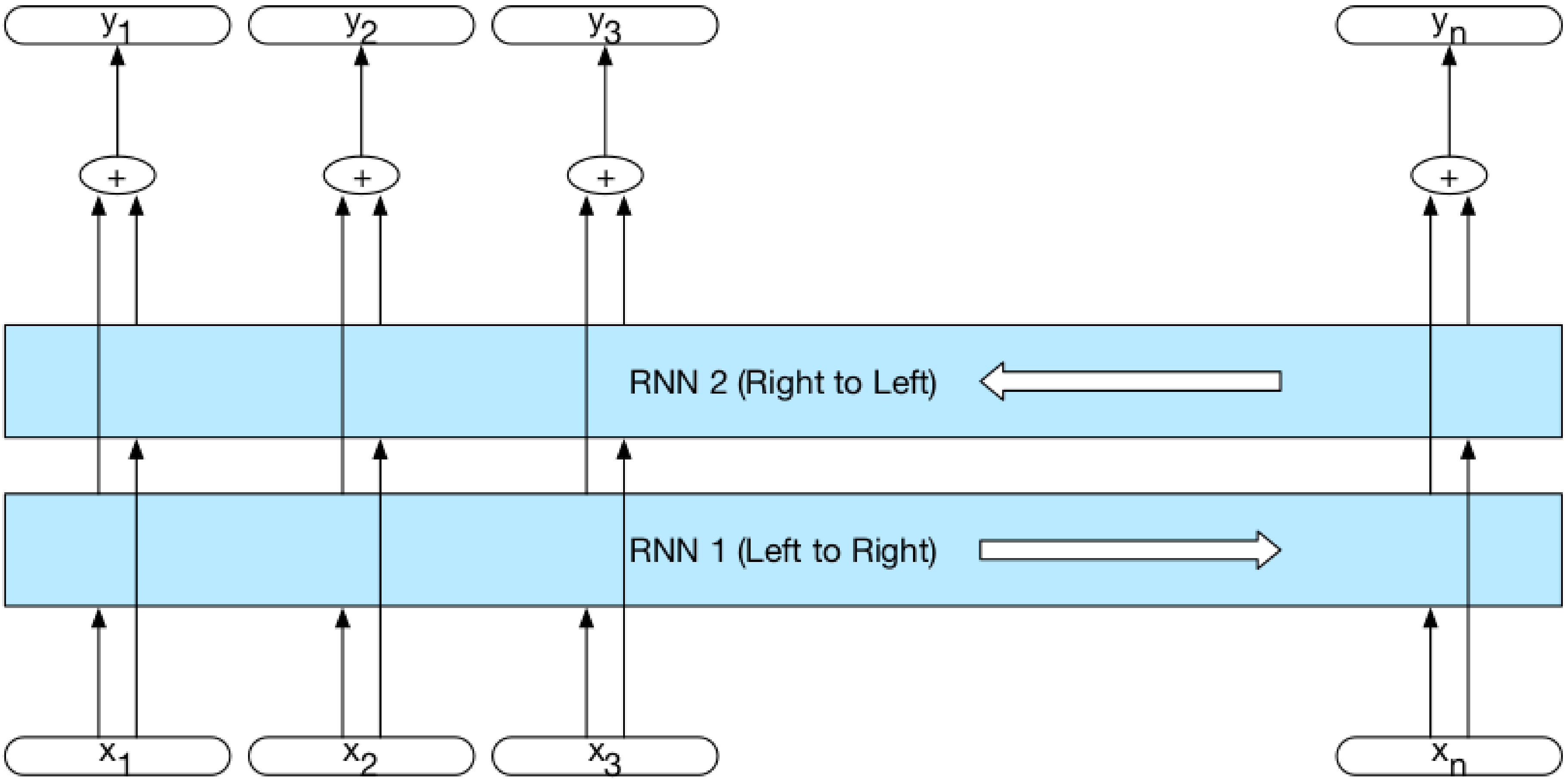
https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Bidirectional RNNs



Jurafsky and Martin (2020) Ch. 9

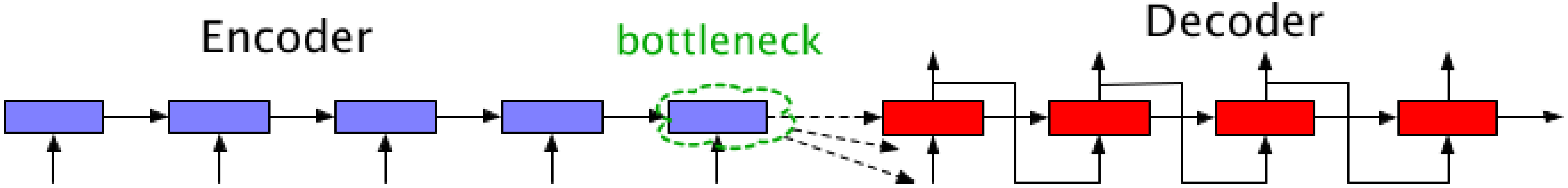
Bidirectional RNNs



Jurafsky and Martin (2020) Ch. 9

Attention

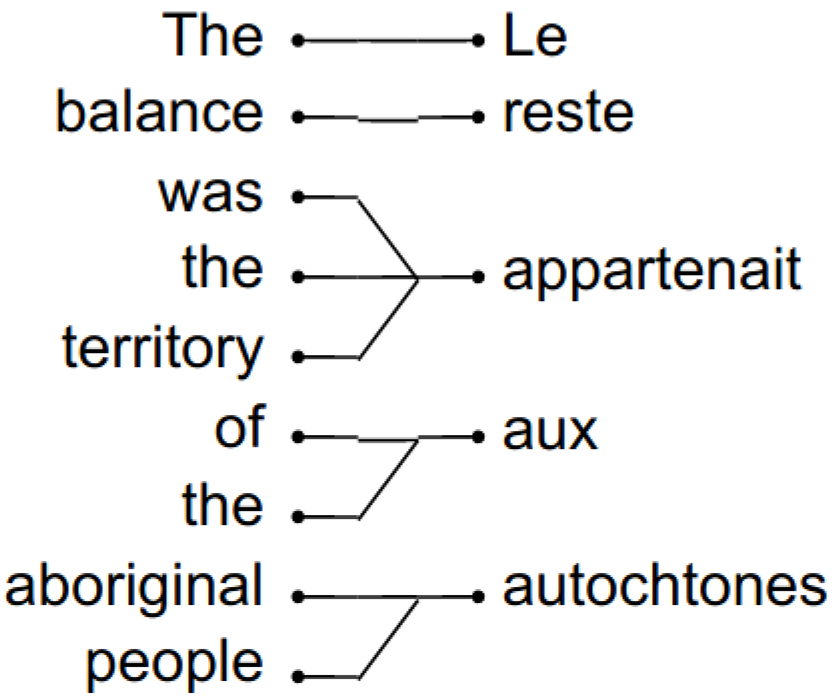
- Even with LSTMs and bidirectional encoders, sequence-to-sequence models still have a bottleneck limiting their capacity

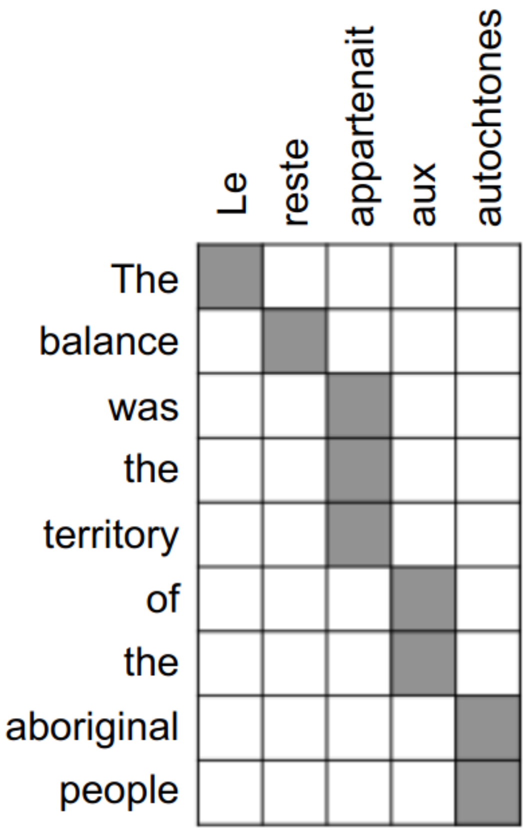


Jurafsky and Martin (2020) Ch. 11

Attention

- Word alignments for Machine Translation

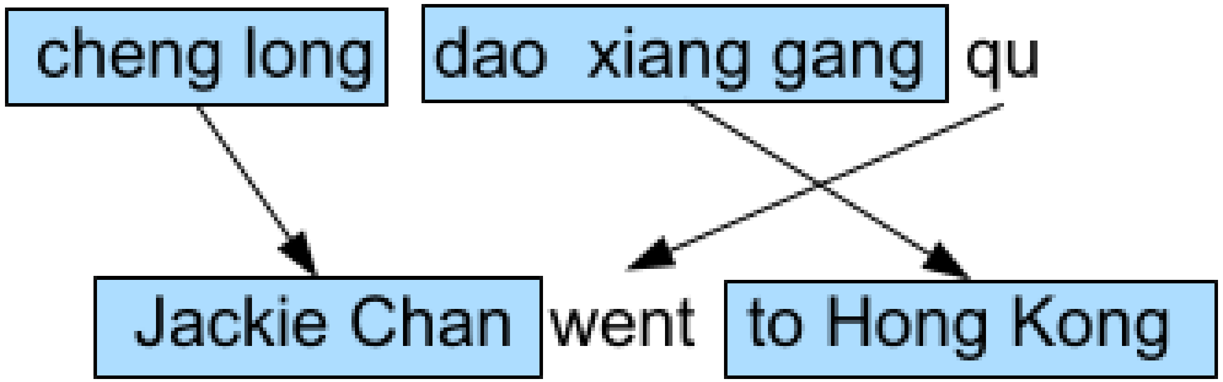




Example from Luong, Cho and Manning (2016)

Attention

- Phrase alignments for Machine Translation

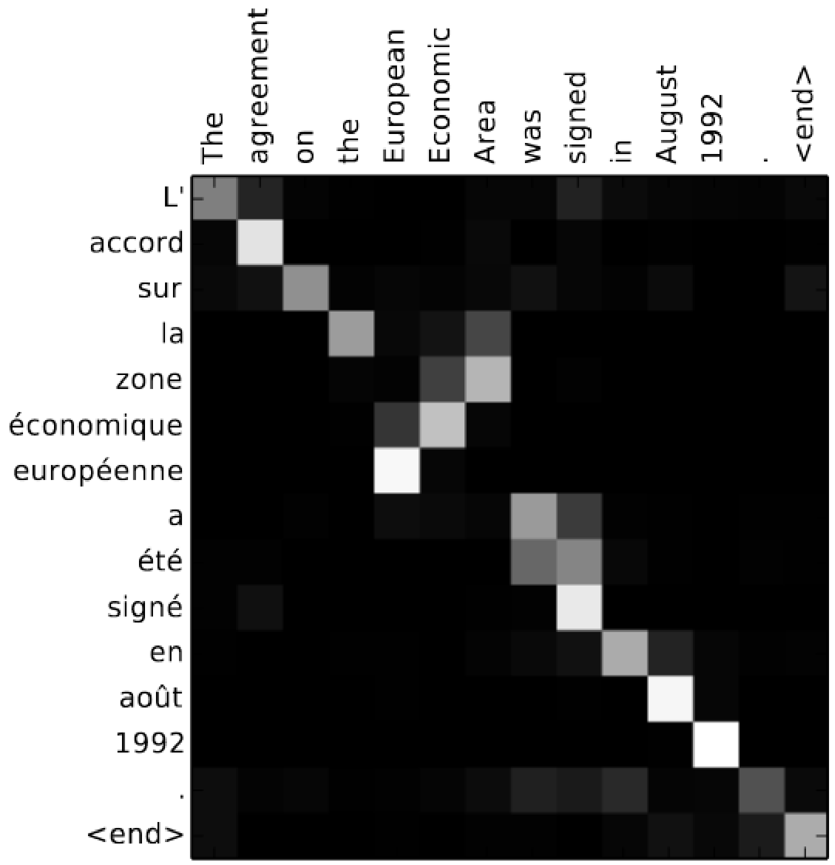


Attention

- Let a neural network predict the alignment between the input and output tokens:

$$z_j = \tanh([s_i^t, s_j^s]W + b)$$
$$j = \text{argmax}_j z_j$$

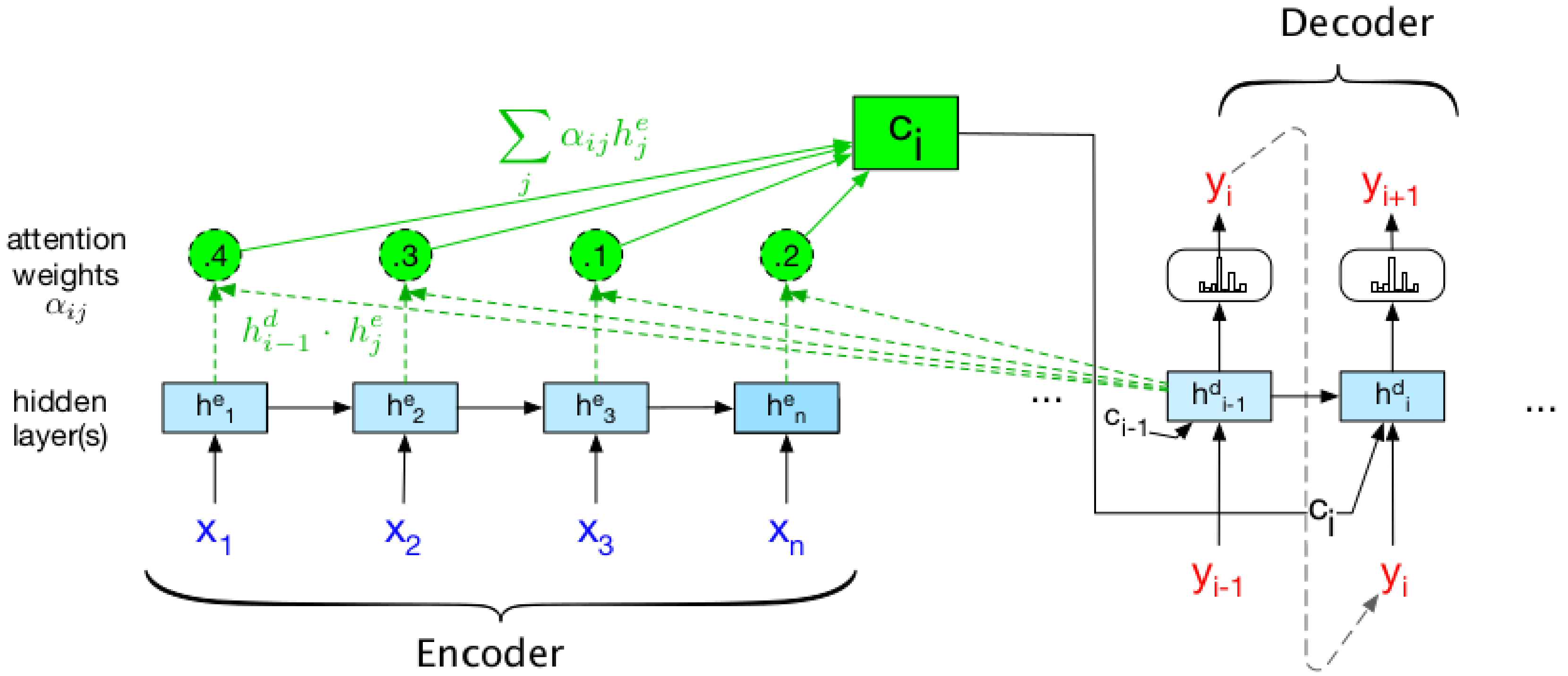
  - Output at position *i* is aligned to input at position *j*



Bahdanau et al. (2015)

Attention

- An attention-based context vector is computed at each output step and fed into the decoder



Jurafsky and Martin (2020) Ch. 11

Attention

- Compute a soft alignment *z* between the input and output
- Compute a weighted average of the encoder hidden states, where the weights are the alignment probabilities α
- Integrate the context vector into the decoder and train jointly

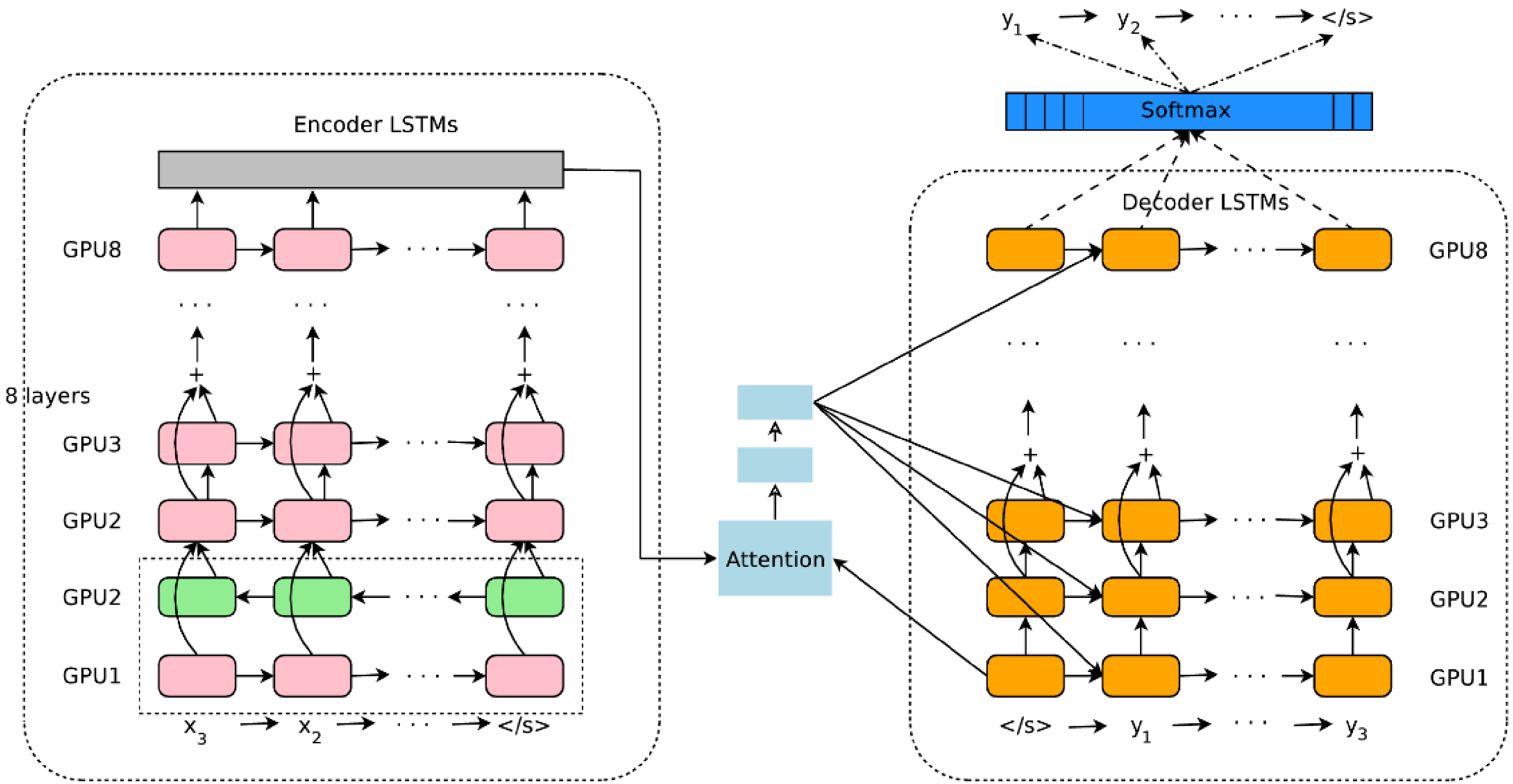
$$z_j = \tanh([s_i^t, s_j^s]W + b)$$

$$\alpha = \mathrm{softmax}(z)$$

$$c = \sum_j \alpha_j s_j^s$$

Attention

- Google's Neural Machine Translation System (2016)



3. Transformers and contextualized representations

Transformers

Disadvantages of RNN-based models:

- Limited ability to model very long contexts, even when using LSTMs
- Computation cannot be parallelized across time steps, which makes GPU training less efficient

New architecture: Transformers (Vaswani et al., 2017)

Transformers: Attention is all you need

- No recurrence, uses only attention to model interaction between different time steps
- Key idea: *self*-attention among all the elements in a sequence

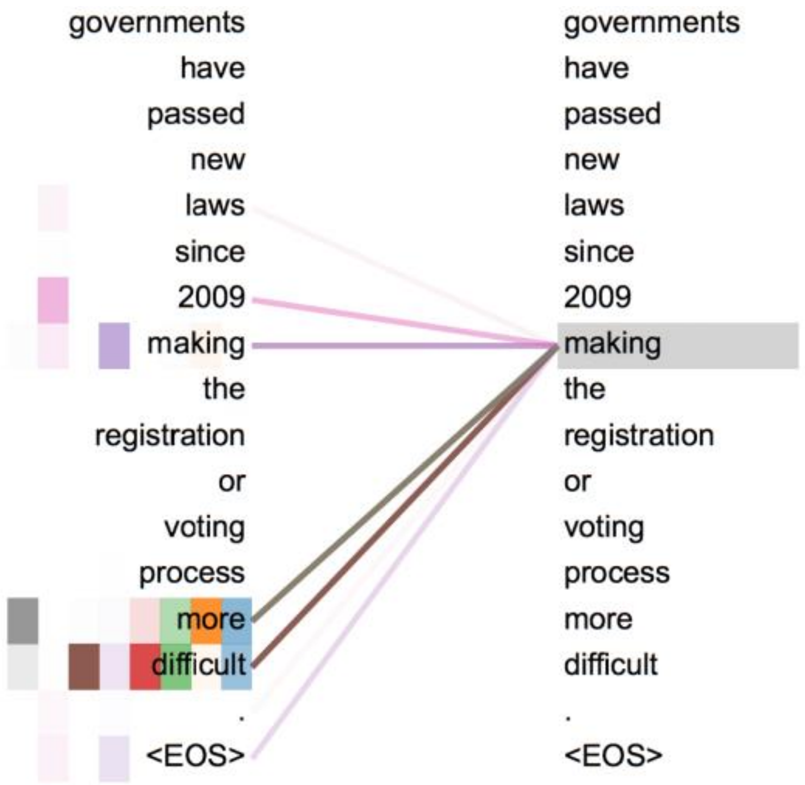


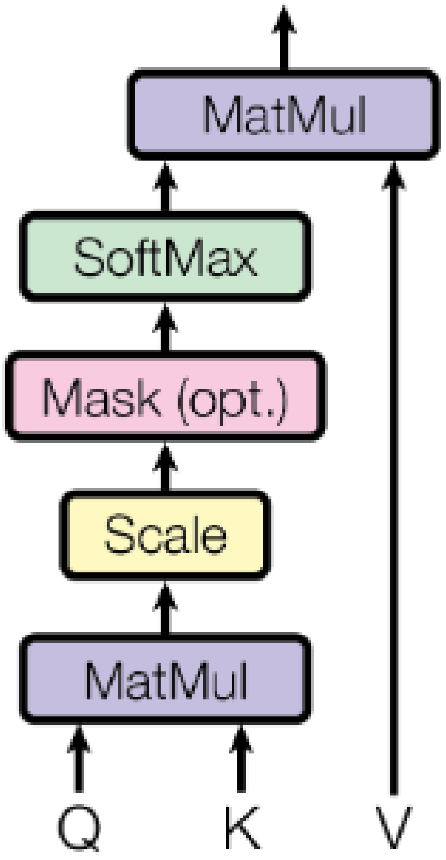
Diagram from Chris Manning

Transformers

Scalar dot-product attention

- Query, Key, Value matrices

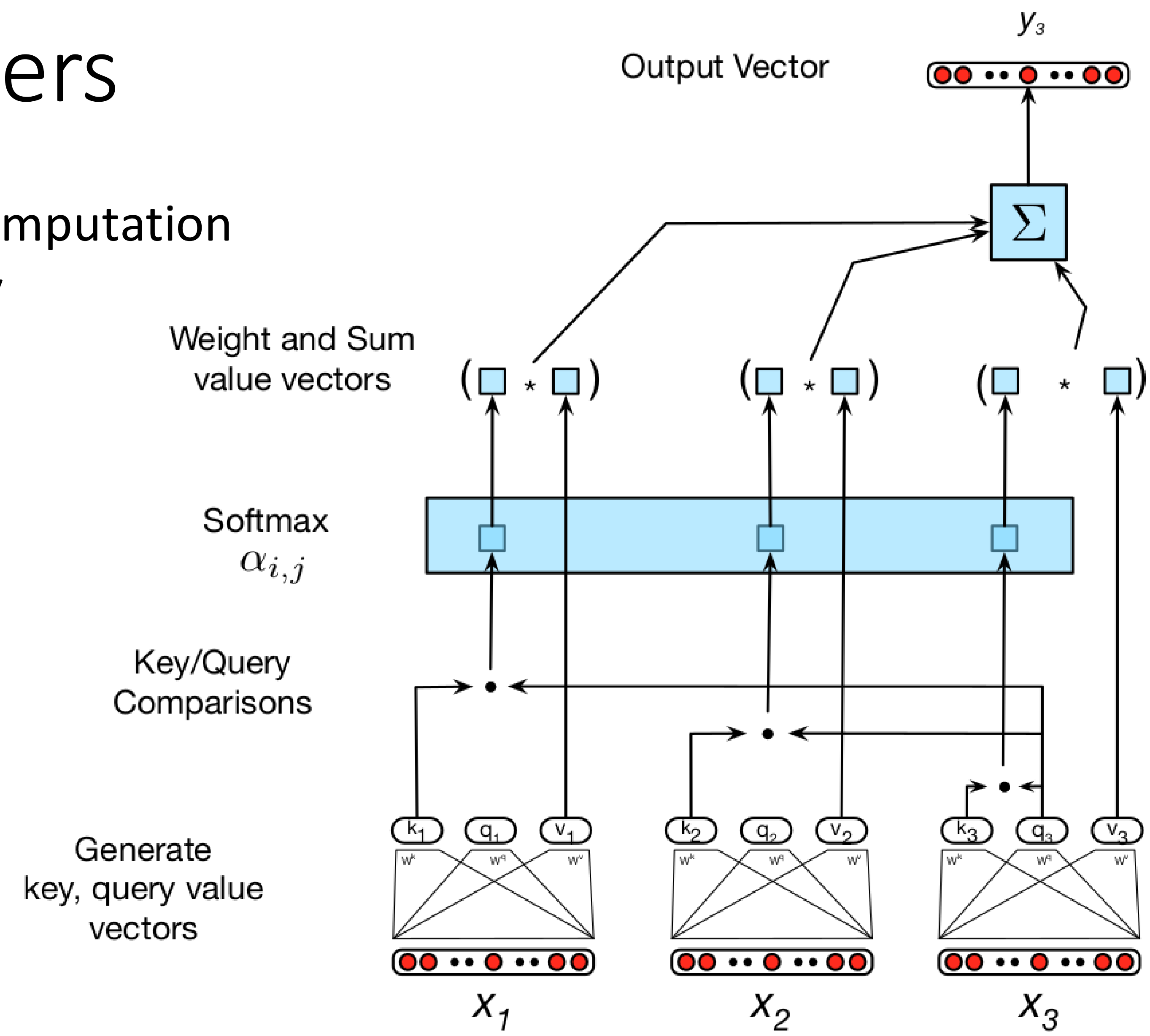
$$\mathrm{Attention}(Q, K, V) = \mathrm{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$



Vaswani et al. (2017)

Transformers

- Self-attention computation using x_3 as query



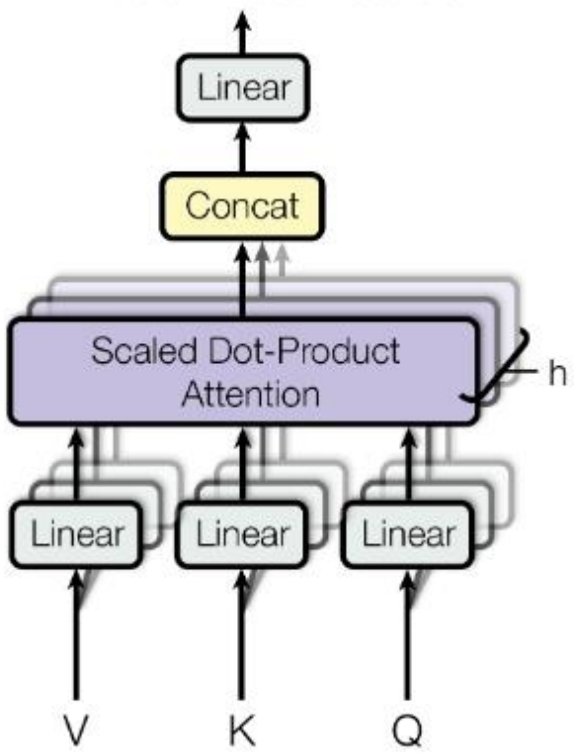
Jurafsky and Martin (2020) Ch. 9

Transformers

Multi-head attention

- Each head has its own parameters

$$\mathrm{MultiHead}(Q, K, V) = \mathrm{Concat}(\mathrm{head_1}, ..., \mathrm{head_h})W^O$$
$$\text{where } \mathrm{head_i} = \mathrm{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$



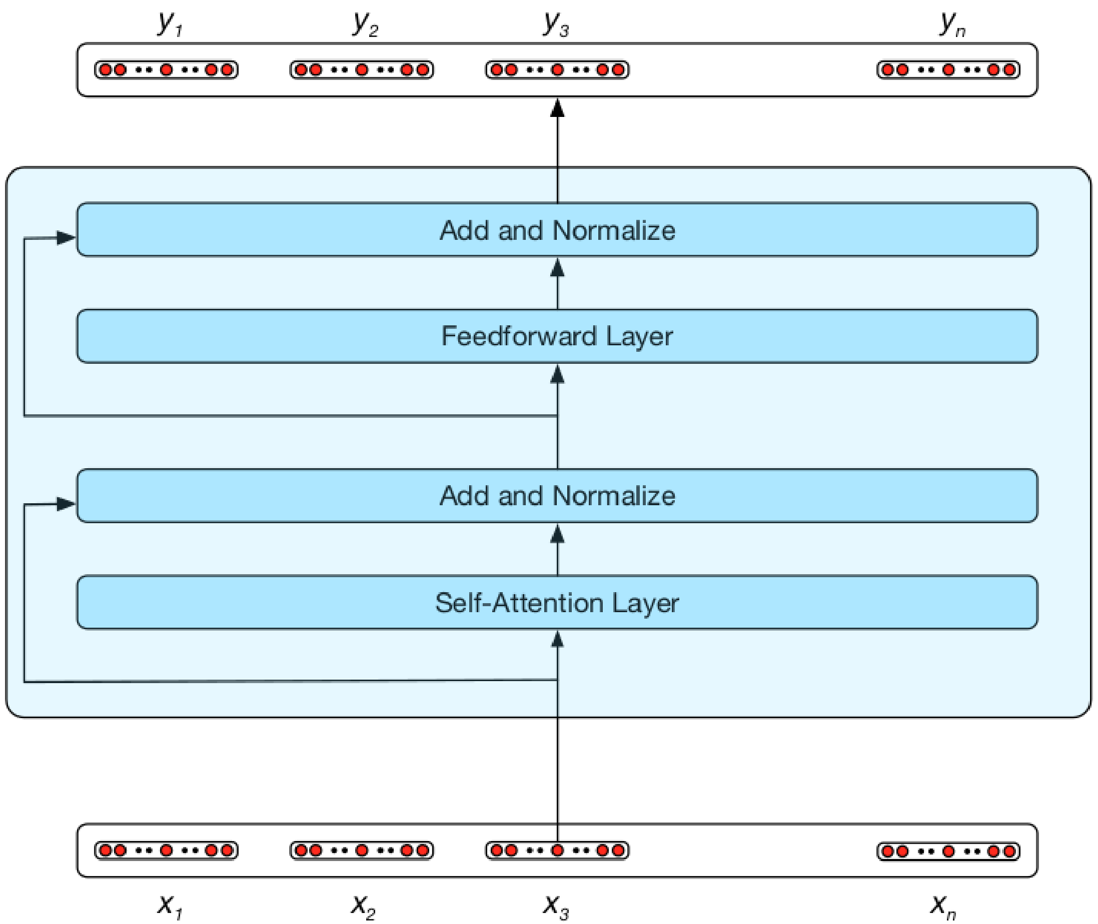
Vaswani et al. (2017)

Transformers

Transformer Block

- Residual connection and layer normalization
- Position-wise Feedforward network

$$\mathrm{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2$$



Jurafsky and Martin (2020) Ch. 9

Transformers

Positional embeddings

- Unlike RNNs, the architecture itself does not encode relative positions
- Add positional embeddings to input
- Embeddings can be learned or predefined based on sin/cos functions

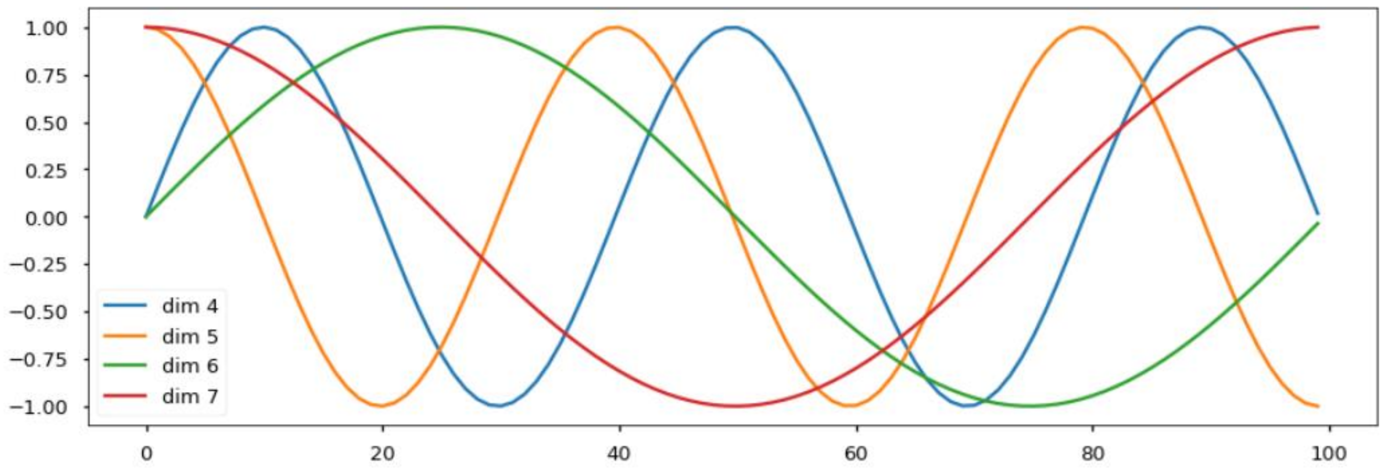
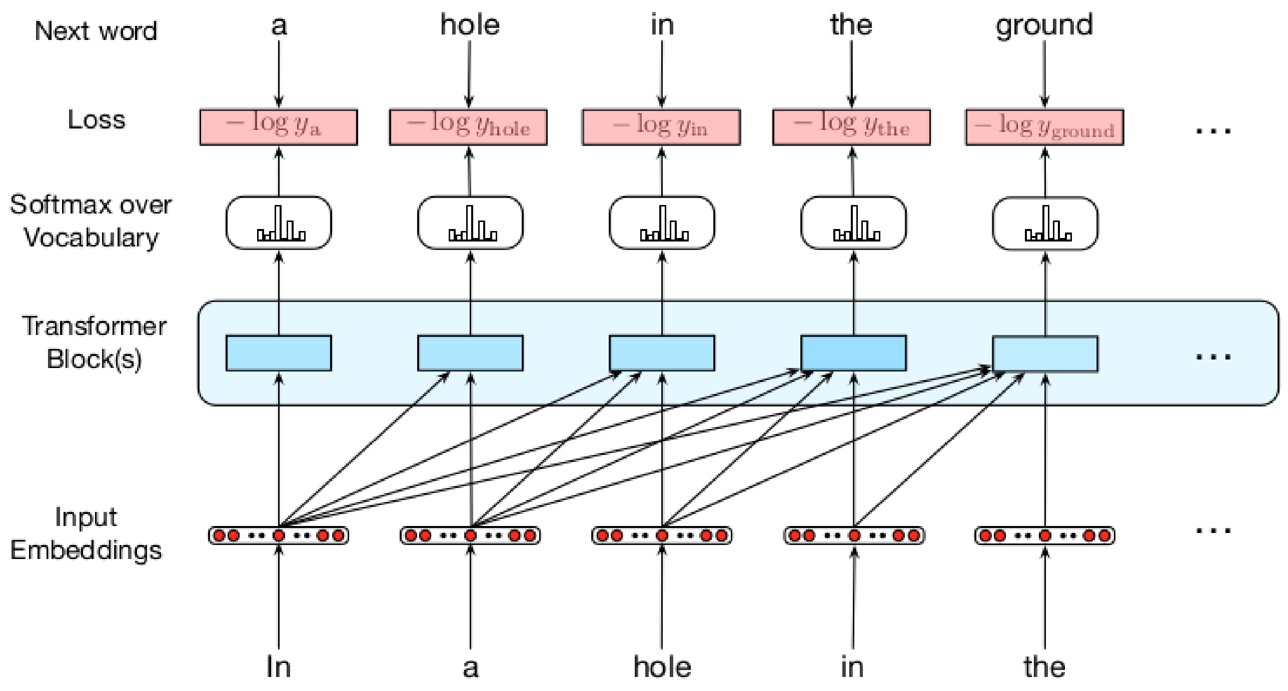


Figure: https://nlp.seas.harvard.edu/2018/04/03/attention.html

Transformers

Transformer Language Model

- "Causal" attention: do not attend to future words (also in decoders)



Jurafsky and Martin (2020) Ch. 9

Transformers

- Application: Summarization

Input document:

Will the litany of bad economic news ever end? The Gauteng High Court has ordered that Eskom can reap another R10-billion in the 2021/22 financial year, which means an effective increase in power prices of more than 15%. Eskom's woes translate into woes for the entire economy of South Africa. The state-run power utility's inability to provide reliable power is a huge obstacle to investment, economic growth and job creation. Then there is the issue of soaring prices for its sputtering service. Prices are about to jump by more than 15% in the coming financial year, adding an additional cost burden to South African industry and consumers just when they are least able to absorb it. "The National Energy Regulator of South Africa (Nersa) announced… that the High Court of South Africa (Gauteng Division) has ordered that an amount of R10-billion be added to Eskom's allowable revenue to be recovered from tariff customers in the 2021/22 financial year," Nersa said in a terse statement on Tuesday. Eskom has long complained that Nersa has awarded it lower increases than it had applied for, worsening a spiralling financial crisis. The upshot is that "this will result in an average tariff percentage increase of 15.63% in the 2021/22 financial year". Inflation in December was running at 3.1%, so such an increase will effectively be five times the current inflation rate. Inflation remains muted against the backdrop of a fragile economy with an unemployment rate well above 40%, based on its most telling definition. But the South African Reserve Bank has warned that administered prices – which include power tariffs – are an upside risk to the inflation outlook. Petrol prices are also bubbling at the moment, with more increases foreseen at the pumps in the coming months. So this has the potential to nip further interest rate cuts in the bud. Then there are the rising power costs to business at a time when one of the many pledges to come out of President Cyril Ramaphosa's administration is to reduce the cost of doing business. That is not about to happen for power-intensive industries such as mining, manufacturing and large-scale commercial agriculture. If there is a light at the end of this tunnel, it can't be switched on, thanks to Eskom.

Text: www.dailymaverick.co.za

Transformers

- Application: Summarization

Output summary:

Eskom’s woes translate into woes for the entire economy of South Africa.

Will the litany of bad economic news ever end?

The Gauteng High Court has ordered that Eskom can reap another R10-billion in the 2021/22 financial year, which means an effective increase in power prices of more than 15%.

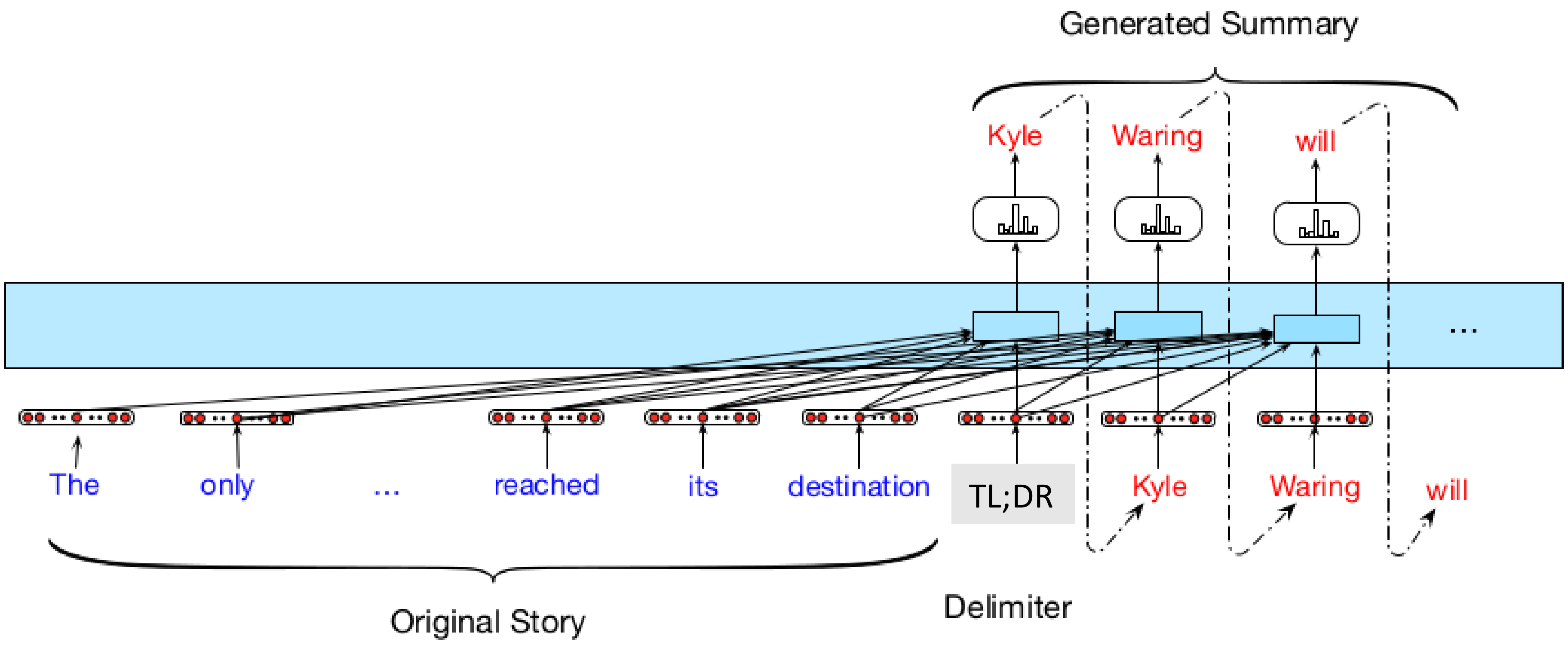
Inflation in December was running at 3.1%, so such an increase will effectively be five times the current inflation rate.

“The National Energy Regulator of South Africa (Nersa) announced… that the High Court of South Africa (Gauteng Division) has ordered that an amount of R10-billion be added to Eskom’s allowable revenue to be recovered from tariff customers in the 2021/22 financial year,” Nersa said in a terse statement on Tuesday.

http://textsummarization.net/text-summarizer

Transformers

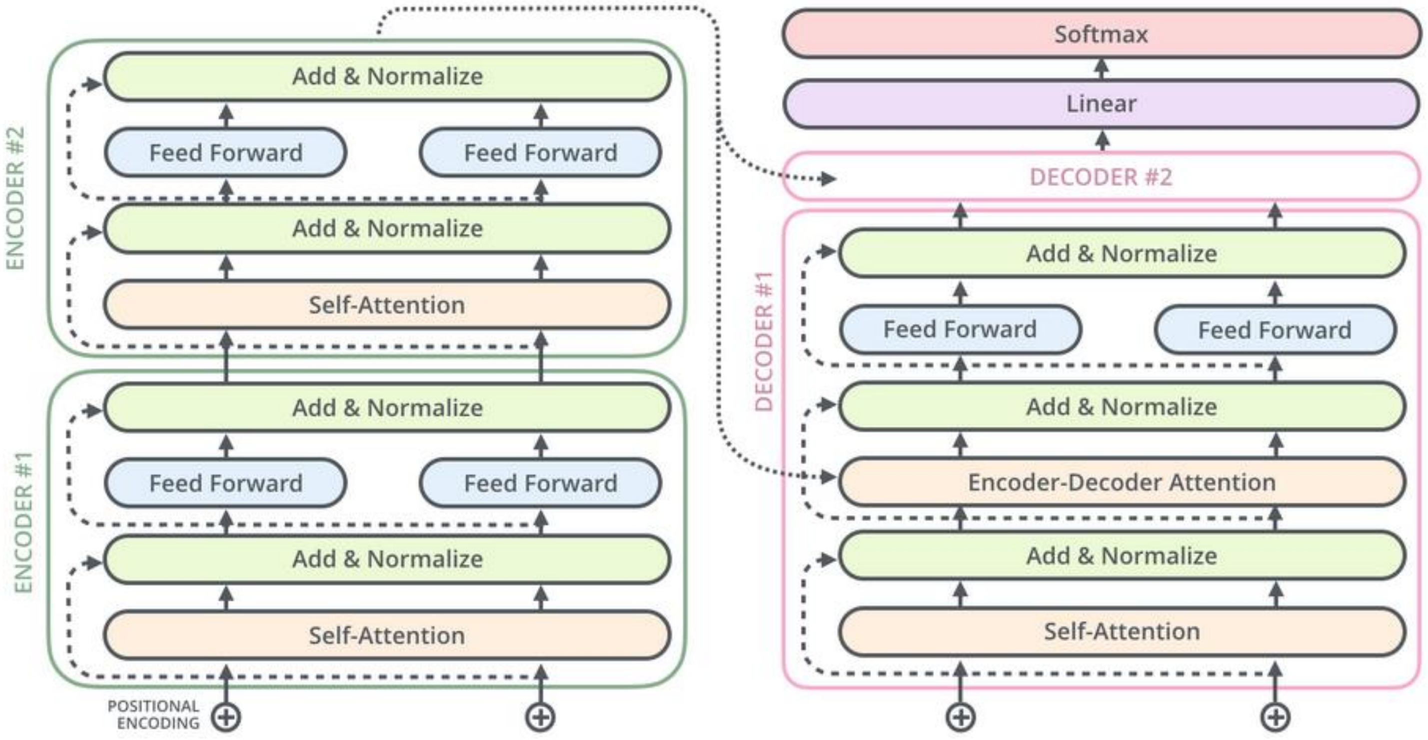
- Application: Summarization



Jurafsky and Martin (2020) Ch. 9

Encoder-decoder Transformers

- Decoder also has both self-attention and attention between the encoder and decoder



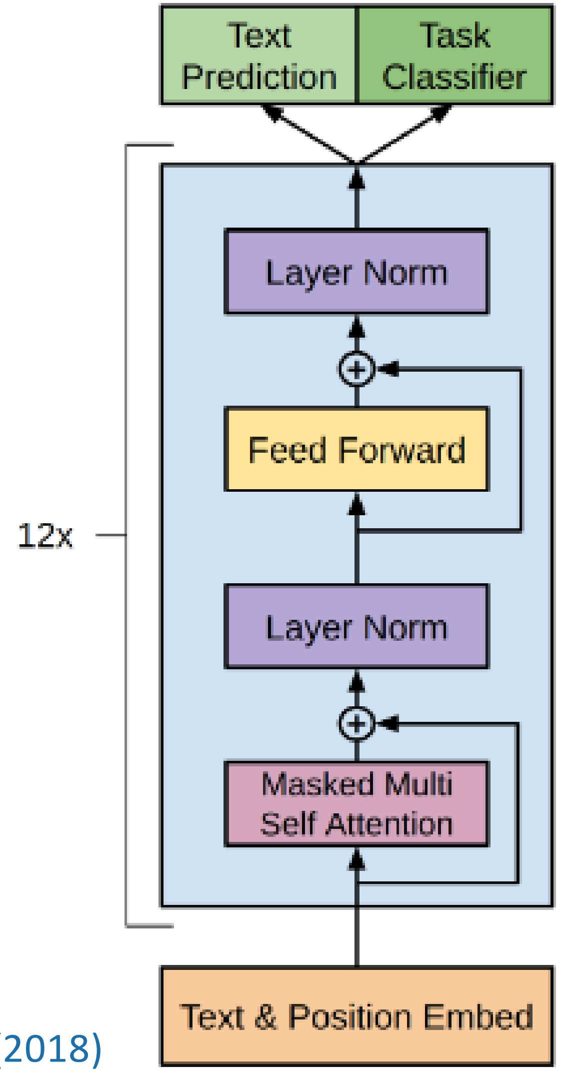
https://jalammar.github.io/illustrated-transformer/

Contextualized Representations

- So far, we have seen pre-trained word embeddings that can be re-used across tasks and models
- But these embeddings are context independent – one learned embedding per word regardless of context
- RNN and Transformer hidden states do give us context-dependent vectors corresponding to each state
- Can we utilise these representations as reusable, pre-trained contextualized embeddings?

GPT: Generative Pre-trained Transformers

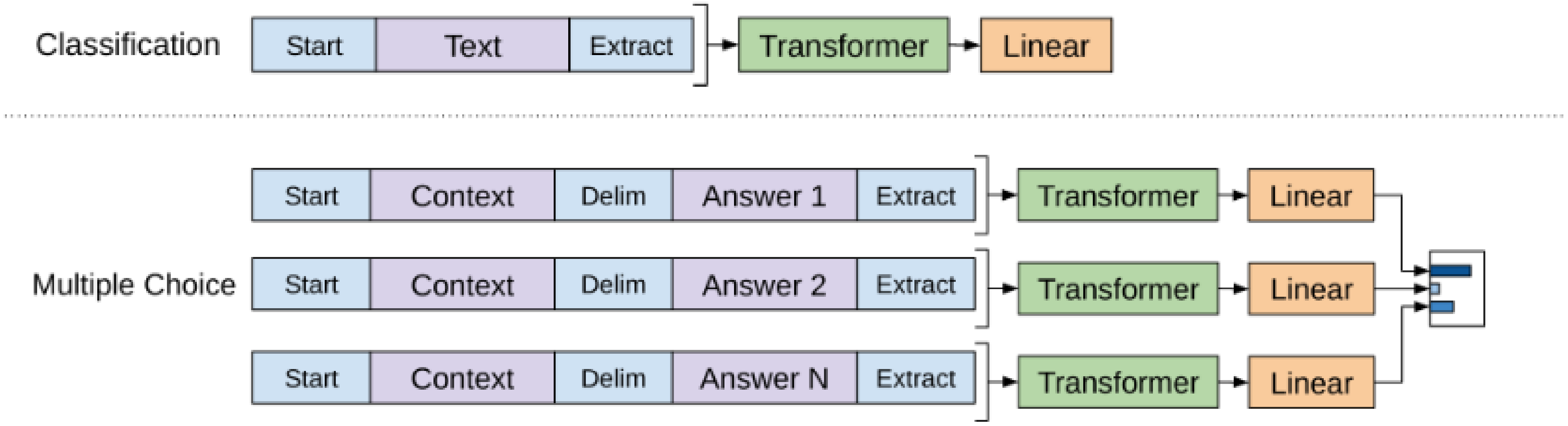
- Transformer language model trained on ~1B word corpus
- Forward model only (causal attention)
- Add task-specific output layer
- Fine-tune all parameters on language understanding tasks



Radford et al. (2018)

GPT: Generative Pre-trained Transformers

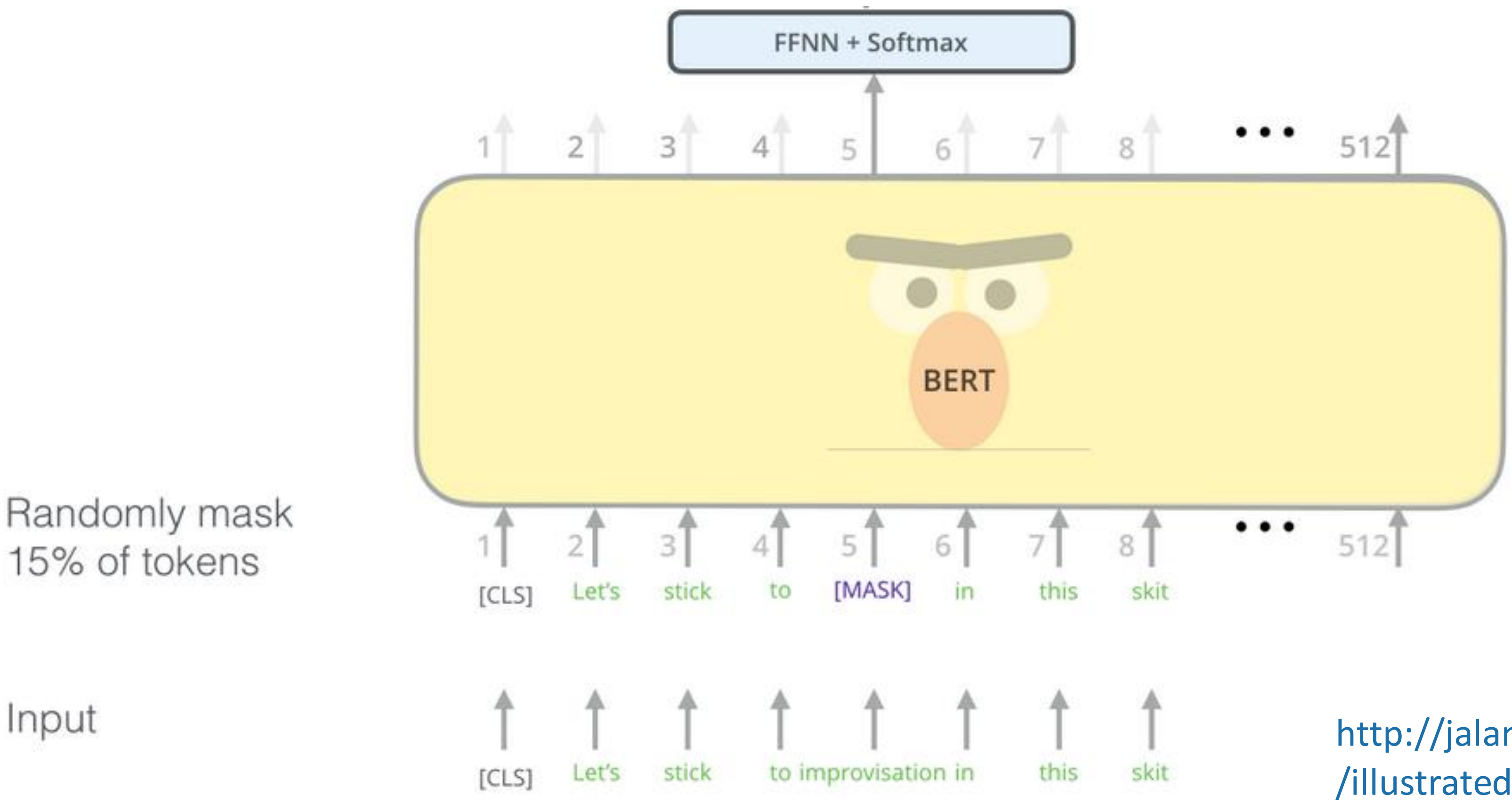
- For different tasks, just change the input sequence and fine-tune:



Radford et al. (2018)

BERT: Bidirectional Encoder Representations from Transformers

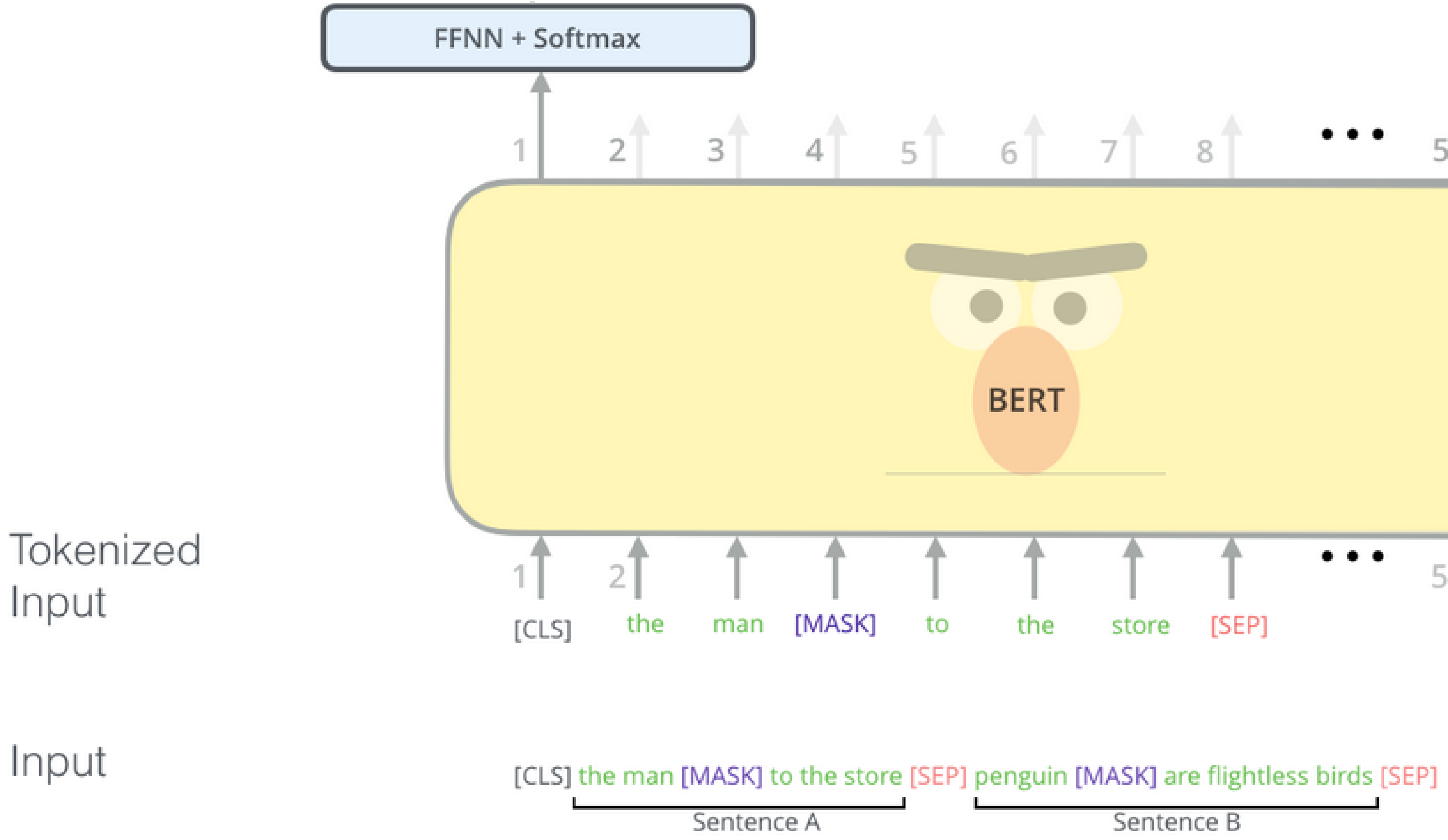
- Pretraining with *masked* language modelling



http://jalammar.github.io/illustrated-bert/

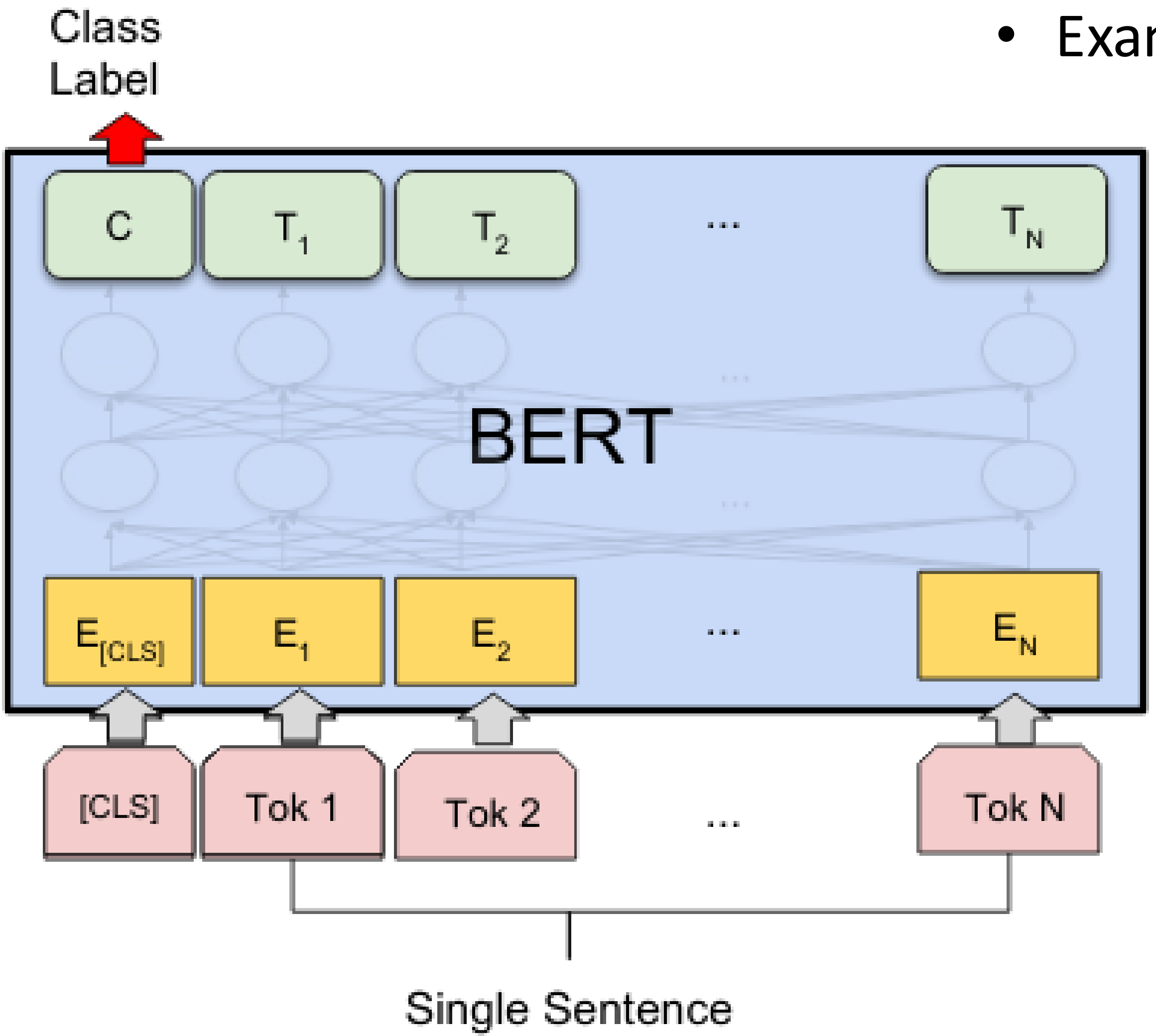
BERT: Bidirectional Encoder Representations from Transformers

- Pretraining with next sentence prediction



http://jalammar.github.io/illustrated-bert/

BERT for Classification

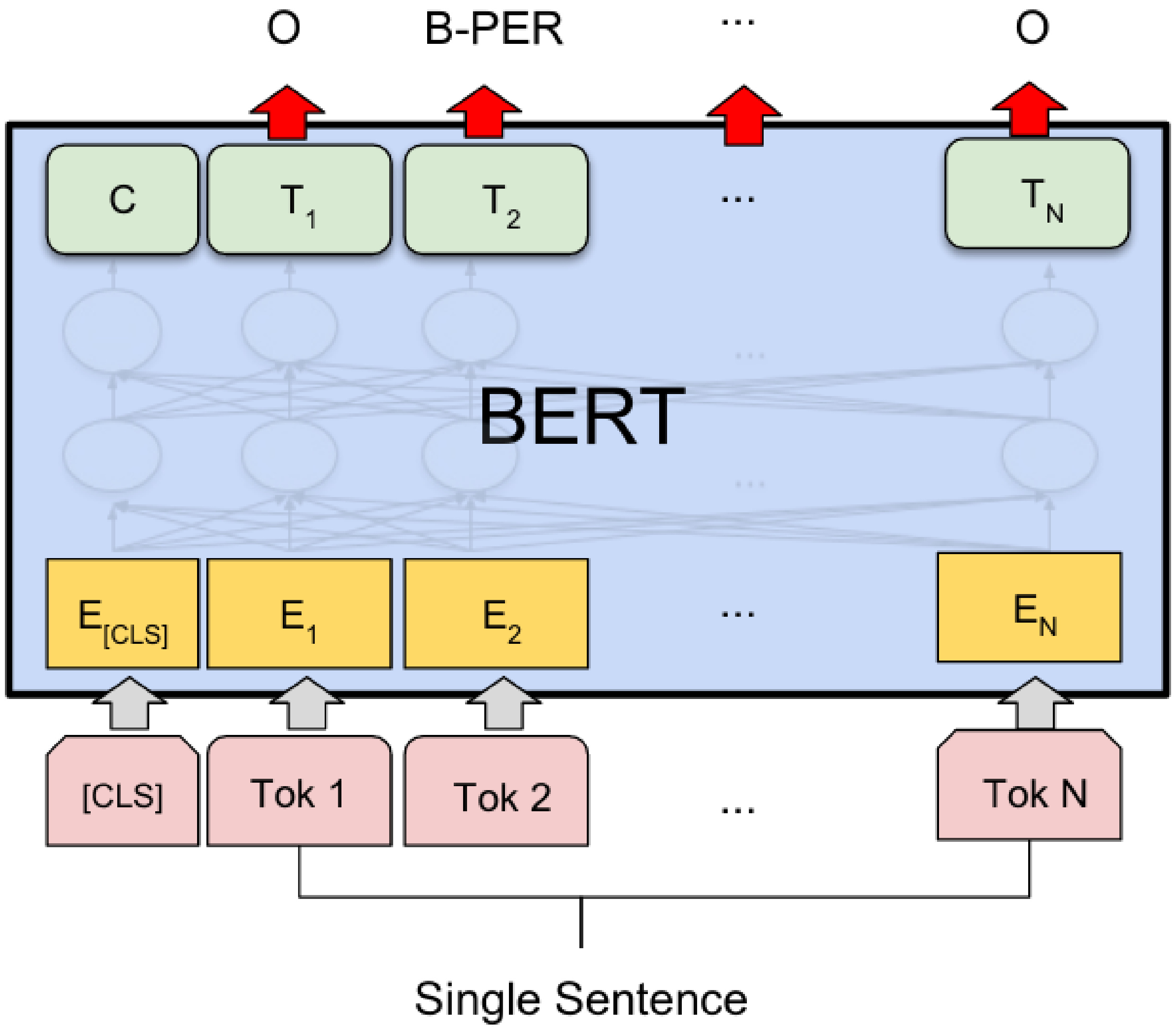


- Example: Sentiment analysis

Devlin et al. (2018)

BERT for Sequence Labelling

- Example: Named Entity Recognition



Devlin et al. (2018)

BERT for Question Answering

Question

How much will Eskom increase power prices?

Run Model

Model Output

Share

Answer

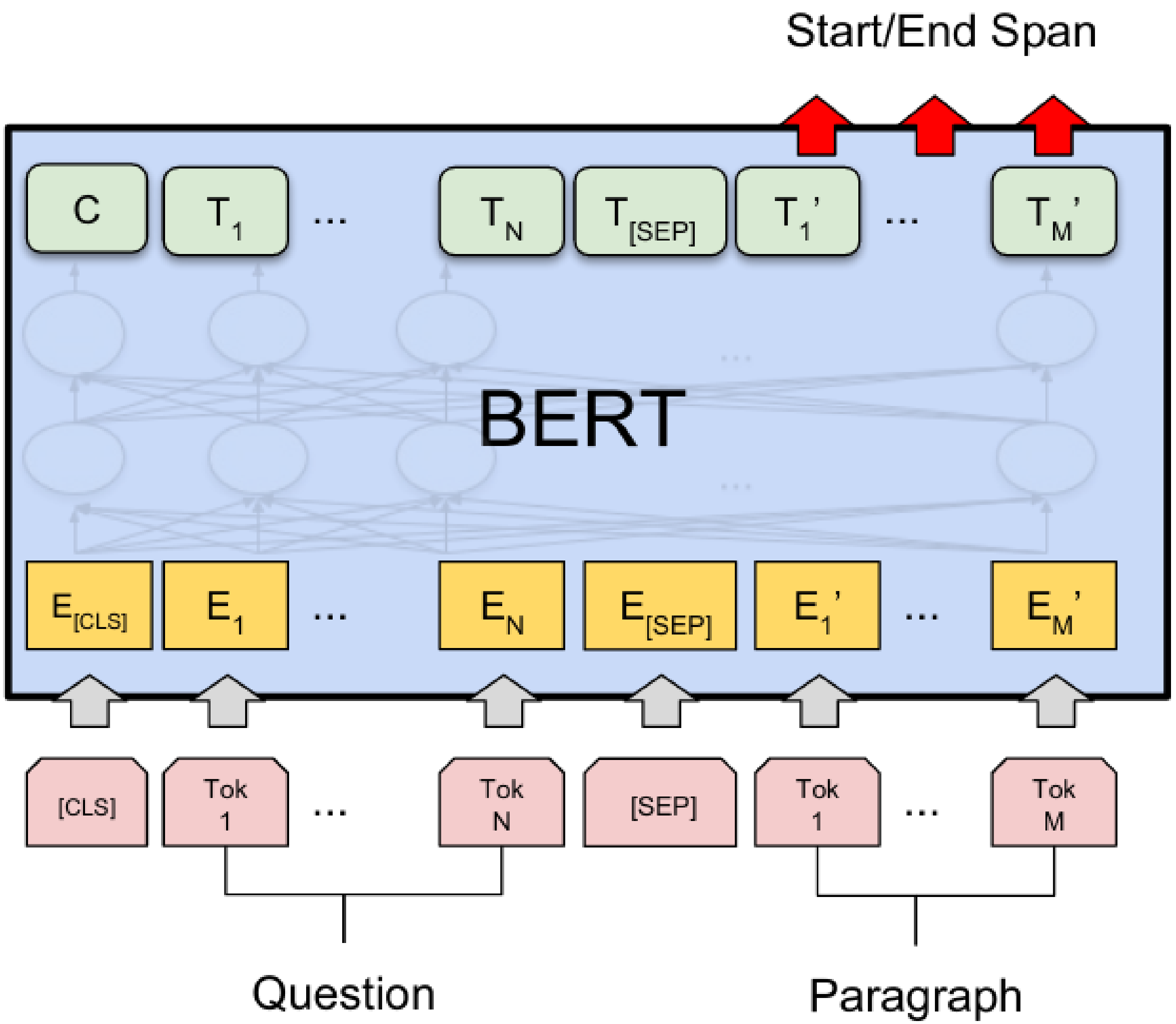
more than 15%

Passage Context

Will the litany of bad economic news ever end? The Gauteng High Court has ordered that Eskom can reap another R10-billion in the 2021/22 financial year, which means an effective increase in power prices of more than 15% . Eskom's woes translate into woes for the entire economy of South Africa. The state-run power utility's inability to provide reliable power is a huge obstacle to investment, economic growth and job creation. Then there is the issue of soaring prices for its sputtering service. Prices are about to jump by more than 15% in the coming financial year, adding an additional cost burden to South African industry and consumers just when they are least able to absorb it. "The National Energy Regulator of South Africa (Nersa) announced... that the High Court of South Africa (Gauteng Division) has ordered that an amount of R10-billion be added to Eskom's allowable revenue to be recovered from tariff customers in the 2021/22 financial year," Nersa said in a terse statement on Tuesday. Eskom has long complained that Nersa has

https://demo.allennlp.org/reading-comprehension/bidaf-elmo

BERT for Question Answering



Devlin et al. (2018)

GPT-2 and GPT-3

- Large-scale training

GPT-2: 40GB
GPT-3: >1 TB

GPT-2 (2019): 1.5 billion parameters
GPT-3 (2020): 175 billion parameters

GPT-2 Text Generation

- Automatic news article generation

Prompt: *In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English.*

Machine-written continuation: The scientist named the population, after their distinctive horn, Ovid's Unicorn. These four-horned, silver-white unicorns were previously unknown to science. Now, after almost two centuries, the mystery of what sparked this odd phenomenon is finally solved. Dr. Jorge Pérez, an evolutionary biologist from the University of La Paz, and several companions, were exploring the Andes Mountains when they found a small valley, with no other animals or humans.

...

Radford et al. (2019)

GPT-3 Zero-shot and few-shot learning

- Question Answering

Context →	Q: Who played tess on touched by an angel? A:
Target Completion →	Delloreese Patricia Early (July 6, 1931 { November 19, 2017), known professionally as Della Reese

- Machine Translation

Context →	Keinesfalls dürfen diese für den kommerziellen Gebrauch verwendet werden. =
Target Completion →	In no case may they be used for commercial purposes.

Brown et al. (2020)

Conclusion

- Deep learning methods can learn powerful representations of word and sequences
- This has enabled state-of-the-art performance on most NLP tasks, and enabled new applications not previously feasible
- But deep learning methods are not perfect and many interesting research questions remain open

References

- Dan Jurafsky and James H. Martin. *Speech and Language Processing.* 3rd edition draft, December 2020. Available at: https://web.stanford.edu/~jurafsky/slp3/
- Yoav Goldberg. *Neural Network Methods for Natural Language Processing*. Synthesis Lectures on Human Language Technologies. 2017.
- Ian Goodfellow, Yoshua Bengio and Aaron Courville. *Deep Learning*. MIT Press. 2016.
- Yejin Choi. *Deep Learning for NLP*. Lectures at the University of Washington, 2020. Available at: https://courses.cs.washington.edu/courses/cse447/20wi/
- Thang Luong, Kyunghyun Cho, Christopher Manning. *Neural Machine Translation*. Tutorial at ACL 2016. Available at: https://sites.google.com/site/acl16nmt/home

References

- Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado and Jeffrey Dean. *Distributed Representations of Words and Phrases and their Compositionality.* NeurIPS 2013.
- Dzmitry Bahdanau, Kyunghyun Cho and Yoshua Bengio. *Neural Machine Translation by Jointly Learning to Align and Translate.* ICLR 2015.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser and Illia Polosukhin. *Attention Is All You Need.* NeurIPS 2017.
- Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee and Luke Zettlemoyer. *Deep contextualized word representations.* NAACL 2018.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee and Kristina Toutanova. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.* NAACL 2019.
- Alec Radford, Karthik Narasimhan, Tim Salimans and lya Sutskever. *Improving Language Understanding by Generative Pre-Training.* Technical Report, OpenAI, 2018.

References

- https://colah.github.io/posts/2015-08-Understanding-LSTMs/
- https://jalammar.github.io/illustrated-transformer/
- https://nlp.seas.harvard.edu/2018/04/03/attention.html
- https://jalammar.github.io/illustrated-word2vec/
- https://demo.allennlp.org