CS11-737 Multilingual NLP

Machine Translation/ Sequence-to-sequence Models

Graham Neubig

Carnegie Mellon University
Language Technologies Institute

Site
http://phontron.com/class/multiling2022/

Language Models

- Language models are generative models of text

$$s \sim P(x)$$

↓

"The Malfoys!" said Hermione.

Harry was watching him. He looked like Madame Maxime. When she strode up the wrong staircase to visit himself.

"I'm afraid I've definitely been suspended from power, no chance—indeed?" said Snape. He put his head back behind them and read groups as they crossed a corner and fluttered down onto their ink lamp, and picked up his spoon. The doorbell rang. It was a lot cleaner down in London.

Text Credit: Max Deutsch (https://medium.com/deep-writing/)

Conditioned Language Models

- Not just generate text, generate text according to some specification

Input *X*	Output *Y* (**Text**)	Task
Structured Data	NL Description	NL Generation
English	Japanese	Translation
Document	Short Description	Summarization
Utterance	Response	Response Generation
Image	Text	Image Captioning
Speech	Transcript	Speech Recognition

Formulation and Modeling

Calculating the Probability of a Sentence

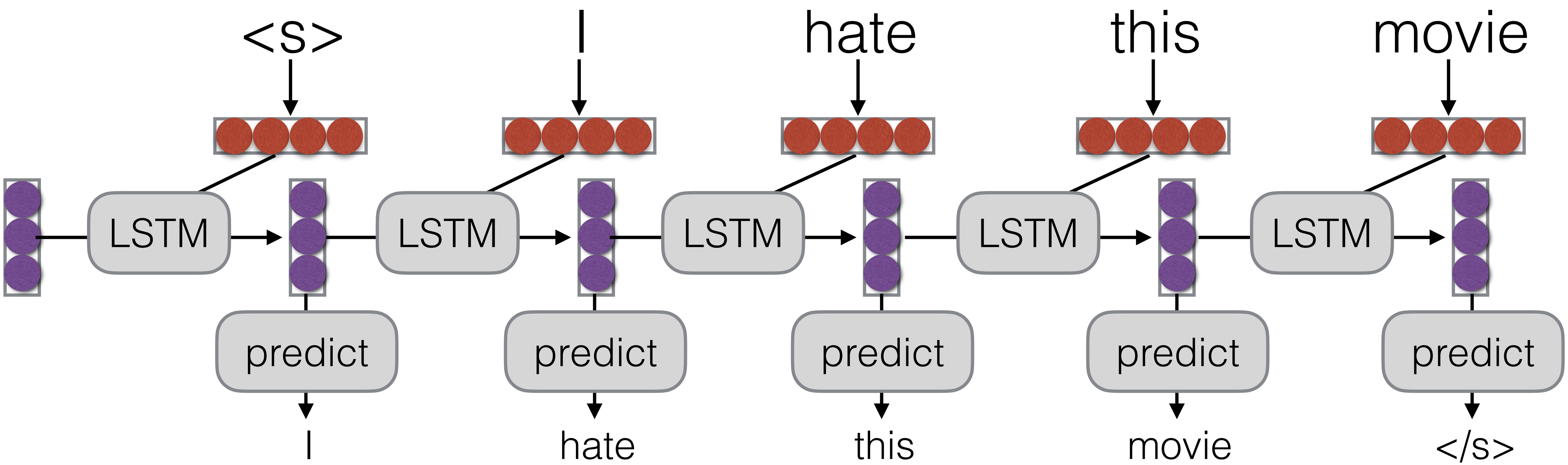
$$P(X) = \prod_{i=1}^{I} P(\underbrace{x_i}_{\text{Next Word}} \mid \underbrace{x_1, \ldots, x_{i-1}}_{\text{Context}})$$

Conditional Language Models

$$P(Y|X) = \prod_{j=1}^{J} P(y_j \mid \underline{X}, y_1, \ldots, y_{j-1})$$

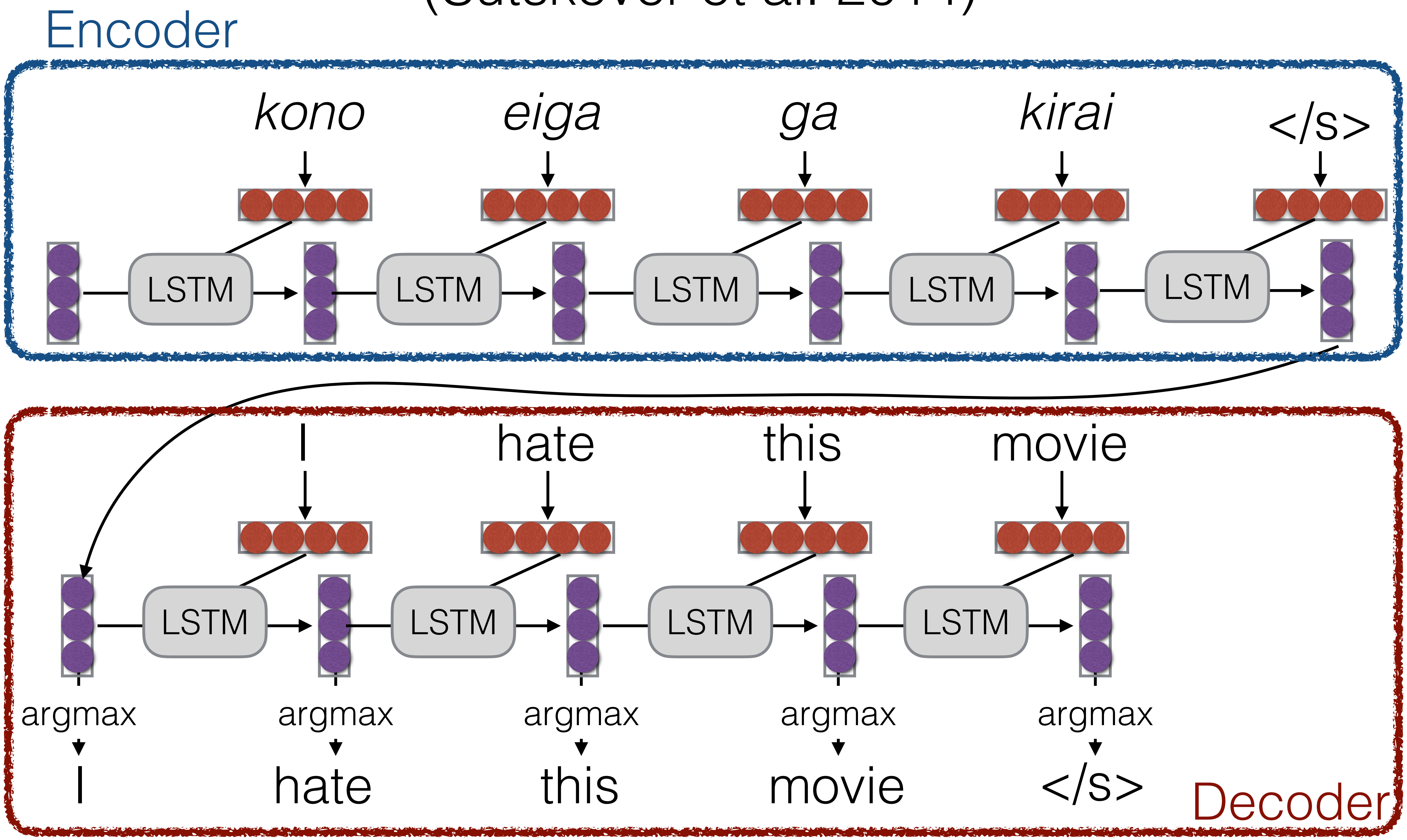
Added Context!

(One Type of) Language Model (Mikolov et al. 2011)



Mikolov, Tomáš, et al. "Extensions of recurrent neural network language model." *2011 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2011.

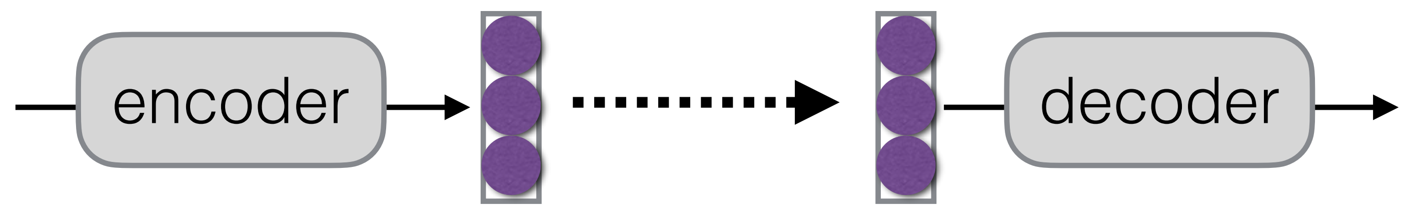
(One Type of) Conditional Language Model
(Sutskever et al. 2014)



Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. "Sequence to sequence learning with neural networks." *Advances in neural information processing systems*. 2014.

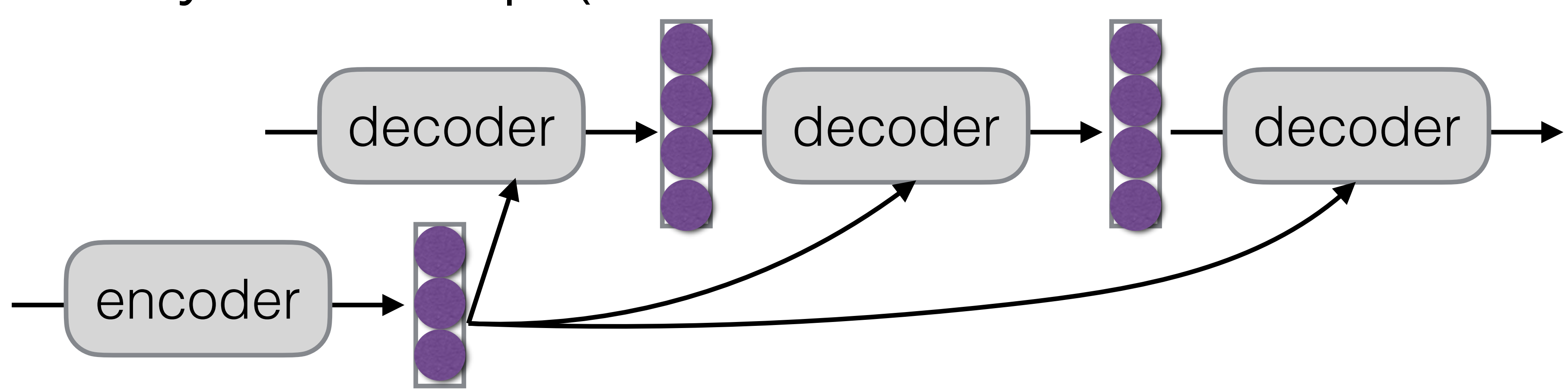
How to Pass Hidden State?

- Initialize decoder w/ encoder (Sutskever et al. 2014)



- Transform (can be different dimensions)

- Input at every time step (Kalchbrenner & Blunsom 2013)



Kalchbrenner, Nal, and Phil Blunsom. "Recurrent continuous translation models." *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. 2013.

Methods of Generation

The Generation Problem

- We have a model of P(Y|X), how do we use it to generate a sentence?
- Two methods:
  - **Sampling:** Try to generate a *random* sentence according to the probability distribution.
  - **Argmax:** Try to generate the sentence with the *highest* probability.

Ancestral Sampling

- **Randomly generate** words one-by-one.

$$
\boxed{\begin{array}{l} \text{while } y_{j-1} \mathrel{!}= \text{“</s>”:} \\ \quad y_j \sim P(y_j \mid X, y_1, \ldots, y_{j-1}) \end{array}}
$$

- An **exact method** for sampling from P(Y|X), no further work needed.

Greedy Search

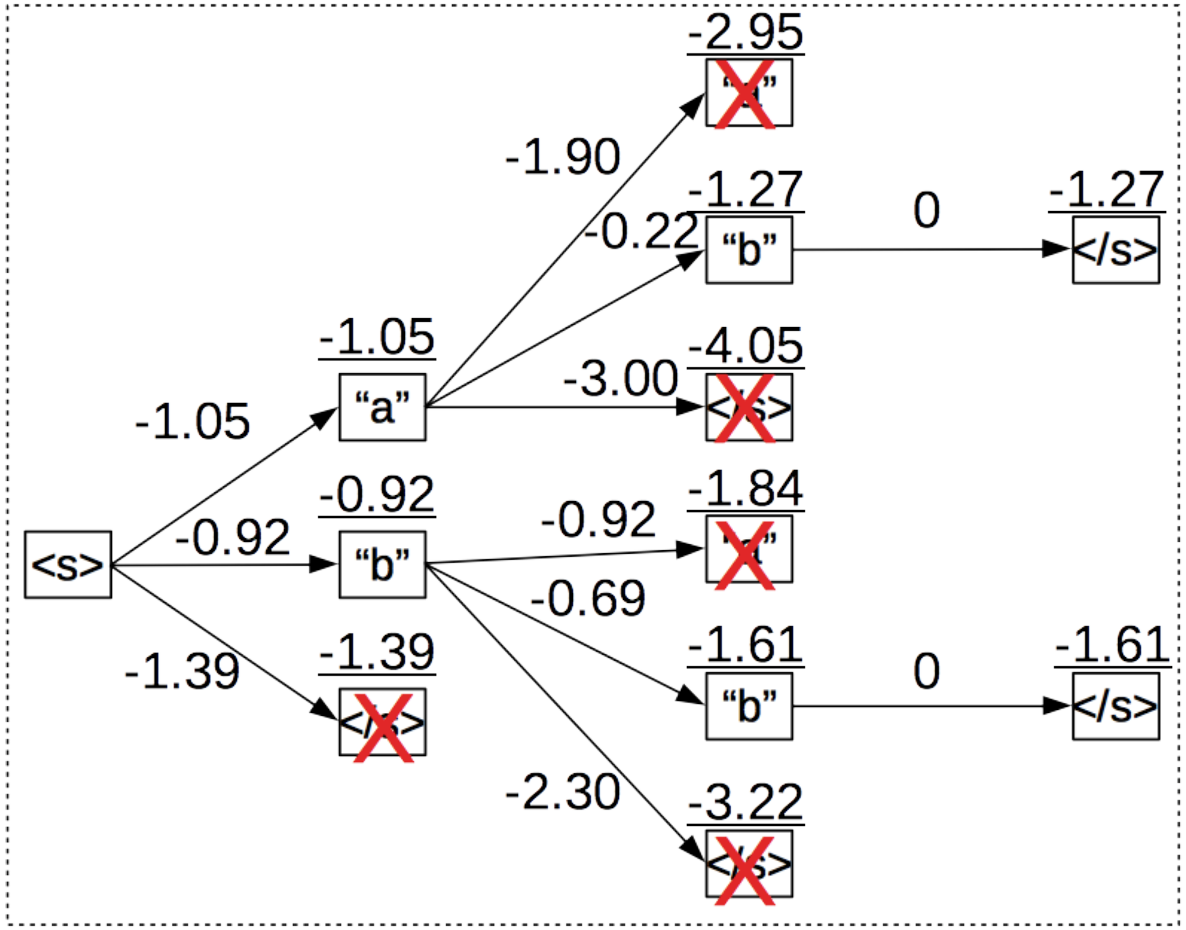
- One by one, pick the single highest-probability word

```
while y_{j-1} != "</s>":
  y_j = argmax P(y_j | X, y_1, …, y_{j-1})
```

- **Not exact, real problems:**
  - Will often generate the "easy" words first
  - Will prefer multiple common words to one rare word

Beam Search

- Instead of picking one high-probability word, maintain several paths



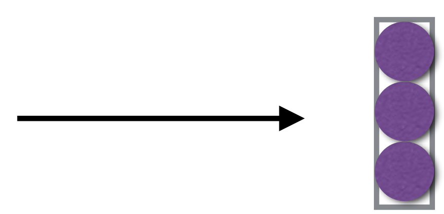
Attention

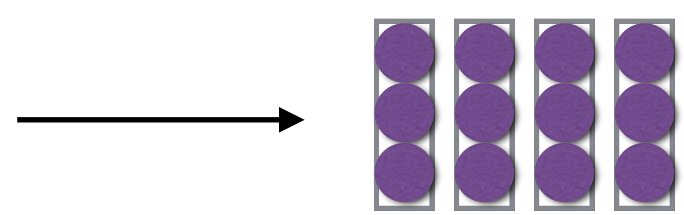
Sentence Representations

Problem!

> "You can't cram the meaning of a whole %&!$ing sentence into a single $&!*ing vector!"
> — Ray Mooney

- But what if we could use multiple vectors, based on the length of the sentence.

this is an example 

this is an example 

Attention: Basic Idea

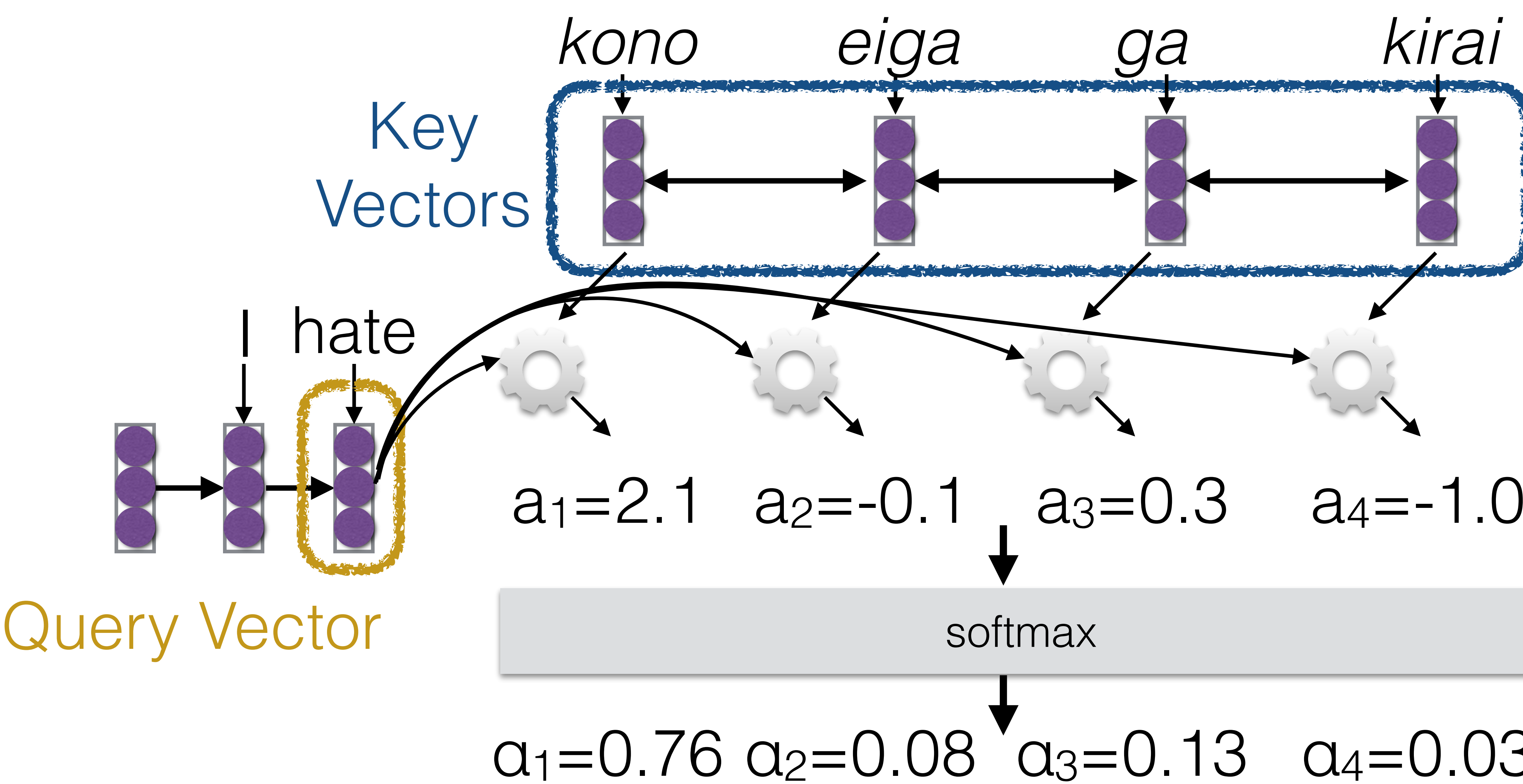
(Bahdanau et al. 2015)

- Encode each word in the sentence into a vector
- When decoding, perform a linear combination of these vectors, weighted by “attention weights”
- Use this combination in picking the next word

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." ICLR 2015.

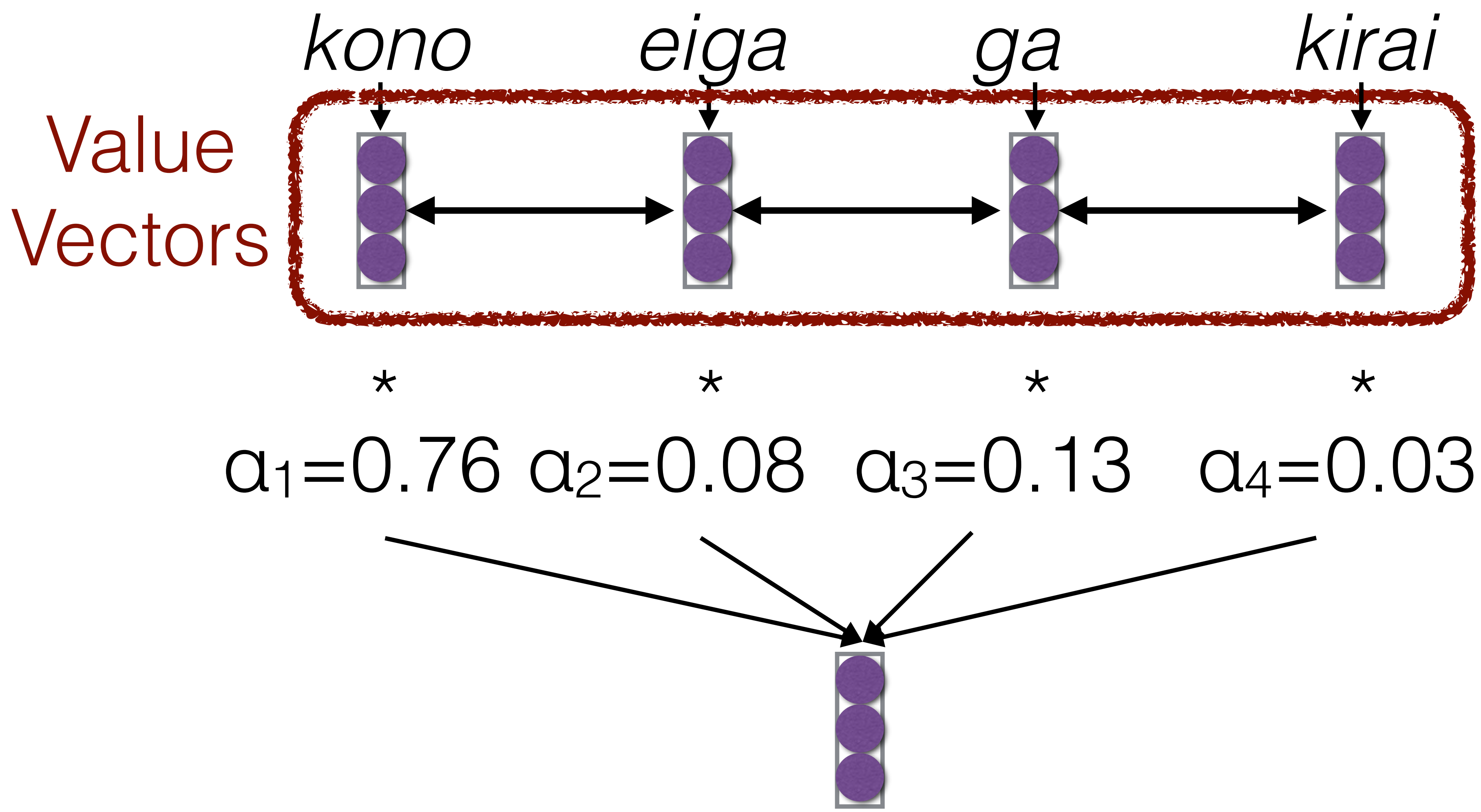
Calculating Attention (1)

- Use “query” vector (decoder state) and “key” vectors (all encoder states)
- For each query-key pair, calculate weight
- Normalize to add to one using softmax



Calculating Attention (2)

- Combine together value vectors (usually encoder states, like key vectors) by taking the weighted sum



- Use this in any part of the model you like

A Graphical Example

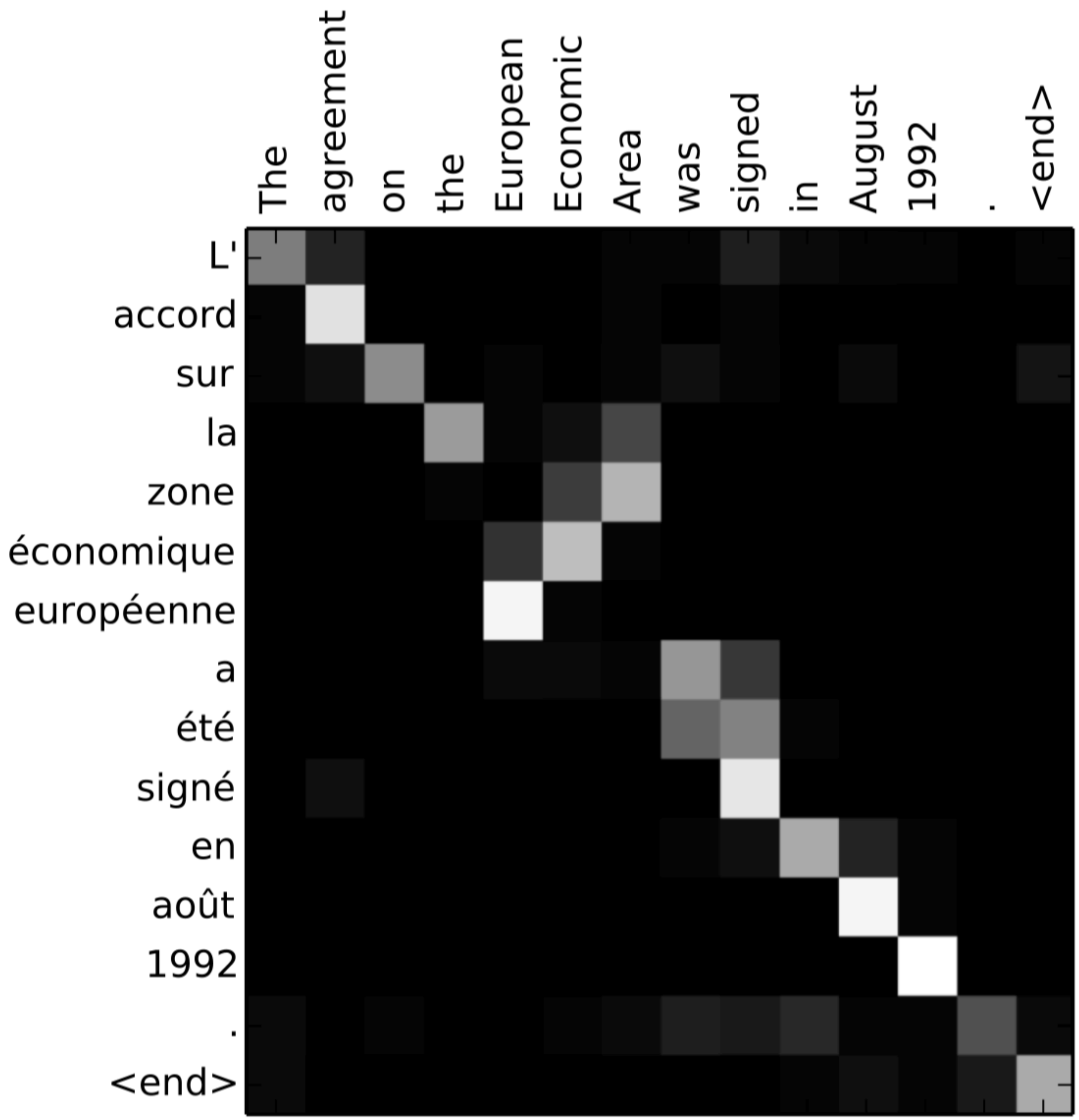


Image from Bahdanau et al. (2015)

Attention Score Functions (1)

- $\boldsymbol{q}$ is the query and $\boldsymbol{k}$ is the key
- **Multi-layer Perceptron** (Bahdanau et al. 2015)

$$a(\boldsymbol{q}, \boldsymbol{k}) = \boldsymbol{w}_2^\intercal \tanh(W_1[\boldsymbol{q}; \boldsymbol{k}])$$

  - Flexible, often very good with large data
- **Bilinear** (Luong et al. 2015)

$$a(\boldsymbol{q}, \boldsymbol{k}) = \boldsymbol{q}^\intercal W \boldsymbol{k}$$

Luong, Minh-Thang, Hieu Pham, and Christopher D. Manning. "Effective approaches to attention-based neural machine translation." *EMNLP 2015*.

Attention Score Functions (2)

- **Dot Product** (Luong et al. 2015)

$$a(\boldsymbol{q}, \boldsymbol{k}) = \boldsymbol{q}^{\intercal}\boldsymbol{k}$$

  - No parameters! But requires sizes to be the same.

- **Scaled Dot Product** (Vaswani et al. 2017)
  - *Problem:* scale of dot product increases as dimensions get larger
  - *Fix:* scale by size of the vector

$$a(\boldsymbol{q}, \boldsymbol{k}) = \frac{\boldsymbol{q}^{\intercal}\boldsymbol{k}}{\sqrt{|\boldsymbol{k}|}}$$

Improvements to Attention

Coverage

- **Problem:** Neural models tends to drop or repeat content
- **Solution:** Model how many times words have been covered
  - Impose a penalty if attention not approx.1 over each word (Cohn et al. 2015)
  - Add embeddings indicating coverage (Mi et al. 2016)

Cohn, Trevor, et al. "Incorporating structural alignment biases into an attentional neural translation model." *NAACL 2016*.
Mi, Haitao, et al. "Coverage embedding models for neural machine translation." EMNLP 2016.

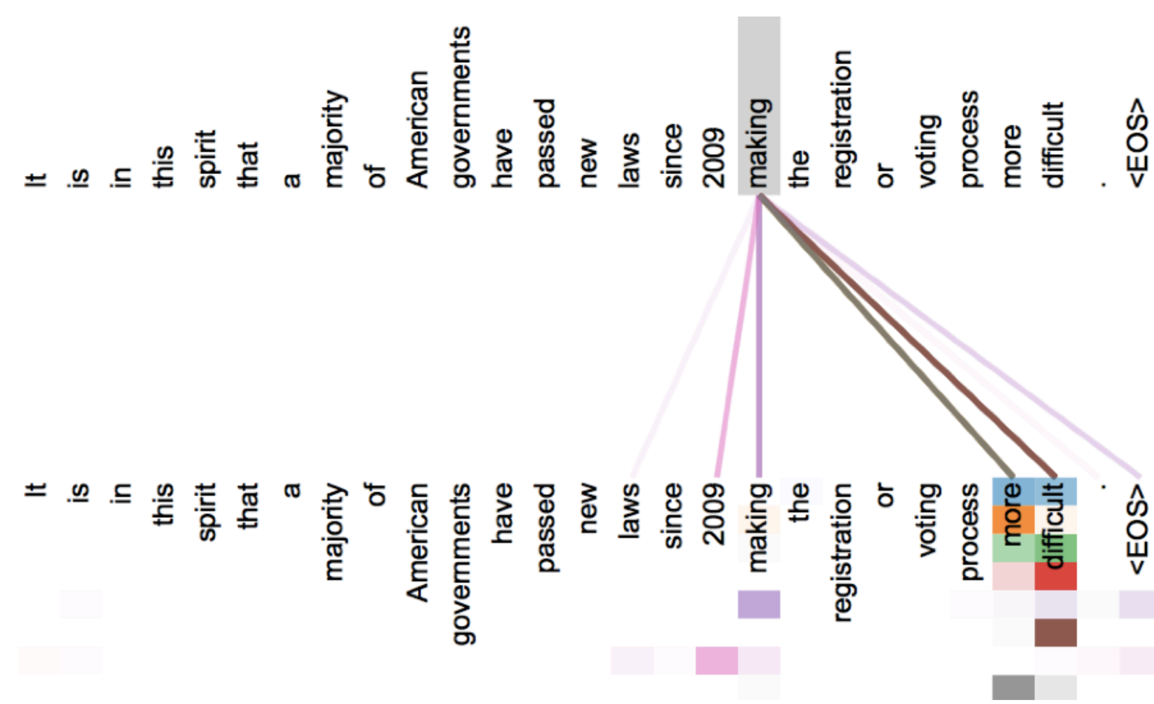
Multi-headed Attention

- **Idea:** multiple attention “heads” focus on different parts of the sentence

- e.g. Different heads for “copy” vs regular (Allamanis et al. 2016)

	Target		Attention Vectors	λ
m_1	set	$\boldsymbol{\alpha} =$	<s> { this . use Browser Cache = use Browser Cache ; } </s>	0.012
		$\boldsymbol{\kappa} =$	<s> { this . use Browser Cache = use Browser Cache ; } </s>	
m_2	use	$\boldsymbol{\alpha} =$	<s> { this . use Browser Cache = use Browser Cache ; } </s>	0.974
		$\boldsymbol{\kappa} =$	<s> { this . use Browser Cache = use Browser Cache ; } </s>	
m_3	browser	$\boldsymbol{\alpha} =$	<s> { this . use Browser Cache = use Browser Cache ; } </s>	0.969
		$\boldsymbol{\kappa} =$	<s> { this . use Browser Cache = use Browser Cache ; } </s>	
m_4	cache	$\boldsymbol{\alpha} =$	<s> { this . use Browser Cache = use Browser Cache ; } </s>	0.583
		$\boldsymbol{\kappa} =$	<s> { this . use Browser Cache = use Browser Cache ; } </s>	
m_5	END	$\boldsymbol{\alpha} =$	<s> { this . use Browser Cache = use Browser Cache ; } </s>	0.066
		$\boldsymbol{\kappa} =$	<s> { this . use Browser Cache = use Browser Cache ; } </s>	

- Or multiple independently learned heads (Vaswani et al. 2017)



Allamanis, Miltiadis, Hao Peng, and Charles Sutton. "A convolutional attention network for extreme summarization of source code." *ICML* 2016.
Vaswani, Ashish, et al. "Attention is all you need." NeurIPS 2017.

Supervised Training
(Liu et al. 2016)

- Sometimes we can get “gold standard” alignments *a-priori*
    - Manual alignments
    - Pre-trained with strong alignment model
- **Train the model to match** these strong alignments

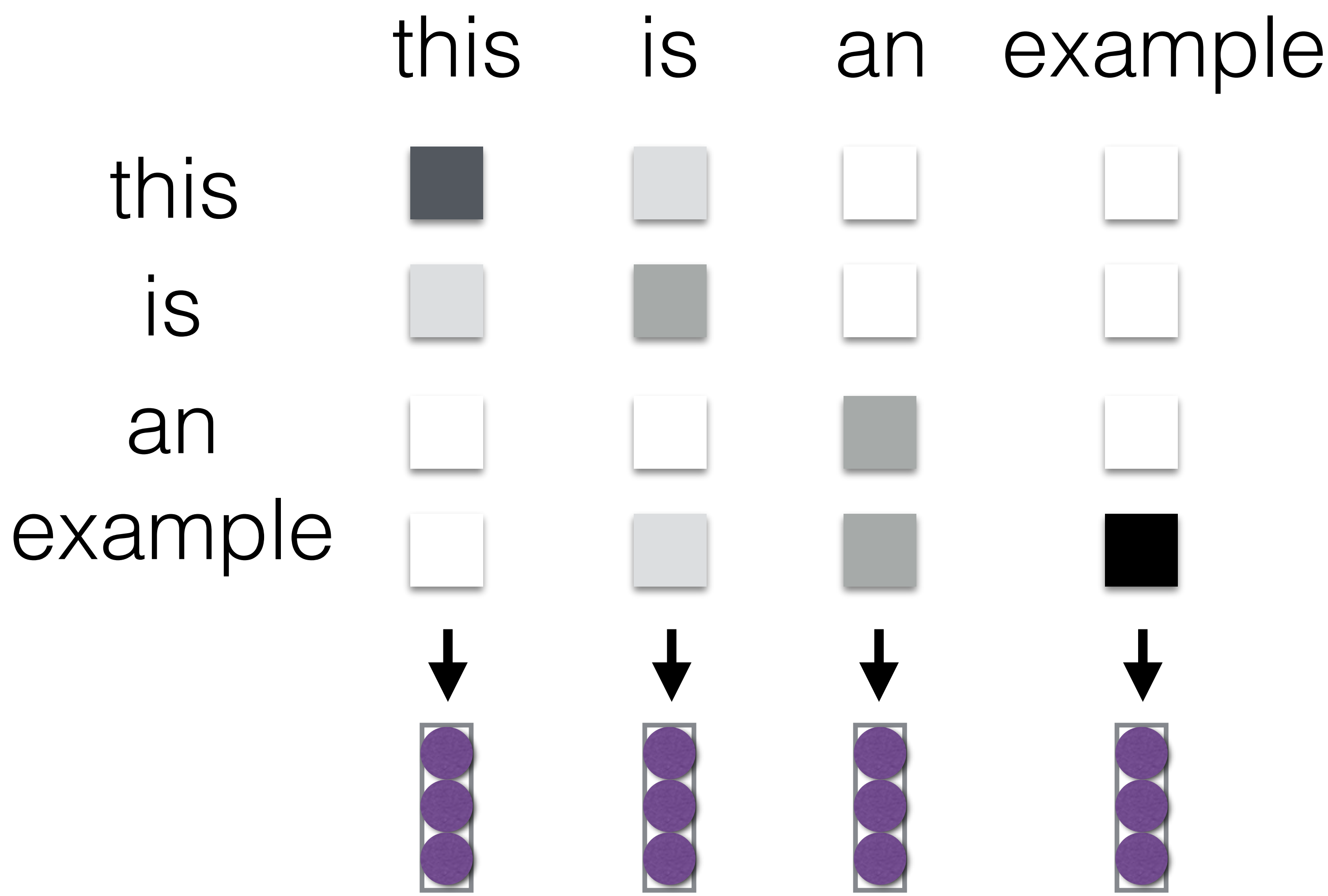
Liu, Lemao, et al. "Neural machine translation with supervised attention." *EMNLP* 2016.

Self Attention/Transformers

Self Attention

(Cheng et al. 2016)

- Each element in the sentence attends to other elements → context sensitive encodings!



- Can be used as drop-in replacement for other sequence models, e.g. RNNs, CNNs

Cheng, Jianpeng, Li Dong, and Mirella Lapata. "Long short-term memory-networks for machine reading." *EMNLP 2016.*

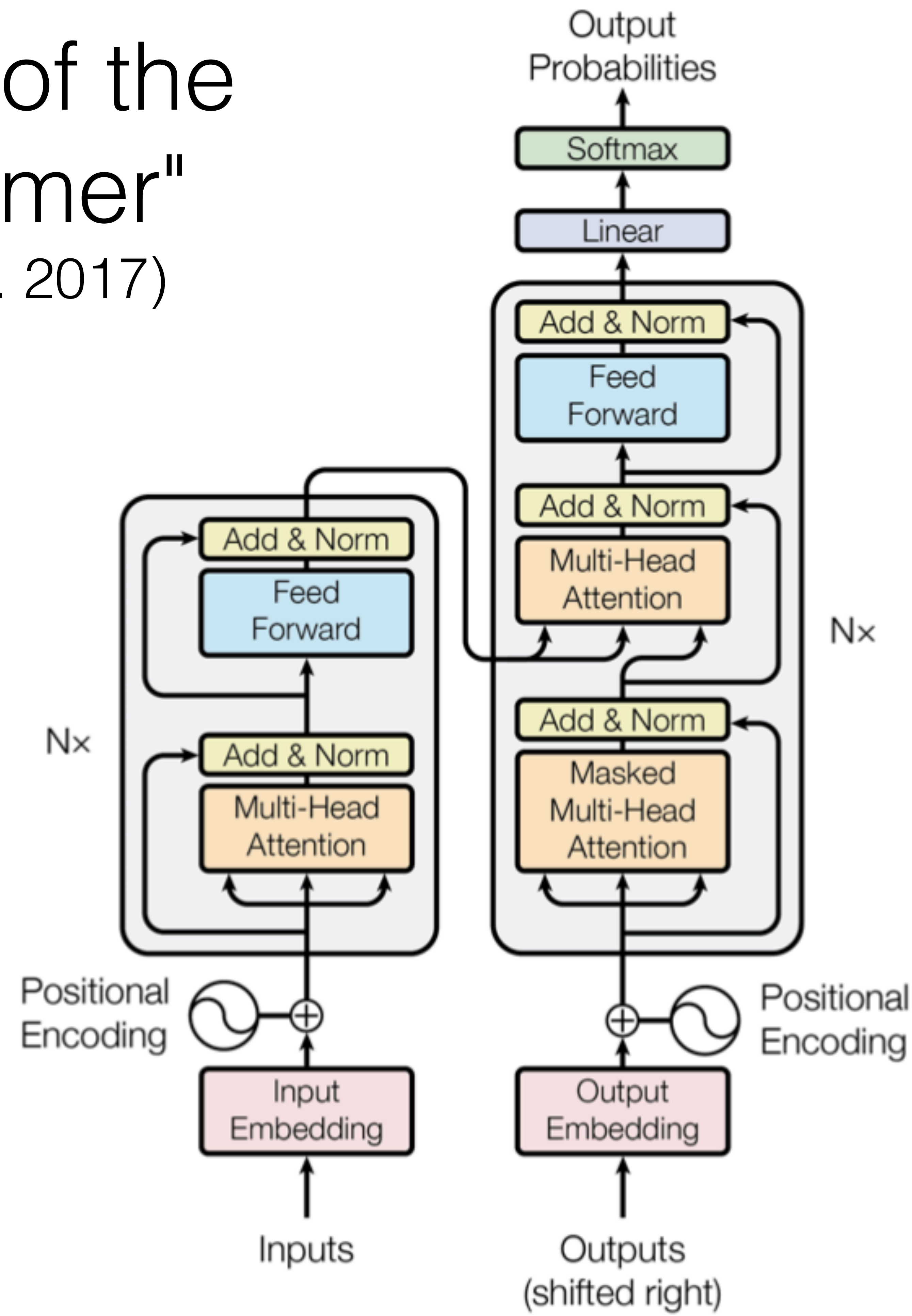
Why Self Attention?

- Unlike RNNs, parallelizable -> fast training on GPUs!
- Unlike CNNs, easily capture global context
- In general, high accuracy, although not 100% clear when all things being held equal (Chen et al. 2018)
- *Downside:* quadratic computation time

Chen, Mia Xu, et al. "The best of both worlds: Combining recent advances in neural machine translation." *ACL 2018*.

Summary of the "Transformer" (Vaswani et al. 2017)

- A sequence-to-sequence model based entirely on attention
- Strong results on standard WMT datasets
- Fast: only matrix multiplications



Transformer Attention Tricks

- **Self Attention:** Each layer combines words with others
- **Multi-headed Attention:** 8 attention heads learned independently
- **Normalized Dot-product Attention:** Remove bias in dot product when using large networks
- **Positional Encodings:** Make sure that even if we don’t have RNN, can still distinguish positions

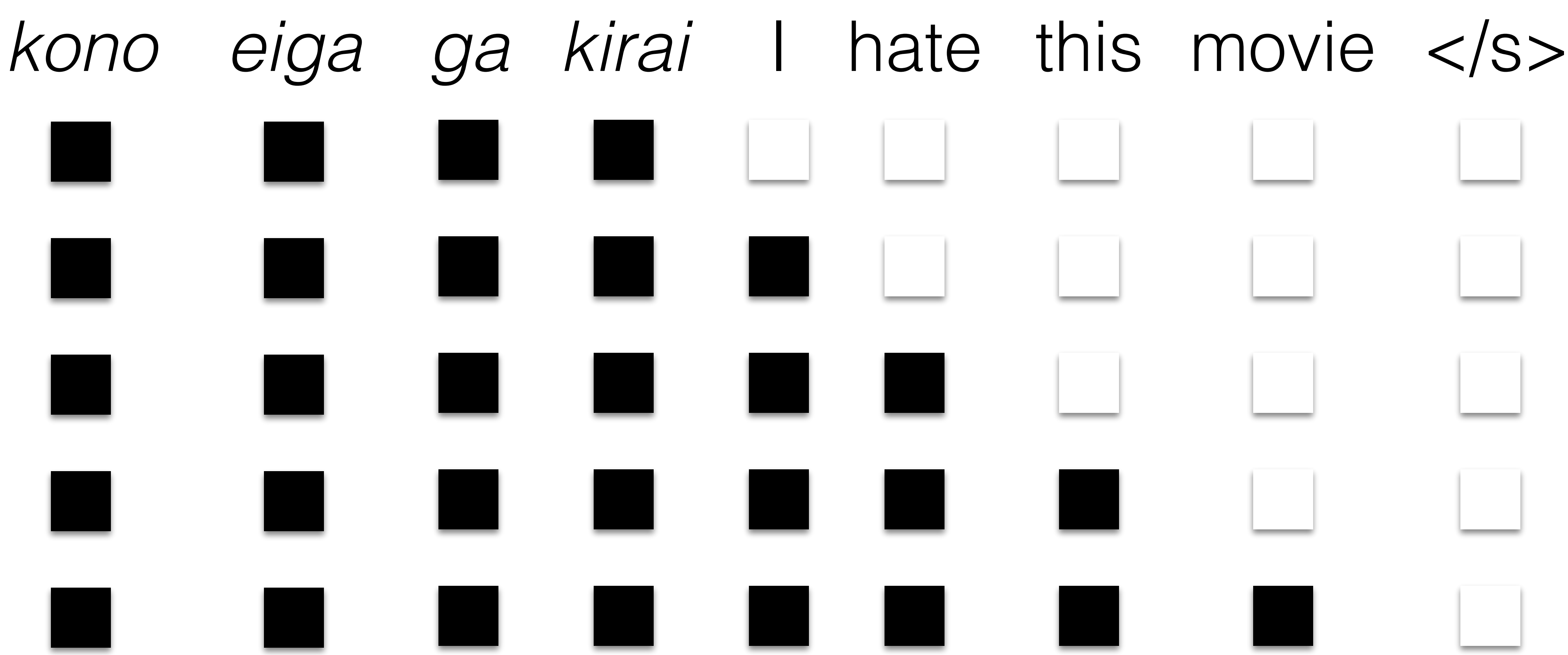
Transformer Training Tricks

- **Layer Normalization:** Help ensure that layers remain in reasonable range
- **Specialized Training Schedule:** Adjust default learning rate of the Adam optimizer
- **Label Smoothing:** Insert some uncertainty in the training process
- **Masking for Efficient Training**

Masking for Training

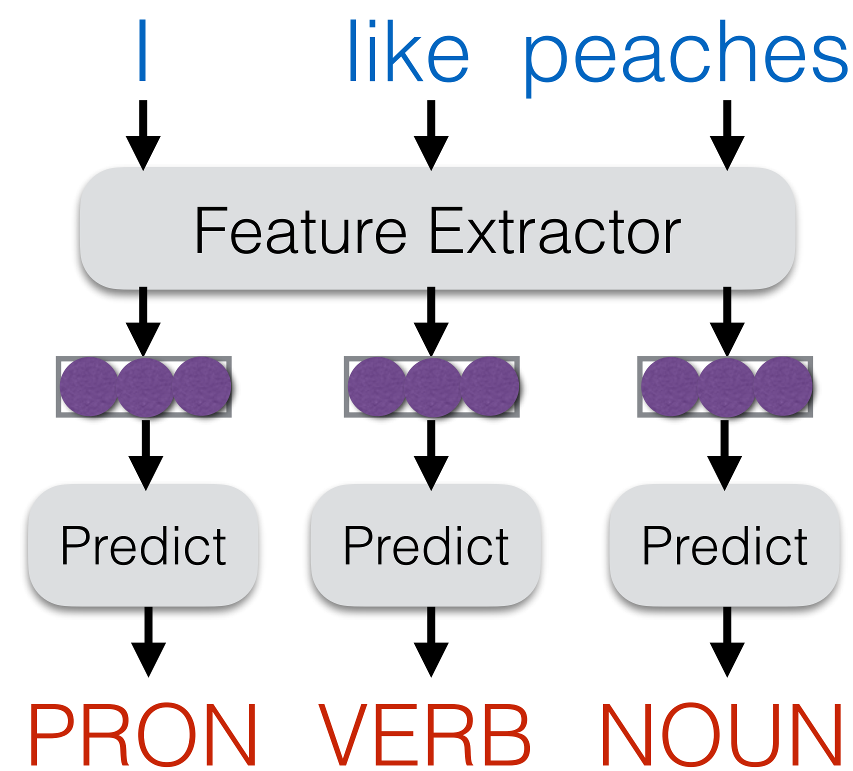
- We want to perform training in as few operations as possible using big matrix multiplies

- We can do so by “masking” the results for the output

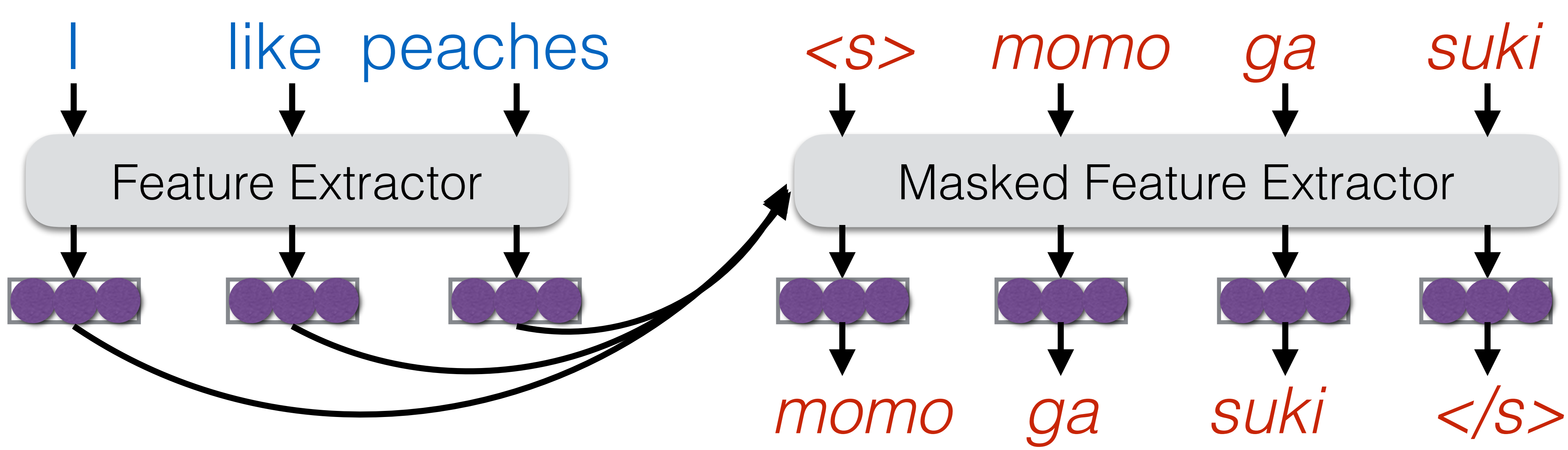


A Unified View of Sequence-to-sequence Models

- Review: sequence labeling



- Sequence-to-sequence modeling



In-class Assignment

Code Walk

- There will be no graded discussion, but we'll have a code walk through The Annotated Transformer https://nlp.seas.harvard.edu/2018/04/03/attention.html

- We'll go into depth into some of the design decisions, their motivation, etc.

- Then we'll discuss Assignment 2!