Summer semester 2022

344.063 KV Special Topic:

Natural Language Processing with Deep Learning

Neural Machine Translation with Attention Networks

Navid Rekab-saz

navid.rekabsaz@jku.at

JOHANNES KEPLER UNIVERSITY LINZ

Institute of Computational Perception

Agenda

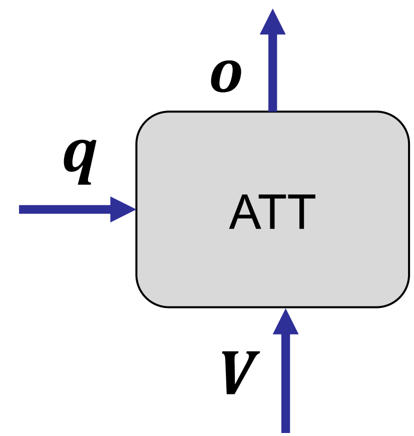
- Attention Networks
- Machine Translation
  - Seq2seq with Attention
- Hierarchical document classification

Agenda

- **Attention Networks**
- Machine Translation
  - Seq2seq with Attention
- Hierarchical document classification

Attention Networks

- Attention is a deep learning architecture …
  - to obtain a composed output embedding $\boldsymbol{o}$ …
  - from a set (matrix) of input values $\boldsymbol{V}$ …
  - based on a given query embedding $\boldsymbol{q}$



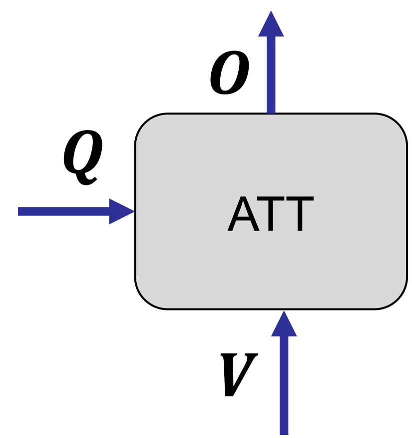
- General form of an attention network:

$$\boldsymbol{o} = \mathrm{ATT}(\boldsymbol{q}, \boldsymbol{V})$$

- If a set/matrix of queries $\boldsymbol{Q}$ is given, the output will become a set/matrix $\boldsymbol{O}$:

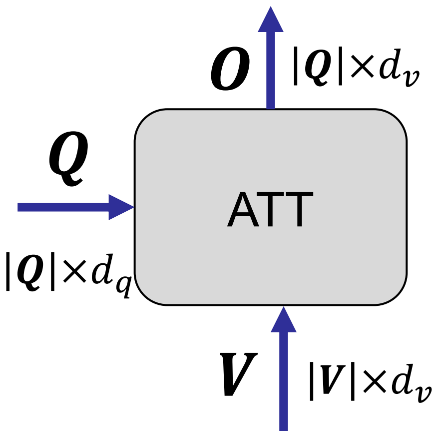
$$\boldsymbol{O} = \mathrm{ATT}(\boldsymbol{Q}, \boldsymbol{V})$$

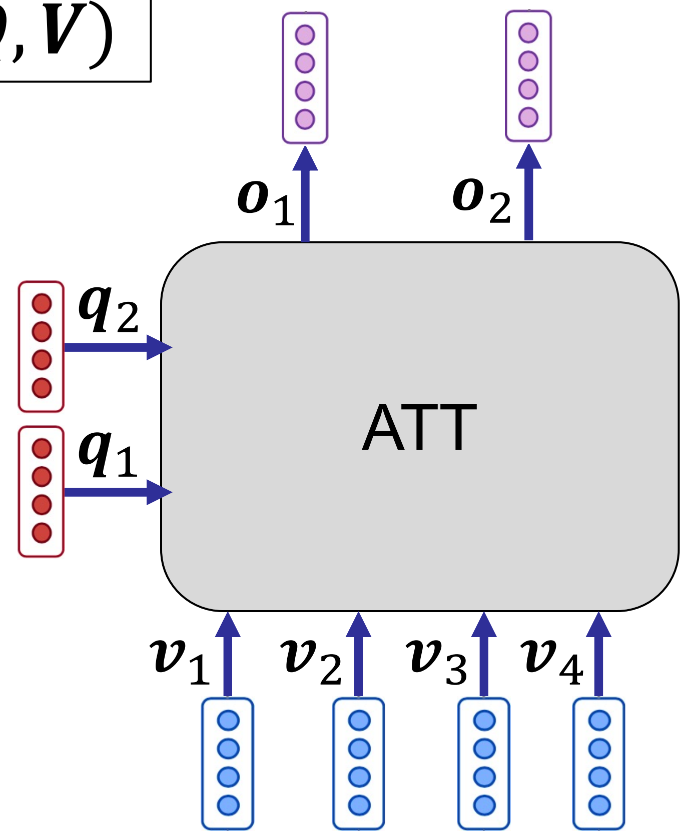
  - where each output vector belongs to its respective query vector



Attention Networks

$$\boldsymbol{O} = \mathrm{ATT}(\boldsymbol{Q}, \boldsymbol{V})$$

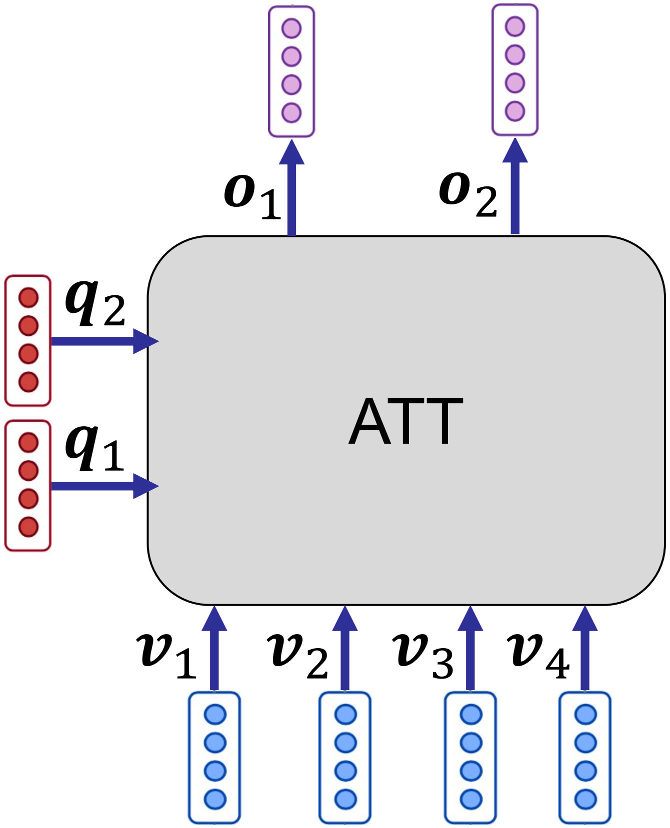




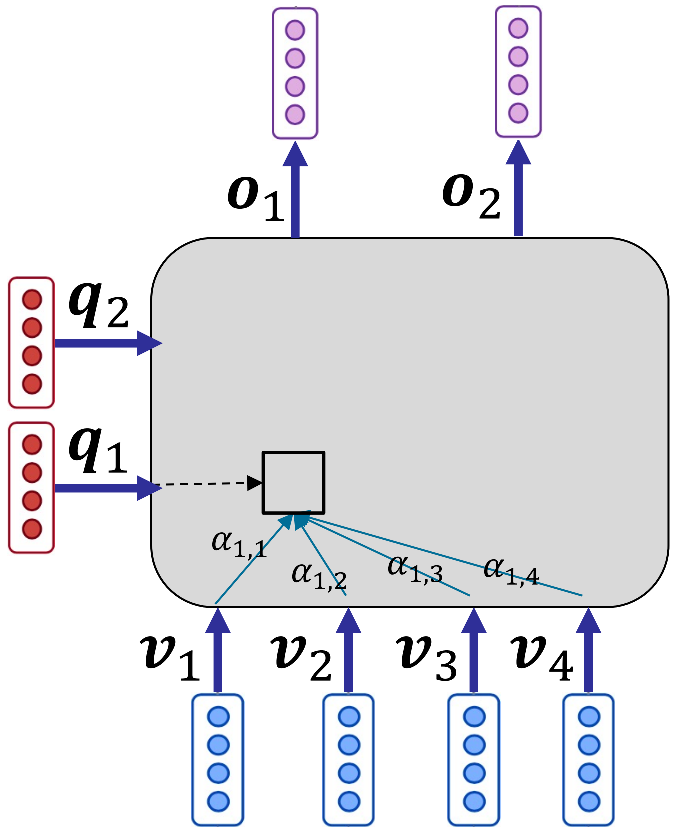
- d_q, d_v are embedding dimensions of query and value vectors, respectively

We sometime say, each query vector $\boldsymbol{q}$ "*attends*" to value vectors

Attention Networks

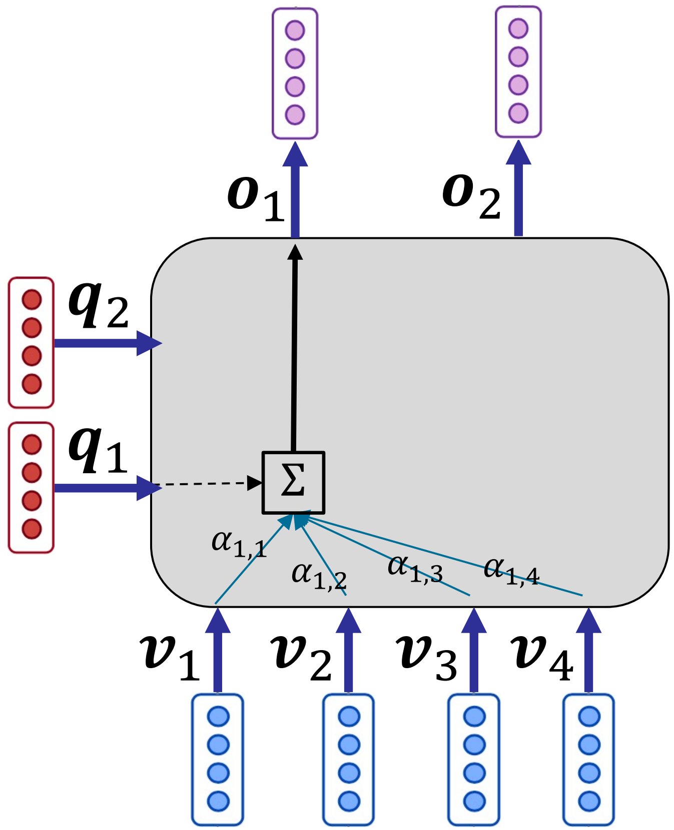


Attention Networks



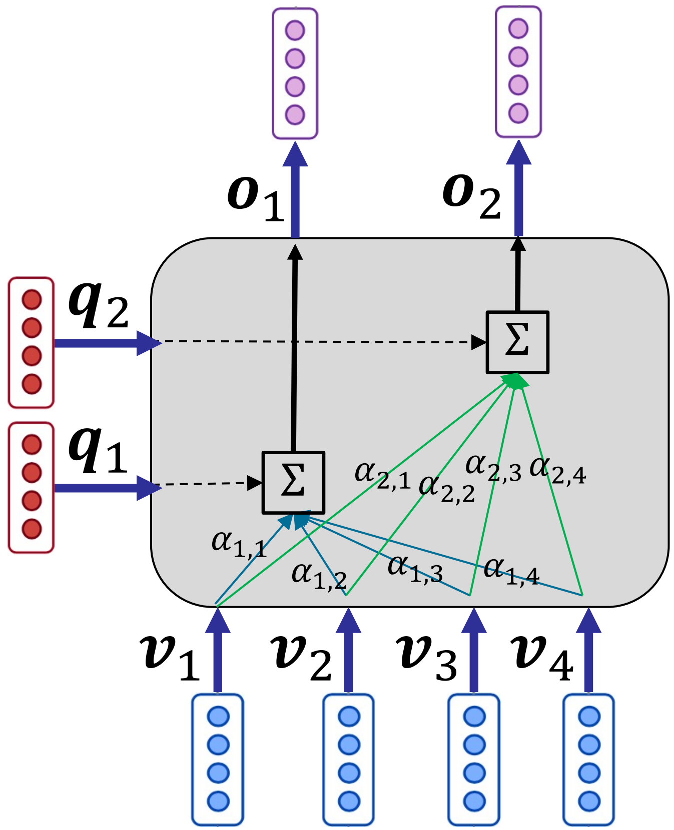
$\alpha_{i,j}$ is the attention of query $\boldsymbol{q}_i$ on value $\boldsymbol{v}_j$

Attention Networks



$\alpha_{i,j}$ is the attention of query $\boldsymbol{q}_i$ on value $\boldsymbol{v}_j$

Attention Networks



$\alpha_{i,j}$ is the attention of query $\boldsymbol{q}_i$ on value $\boldsymbol{v}_j$

Attention Networks – definition

- Given a matrix of values $\boldsymbol{V}$ and a matrix of queries $\boldsymbol{Q}$, for each query vector $\boldsymbol{q} \in \boldsymbol{Q}$, an **attention network** …
  - first assigns an attention score to each value vector $\boldsymbol{v} \in \boldsymbol{V}$ based on the similarity of $\boldsymbol{q}$ to $\boldsymbol{v}$,…
  - then turns the attention scores to a probability distribution of attentions over value vectors, …
  - and finally uses the attentions to calculate the weighted sum of the value vectors as the corresponding output $\boldsymbol{o}$ of the query vector $\boldsymbol{q}$

- The output of attention networks can be viewed as a weighted aggregation of the value vectors, where the query (through attentions) defines the proportion of the contribution of each value vector.

Attention Networks – formulation

- Given query vector $\boldsymbol{q}_i$, an attention network uses the attention similarity function f to assign a non-normalized attention score $\tilde{\alpha}_{i,j}$ to value vector $\boldsymbol{v}_j$:

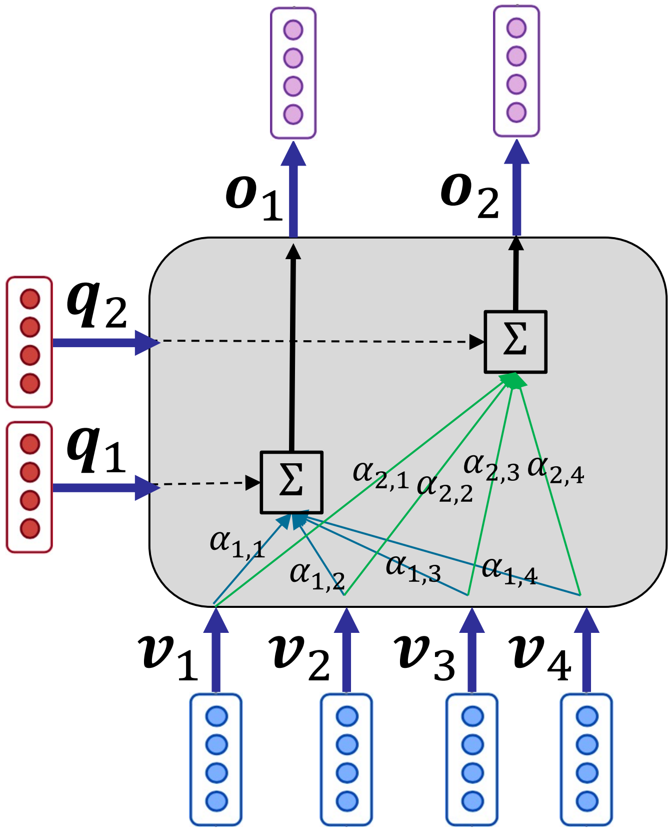
$$\tilde{\alpha}_{i,j} = f(\boldsymbol{q}_i, \boldsymbol{v}_j)$$

- Then, the attention scores over values are turned to a probability distribution using softmax:

$$\boldsymbol{\alpha}_i = \text{softmax}(\tilde{\boldsymbol{\alpha}}_i), \qquad \sum_{j=1}^{|\boldsymbol{V}|} \alpha_{i,j} = 1$$

- Finally, output vector $\boldsymbol{o}_i$ regarding query $\boldsymbol{q}_i$ is defined as the sum of the value vectors weighted by their corresponding attentions:

$$\boldsymbol{o}_i = \sum_{j=1}^{|\boldsymbol{V}|} \alpha_{i,j} \boldsymbol{v}_j$$



Attention – first implementation

Basic dot-product attention

- Non-normalized attention scores:

$$\tilde{\alpha}_{i,j} = f(\boldsymbol{q}_i, \boldsymbol{v}_j)$$

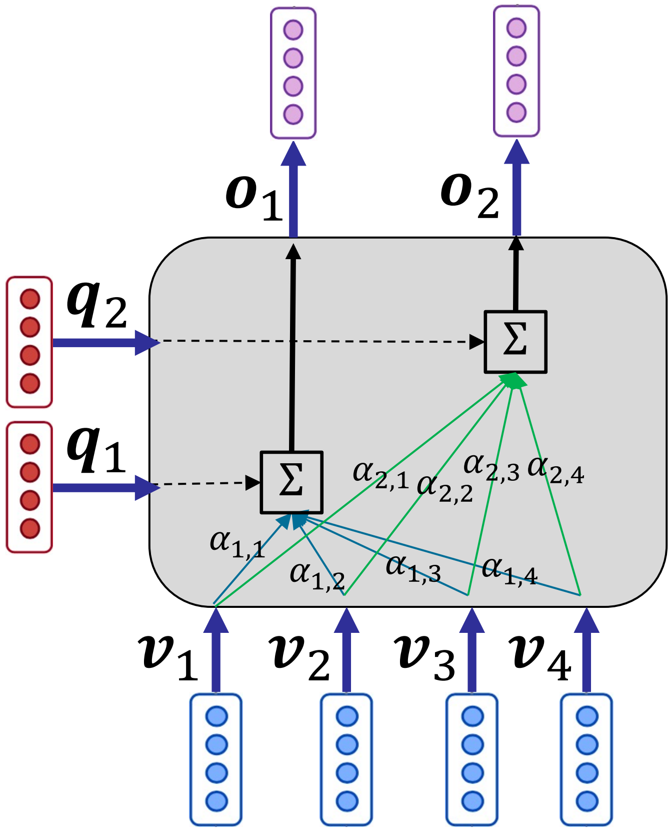
$$\tilde{\alpha}_{i,j} = \boldsymbol{q}_i \boldsymbol{v}_i^{\mathrm{T}}$$

  - In this case, $d_q = d_v$
  - Attention network has no parameter to learn!

- Softmax over value vectors:

$$\boldsymbol{\alpha}_i = \mathrm{softmax}(\tilde{\boldsymbol{\alpha}}_i)$$

- Output (weighted sum): $\boldsymbol{o}_i = \sum_{j=1}^{|V|} \alpha_{i,j} \boldsymbol{v}_j$

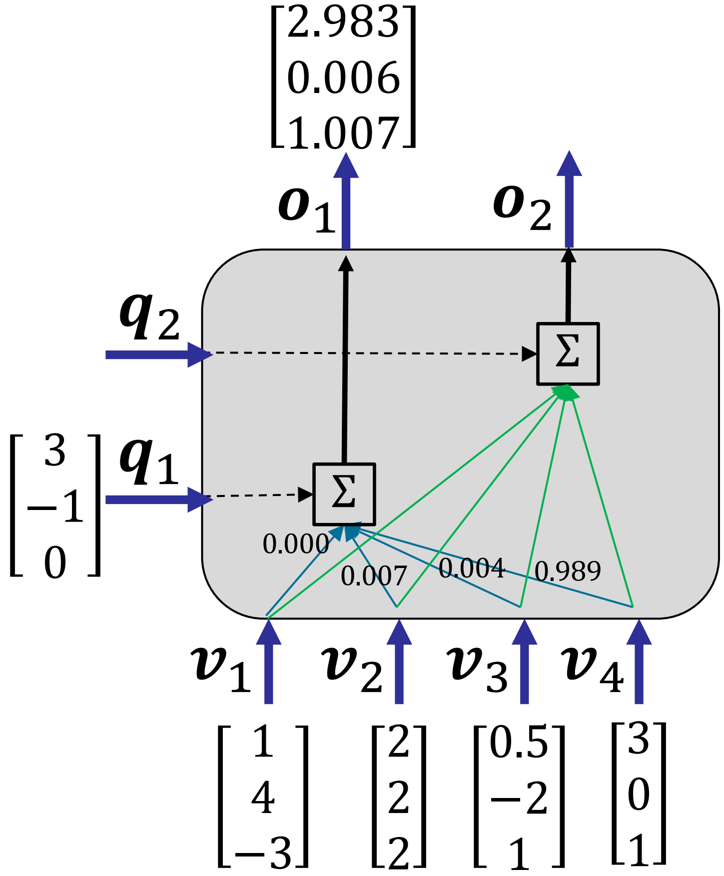


Example

$$\tilde{\boldsymbol{\alpha}}_1 = \begin{bmatrix} \boldsymbol{q}_1\boldsymbol{v}_1^{\mathrm{T}} = -1 \\ \boldsymbol{q}_1\boldsymbol{v}_2^{\mathrm{T}} = 4 \\ \boldsymbol{q}_1\boldsymbol{v}_3^{\mathrm{T}} = 3.5 \\ \boldsymbol{q}_1\boldsymbol{v}_4^{\mathrm{T}} = 9 \end{bmatrix} \rightarrow \boldsymbol{\alpha}_1 = \begin{bmatrix} 0.000 \\ 0.007 \\ 0.004 \\ 0.989 \end{bmatrix}$$

$$\boldsymbol{o}_1 = 0.000\begin{bmatrix} 1 \\ 4 \\ -3 \end{bmatrix} + 0.007\begin{bmatrix} 2 \\ 2 \\ 2 \end{bmatrix} + 0.004\begin{bmatrix} 0.5 \\ -2 \\ 1 \end{bmatrix} + 0.989\begin{bmatrix} 3 \\ 0 \\ 1 \end{bmatrix}$$

$$\boldsymbol{o}_1 = \begin{bmatrix} 2.983 \\ 0.006 \\ 1.007 \end{bmatrix}$$

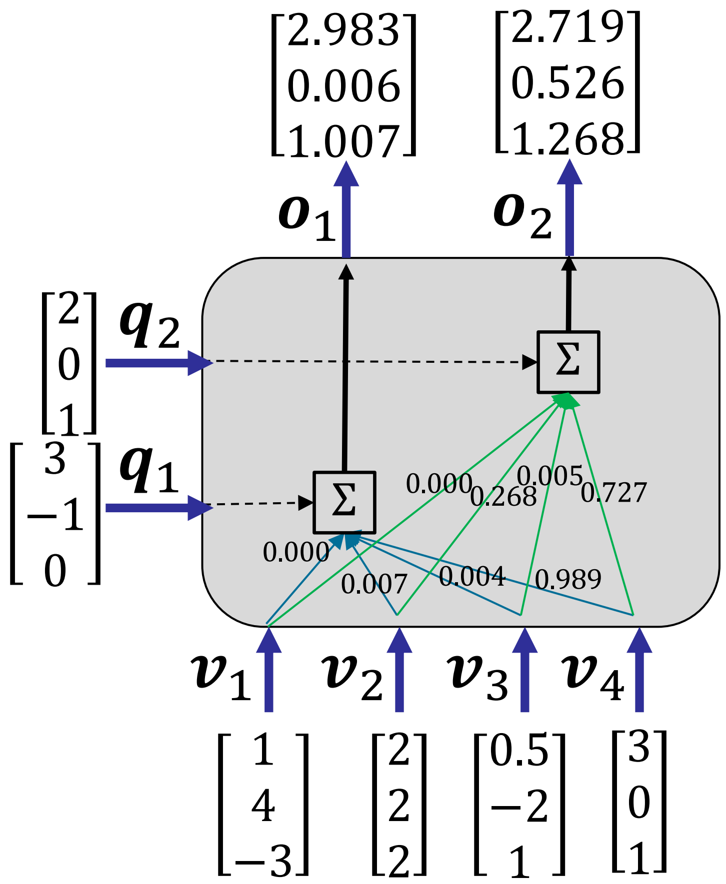


Example

$$\tilde{\boldsymbol{\alpha}}_2 = \begin{bmatrix} \boldsymbol{q}_2\boldsymbol{v}_1^{\mathrm{T}} = -1 \\ \boldsymbol{q}_2\boldsymbol{v}_2^{\mathrm{T}} = 6 \\ \boldsymbol{q}_2\boldsymbol{v}_3^{\mathrm{T}} = 2 \\ \boldsymbol{q}_2\boldsymbol{v}_4^{\mathrm{T}} = 7 \end{bmatrix} \rightarrow \quad \boldsymbol{\alpha}_2 = \begin{bmatrix} 0.000 \\ 0.268 \\ 0.005 \\ 0.727 \end{bmatrix}$$

$$\boldsymbol{o}_2 = 0.000 \begin{bmatrix} 1 \\ 4 \\ -3 \end{bmatrix} + 0.268 \begin{bmatrix} 2 \\ 2 \\ 2 \end{bmatrix} + 0.005 \begin{bmatrix} 0.5 \\ -2 \\ 1 \end{bmatrix} + 0.727 \begin{bmatrix} 3 \\ 0 \\ 1 \end{bmatrix}$$

$$\boldsymbol{o}_2 = \begin{bmatrix} 2.719 \\ 0.526 \\ 1.268 \end{bmatrix}$$



Attention – other implementations

Multiplicative attention

- Non-normalized attention scores:

$$\tilde{\alpha}_{i,j} = f(\boldsymbol{q}_i, \boldsymbol{v}_j)$$

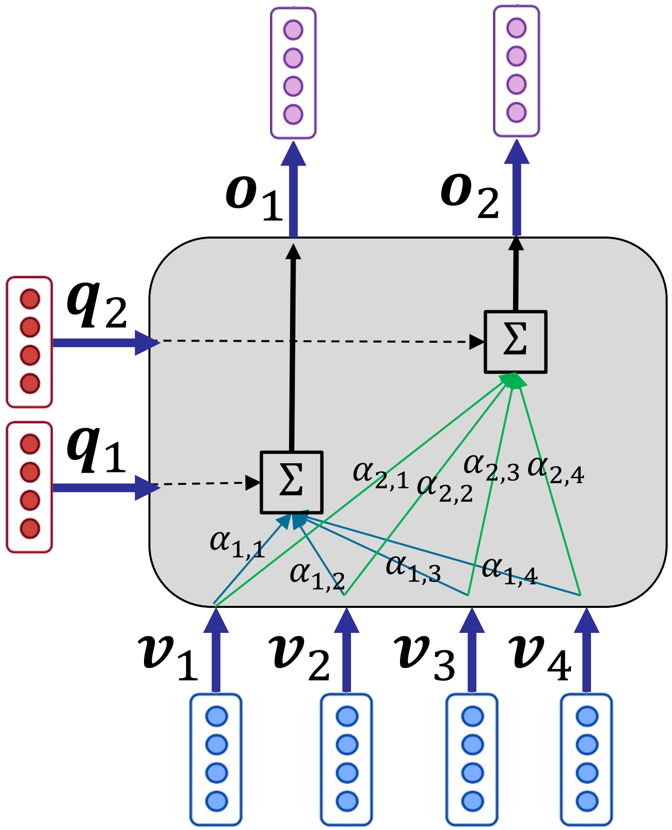
$$\tilde{\alpha}_{i,j} = \boldsymbol{q}_i \boldsymbol{W} \boldsymbol{v}_i^{\mathrm{T}}$$

  - $\boldsymbol{W}$ is a matrix of parameter
  - similarity of query to value is defined as a linear function

- Softmax over values:

$$\boldsymbol{\alpha}_i = \mathrm{softmax}(\tilde{\boldsymbol{\alpha}}_i)$$

- Output (weighted sum): $\boldsymbol{o}_i = \sum_{j=1}^{|V|} \alpha_{i,j} \boldsymbol{v}_j$



Attention – other implementations

Additive attention

- Non-normalized attention scores:

$$\tilde{\alpha}_{i,j} = f(\boldsymbol{q}_i, \boldsymbol{v}_j)$$

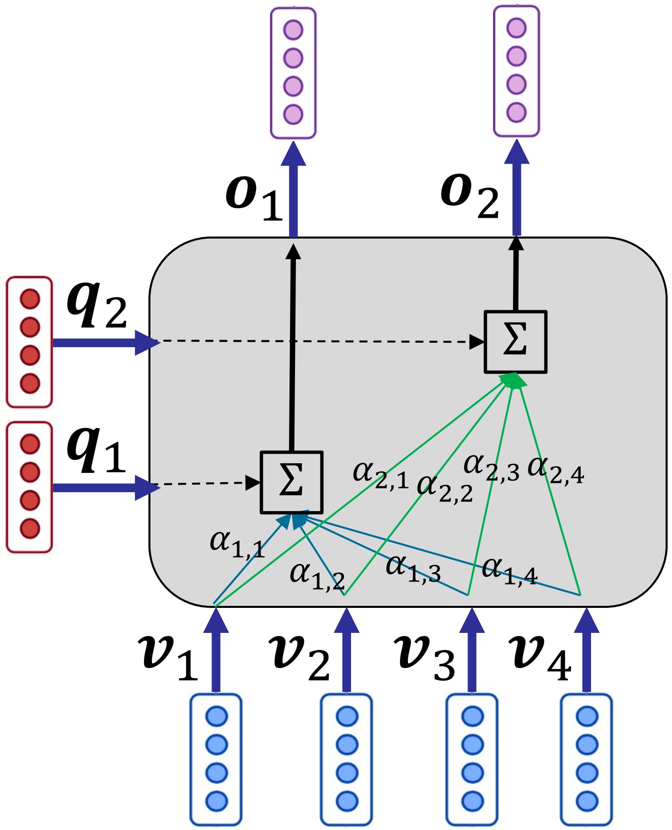
$$\tilde{\alpha}_{i,j} = \boldsymbol{u}^{\mathrm{T}} \tanh(\boldsymbol{q}_i \boldsymbol{W}_1 + \boldsymbol{v}_j \boldsymbol{W}_2)$$

  - $\boldsymbol{W}_1$, $\boldsymbol{W}_2$, and $\boldsymbol{u}$ are model parameters
  - similarity of query to value is defined as a non-linear function

- Softmax over values:

$$\boldsymbol{\alpha}_i = \mathrm{softmax}(\tilde{\boldsymbol{\alpha}}_i)$$

- Output (weighted sum): $\boldsymbol{o}_i = \sum_{j=1}^{|V|} \alpha_{i,j} \boldsymbol{v}_j$



Attention – summary

- Attention is a way to define the distribution of focus on inputs based on a query, and create a compositional embedding of inputs

- Attention networks define an attention distribution over inputs and calculate their weighted sum

- The original definition of attention network has two inputs: key vectors $\boldsymbol{K}$, and value vectors $\boldsymbol{V}$
  - Key vectors are used to calculate attentions
  - and output is the weighted sum of value vectors
  - In practice, in most cases $\boldsymbol{K} = \boldsymbol{V}$.
  - In this course, we use our slightly simplified definition

Agenda

- Attention Networks
- **Machine Translation**
  - Seq2seq with Attention
- Hierarchical document classification

Machine Translation (MT)

- Machine Translation is the task of translating a sentence X from source language to sentence Y in target language

- A long-history (since 1950)
  - Early systems were mostly rule-based

- Challenges:
  - Common sense
  - Idioms!
  - Typological differences between the source and target language
  - Alignment
  - Low-resource language pairs

Statistical Machine Translation (SMT)

- Statistical Machine Translation (1990-2010) learns a probabilistic model using large amount of parallel data
- The model aims to find the best target language sentence Y^*, given the source language sentence X:

$$Y^* = \underset{Y}{\operatorname{argmax}} P(Y|X)$$

- SMT uses Bayes Rule to split this probability into two components that can be learnt separately:

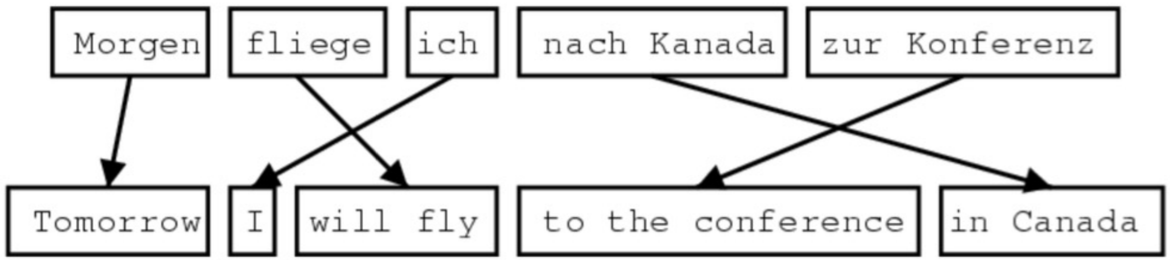
$$= \underset{Y}{\operatorname{argmax}} P(X|Y)P(Y)$$

https://en.wikipedia.org/wiki/Rosetta_Stone

Translation Model	Language Model
The statistical model that defines how words and phrases should be translated (learnt from parallel data)	The statistical model that tells us how to write good sentences in the target language (learnt from monolingual data)

Learning Translation model

- To learn the Translation model $P(Y|X)$, we need to break X and Y down to aligned words and phrases:

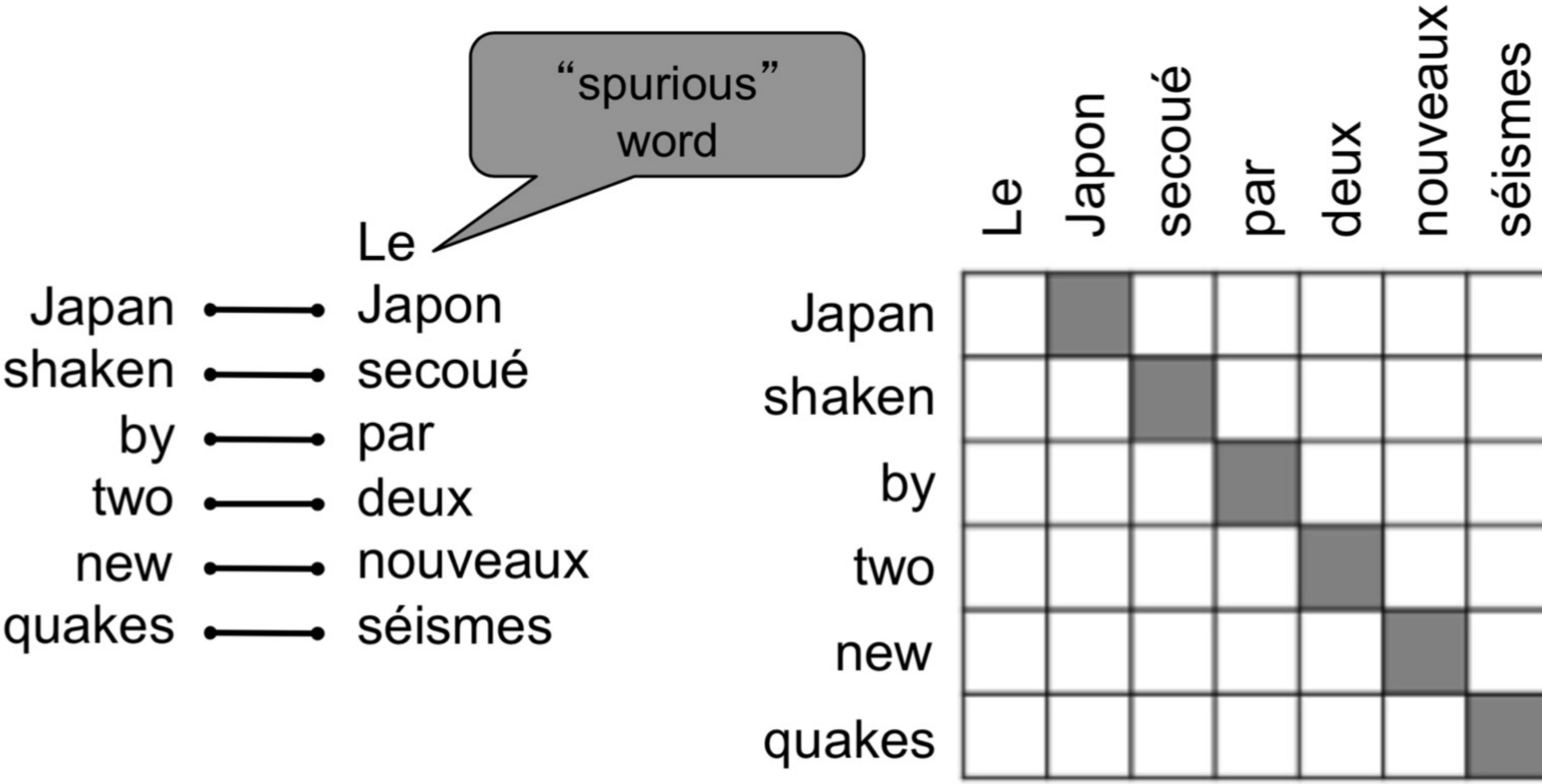


- To this end, the alignment latent variable a is added to the formulation of Translation model:

$$P(X, a|Y)$$

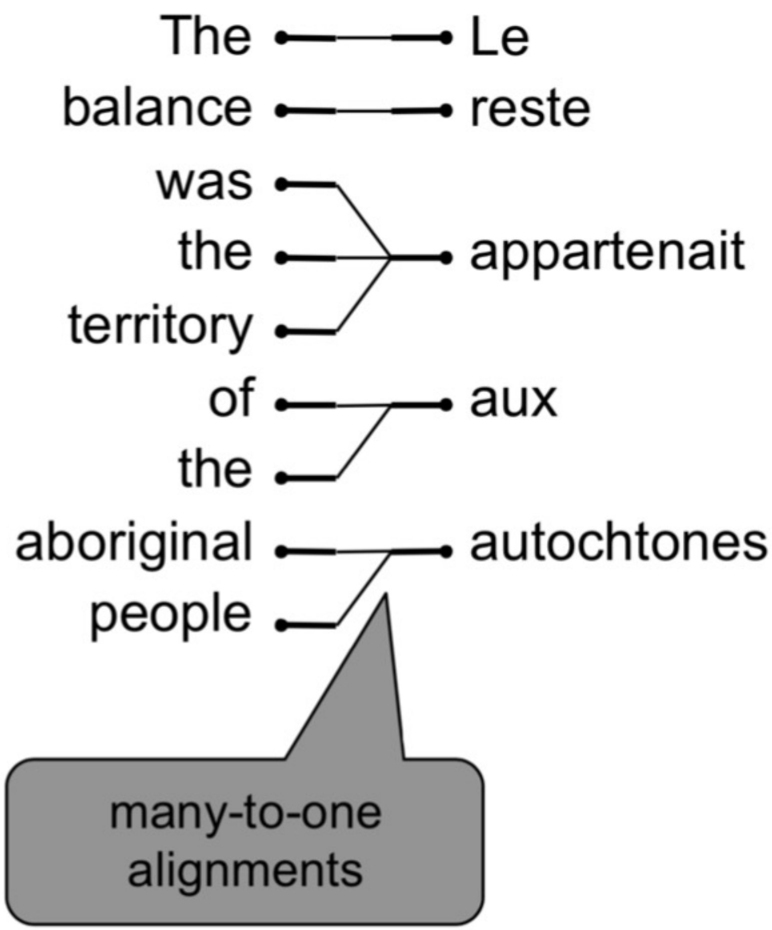
- Alignment …
  - is a latent variable → is not explicitly defined in the data!
  - defines the correspondence between particular words/phrases in the translation sentence pair

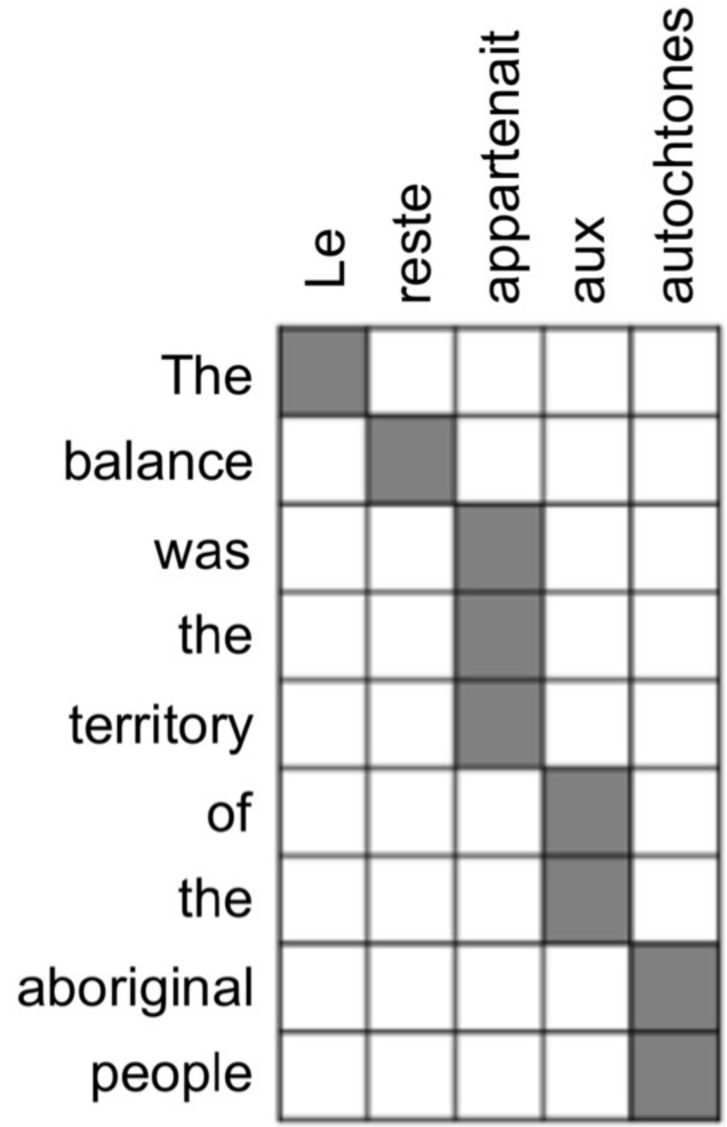
Alignment!



Examples originally from: "The Mathematics of Statistical Machine Translation: Parameter Estimation", Brown et al, 1993.

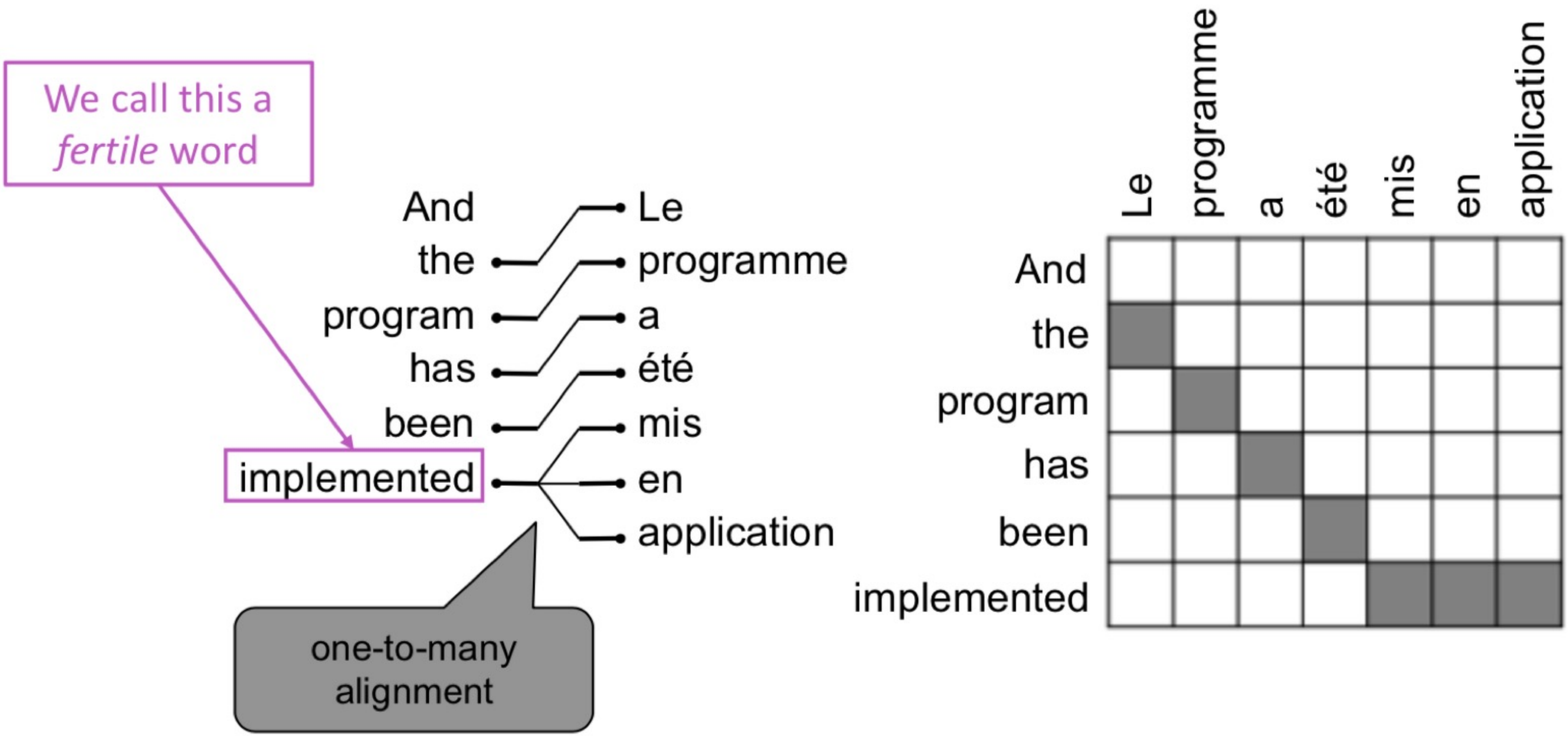
Alignment!





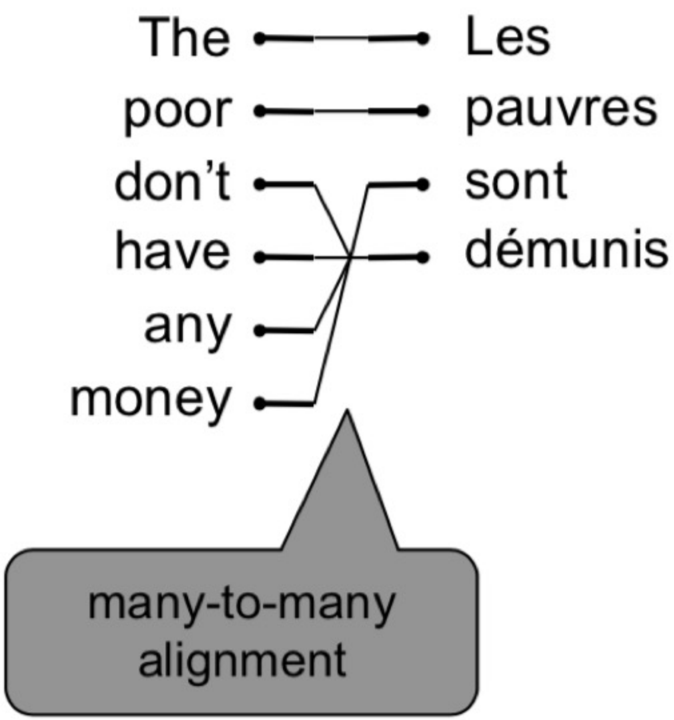
Examples originally from: “The Mathematics of Statistical Machine Translation: Parameter Estimation", Brown et al, 1993.

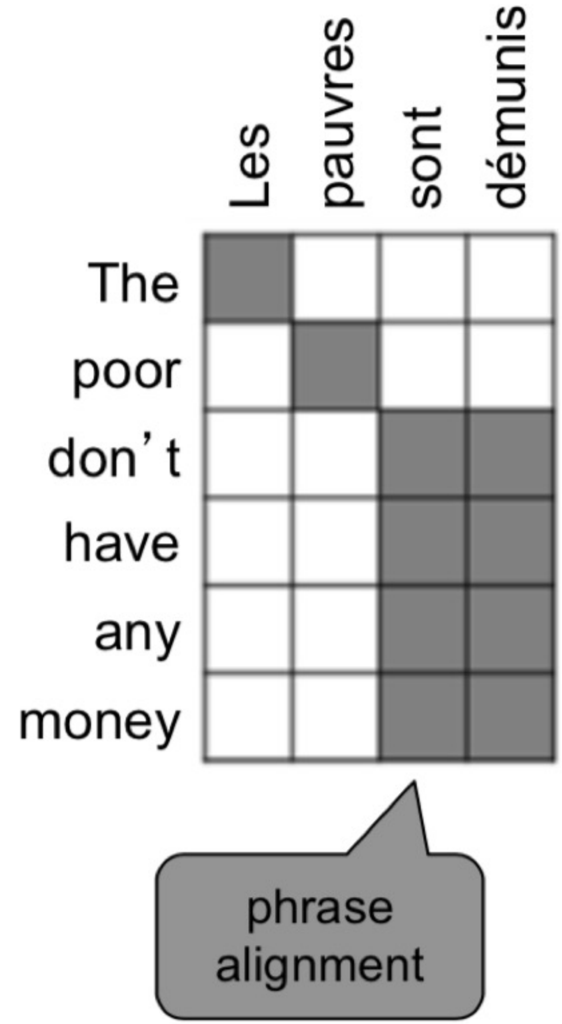
Alignment!



Examples originally from: "The Mathematics of Statistical Machine Translation: Parameter Estimation", Brown et al, 1993.

Alignment!





Examples originally from: “The Mathematics of Statistical Machine Translation: Parameter Estimation", Brown et al, 1993.

SMT – summary

- Defining alignment is complex!
  - The Translation model should jointly estimate distributions of both variables (X and a)

- SMT systems …
  - were extremely complex with lots of features engineering
  - required extra resources like dictionaries and mapping tables between phrases and words
  - required “special attention” for each language pair and lots of human efforts

MT – Evaluation

- BLEU (Bilingual Evaluation Understudy)

- BLEU computes a similarity score between the machine-written translation to one or several human-written translation(s), based on:
  - *n*-gram precision (usually for 1, 2, 3 and 4-grams)
  - plus a penalty for too-short machine translations

- BLEU is precision-based, while ROUGE is recall-based

Details of how to calculate BLEU: https://www.coursera.org/lecture/nlp-sequence-models/bleu-score-optional-kC2HD

Agenda

- Attention Networks
- **Machine Translation**
    - **Seq2seq with Attention**
- Hierarchical document classification

Neural Machine Translation (NMT)

- Given the source language sentence X and target language sentence Y, NMT uses seq2seq models to calculate the conditional language model:

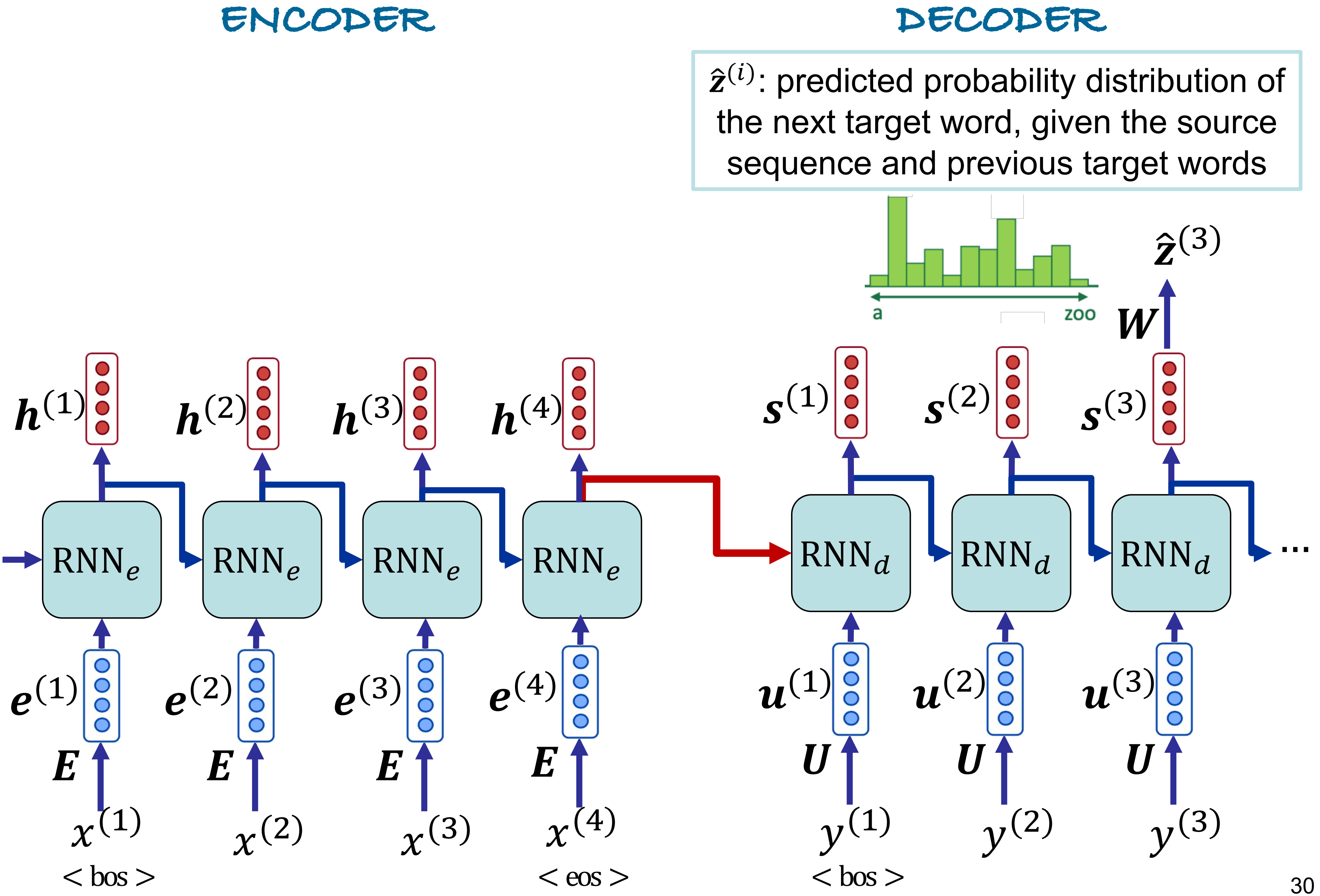
$$P(Y|X)$$

  - A language model of the target language
  - Conditioned on the source language

- In contrast to SMT, no need for pre-defined alignments! 🎉

- We can simply use a seq2seq with two RNNs

Seq2seq with two RNNs (recap)



Seq2seq with two RNNs – training (recap)

Look here for more: https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

Seq2seq – decoding / beam search (recap)

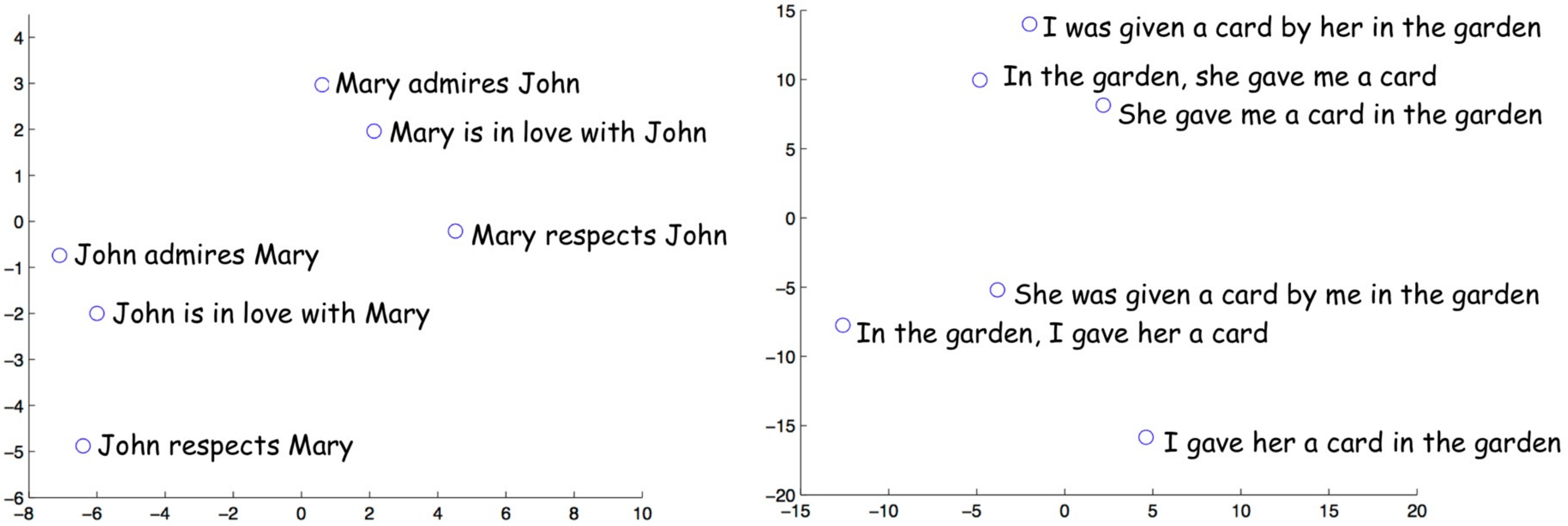
<bos>

Start with the begin of sentence token or with an empty sequence

Look here for more: https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

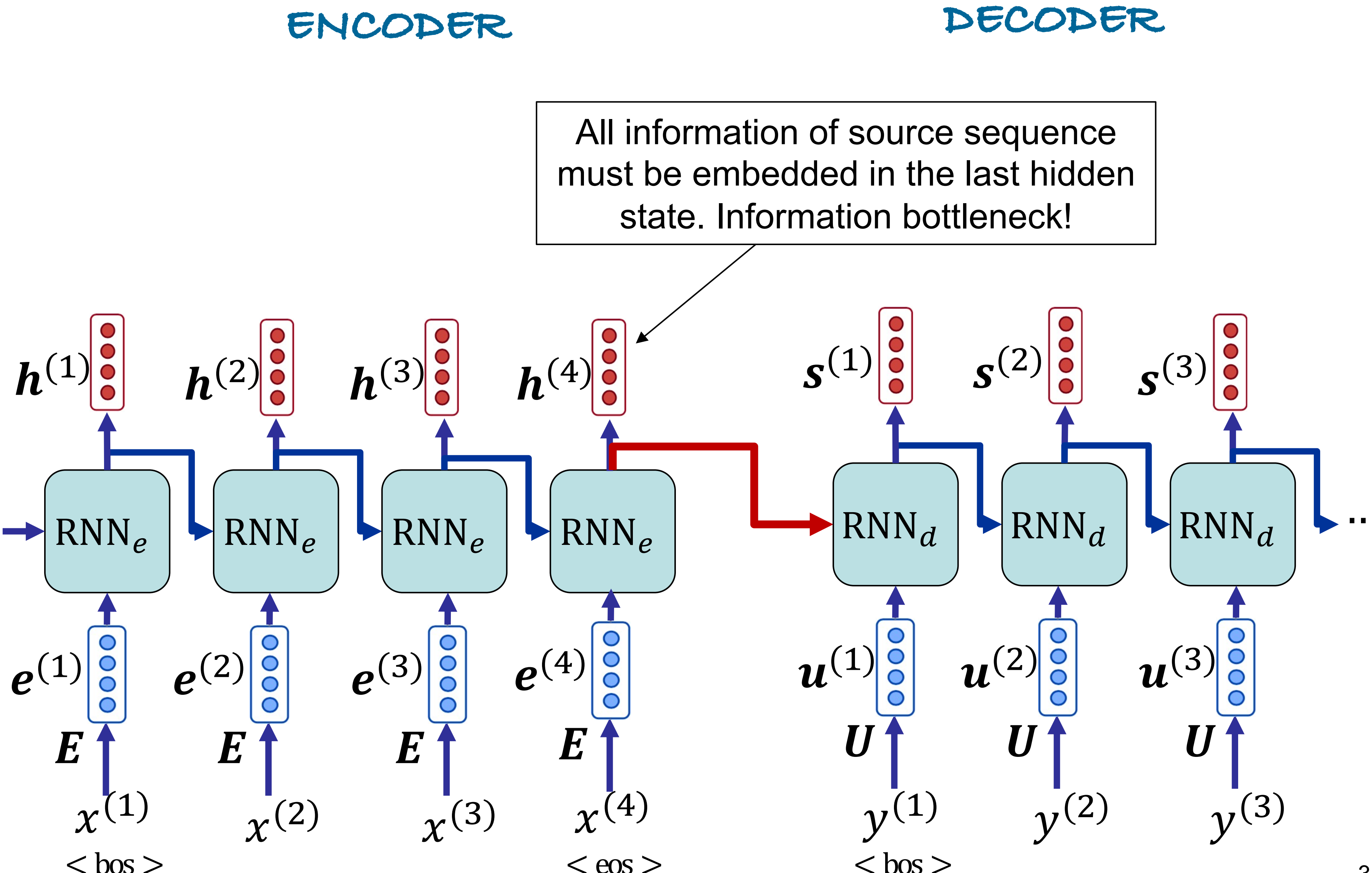
Sentence-level semantic representations (recap)

- Two-dimensional projection of the last hidden states $\boldsymbol{h}^{(L)}$ of RNN_e, obtained from different phrases



Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. "Sequence to Sequence Learning with Neural Networks." *Advances in Neural Information Processing Systems* 27 (2014): 3104-3112.

Bottleneck problem in seq2seq with two RNNs

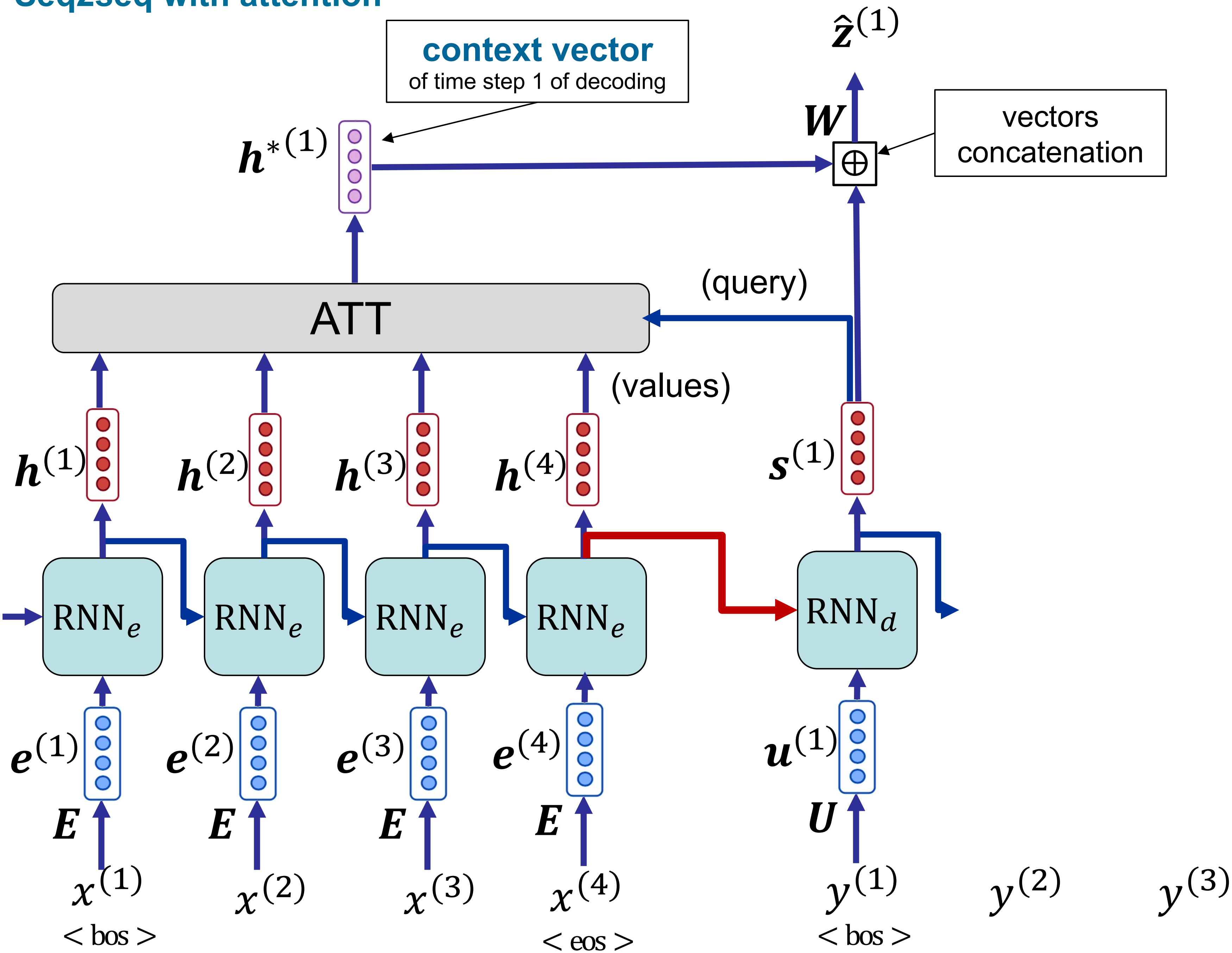


Seq2seq + Attention

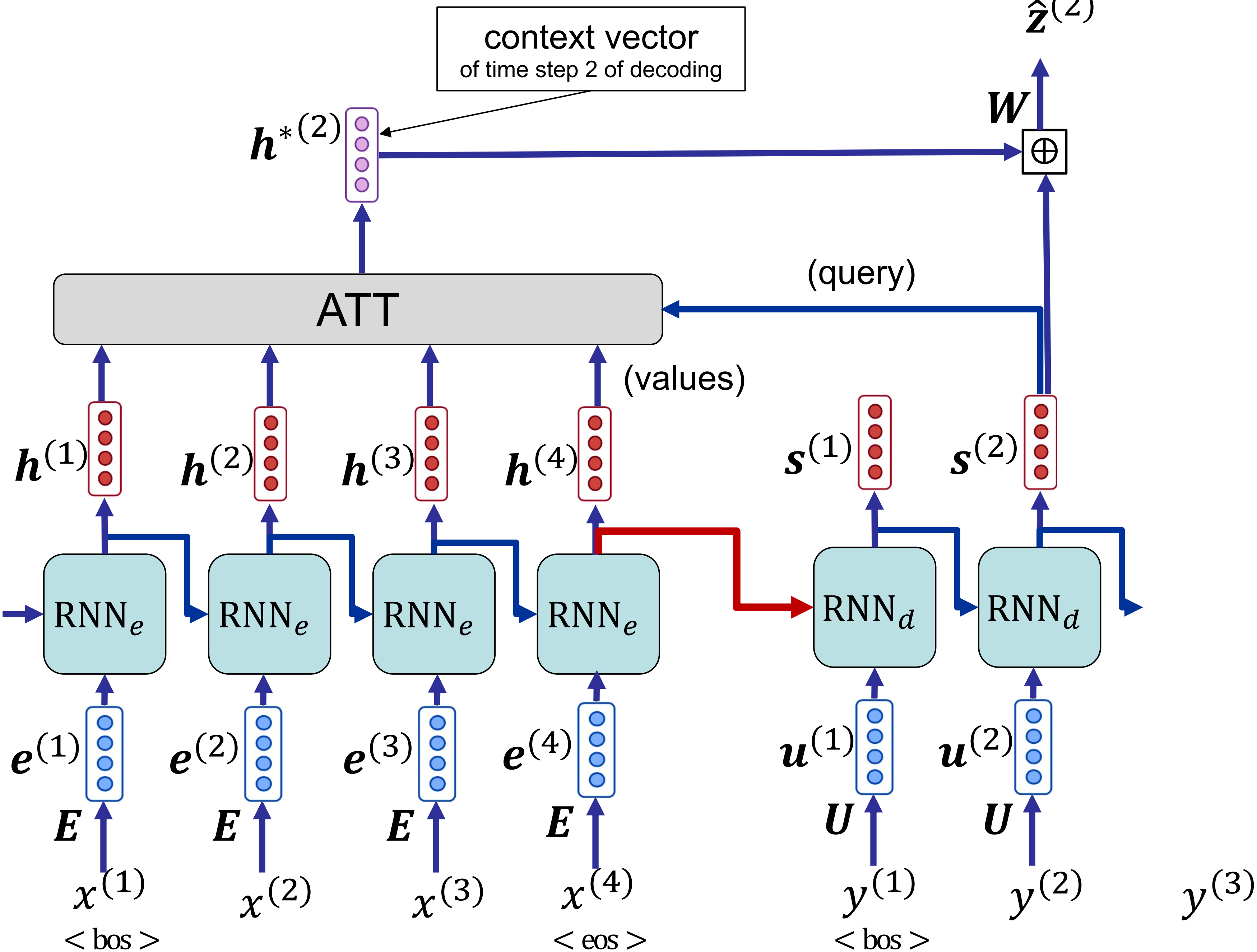
- It can be useful, if we allow decoder the direct access to all elements of source sequence,
  - Decoder can decide on which element of source sequence, it wants to put attention
- Attention is a solution to the bottleneck problem
- Seq2seq with attention …
  - adds an attention network to the architecture of basic seq2seq (two RNNs)
  - At each time step, decoder uses the attention network to attend to all contextualized vectors of the source sequence
  - Training and inference (decoding) processes are the same as basic seq2seq

Bahdanau, Dzmitry, Kyung Hyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." *3rd International Conference on Learning Representations, ICLR*. 2015.

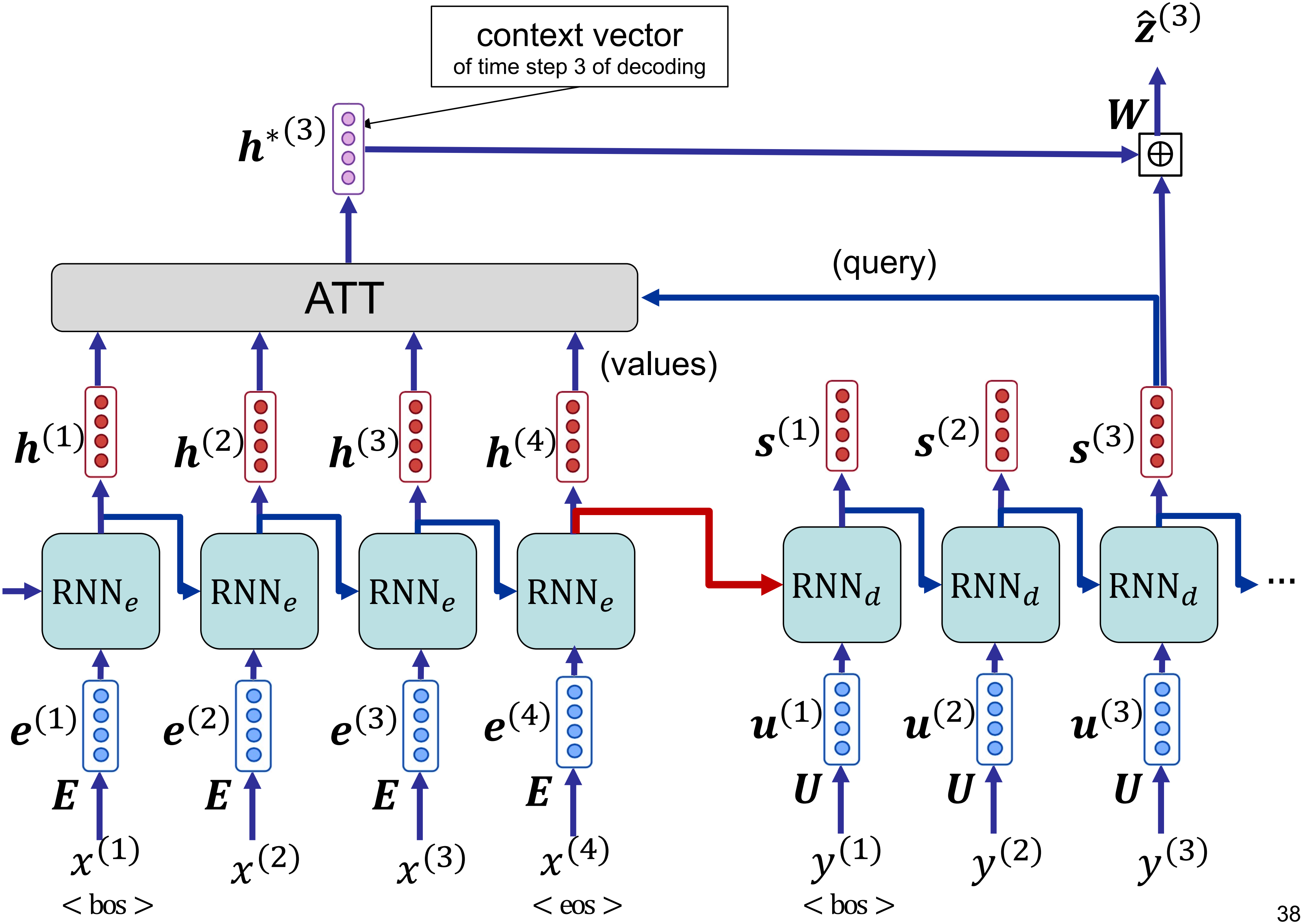
Seq2seq with attention



Seq2seq with attention



Seq2seq with attention



Seq2seq with attention – formulation

- Two sets of vocabularies
  - $\mathbb{V}_e$ is the set of vocabularies for source sequences
  - $\mathbb{V}_d$ is the set of vocabularies for target sequences

Encoder

- From words to word embeddings:
  - Encoder embeddings of source words $(\mathbb{V}_e) \rightarrow \boldsymbol{E}$
  - Embedding of the source word $x^{(l)}$ (at time step l) $\rightarrow \boldsymbol{e}^{(l)}$

- Encoder RNN:

$$\boldsymbol{h}^{(l)} = \mathrm{RNN}_e\left(\boldsymbol{h}^{(l-1)}, \boldsymbol{e}^{(l)}\right)$$

Parameters are shown in red

Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. "Sequence to Sequence Learning with Neural Networks." *Advances in Neural Information Processing Systems* 27 (2014): 3104-3112.

Seq2seq with attention – formulation

Decoder

- From words to word embeddings:
  - Decoder embeddings of target words ($\mathbb{V}_d$) at input $\rightarrow$ $\boldsymbol{U}$
  - Embedding of the target word $y^{(t)}$ at time step $t \rightarrow \boldsymbol{u}^{(t)}$
- Decoder RNN: $\boldsymbol{s}^{(t)} = \mathrm{RNN}_d(\boldsymbol{s}^{(t-1)}, \boldsymbol{u}^{(t)})$
  - where the initial hidden state of the decoder RNN is set to the last hidden state of the encoder RNN: $\boldsymbol{s}^{(0)} = \boldsymbol{h}^{(L)}$

Parameters are shown in red

Seq2seq with attention – formulation

Decoder (cont.)

- Attention context vector

$$\boldsymbol{h}^{*(t)} = \mathrm{ATT}\left(\boldsymbol{s}^{(t)}, \left\{\boldsymbol{h}^{(1)}, \dots, \boldsymbol{h}^{(L)}\right\}\right)$$

For instance, if ATT is a "basic dot-product attention", this is done by:

- First calculating non-normalized attentions:

$$\tilde{\alpha}_l^{(t)} = \boldsymbol{s}^{(t)^{\mathrm{T}}} \boldsymbol{h}_l$$

- Then, normalizing the attentions:

$$\boldsymbol{\alpha}^{(t)} = \mathrm{softmax}(\tilde{\boldsymbol{\alpha}}^{(t)})$$

- and finally calculating the weighted sum of encoder hidden states

$$\boldsymbol{h}^{*(t)} = \sum_{l=1}^{\boldsymbol{L}} \alpha_l^{(t)} \boldsymbol{h}_l$$

Parameters are shown in red

Seq2seq with attention – formulation

Decoder (cont.)

- Decoder output prediction
  - Predicted probability distribution of words at the next time step:

$$\hat{\boldsymbol{z}}^{(t)} = \mathrm{softmax}\left(\boldsymbol{W}\left[\boldsymbol{s}^{(t)}; \boldsymbol{h}^{*(t)}\right] + \boldsymbol{b}\right) \in \mathbb{R}^{|\mathbb{V}_d|}$$

$[;]$ denotes the concatenation of two vectors

  - Probability of the next target word (at time step $t+1$):

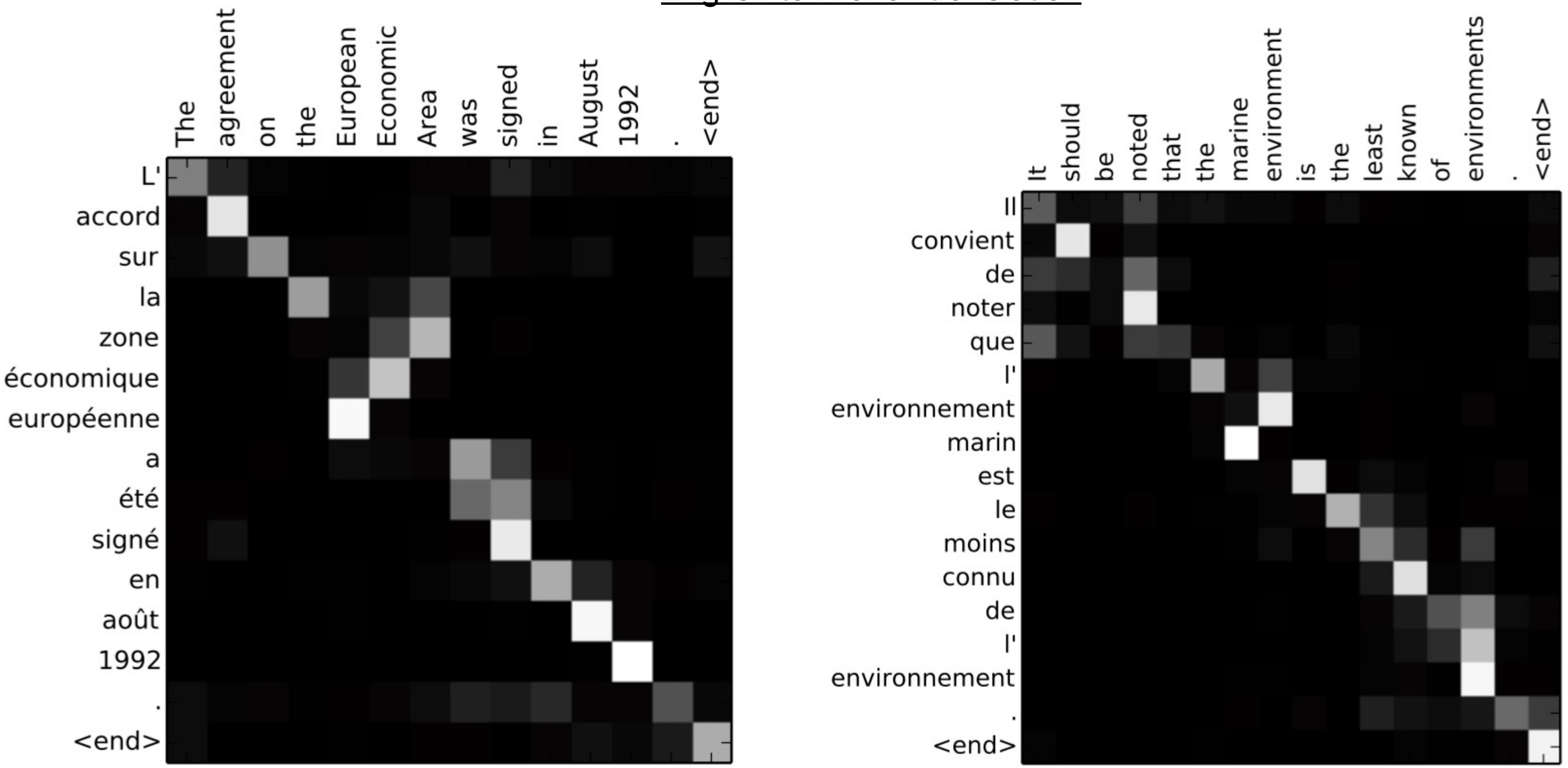
$$P\left(y^{(t+1)} \middle| X, y^{(1)}, \ldots, y^{(t-1)}, y^{(t)}\right) = \hat{z}^{(t)}_{y^{(t+1)}}$$

Parameters are shown in red

Alignment in NMT (seq2seq with attention)

- Attention automatically learns (nearly) alignment

English to French translation



Bahdanau et al. [2015]
Try more here: https://distill.pub/2016/augmented-rnns/#attentional-interfaces

Seq2seq with attention – summary

- Attention on source sequence facilitates the focus on relevant words and a better flow of information

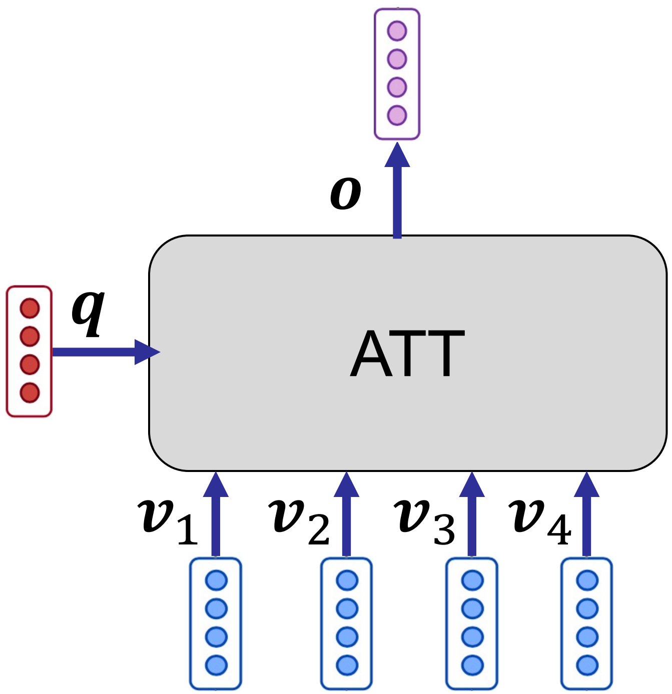
- Adding the attention network also helps avoiding vanishing gradient problem by providing a shortcut to faraway states

Agenda

- Attention Networks
- Machine Translation
  - Seq2seq with Attention
- **Hierarchical document classification**

Attention in practice

- Attention is used to create a compositional embedding of value vectors according to a query
  - as we already saw in seq2seq models …
  - but it can also in tasks like sequence classification

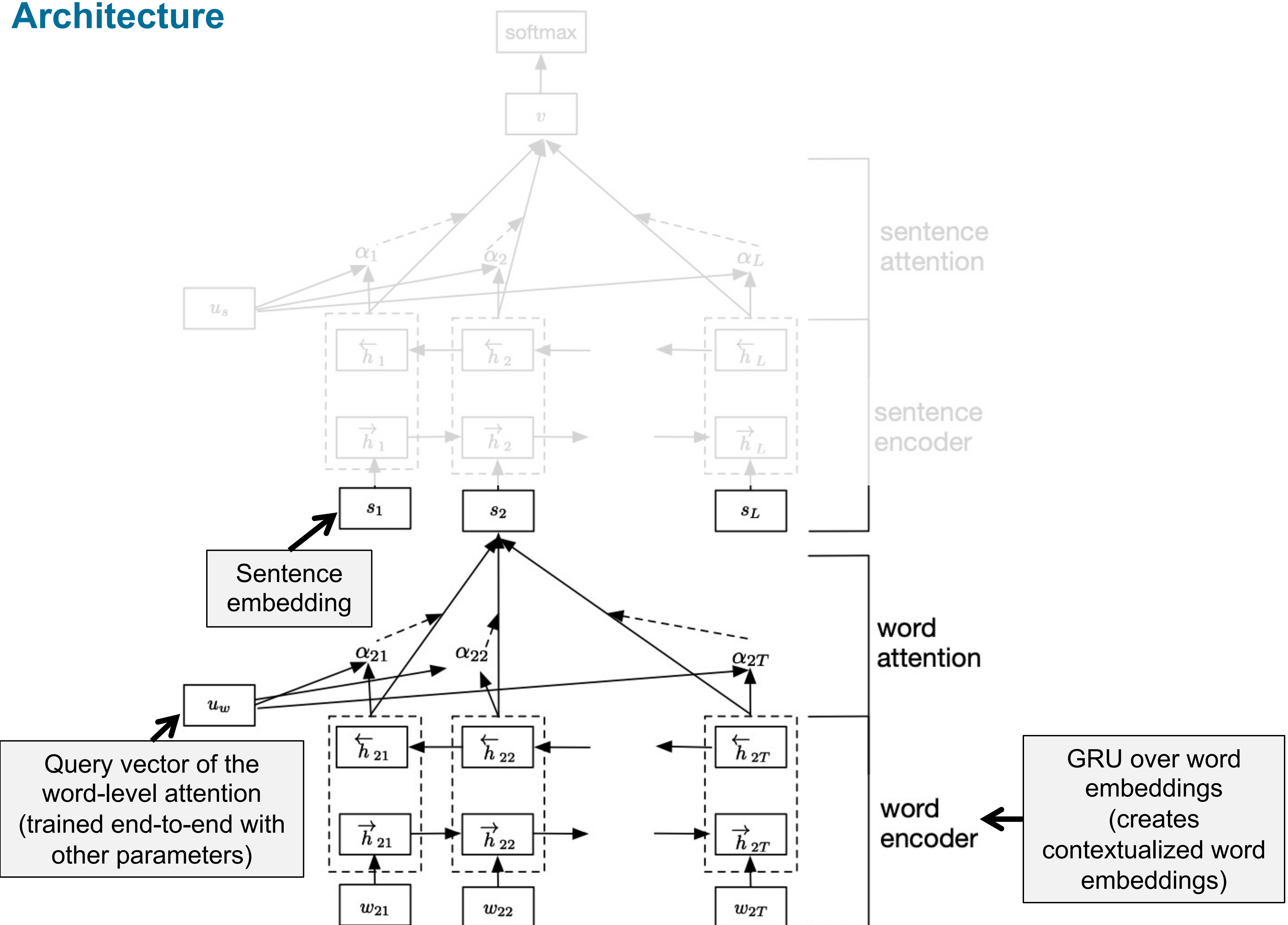


Hierarchical document classification with attention

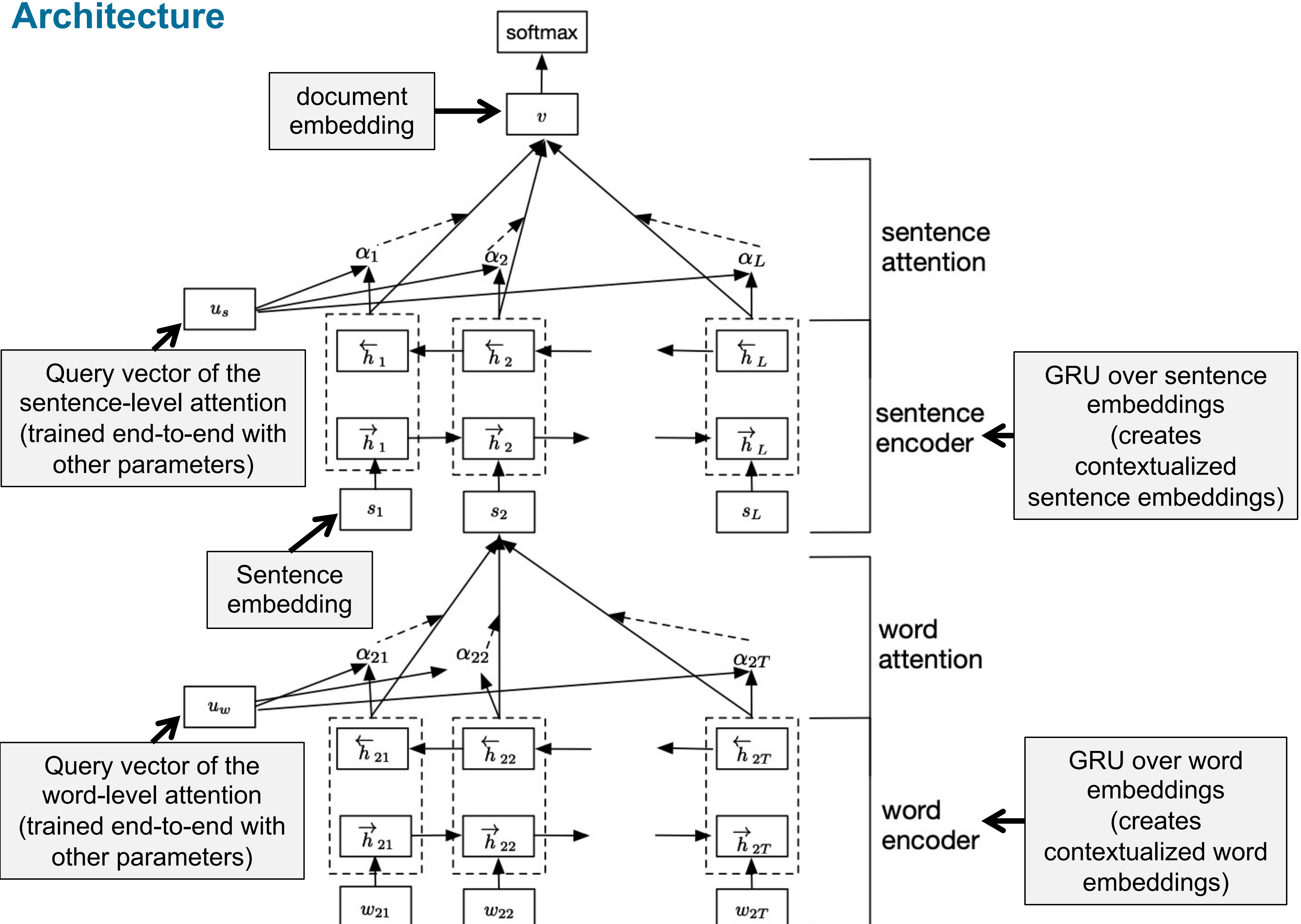
- Document classification with attention
  - An attention network is applied to word embeddings as **values** (inputs) to compose a document vector (output)
    - Document embedding is then used as features for classification
  - The **query** of the attention network is a randomly initialized parameter vector, whose weights are trained end-to-end with the model
- Hierarchical document classification
  - Split the document into sentences
  - Use a word-level attention to create a sentence embedding from the word embeddings of each sentence
  - Use a sentence-level attention to create the document embedding from the sentence embeddings

Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., & Hovy, E. (2016, June). Hierarchical attention networks for document classification. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*

Architecture



Architecture



Examples

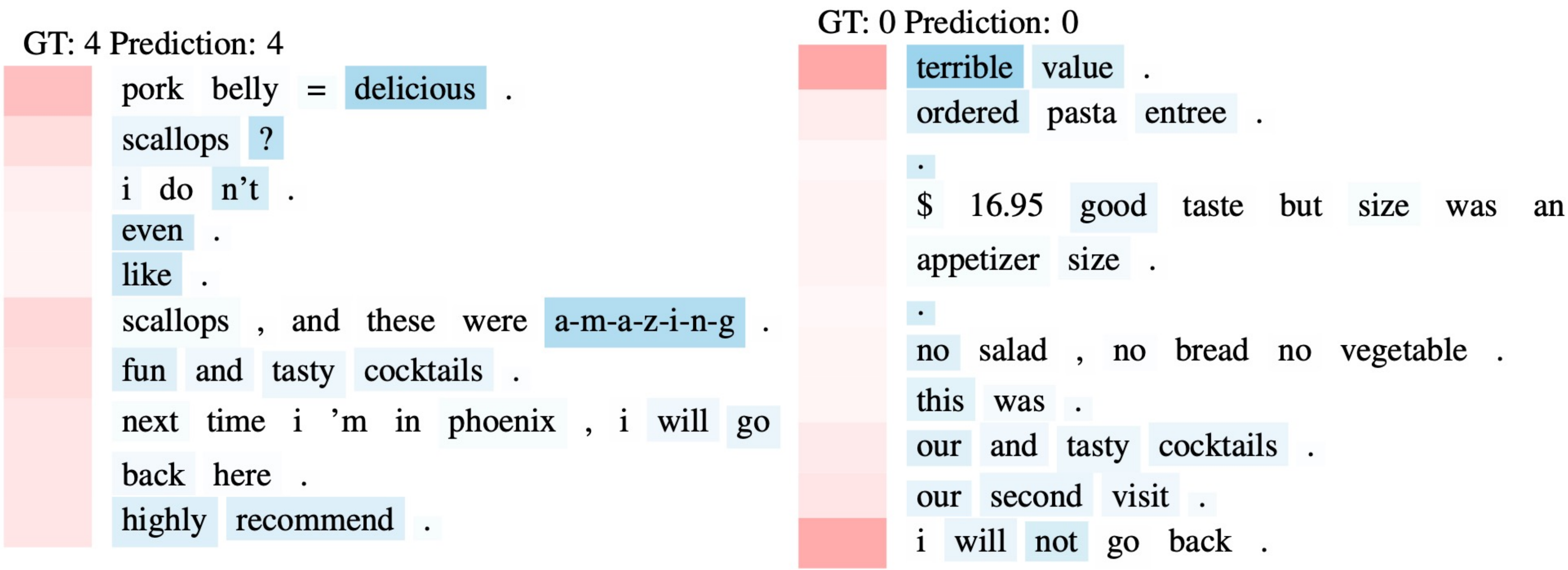


Figure 5: Documents from Yelp 2013. Label 4 means star 5, label 0 means star 1.

https://www.aclweb.org/anthology/N16-1174.pdf

Example

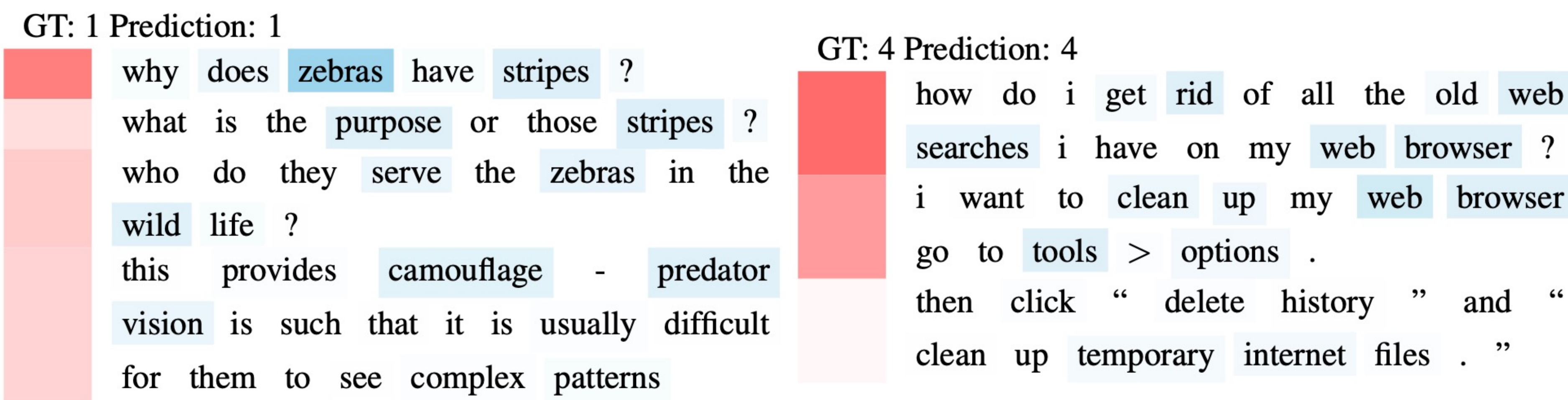


Figure 6: Documents from Yahoo Answers. Label 1 denotes Science and Mathematics and label 4 denotes Computers and Internet.

https://www.aclweb.org/anthology/N16-1174.pdf

Sequence classification with attention – summary

- Attention can be used to compose a sequence vector from its token vectors
  - In this case, the query vector is a set of parameters that will be trained with other model parameters
  - The composed vector is in fact the weighted average of the token vectors based on attention weights

- Attention provides some interpretability
  - Looking at attention distributions, one may assume what the model is focusing on
  - We should however be careful about directly taking attention distributions as model explanations (particularly in Transformers)!
    - Jain, Sarthak, and Byron C. Wallace. "Attention is not Explanation." *In proc. of NAACL-HTL* 2019. https://www.aclweb.org/anthology/N19-1357.pdf
    - Wiegreffe, Sarah, and Yuval Pinter. "Attention is not not Explanation." *In proc. of EMNLP-IJCNLP*. 2019. https://www.aclweb.org/anthology/D19-1002/