Reference: AREL Stories **Edited** by Human					
	BLEU4	**METEOR**	**ROUGE**	**Skip-Thoughts**	**Human Rating**
AREL	0.93	0.91	0.92	0.97	**3.69**
AREL Edited By LSTM(T)	0.21	0.46	0.40	0.76	**3.81**

Table 3: Average evaluation scores for AREL stories, using the human-edited stories as references. All the automatic evaluation metrics generate lower scores when human judges give a higher rating.

CS695-002 Special Topics in NLP

Language Generation and Attention

Antonis Anastasopoulos

https://cs.gmu.edu/~antonis/course/cs695-fall20/

With many slides from Graham Neubig's CMU NN4NLP course

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

$$P(X) = \prod_{i=1}^{I} P(x_i \mid x_1, \ldots, x_{i-1})$$

Next Word

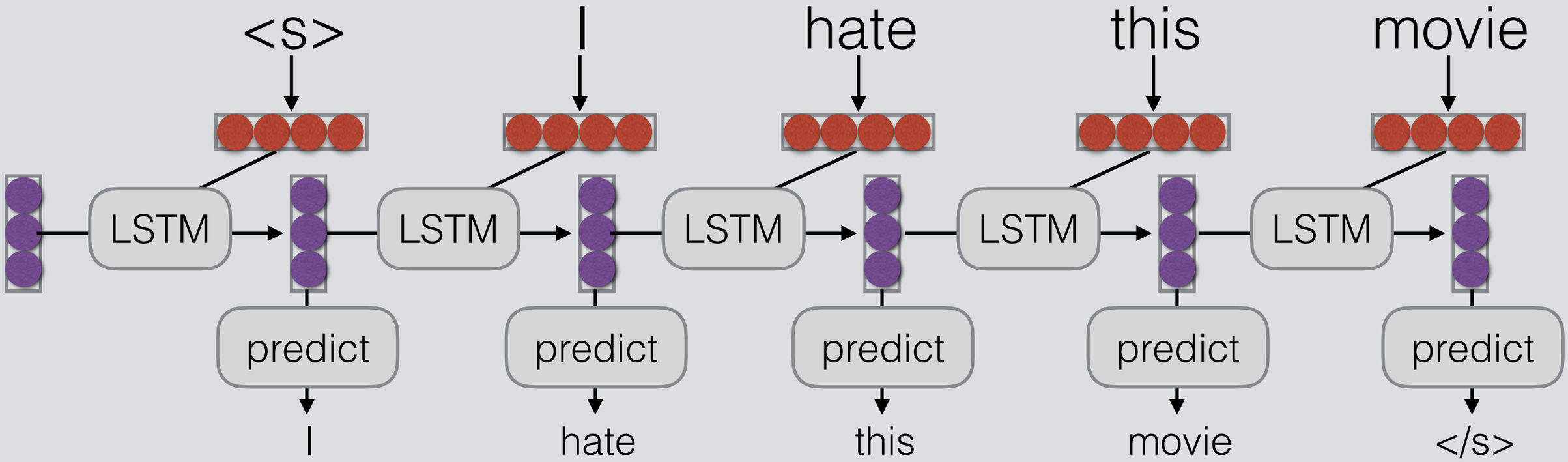
Context

Conditional Language Models

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

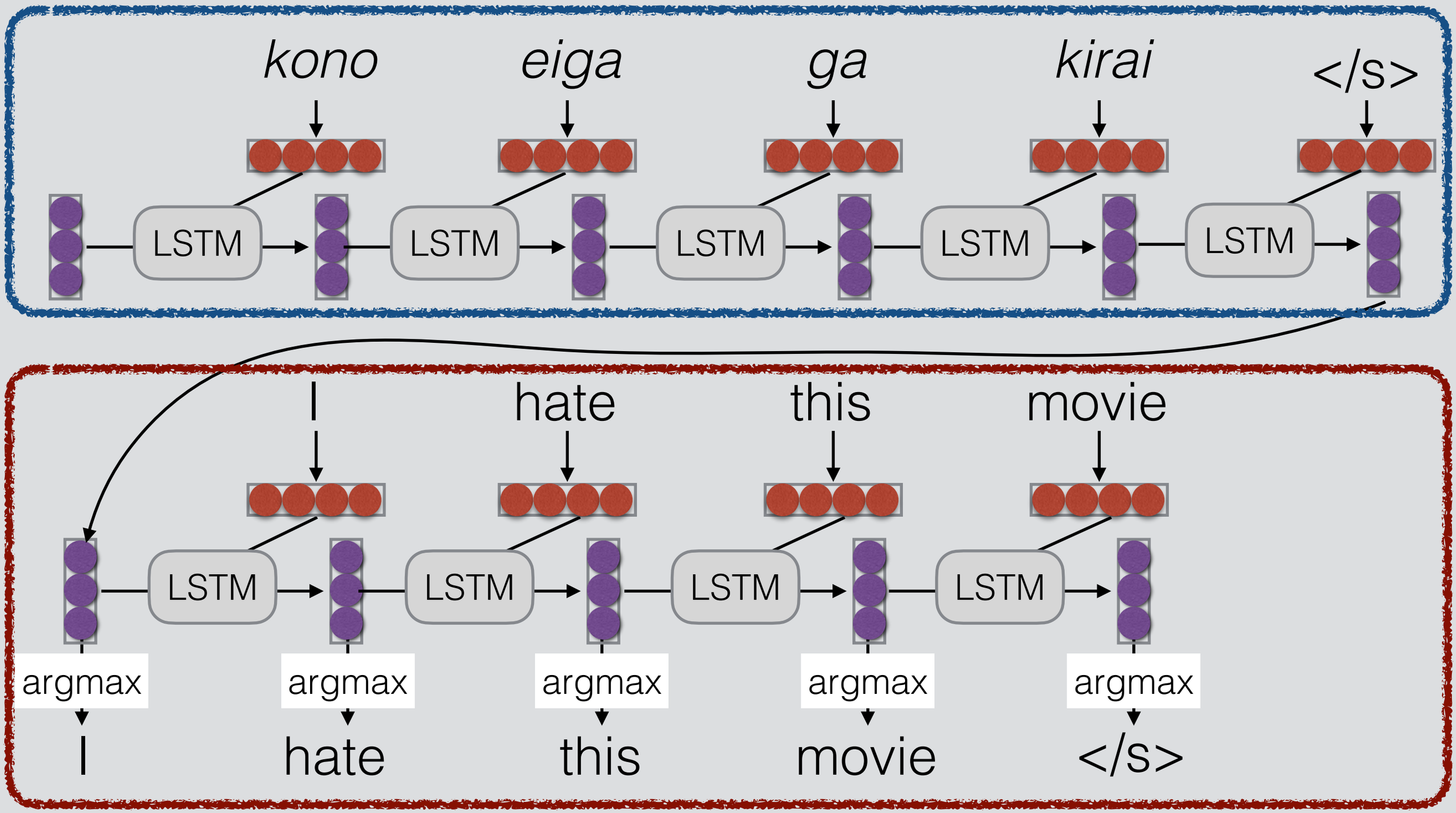
Added Context!

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



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

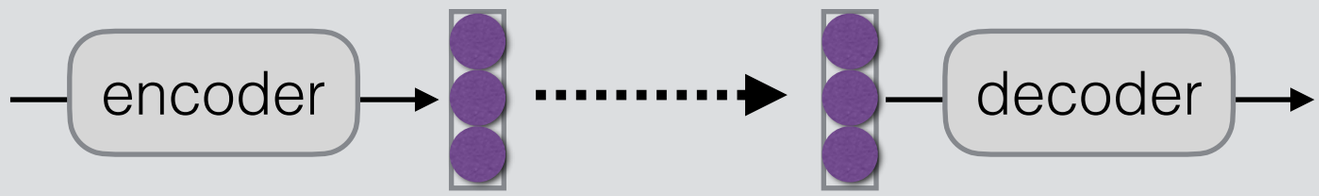
Encoder



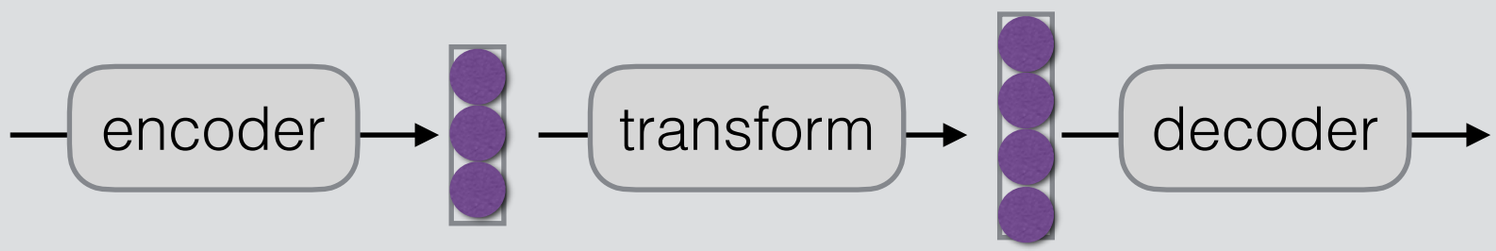
Decoder

How to Pass Hidden State?

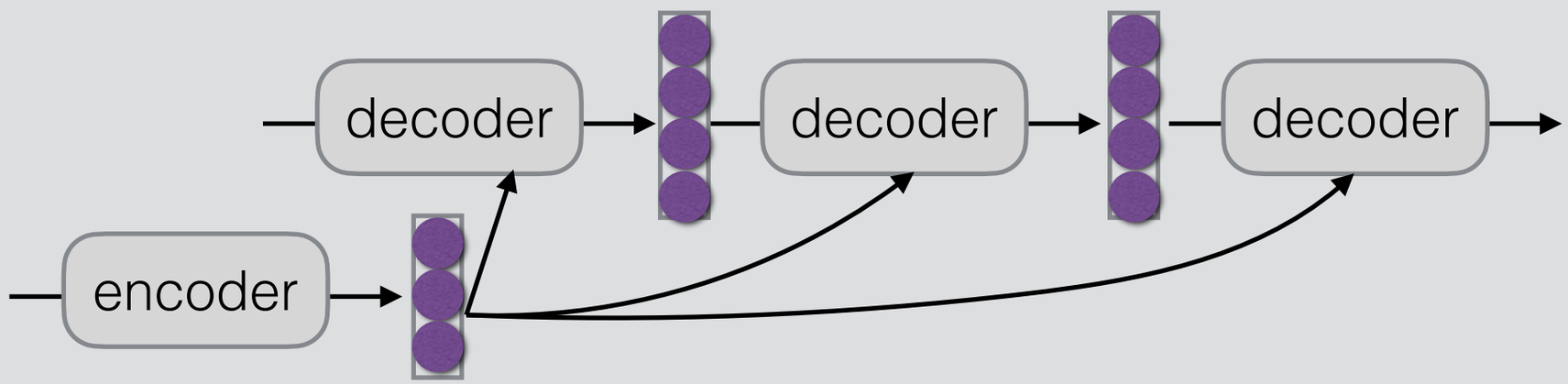
- Initialize decoder w/ encoder (Sutskever et al. 2014)



- Transform (can be different dimensions)



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



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.

while y_{j-1} != “</s>”:
  $y_j \sim P(y_j \mid X, y_1, \ldots, y_{j-1})$

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

Greedy Search

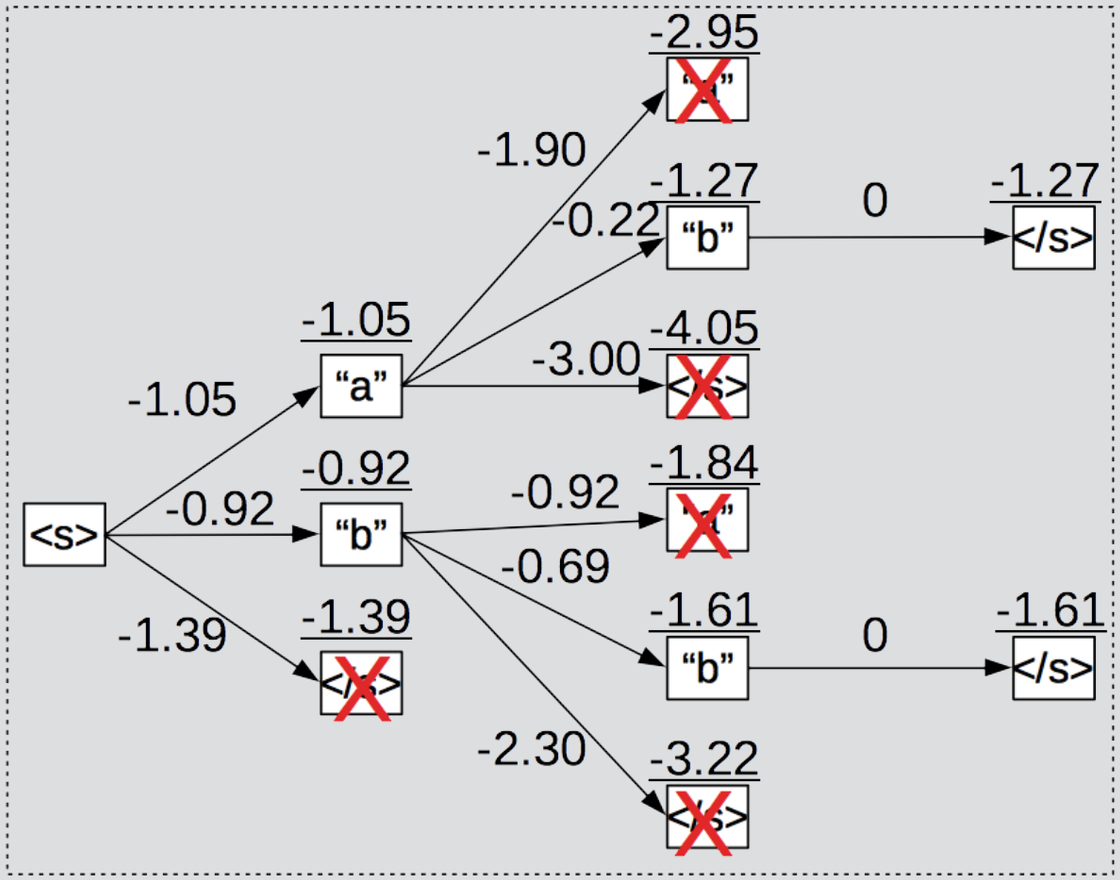
- One by one, pick the single highest-probability word

while y_{j-1} != “</s>”:
  $y_j = \text{argmax}\ P(y_j \mid X, y_1, \ldots, y_{j-1})$

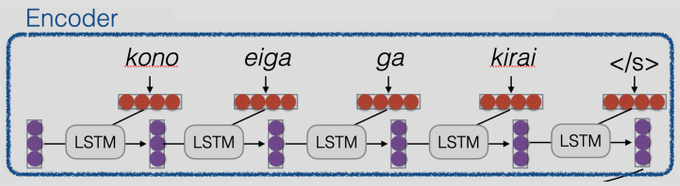
- **Not exact, real problems:**
  - Will often generate the “easy” words first
  - Will prefer multiple common words to one rare word

Beam Search

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

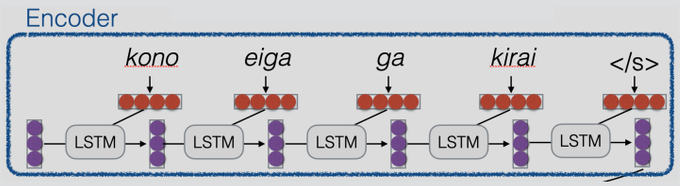


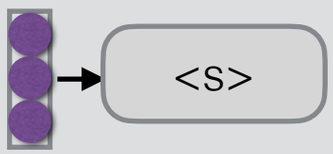
Beam Search


Encoder
kono
eiga
ga
kirai
</s>
LSTM
LSTM
LSTM
LSTM
LSTM


decoder

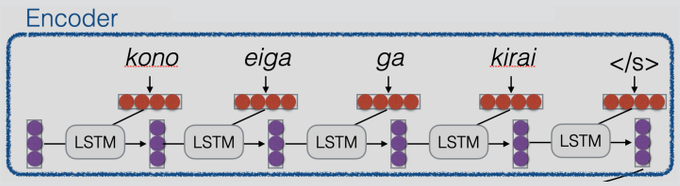
Beam Search



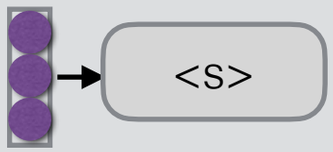


a	0.001
the	0.0002
I	0.12
you	0.04
cat	0.0004
movie	0.01
this	0.02
…	

Beam Search

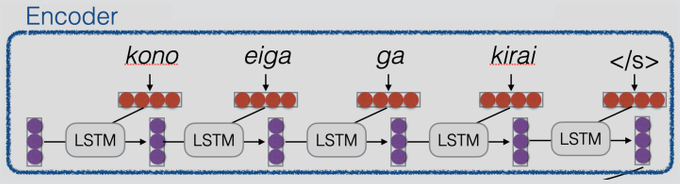


k=2

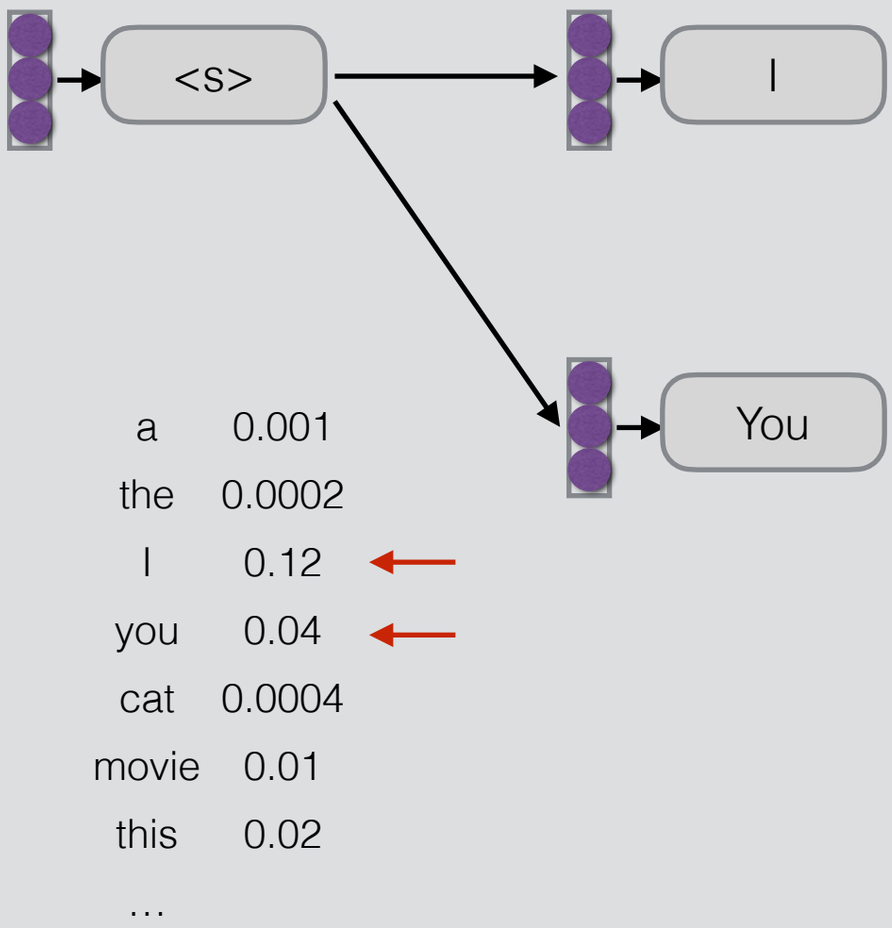


a	0.001
the	0.0002
I	0.12 ←
you	0.04 ←
cat	0.0004
movie	0.01
this	0.02
…	

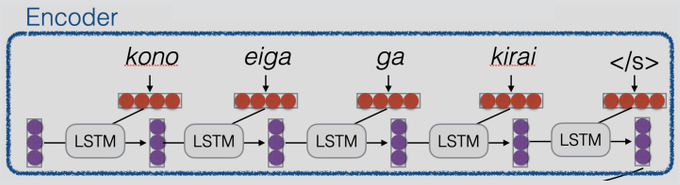
Beam Search



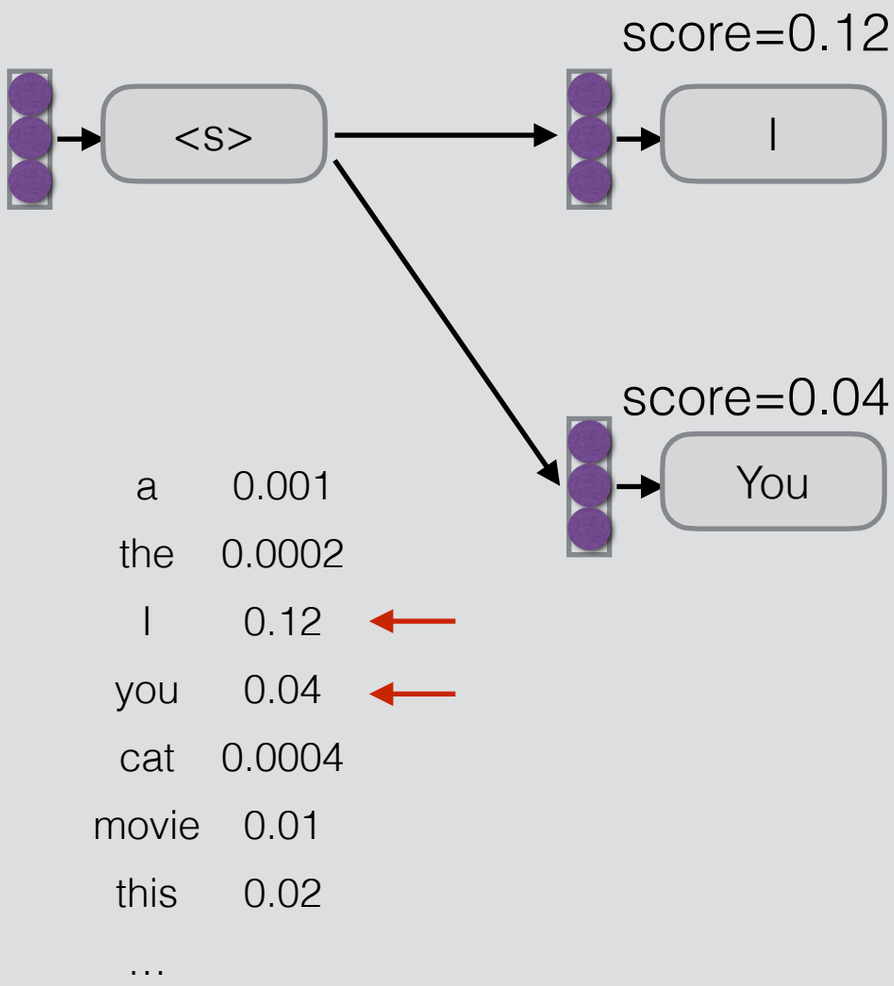
k=2



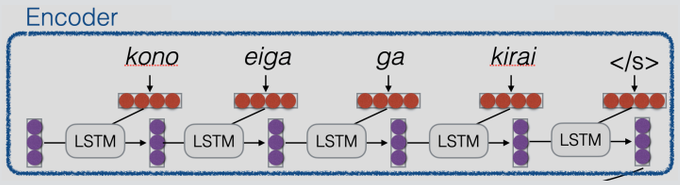
Beam Search



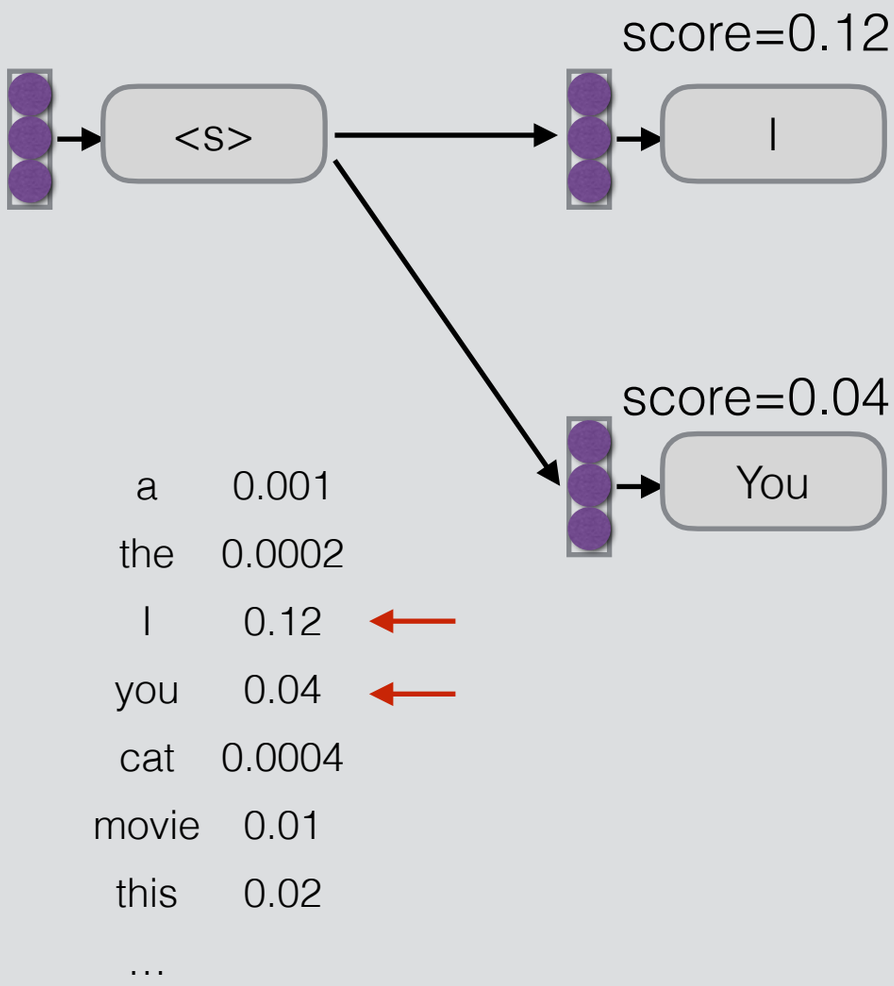
k=2



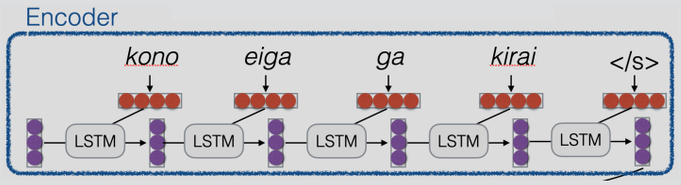
Beam Search



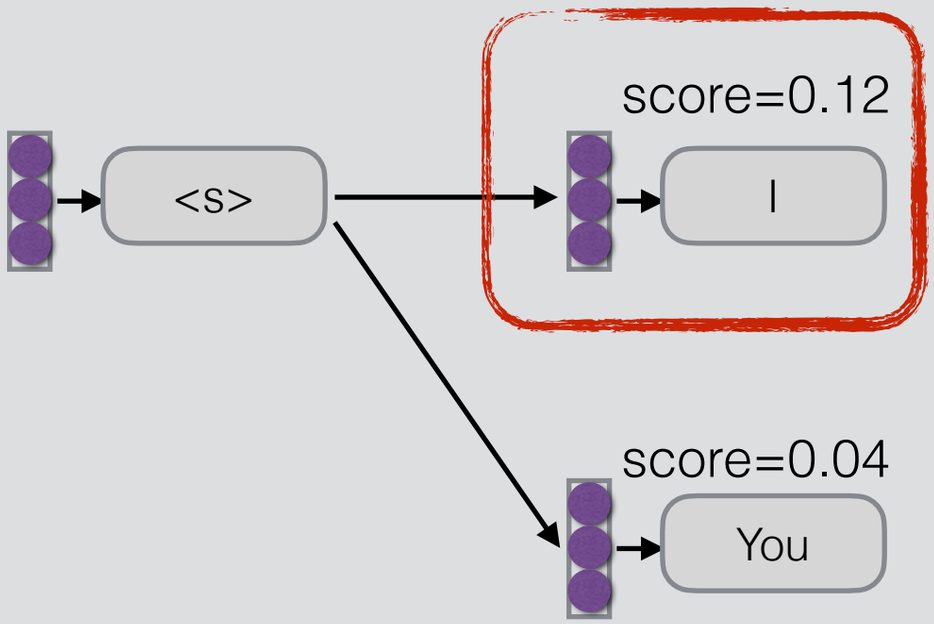
k=2 Expand



Beam Search

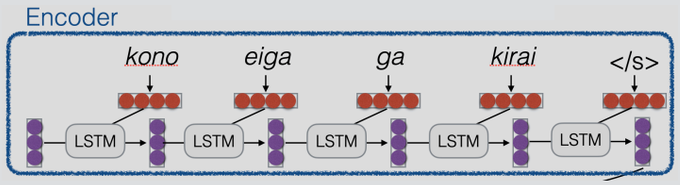


k=2 Expand

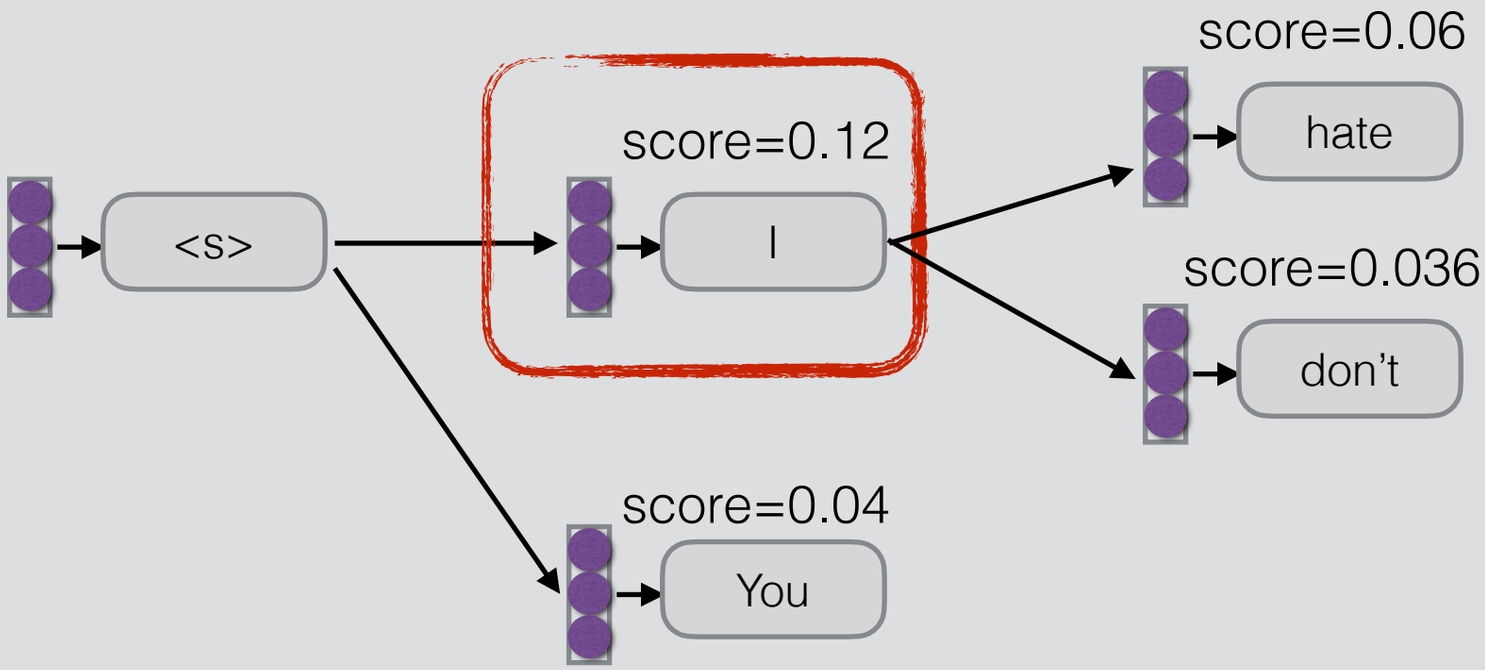


a	0.001
the	0.0002
hate	0.5 ←
this	0.001
cat	0.003
movie	0.07
don't	0.3 ←
…	…

Beam Search

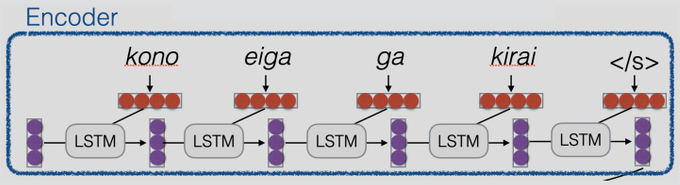


k=2 Expand

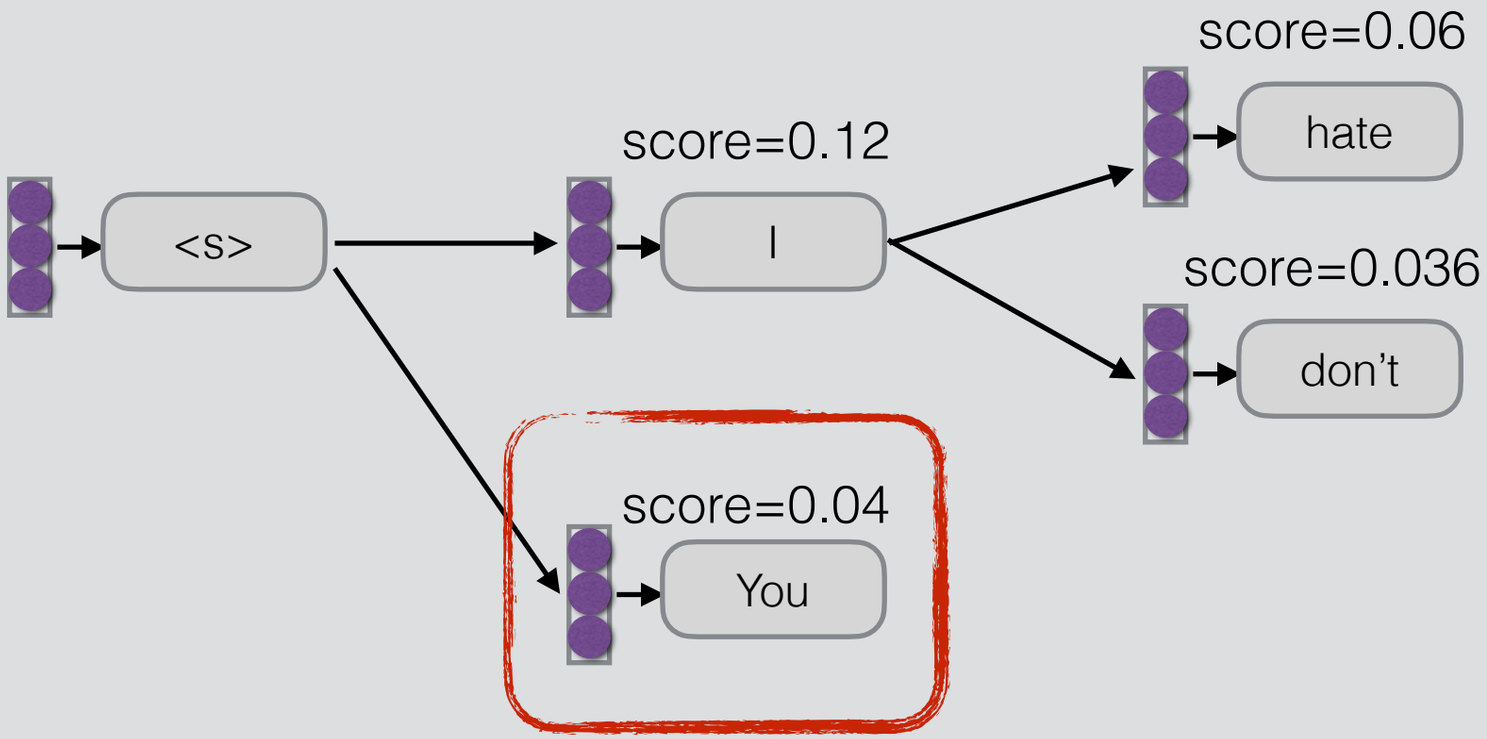


a	0.001
the	0.0002
hate	0.5 ←
this	0.001
cat	0.003
movie	0.07
don't	0.3 ←
…	…

Beam Search

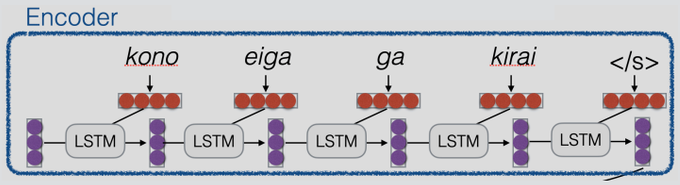


k=2 Expand

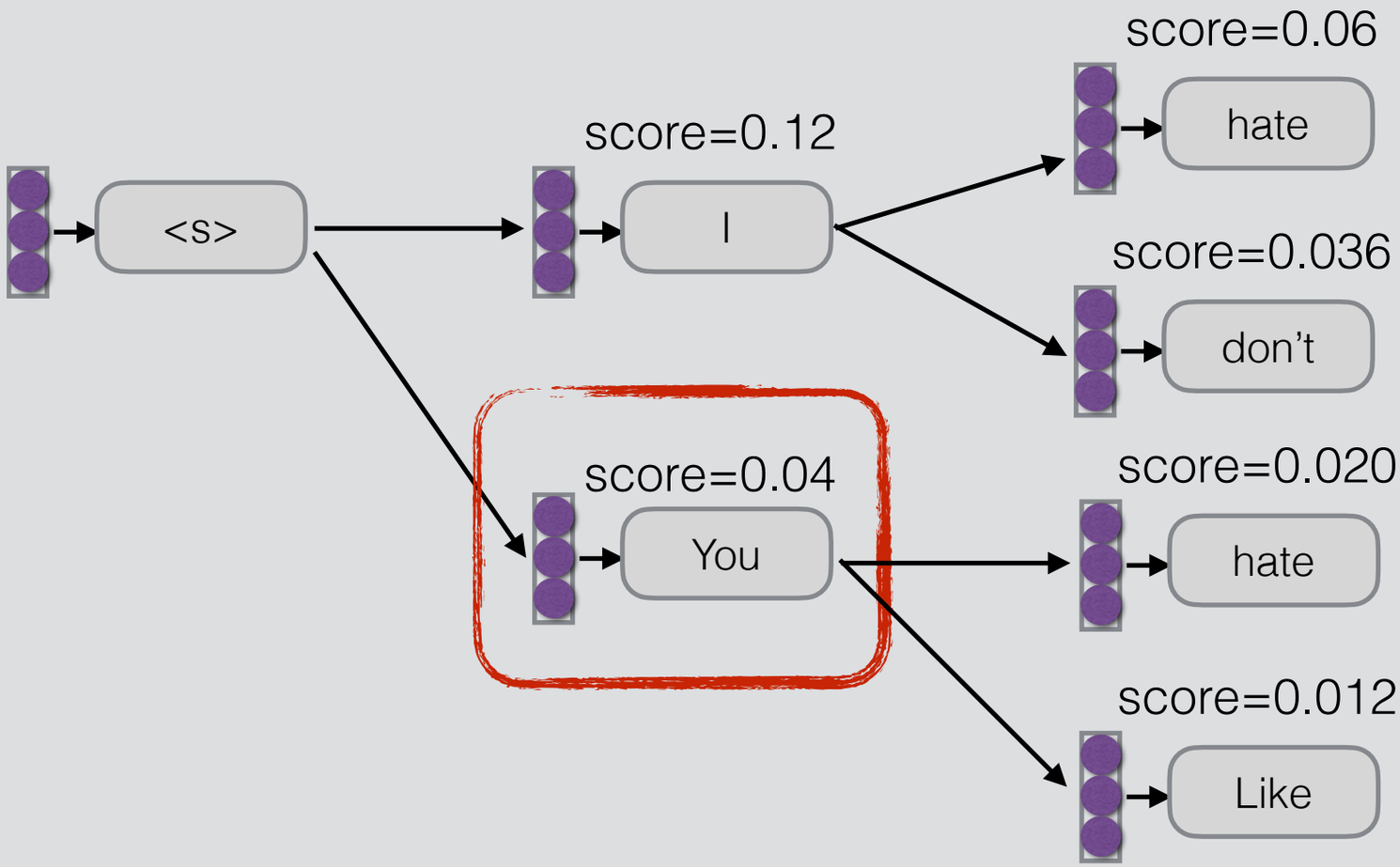


an	0.0012
be	0.0002
hate	0.5 ←
these	0.001
dog	0.003
movie	0.07
like	0.3 ←
…	…

Beam Search

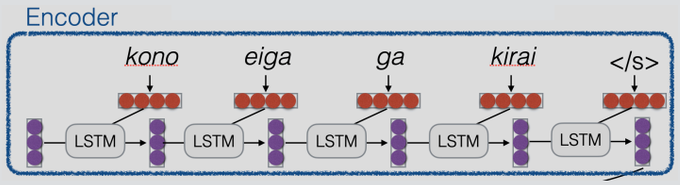


k=2 Expand

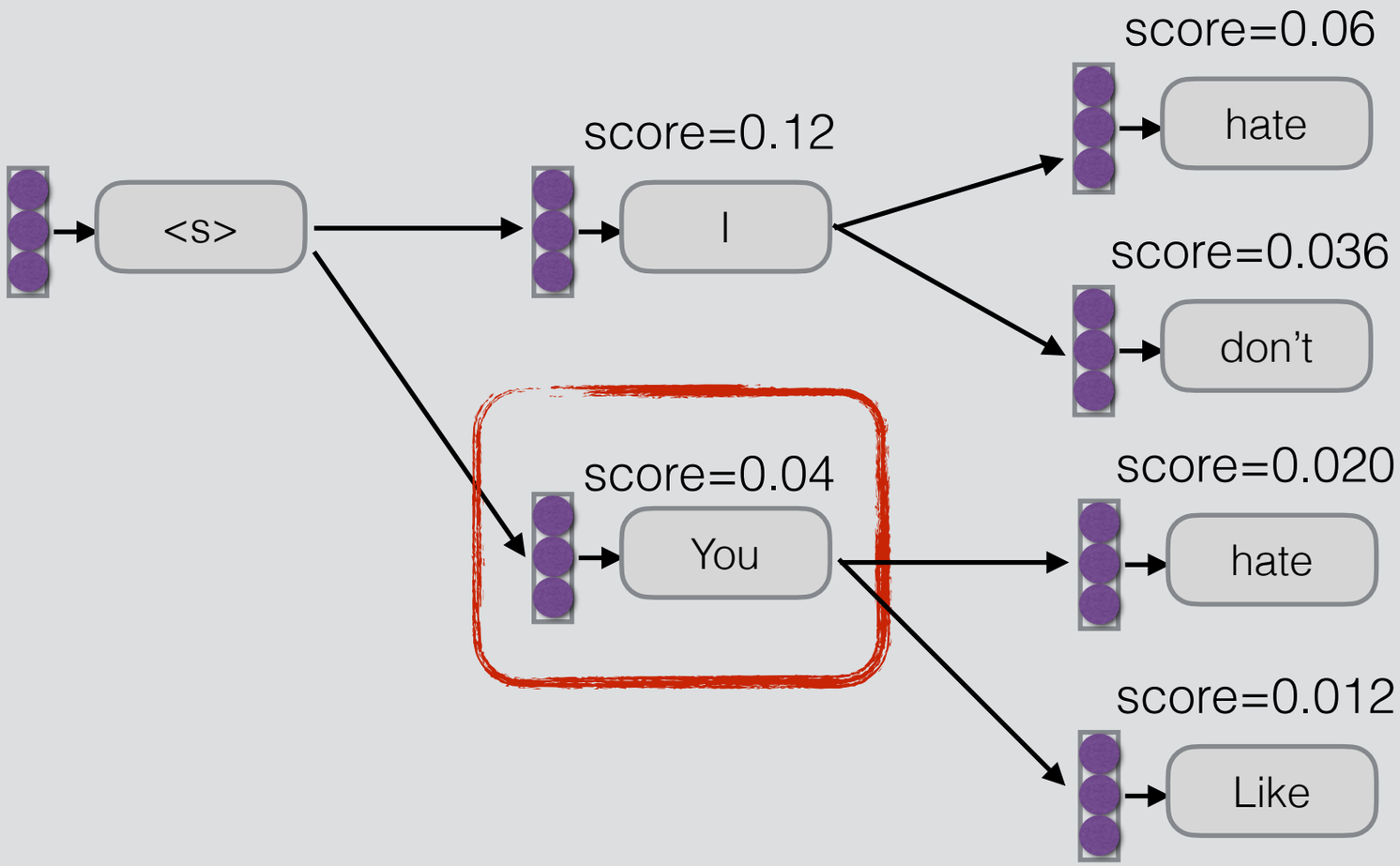


an	0.0012
be	0.0002
hate	0.5 ←
these	0.001
dog	0.003
movie	0.07
like	0.3 ←
…	…

Beam Search

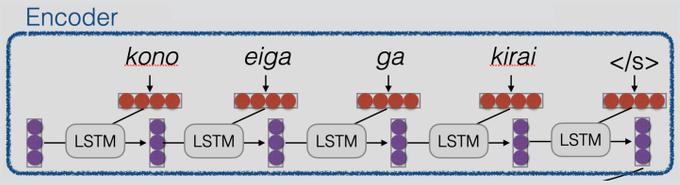


k=2 Expand Prune

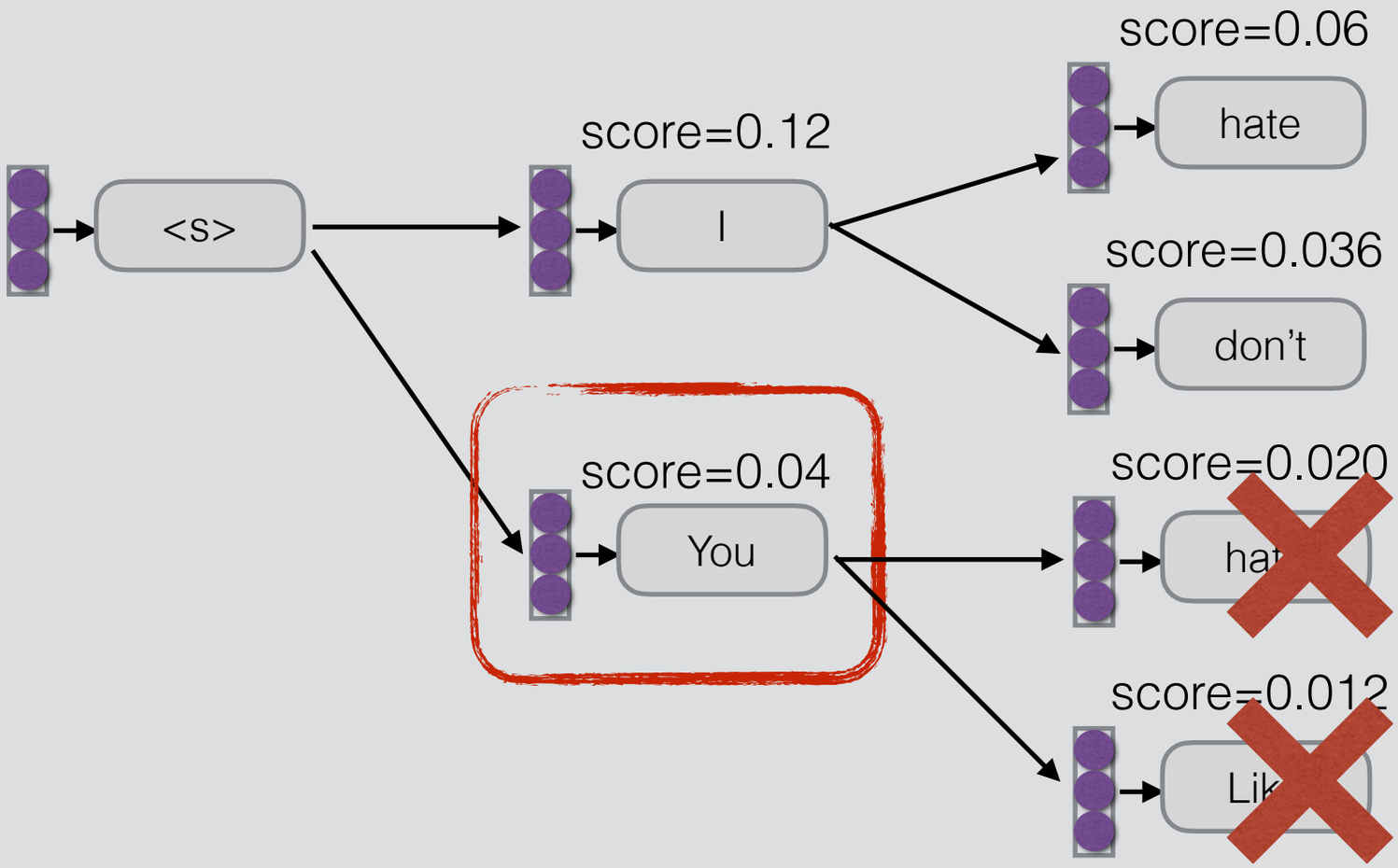


an	0.0012
be	0.0002
hate	0.5 ←
these	0.001
dog	0.003
movie	0.07
like	0.3 ←
…	…

Beam Search

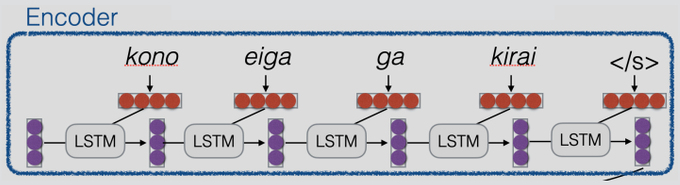


k=2 Expand Prune



an	0.0012
be	0.0002
hate	0.5 ←
these	0.001
dog	0.003
movie	0.07
like	0.3 ←
…	…

Beam Search



k=2 Expand Prune

<s>

I score=0.12

You score=0.04

hate score=0.06

don't score=0.036

hat score=0.020

Lik score=0.012

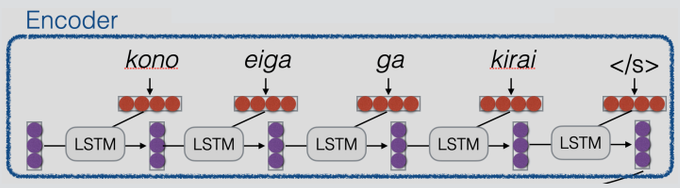
this score=0.003

that score=0.002

like score=0.0024

care score=0.0008

Beam Search



k=2 Expand Prune

<s>

I score=0.12

You score=0.04

hate score=0.06

don't score=0.036

hat score=0.020

Lik score=0.012

this score=0.003

tha score=0.002

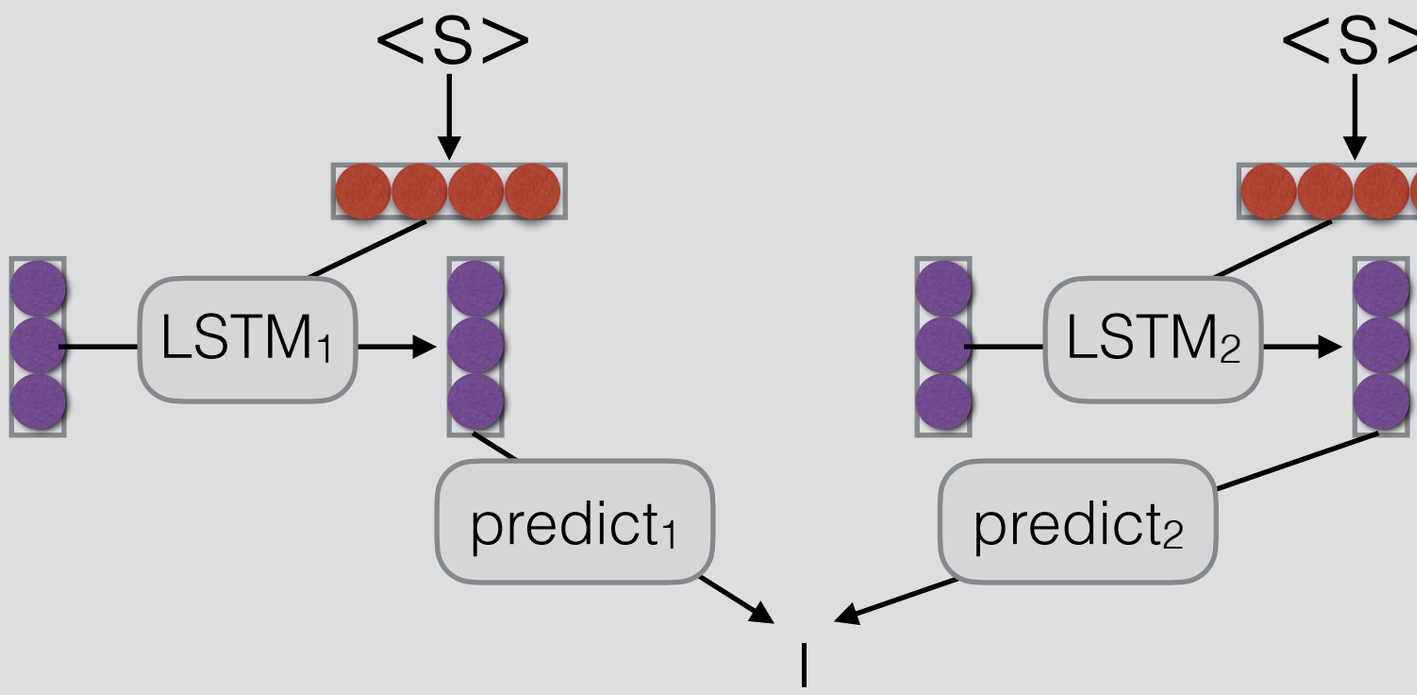
like score=0.0024

score=0.0008

Model Ensembling

Ensembling

- Combine predictions from multiple models



- Why?
  - Multiple models make somewhat uncorrelated errors
  - Models tend to be more uncertain when they are about to make errors
  - Smooths over idiosyncrasies of the model

Linear Interpolation

- Take a weighted average of the M model probabilities

$$P(y_j \mid X, y_1, \ldots, y_{j-1}) = \sum_{m=1}^{M} \underbrace{P_m(y_j \mid X, y_1, \ldots, y_{j-1})}_{\text{Probability according to model } m} \underbrace{P(m \mid X, y_1, \ldots, y_{j-1})}_{\text{Probability of model } m}$$

- Second term often set to uniform distribution 1/M

Log-linear Interpolation

- Weighted combination of log probabilities, normalize

$$P(y_j \mid X, y_1, \ldots, y_{j-1}) = \text{softmax}\left(\sum_{m=1}^{M} \lambda_m(X, y_1, \ldots, y_{j-1}) \log P_m(y_j \mid X, y_1, \ldots, y_{j-1})\right)$$

Normalize

Interpolation coefficient for model *m*

Log probability of model *m*

- Interpolation coefficient often set to uniform distribution 1/M

Linear or Log Linear?

- Think of it in logic!
- **Linear:** “Logical OR”
  - the interpolated model likes any choice that a model gives a high probability
  - use models with models that capture different traits
  - necessary when any model can assign zero probability
- **Log Linear:** “Logical AND”
  - interpolated model only likes choices where all models agree
  - use when you want to restrict possible answers

Parameter Averaging

- **Problem:** Ensembling means we have to use *M* models at test time, increasing our time/memory complexity

- Parameter averaging is a cheap way to get some good effects of ensembling

- Basically, write out models several times near the end of training, and take the average of parameters

Ensemble Distillation (e.g. Kim et al. 2016)

- **Problem:** parameter averaging only works for models within the same run

- Knowledge distillation trains a model to **copy the ensemble**
  - Specifically, it tries to match the description over predicted words
  - Why? We want the model to make the same mistakes as an ensemble

- Shown to increase accuracy notably

Stacking

- What if we have two very different models where prediction of outputs is done in very different ways?
  - e.g. a phrase-based translation model and a neural MT model (Niehues et al. 2017)
  - Stacking uses the **output of one system in calculating features for another** system

How do we Evaluate?

Basic Evaluation Paradigm

- Use parallel test set
- Use system to generate translations
- Compare target translations w/ reference

Human Evaluation

- Ask a human to do evaluation

太郎が花子を訪れた

	Taro visited Hanako	the Taro visited the Hanako	Hanako visited Taro
Adequate?	Yes	Yes	No
Fluent?	Yes	No	Yes
Better?	1	2	3

- Final goal, but slow, expensive, and sometimes inconsistent

BLEU

- Works by comparing n-gram overlap w/ reference

Reference: Taro visited Hanako

System: the Taro visited the Hanako

Brevity: min(1, |System|/|Reference|) = min(1, 5/3)

1-gram: 3/5
2-gram: 1/4
brevity penalty = 1.0

BLEU-2 = $(3/5*1/4)^{1/2} * 1.0$
= 0.387

- **Pros:** Easy to use, good for measuring system improvement
- **Cons:** Often doesn't match human eval, bad for comparing very different systems

METEOR

- Like BLEU in overall principle, with many other tricks: consider paraphrases, reordering, and function word/content word difference

- **Pros:** Generally significantly better than BLEU, esp. for high-resource languages

- **Cons:** Requires extra resources for new languages (although these can be made automatically), and more complicated

Perplexity

- Calculate the perplexity of the words in the held-out set *without* doing generation

- **Pros:** Naturally solves multiple-reference problem!

- **Cons:** Doesn't consider decoding or actually generating output.

- May be reasonable for problems with lots of ambiguity.

A Contrastive Note: Evaluating *Un*conditioned Generation

- How do we evaluate *un*conditioned generation models?

- Not clear! We could do human evaluation.
    - But a model that memorizes the corpus will be too good.

- Perhaps held-out perplexity is as good as we can do?

- Perhaps we should use conditioned generation.

Case Studies in Conditional Language Modeling

From Structured Data

(e.g. Wen et al 2015)

- When you say “Natural Language Generation” to an old-school NLPer, it means this

	SF Restaurant	SF Hotel
act type	inform, inform_only, reject, confirm, select, request, reqmore, goodbye	
shared	name, type, *pricerange, price, phone, address, postcode, *area, *near	
specific	*food *goodformeal ***kids-allowed**	***hasinternet** ***acceptscards** ***dogs-allowed**

bold=binary slots, *=slots can take “don’t care” value

Still a Difficult Problem!

- e.g. "Challenges in data-to-document generation" (Wiseman et al. 2017)

TEAM	WIN	LOSS	PTS	FG_PCT	RB	AS ...
Heat	11	12	103	49	47	27
Hawks	7	15	95	43	33	20

PLAYER	AS	RB	PT	FG	FGA	CITY ...
Tyler Johnson	5	2	27	8	16	Miami
Dwight Howard	4	17	23	9	11	Atlanta
Paul Millsap	2	9	21	8	12	Atlanta
Goran Dragic	4	2	21	8	17	Miami
Wayne Ellington	2	3	19	7	15	Miami
Dennis Schroder	7	4	17	8	15	Atlanta
Rodney McGruder	5	5	11	3	8	Miami
Thabo Sefolosha	5	5	10	5	11	Atlanta
Kyle Korver	5	3	9	3	9	Atlanta
...						

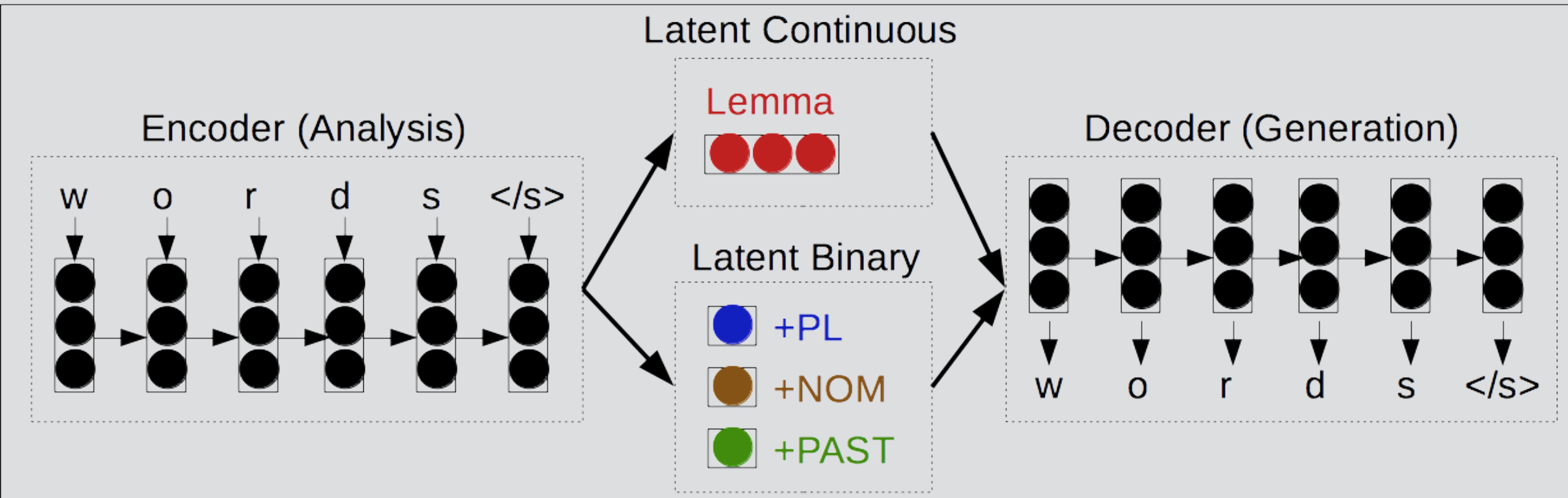
The Utah Jazz (38 - 26) defeated the Houston Rockets (38 - 26) 117 - 91 on Wednesday at Energy Solutions Arena in Salt Lake City . The Jazz got out to a quick start in this one , out - scoring the Rockets 31 - 15 in the first quarter alone . Along with the quick start , the Rockets were the superior shooters in this game , going 54 percent from the field and 43 percent from the three - point line , while the Jazz went 38 percent from the floor and a meager 19 percent from deep . The Rockets were able to out - rebound the Rockets 49 - 49 , giving them just enough of an advantage to secure the victory in front of their home crowd . The Jazz were led by the duo of Derrick Favors and James Harden . Favors went 2 - for - 6 from the field and 0 - for - 1 from the three - point line to score a game - high of 15 points , while also adding four rebounds and four assists ....

Figure 2: Example document generated by the Conditional Copy system with a beam of size 5. Text that accurately reflects a record in the associated box- or line-score is highlighted in blue, and erroneous text is highlighted in red.

From Input + Labels

(e.g. Zhou and Neubig 2017)

- For example, word + morphological tags -> inflected word

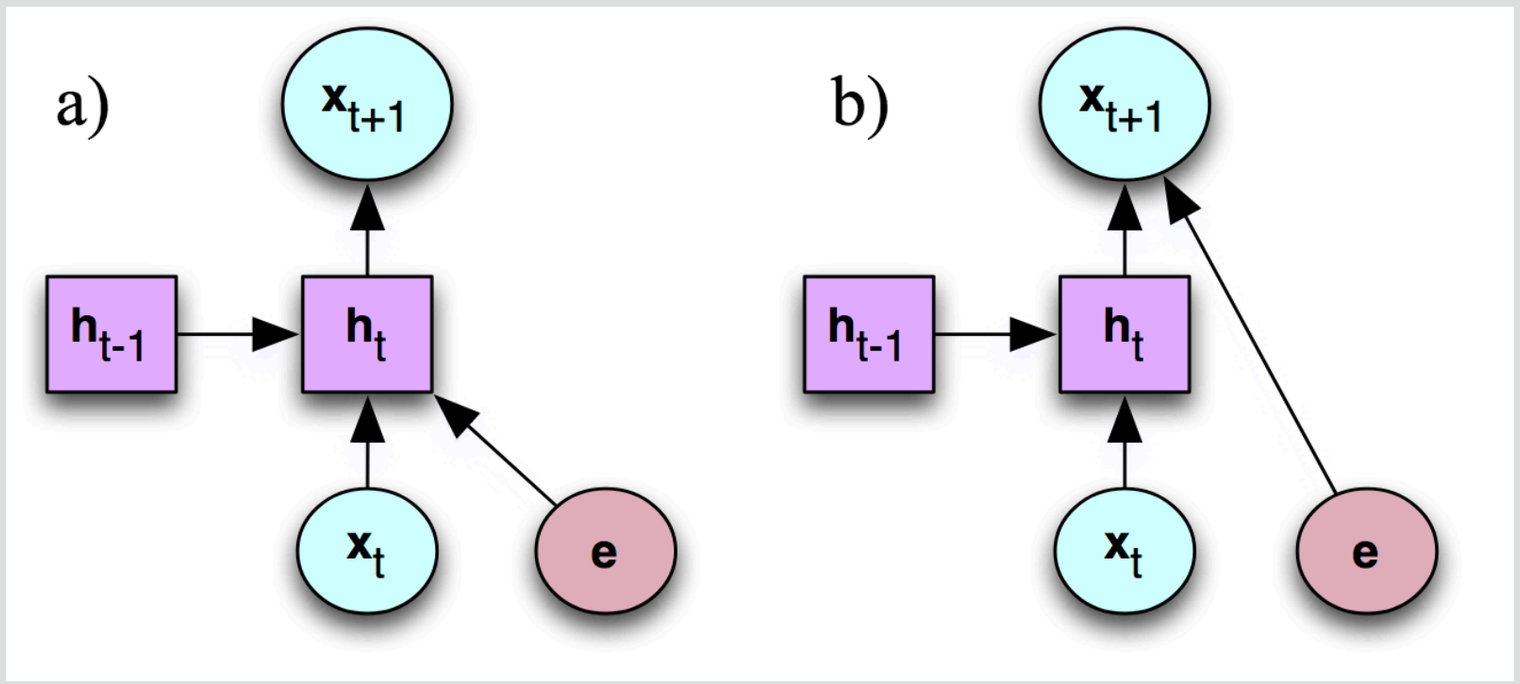


- Other options: politeness/gender in translation, etc.

From Speaker/Document Traits

(Hoang et al. 2016)

- e.g. TED talk description -> TED talk
- Encode title, description, keywords, author embedding
- Various encoding methods: BOW, CNN, RNN
- Various integration methods: in recurrent layer or softmax layer

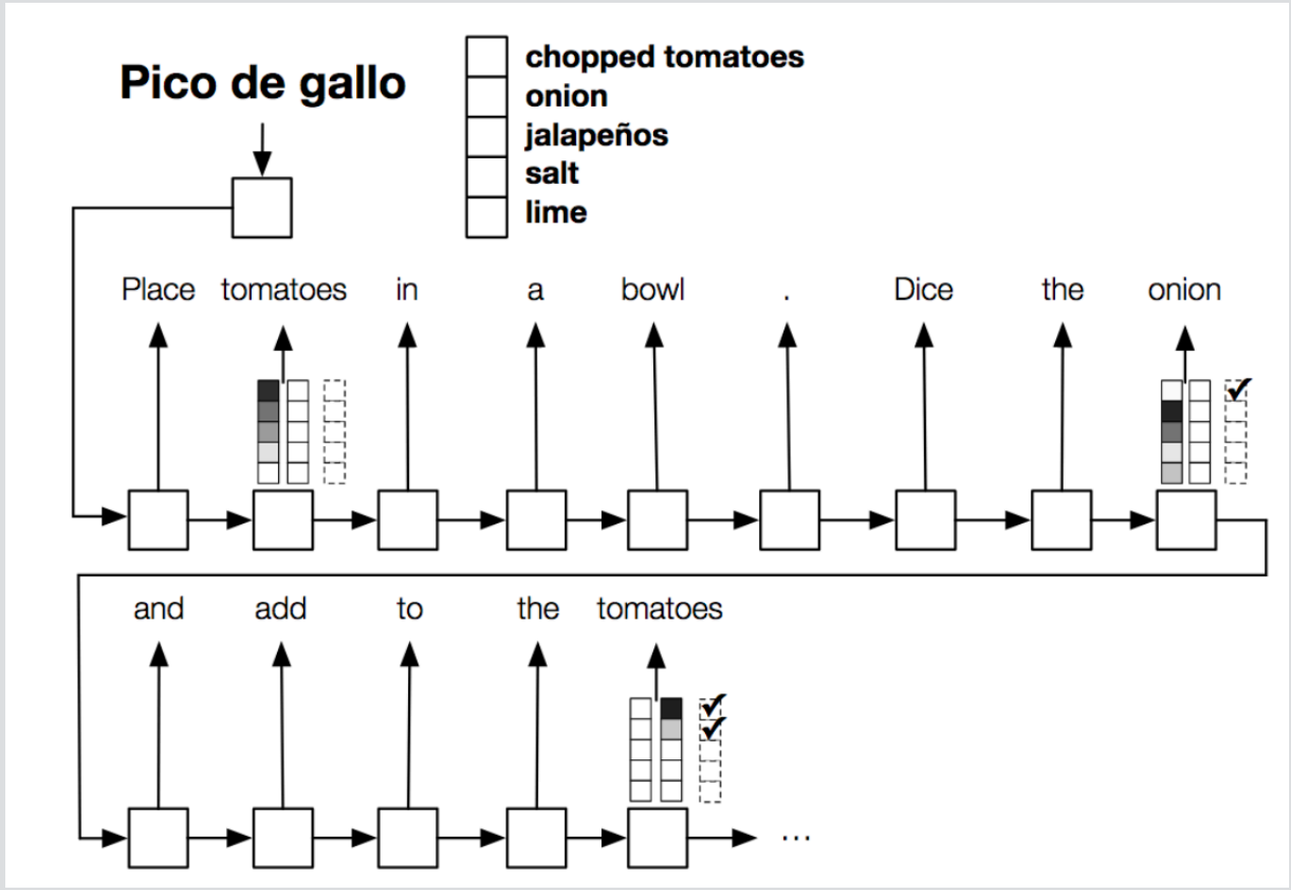


From Lists of Traits

(Kiddon et al. 2016)

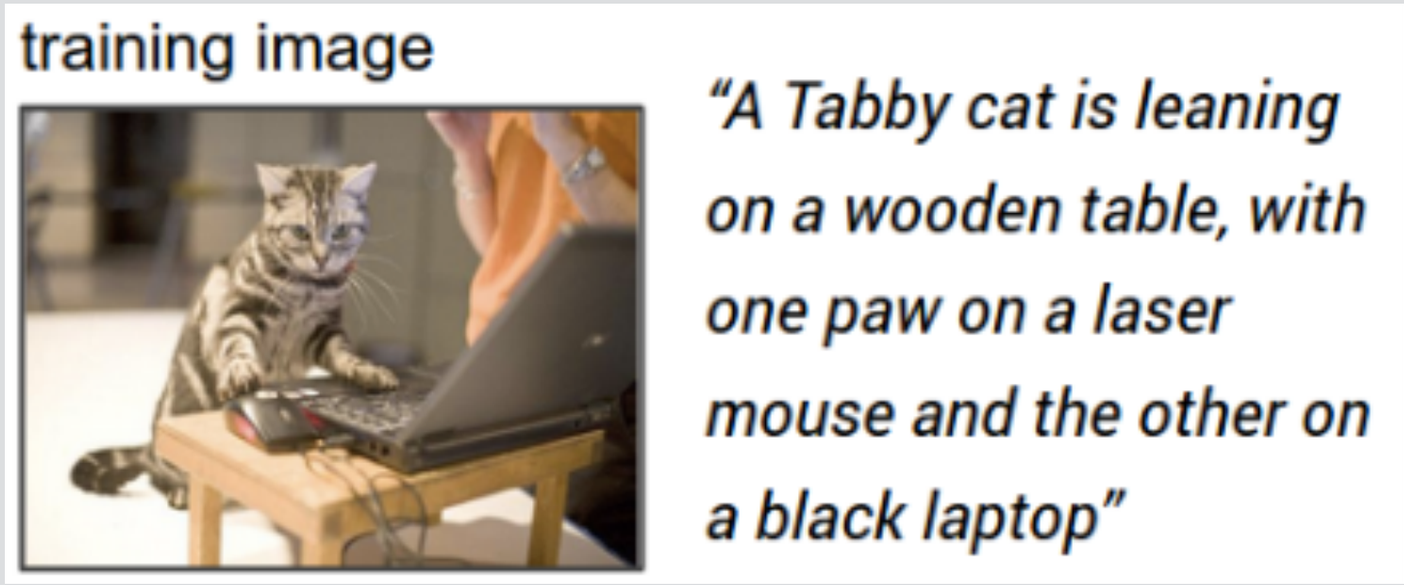
- Name of a recipe + ingredients -> recipe

- "Neural Checklist Model" that tells when a particular item in the list has been generated



From Images
(e.g. Karpathy et al. 2015)

- Input is image features, output is text



- Standard to use CNN-based image encoders

- Often pre-trained on large databases such as ImageNet

From Word Embeddings (Noraset et al. 2017)

Word	Generated definition
brawler	a person who fights
butterfish	a marine fish of the atlantic coast
continually	in a constant manner
creek	a narrow stream of water
feminine	having the character of a woman
juvenility	the quality of being childish
mathematical	of or pertaining to the science of mathematics
negotiate	to make a contract or agreement
prance	to walk in a lofty manner
resent	to have a feeling of anger or dislike
similar	having the same qualities
valueless	not useful

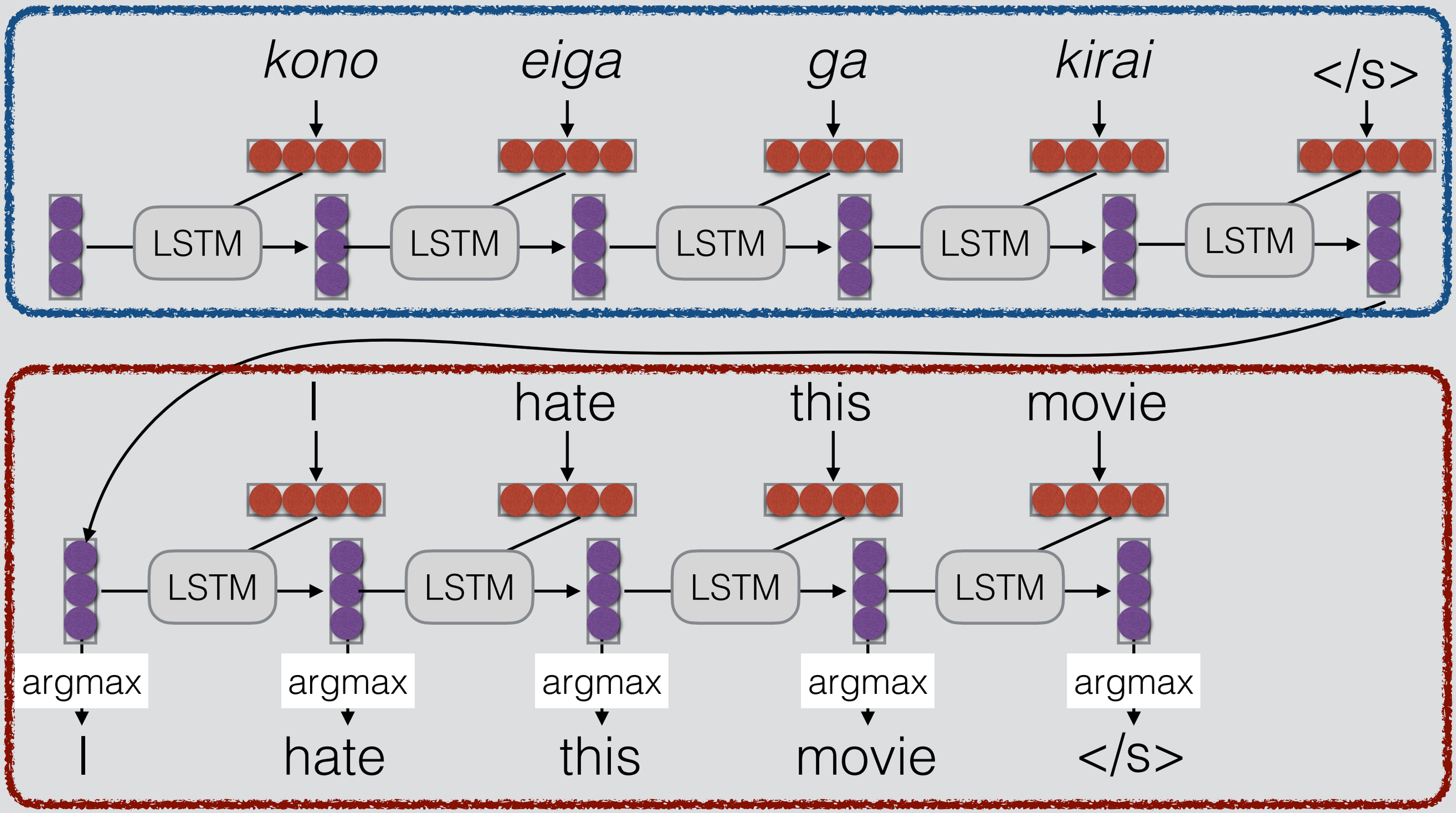
- Baseline: standard sequence-to-sequence model
- Additional information about the affixes and hypernyms

Attention

Encoder-decoder Models

(Sutskever et al. 2014)

Encoder



Decoder

Sentence Representations

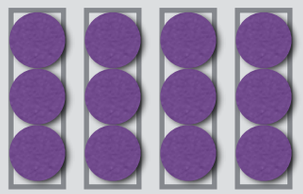
Problem!

"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

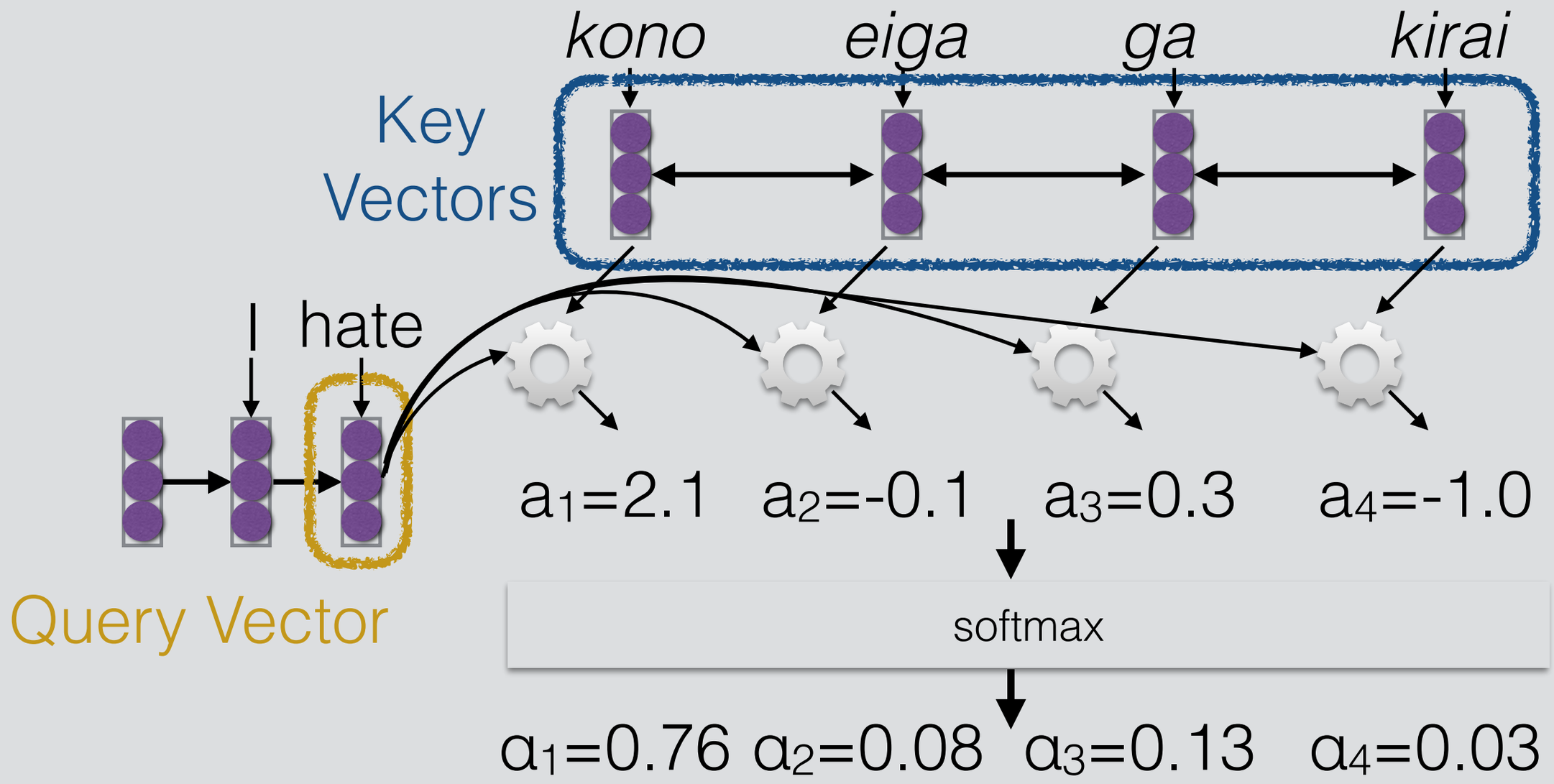
Basic Idea

(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

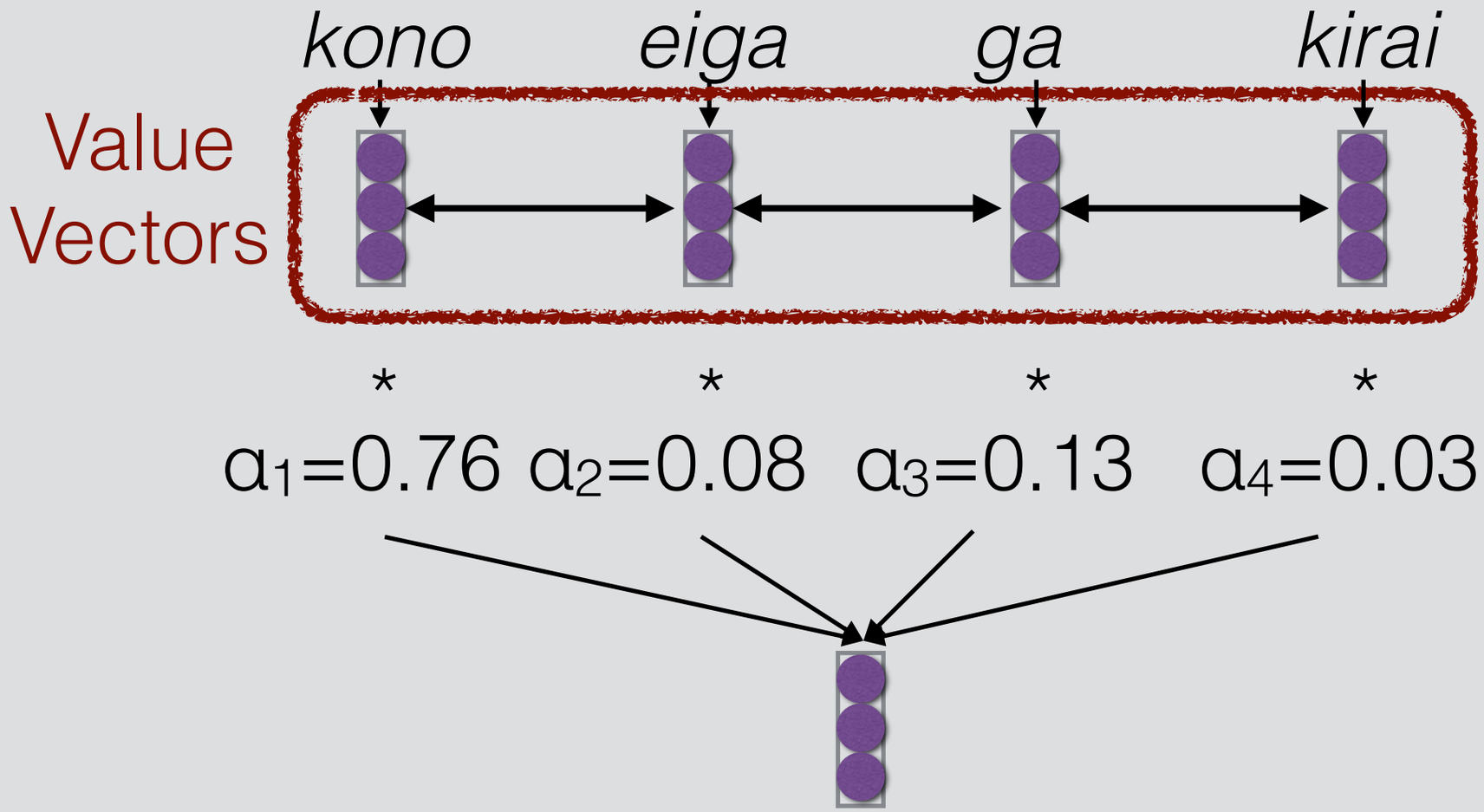
Calculating Attention (1)

- 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

Model Overview

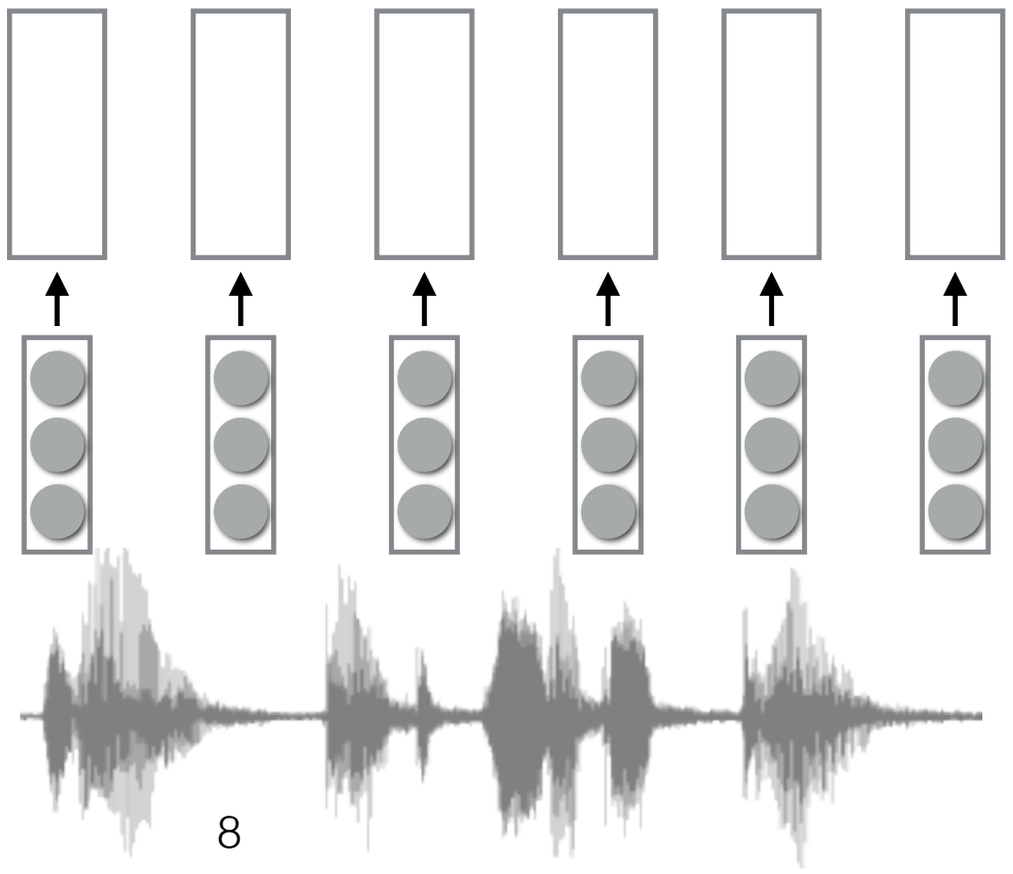
RNN Decoder

Attention

RNN Encoder

Representation

Wave form



Model Overview

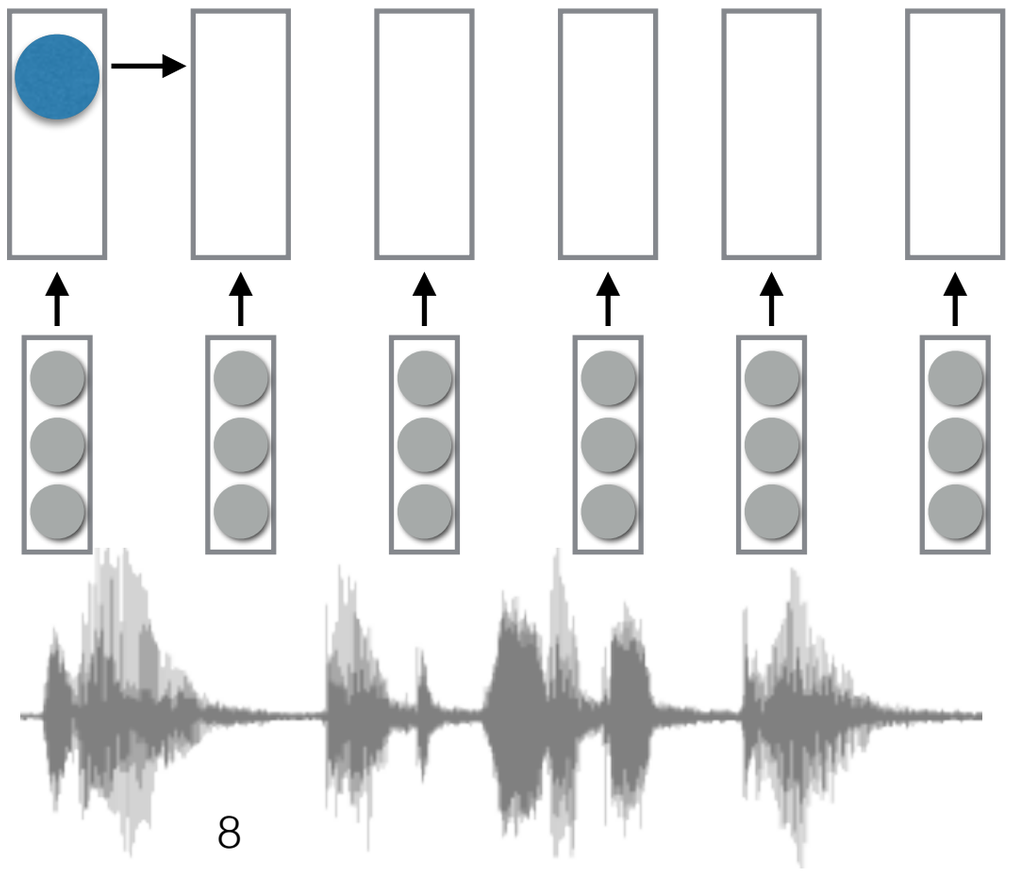
RNN Decoder

Attention

RNN Encoder

Representation

Wave form



Model Overview

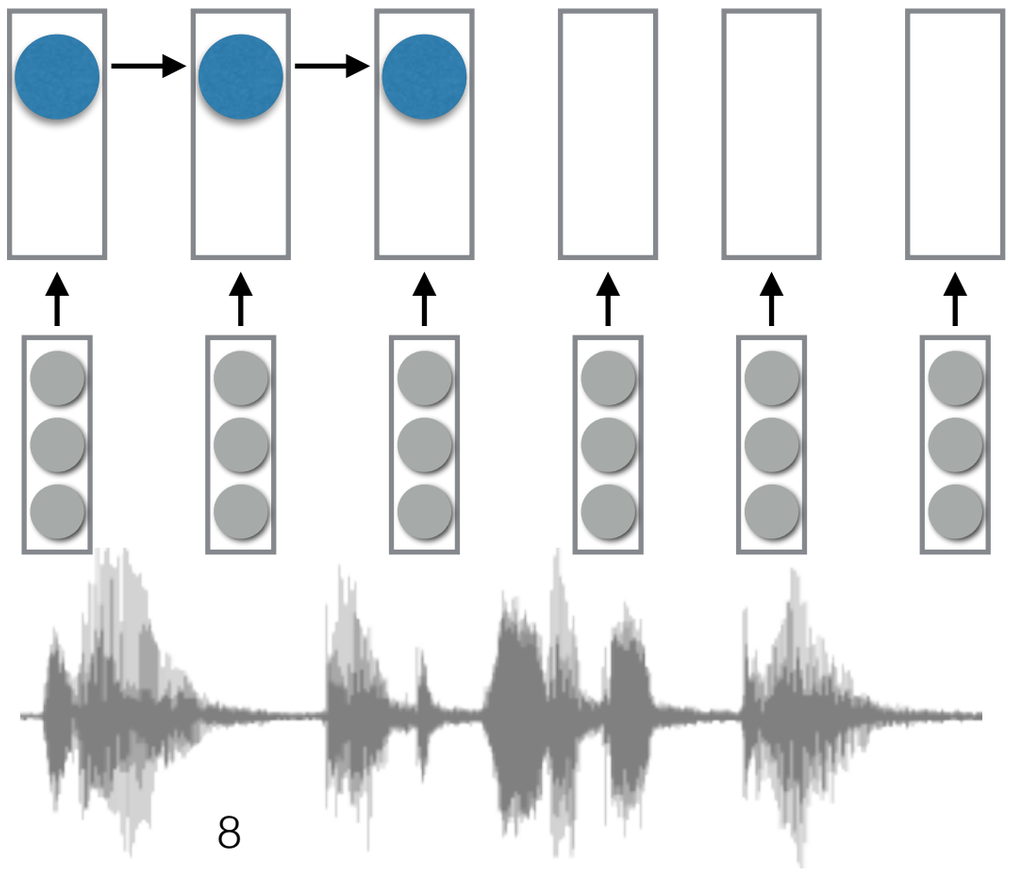
RNN Decoder

Attention

RNN Encoder

Representation

Wave form



Model Overview

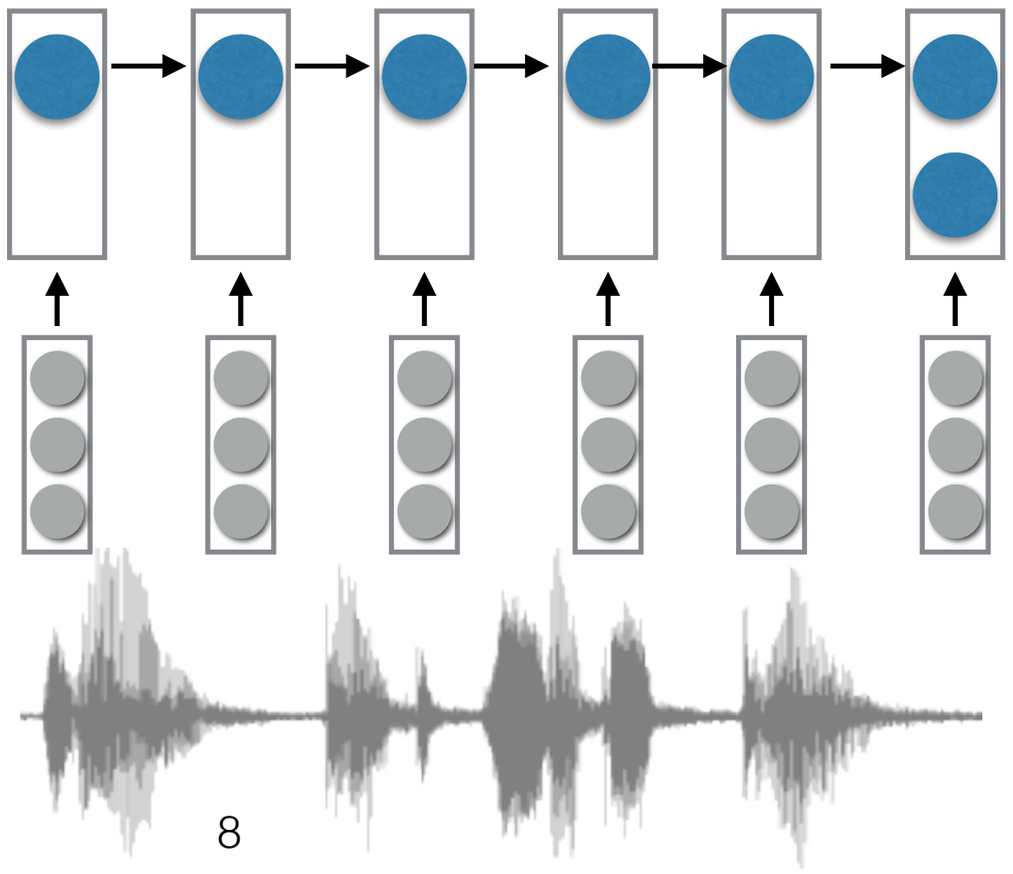
RNN Decoder

Attention

RNN Encoder

Representation

Wave form



Model Overview

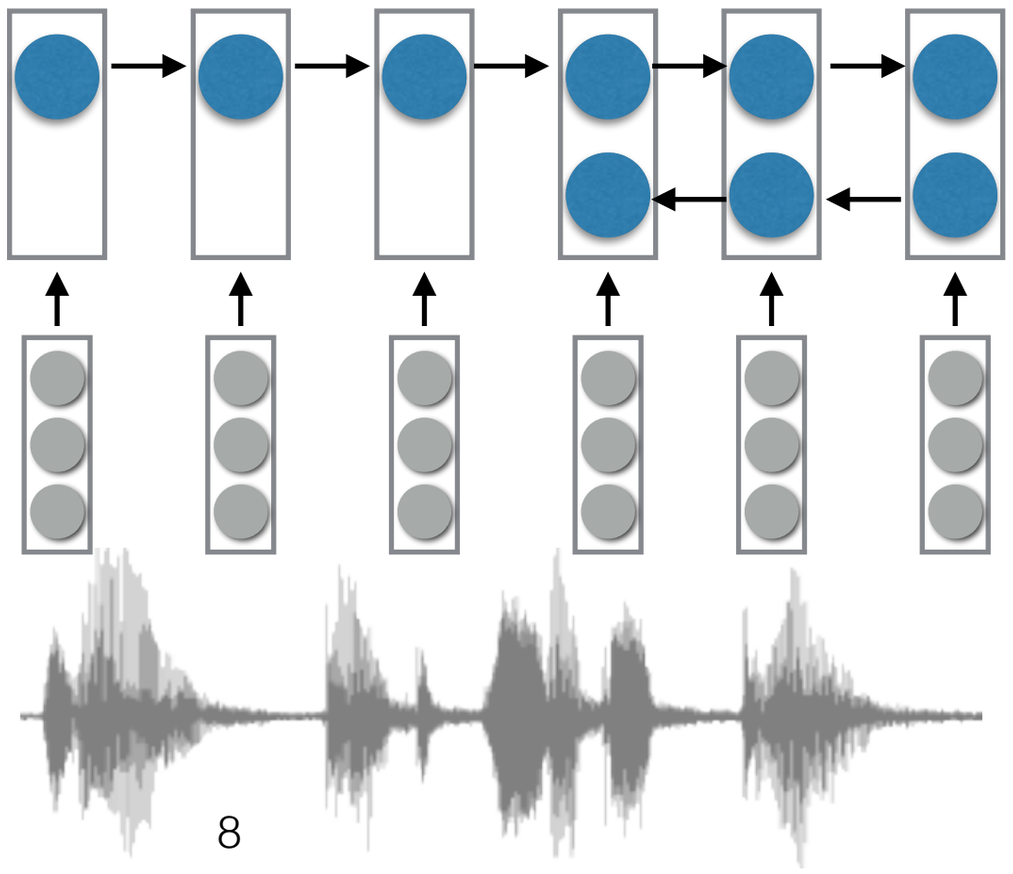
RNN Decoder

Attention

RNN Encoder

Representation

Wave form



Model Overview

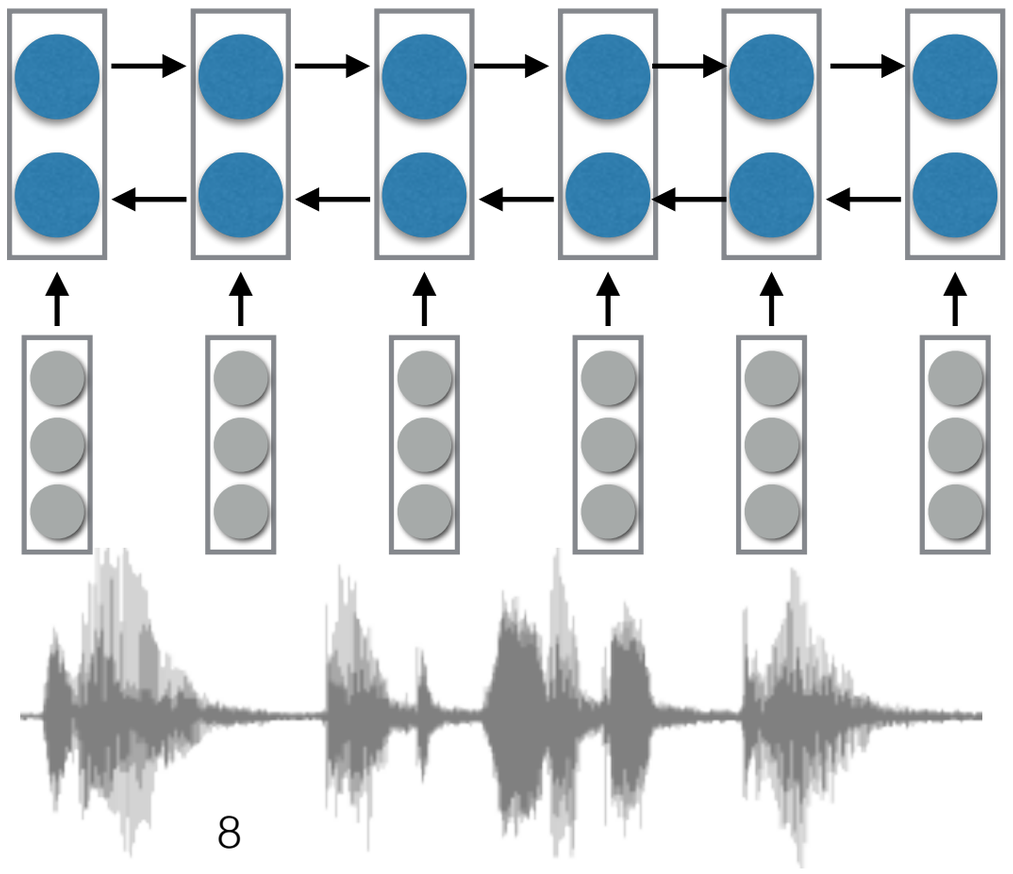
RNN Decoder

Attention

RNN Encoder

Representation

Wave form



Model Overview

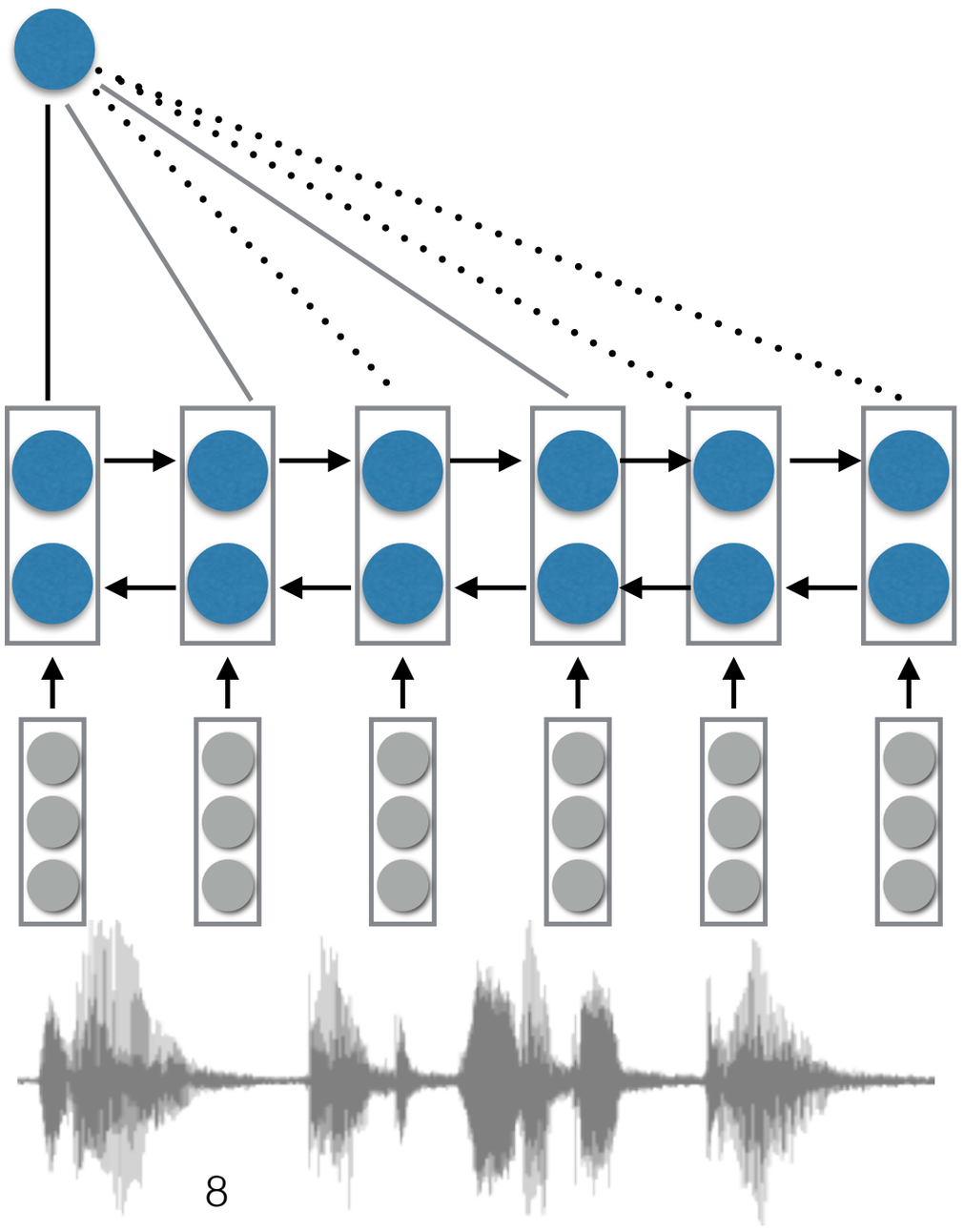
RNN Decoder

Attention

RNN Encoder

Representation

Wave form



Model Overview

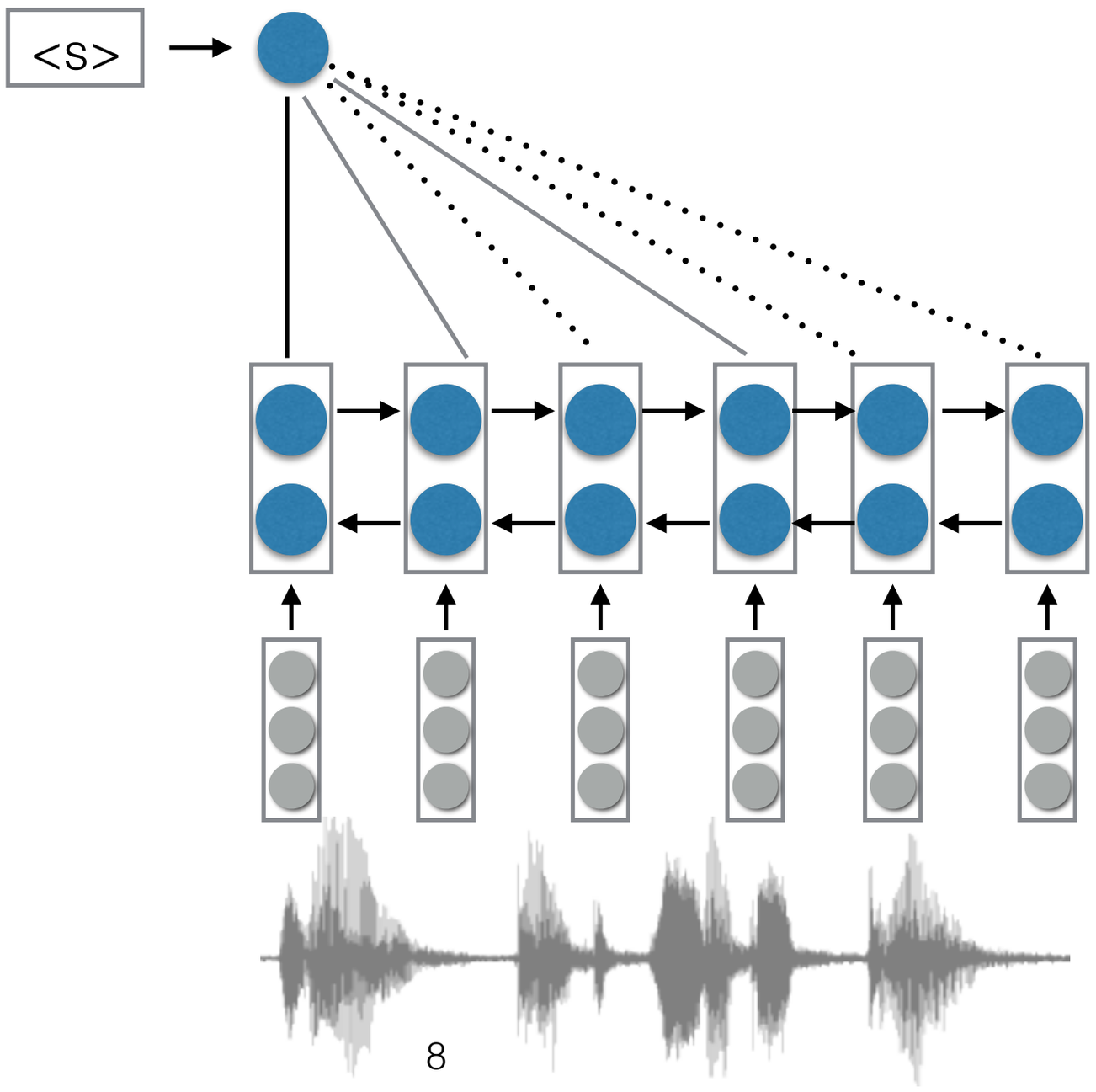
RNN Decoder

Attention

RNN Encoder

Representation

Wave form



Model Overview

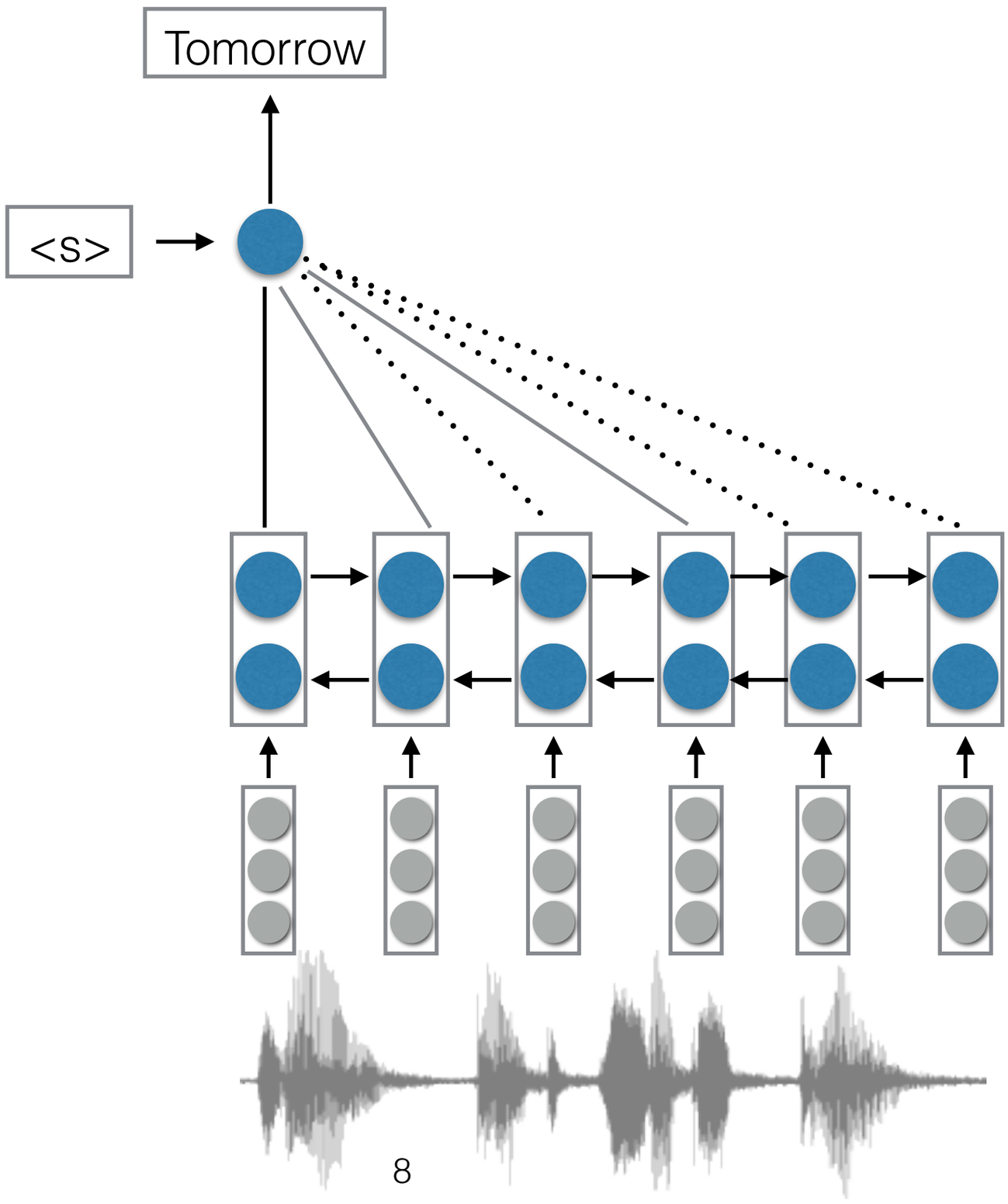
RNN Decoder

Attention

RNN Encoder

Representation

Wave form



Model Overview

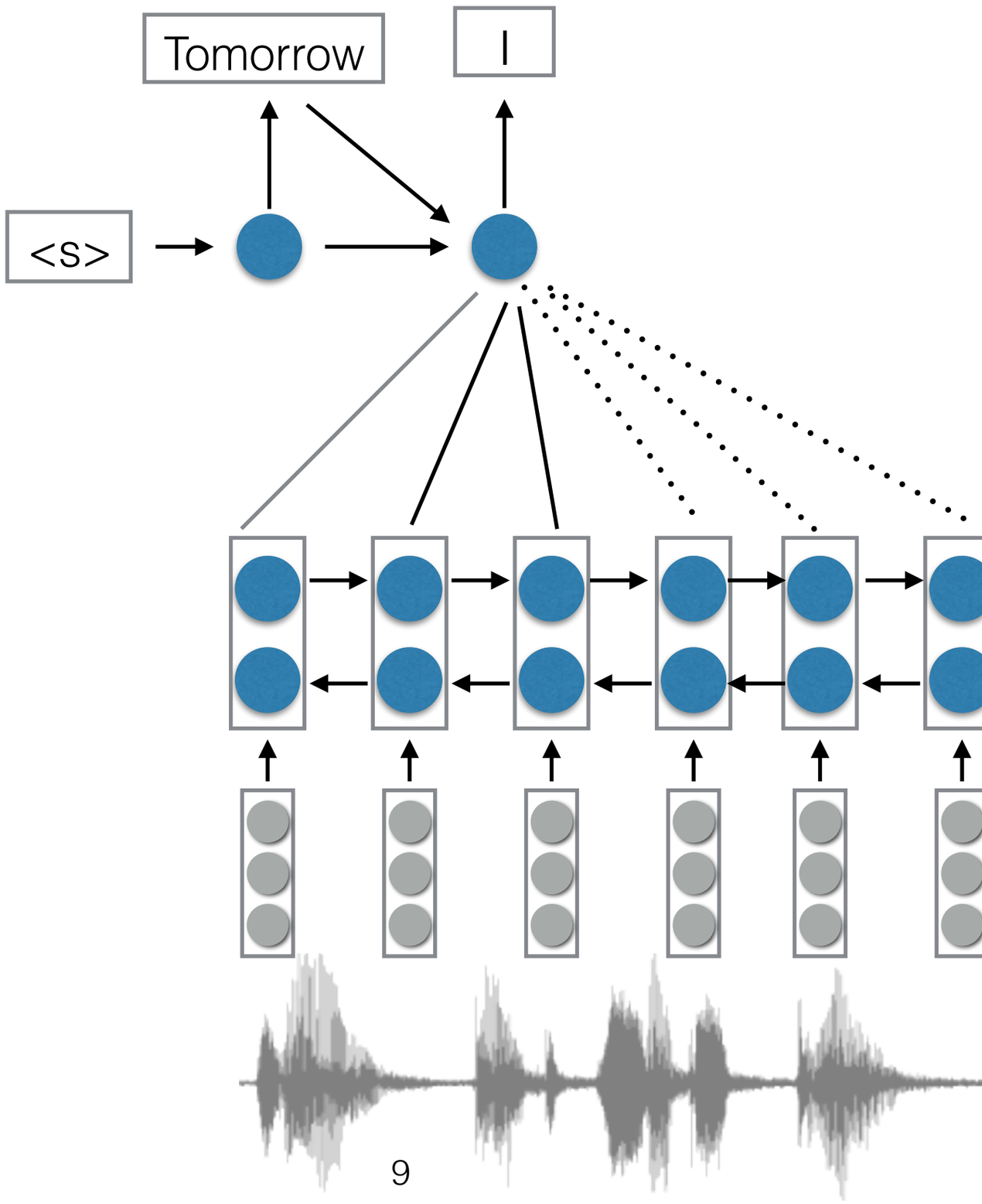
RNN Decoder

Attention

RNN Encoder

Representation

Wave form



Model Overview

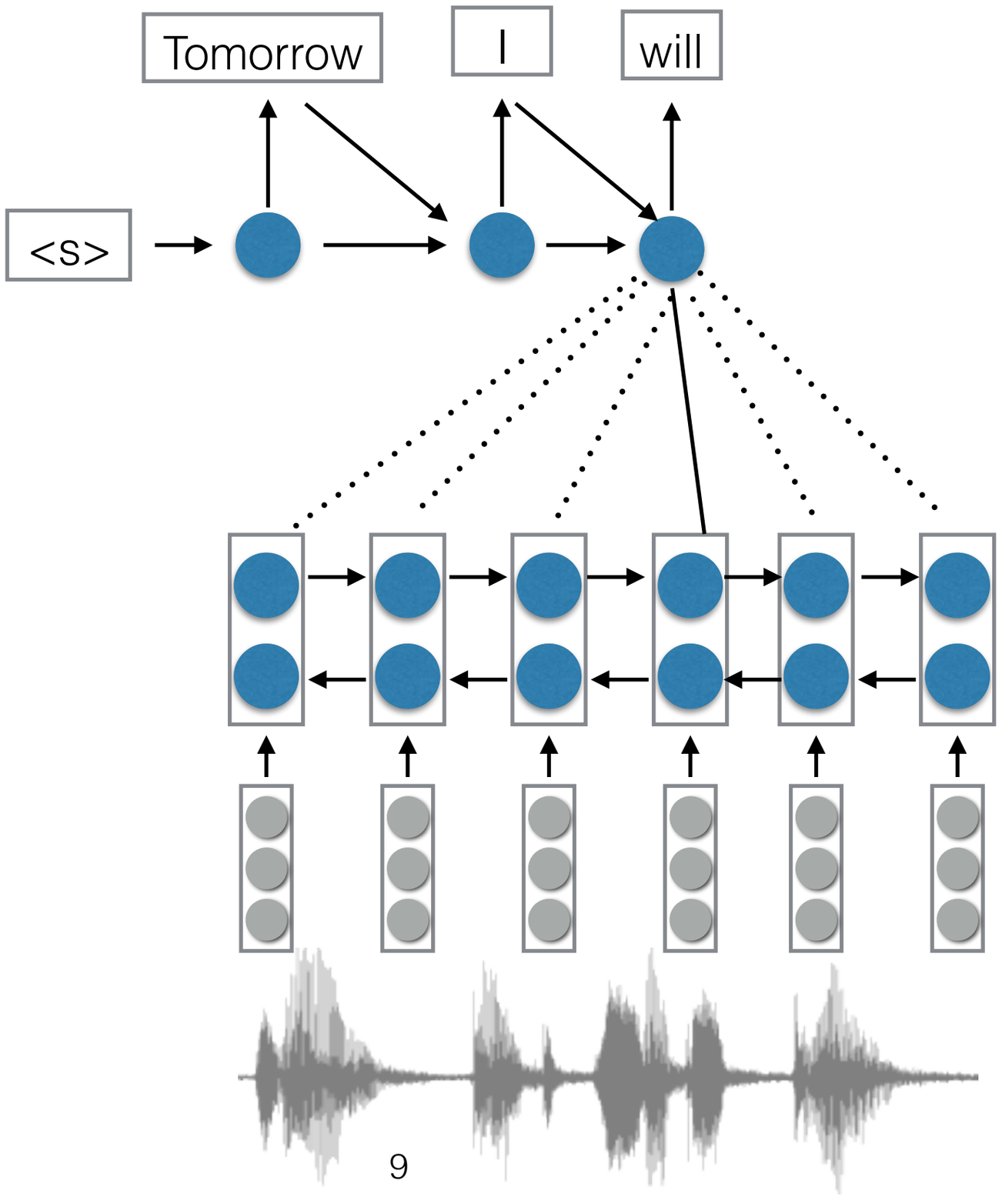
RNN Decoder

Attention

RNN Encoder

Representation

Wave form



Model Overview

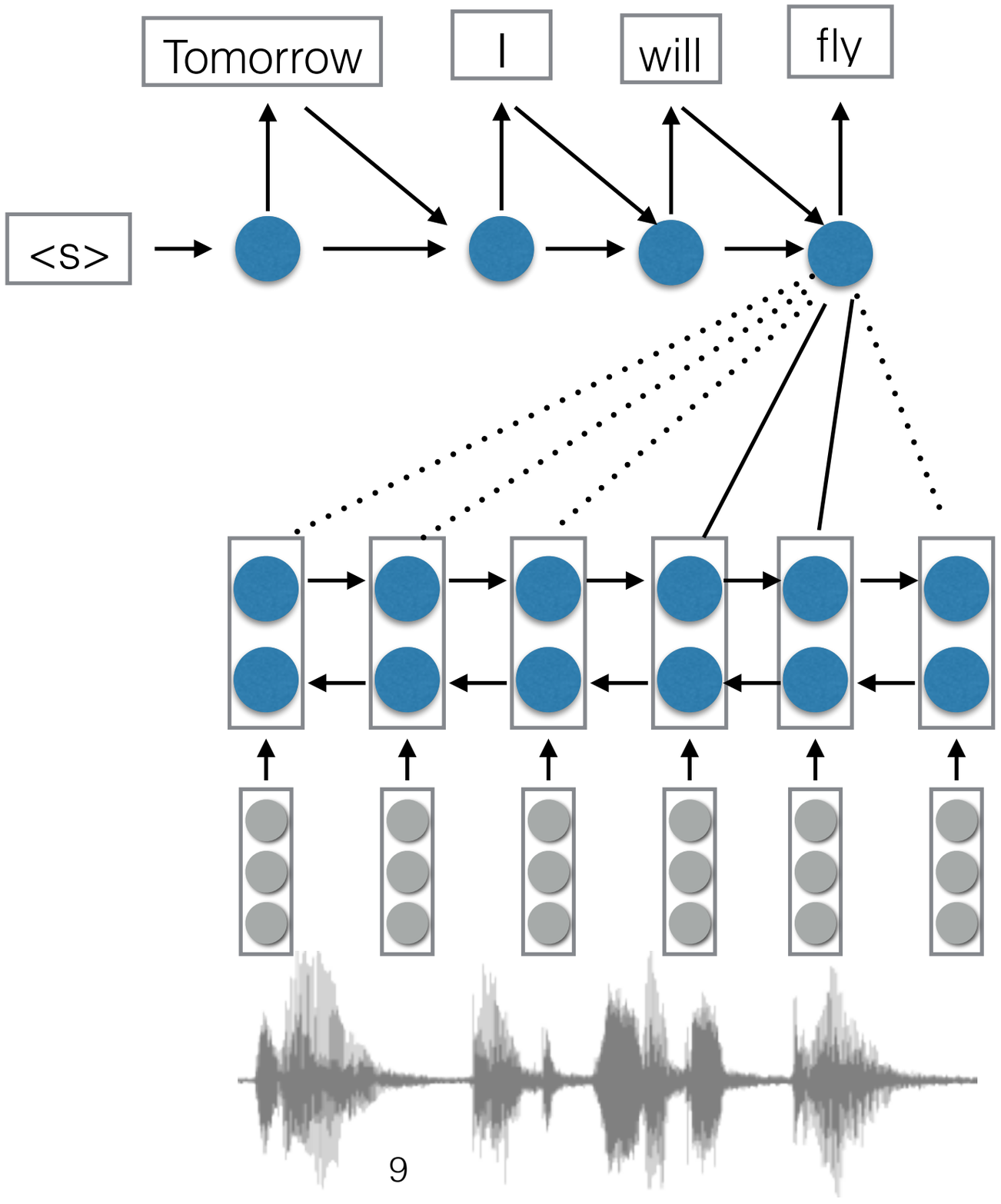
RNN Decoder

Attention

RNN Encoder

Representation

Wave form



Model Overview

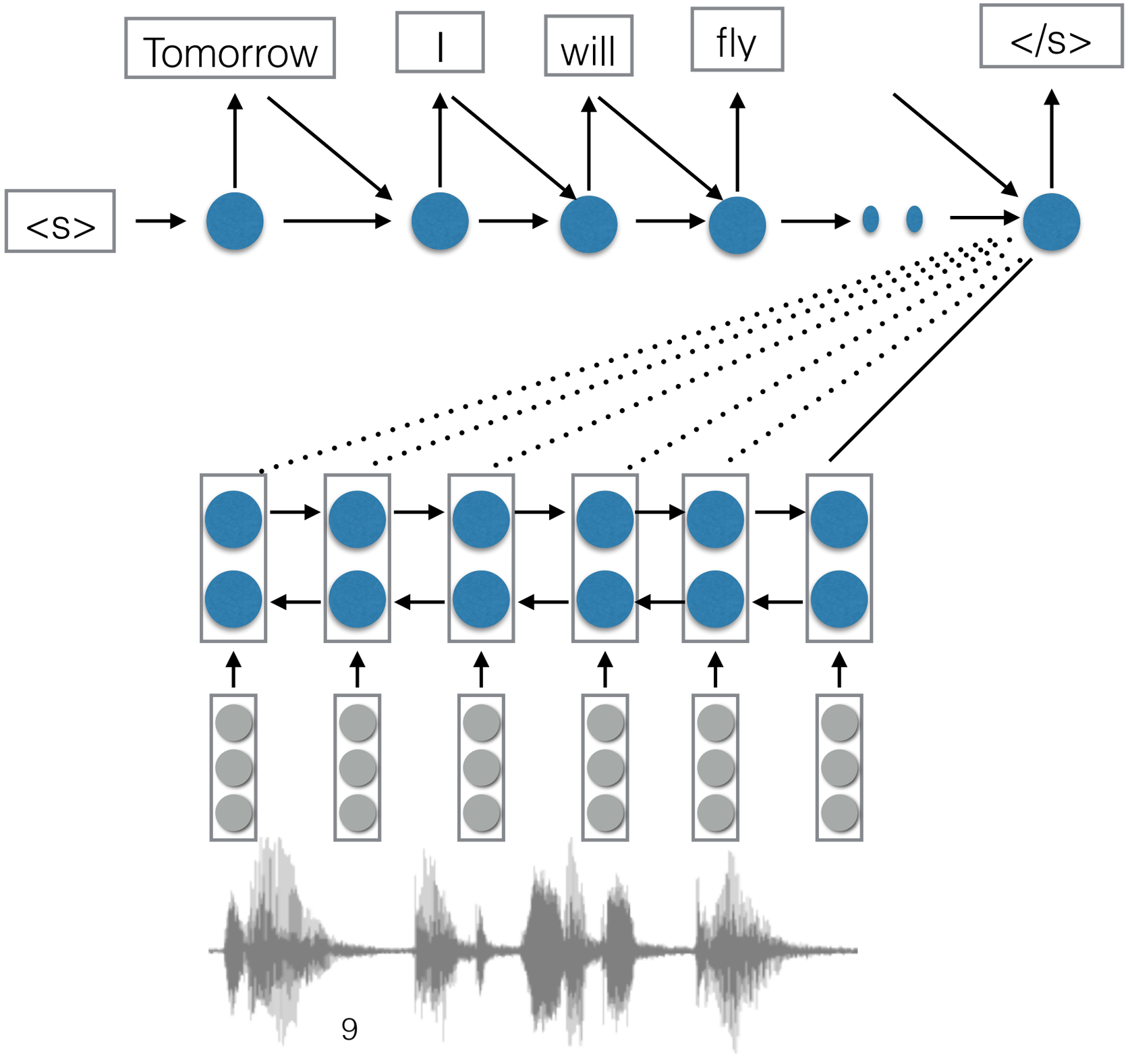
RNN Decoder

Attention

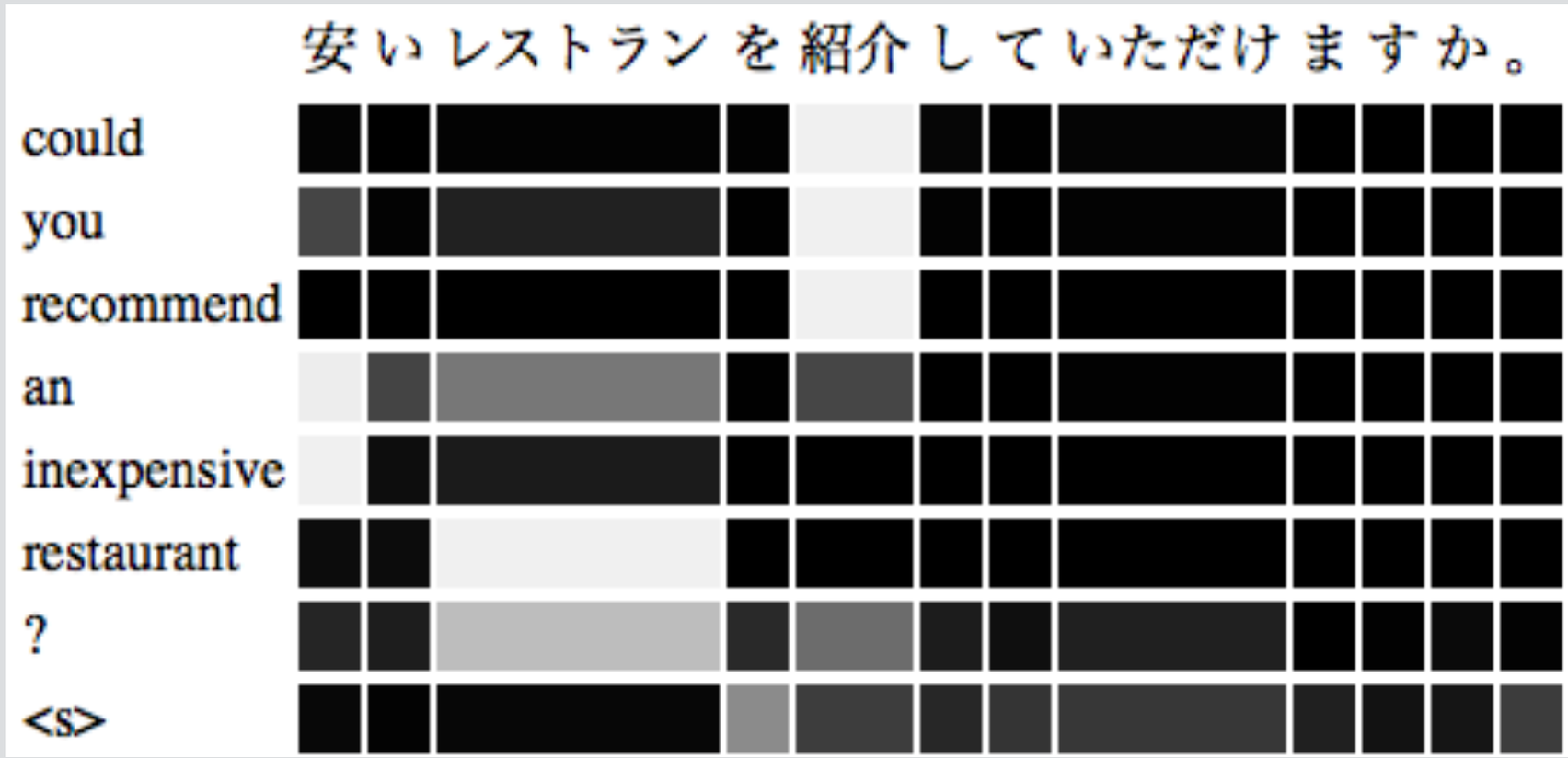
RNN Encoder

Representation

Wave form



A Graphical Example



Attention Score Functions (1)

- ***q*** is the query and ***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)

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

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

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

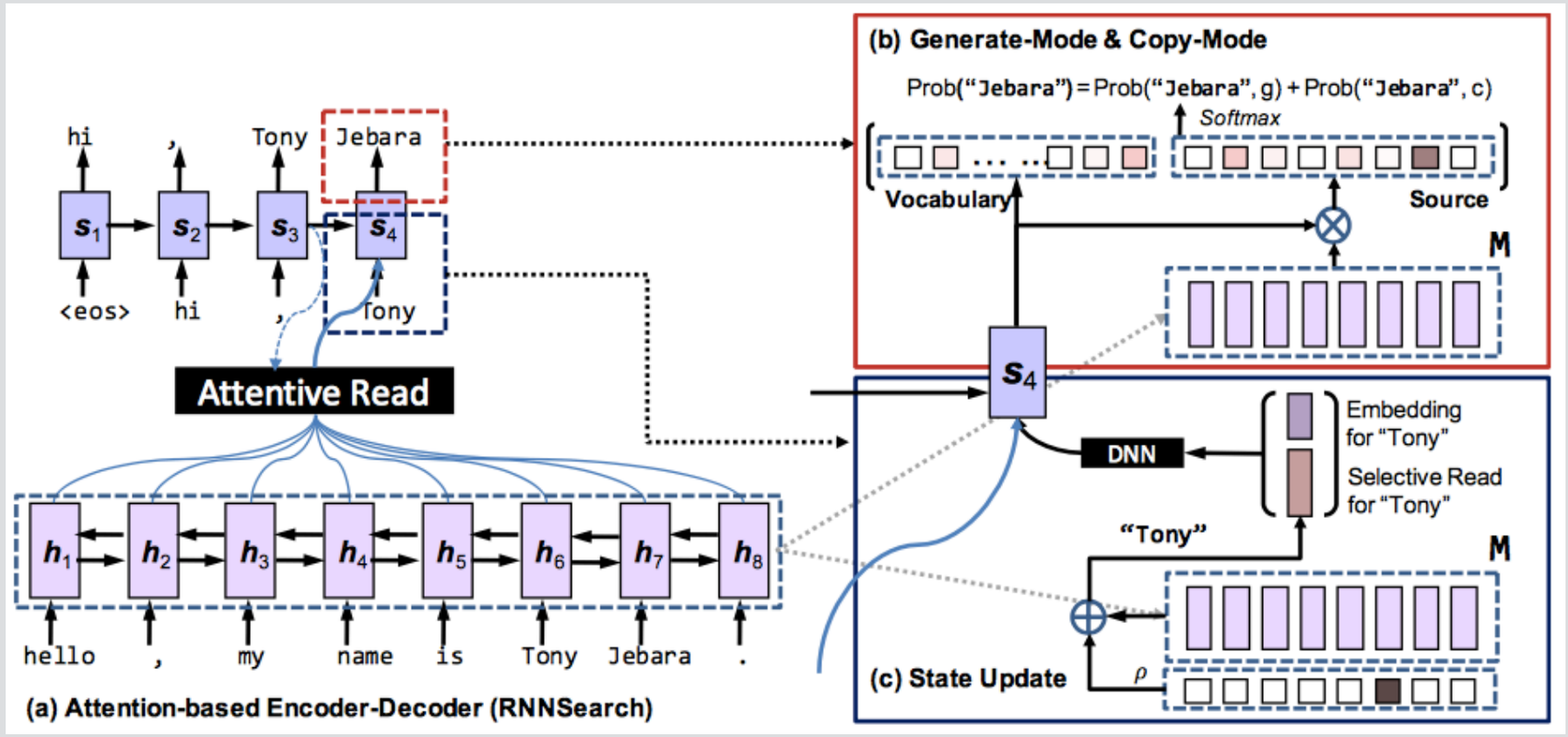
Let’s Try it Out!

`attention.py`

What do we Attend To?

Input Sentence

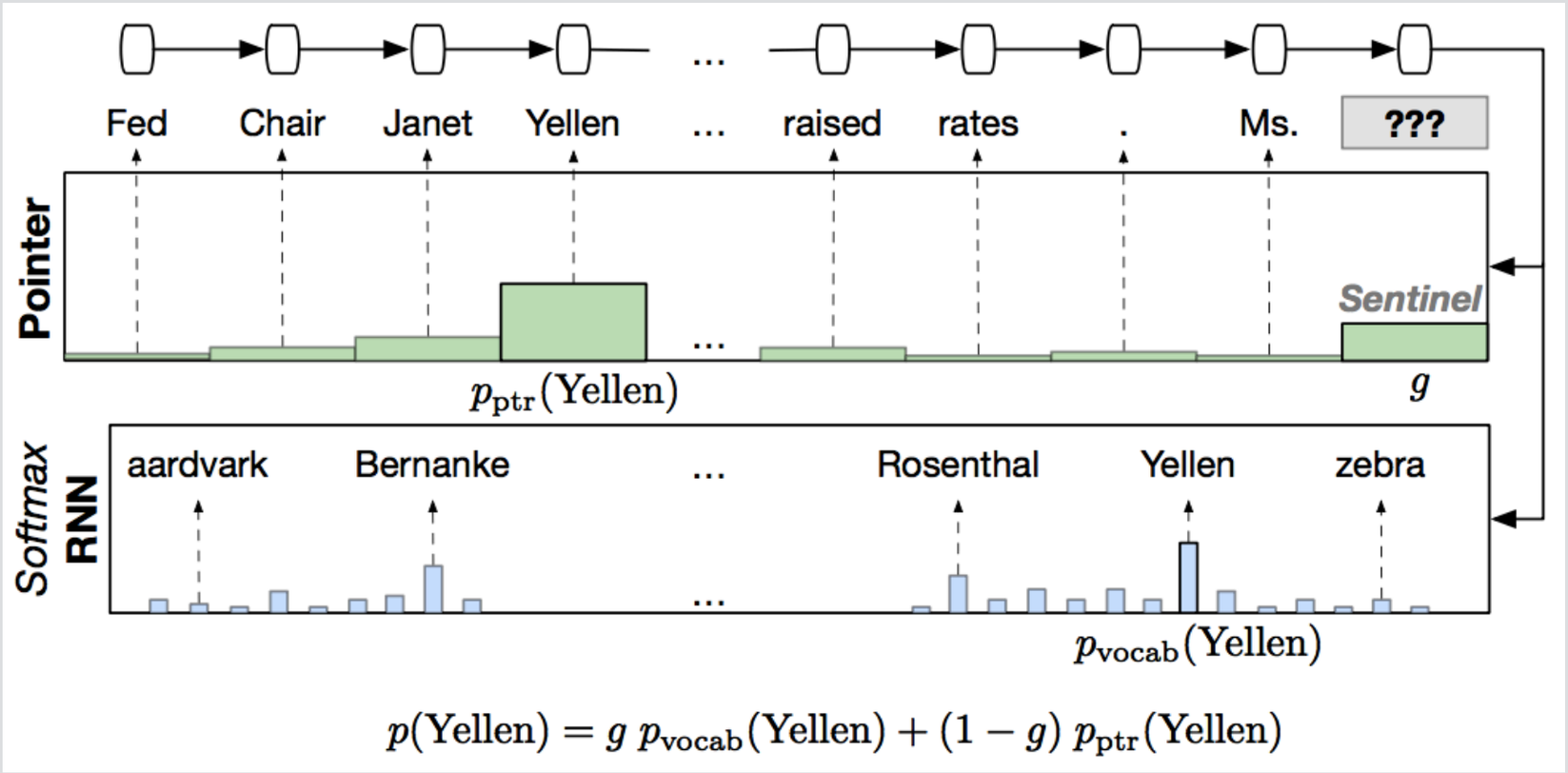
- Like the previous explanation
- But also, more directly
  - **Copying mechanism** (Gu et al. 2016)



- **Lexicon bias** (Arthur et al. 2016)

Previously Generated Things

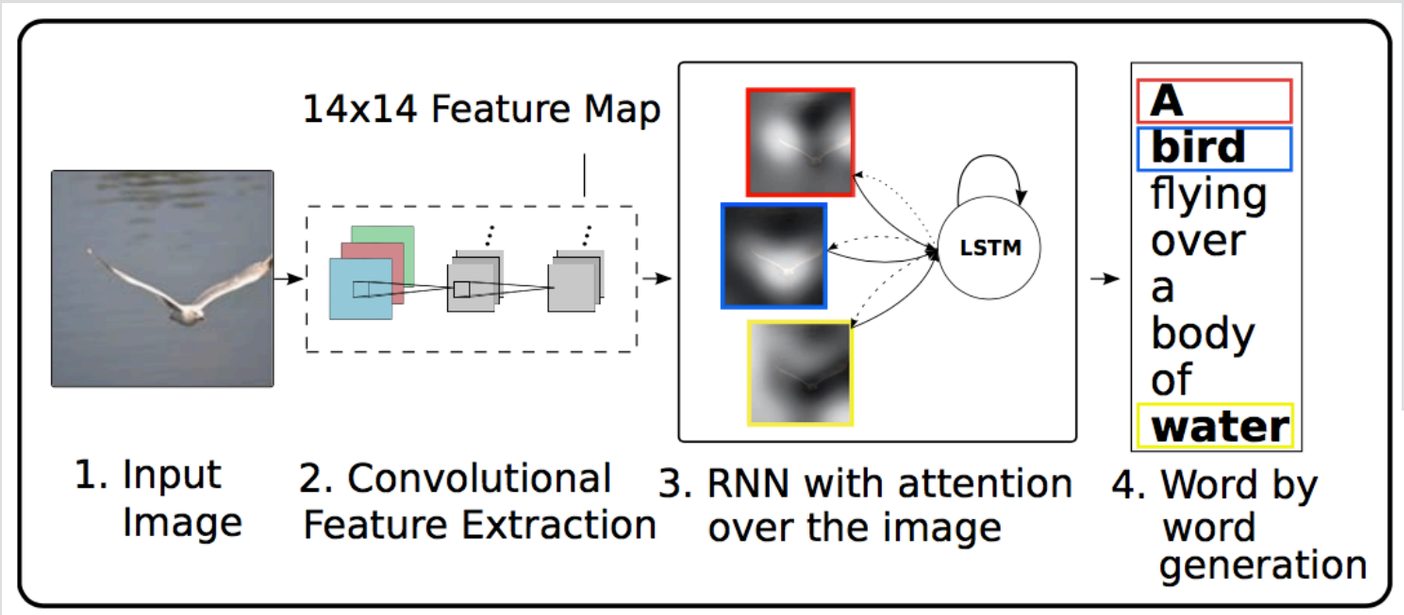
- In language modeling, attend to the previous words (Merity et al. 2016)



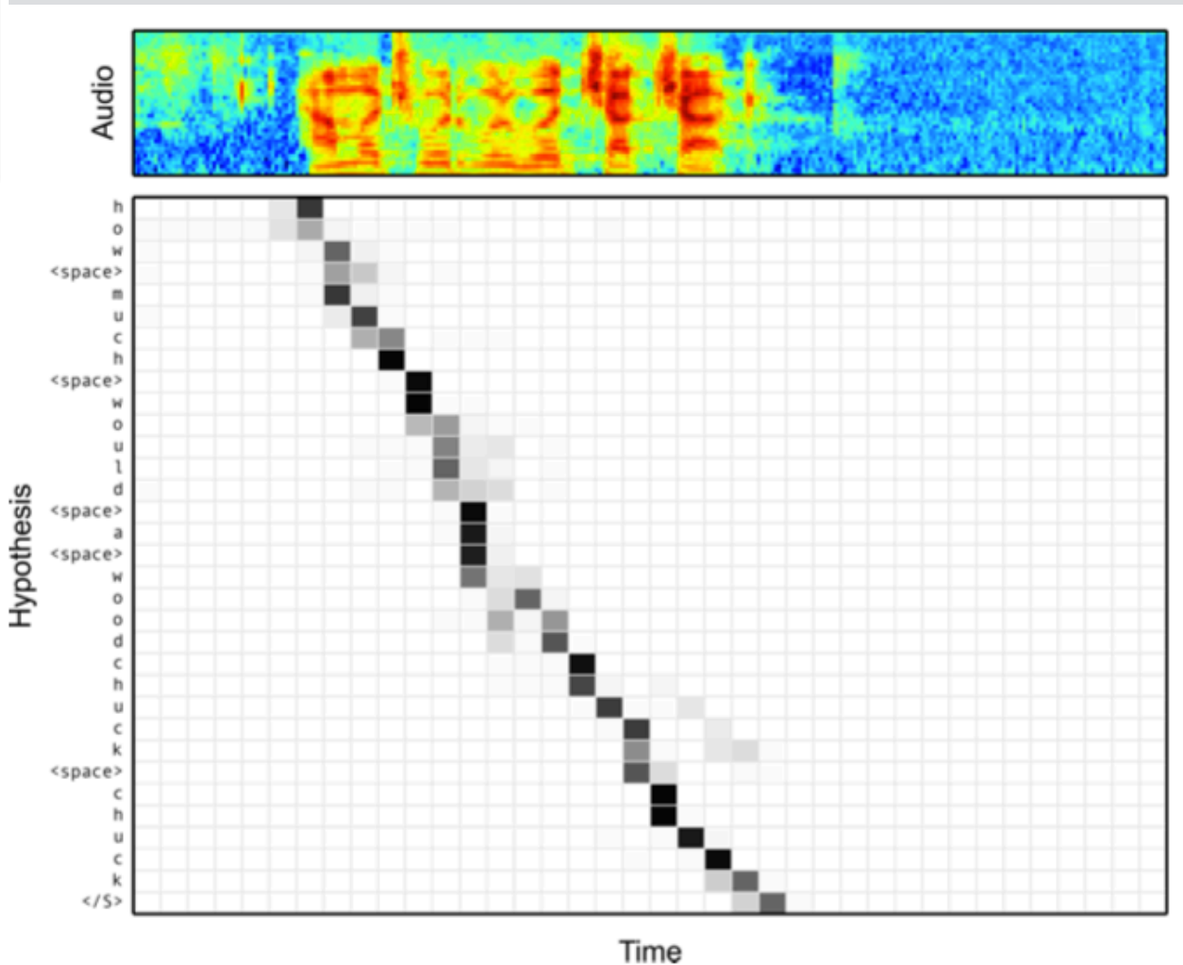
- In translation, attend to either input or previous output (Vaswani et al. 2017)

Various Modalities

- Images (Xu et al. 2015)



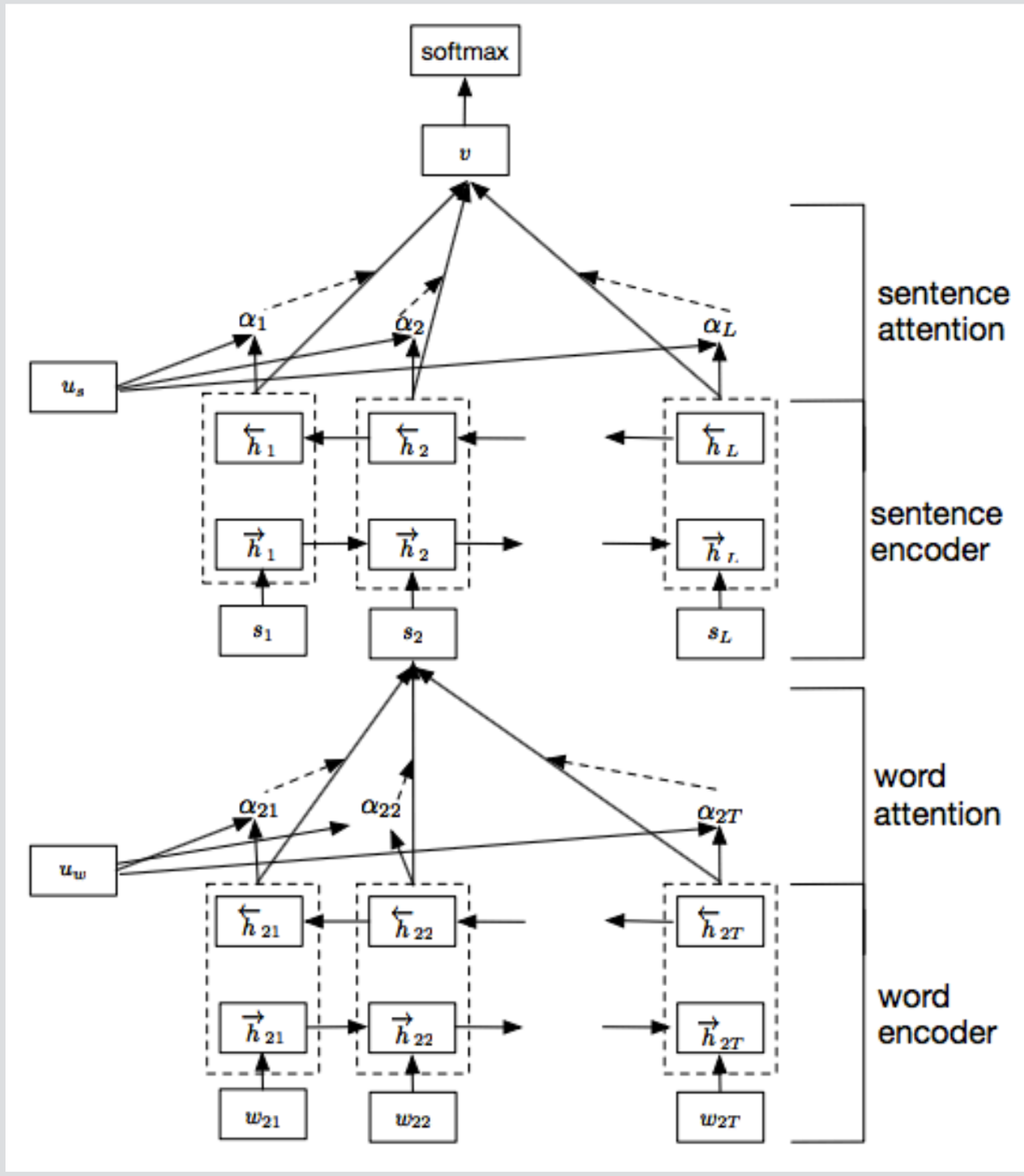
- Speech (Chan et al. 2015)



Hierarchical Structures

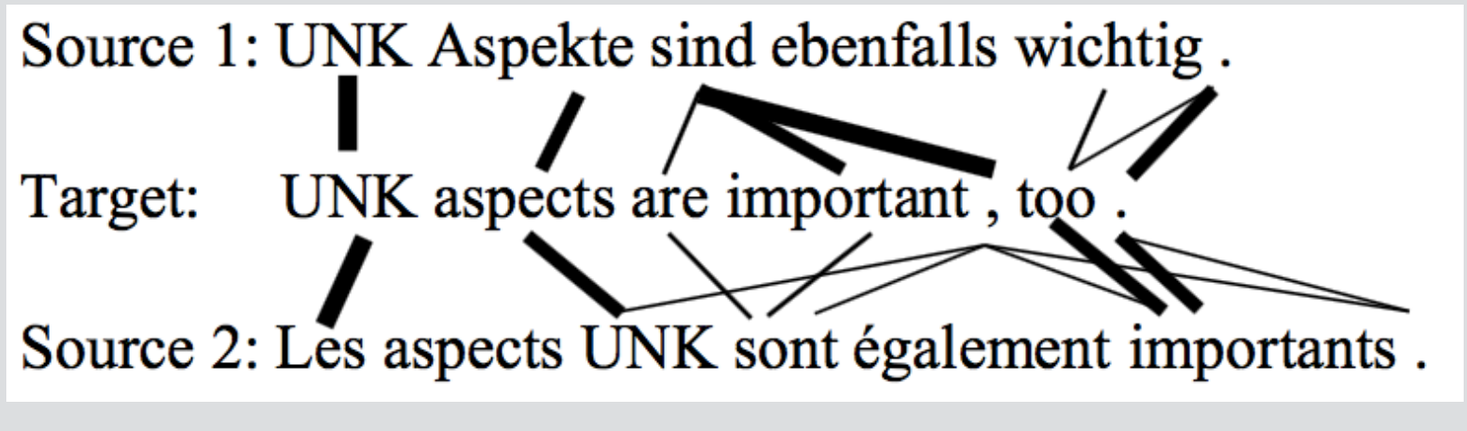
(Yang et al. 2016)

- Encode with attention over each sentence, then attention over each sentence in the document



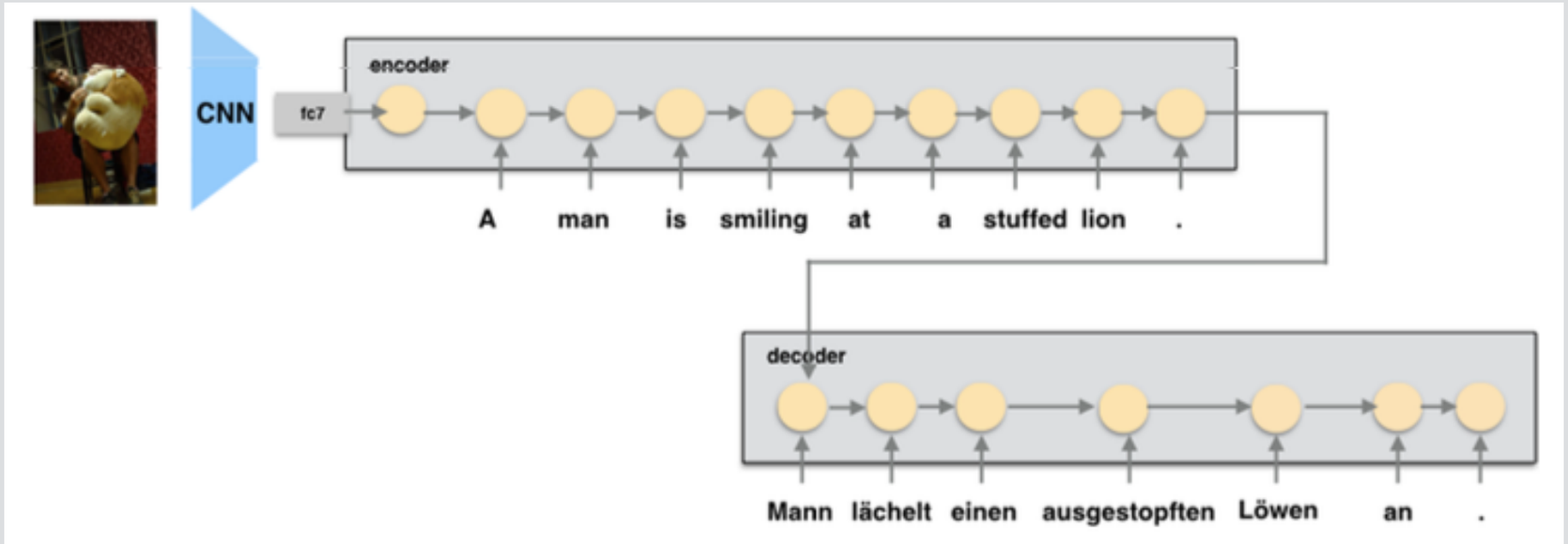
Multiple Sources

- Attend to multiple sentences (Zoph et al. 2015)



  - Libovicky and Helcl (2017) compare multiple strategies

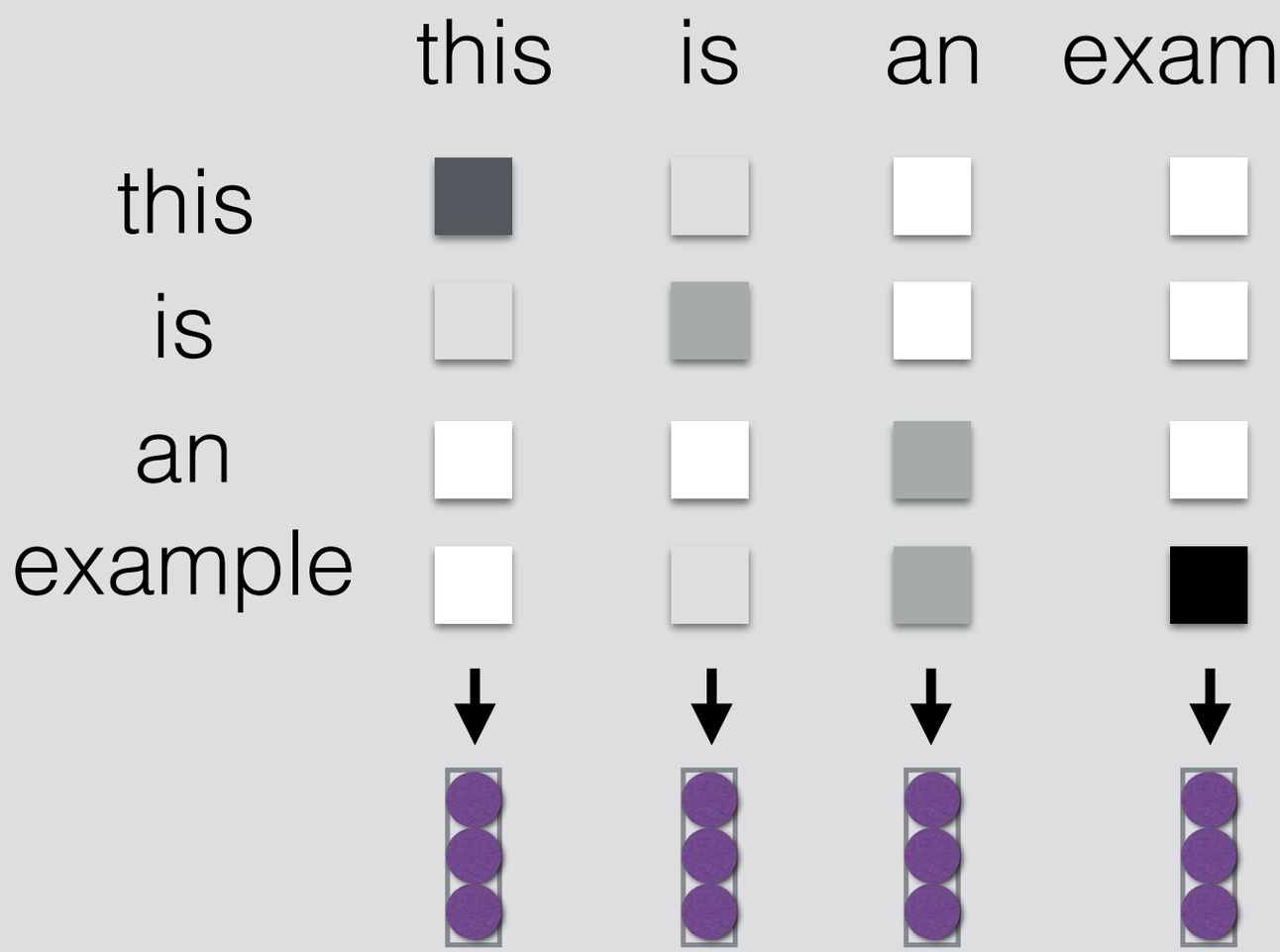
- Attend to a sentence and an image (Huang et al. 2016)



Intra-Attention / Self Attention

(Cheng et al. 2016)

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



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 (Cohn et al. 2015)

  - Add embeddings indicating coverage (Mi et al. 2016)

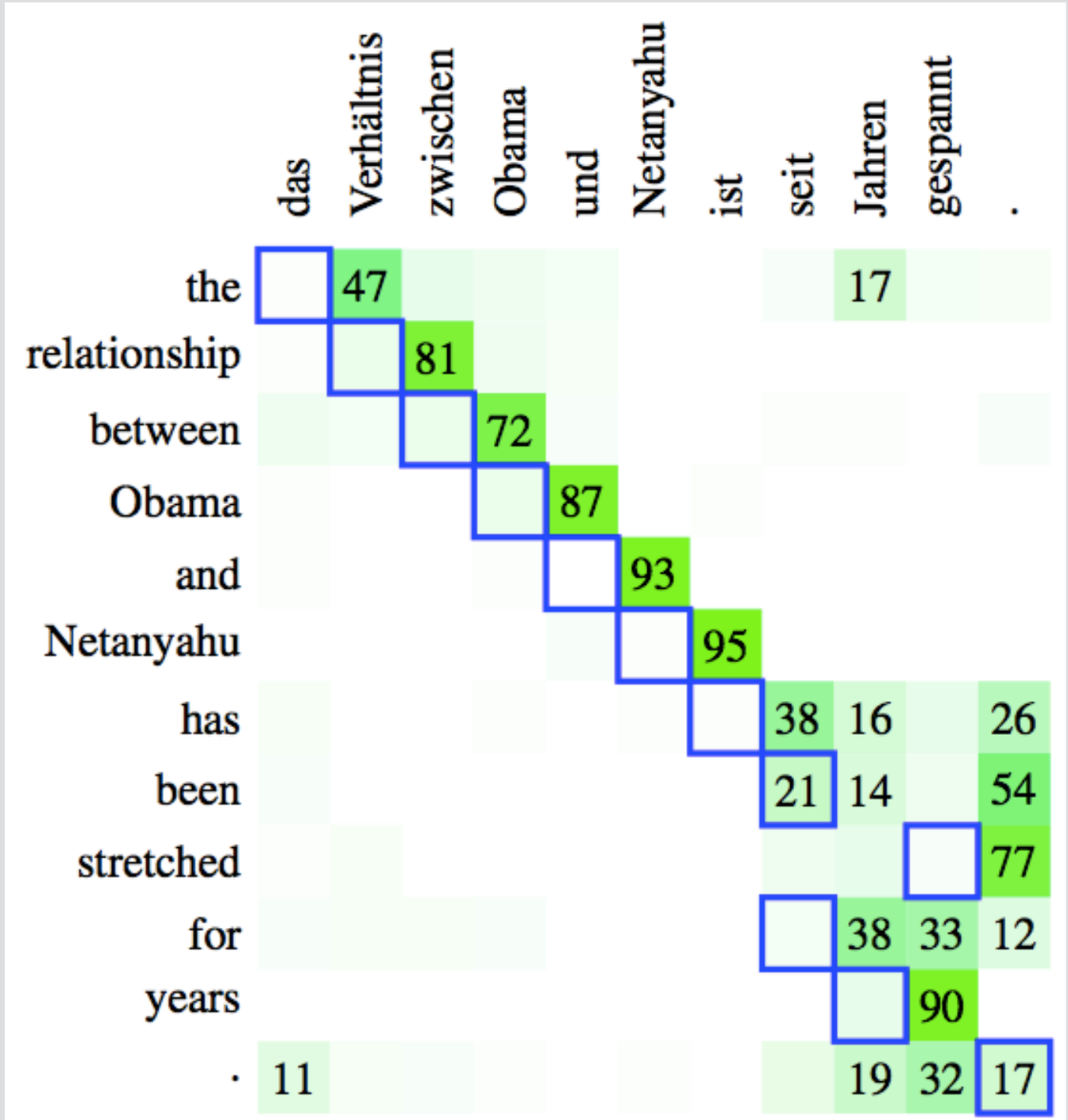
Supervised Training
(Mi et al. 2016)

- 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

Attention is not Alignment!

(Koehn and Knowles 2017)

- Attention is often blurred
- Attention is often off by one



Specialized Attention Varieties

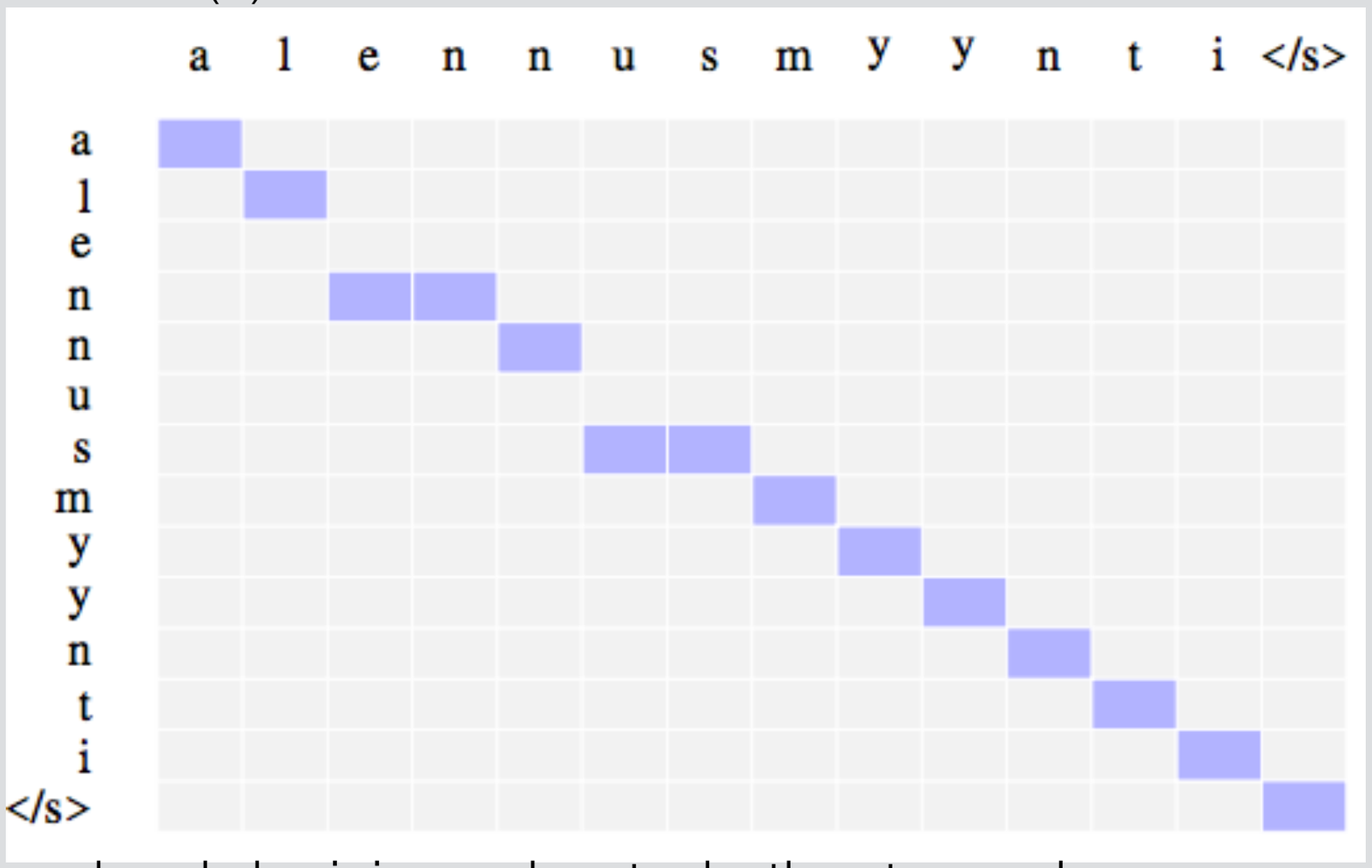
Hard Attention

- Instead of a soft interpolation, make a **zero-one decision** about where to attend (Xu et al. 2015)
  - Harder to train, requires methods such as reinforcement learning (see later classes)
- Perhaps this helps interpretability? (Lei et al. 2016)

Monotonic Attention

(e.g. Yu et al. 2016)

- In some cases, we might know the output will be the same order as the input
  - Speech recognition, incremental translation, morphological inflection (?), summarization (?)



- **Basic idea:** hard decisions about whether to read more

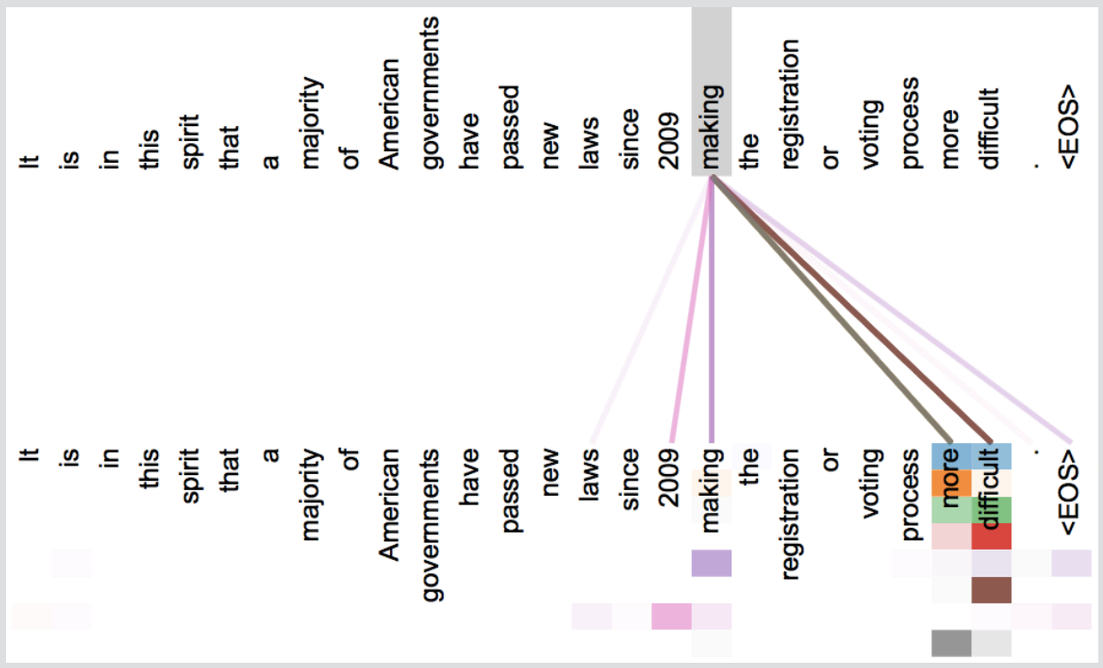
Multi-headed Attention

- **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	$\alpha =$	<s> { this . use Browser Cache = use Browser Cache ; } </s>	0.012
		$\kappa =$	<s> { this . use Browser Cache = use Browser Cache ; } </s>	
m_2	use	$\alpha =$	<s> { this . use Browser Cache = use Browser Cache ; } </s>	0.974
		$\kappa =$	<s> { this . use Browser Cache = use Browser Cache ; } </s>	
m_3	browser	$\alpha =$	<s> { this . use Browser Cache = use Browser Cache ; } </s>	0.969
		$\kappa =$	<s> { this . use Browser Cache = use Browser Cache ; } </s>	
m_4	cache	$\alpha =$	<s> { this . use Browser Cache = use Browser Cache ; } </s>	0.583
		$\kappa =$	<s> { this . use Browser Cache = use Browser Cache ; } </s>	
m_5	END	$\alpha =$	<s> { this . use Browser Cache = use Browser Cache ; } </s>	0.066
		$\kappa =$	<s> { this . use Browser Cache = use Browser Cache ; } </s>	

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



- Or one head for every hidden node! (Choi et al. 2018)

An Interesting Case Study:

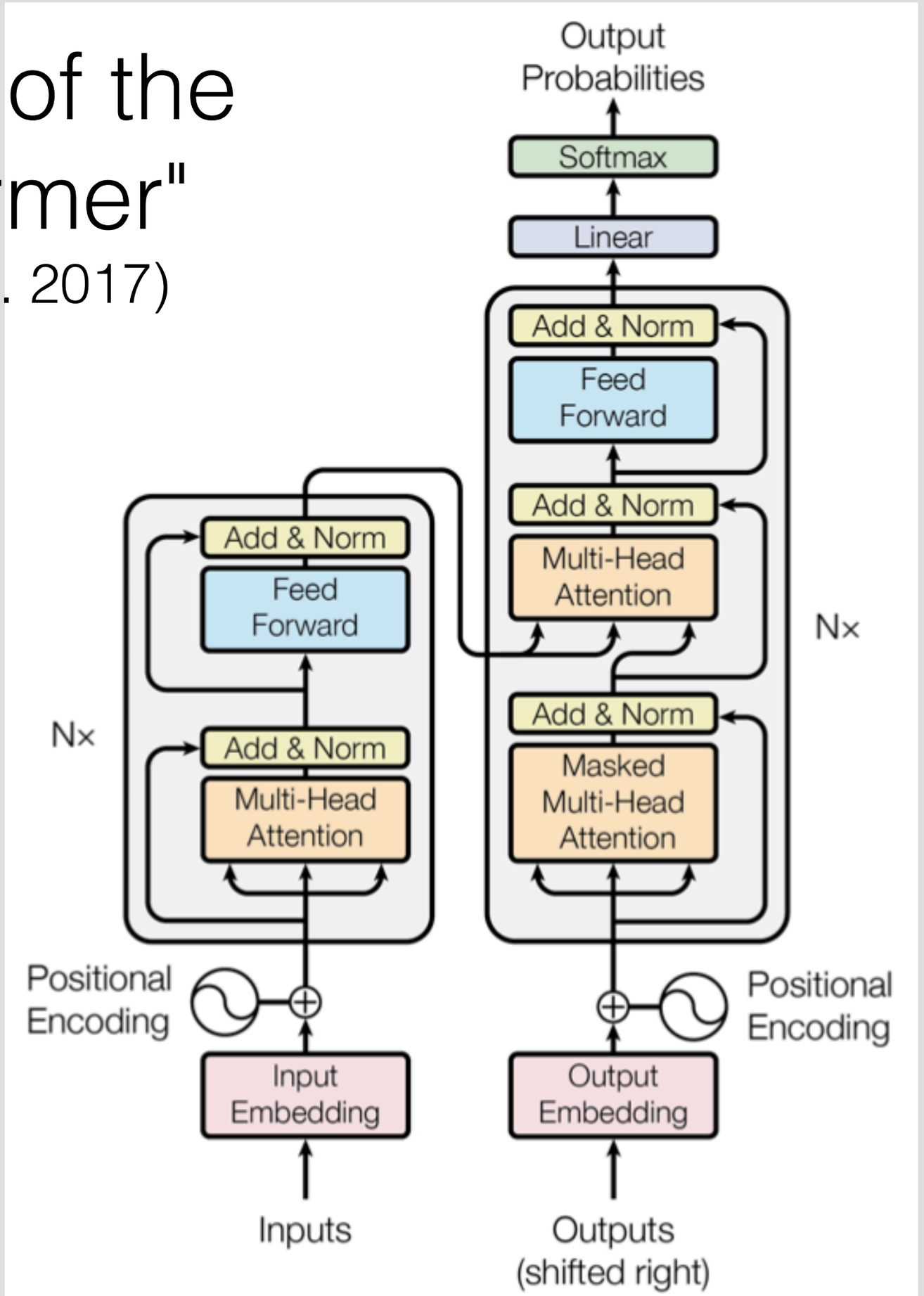
“Attention is All You Need”

(Vaswani et al. 2017)

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



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

Training Tricks

- **Layer Normalization:** Help ensure that layers remain in reasonable range

- **Specialized Training Schedule:** Adjust default learning rate of the Adam optimizer

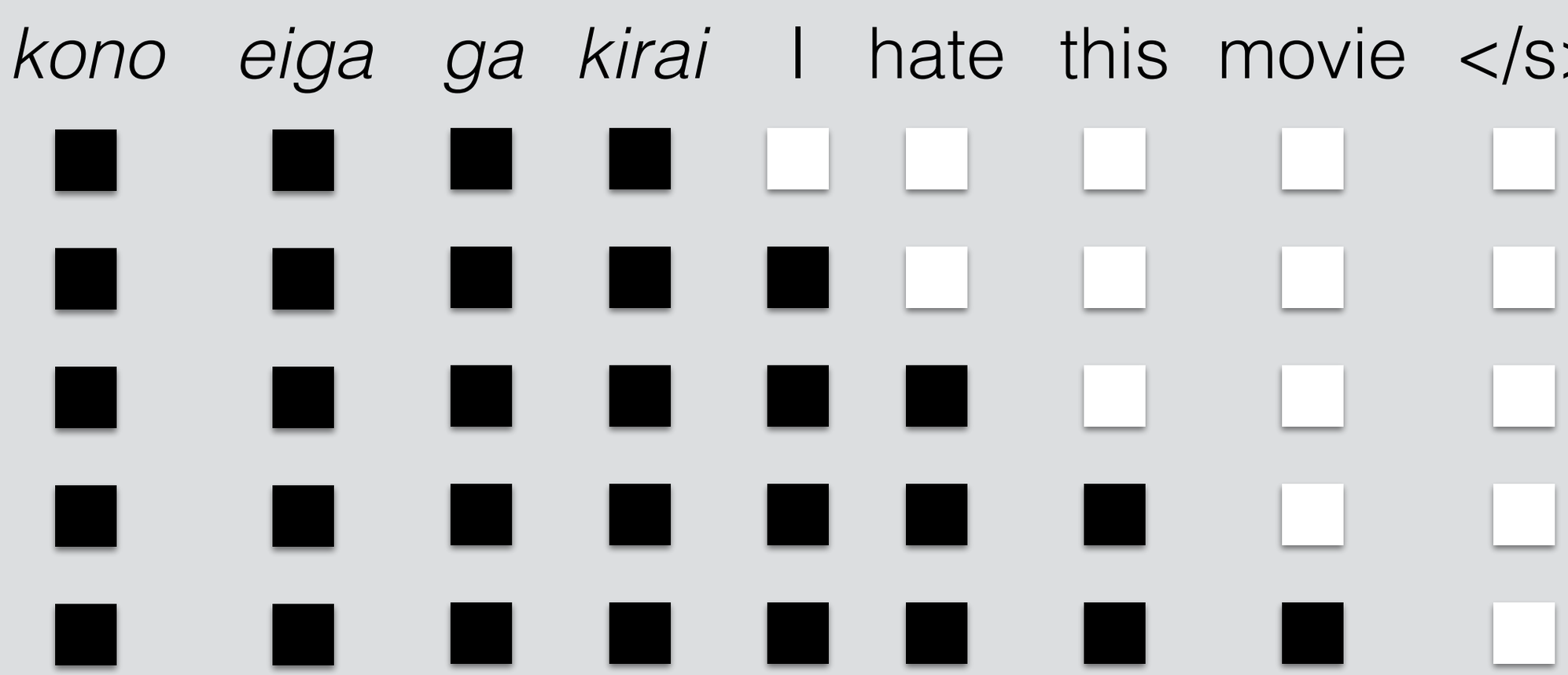
- **Label Smoothing:** Insert some uncertainty in the training process

- **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



Questions?