CS230: Lecture 8
Word2Vec applications + Recurrent Neural Networks with Attention

Kian Katanforoosh, Andrew Ng

Today's outline

We will learn how to:

- **Generalize** results with word vectors
- Augment a RNN with **Attention mechanisms**

I. Word Vector Representation
   i. Training
   ii. Operations
   iii. Applications: debasing / restaurant reviews

II. Attention
   i. Machine Translation
   ii. Image Captioning

Word Vector Representation

vocabulary

$$V_{eng} = \begin{pmatrix} "a" \\ "abs" \\ \vdots \\ "football" \\ \vdots \\ "zone" \\ "zoo" \end{pmatrix}$$

one-hot representation

$$v_{football} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

word-vector representation

$$e_{football} = \begin{pmatrix} 0.11 \\ \vdots \\ 0.31 \\ 0.84 \\ \vdots \\ 0.57 \end{pmatrix}$$

How do you get the vector representation?

Kian Katanforoosh, Andrew Ng

Word Vector Representation: Training

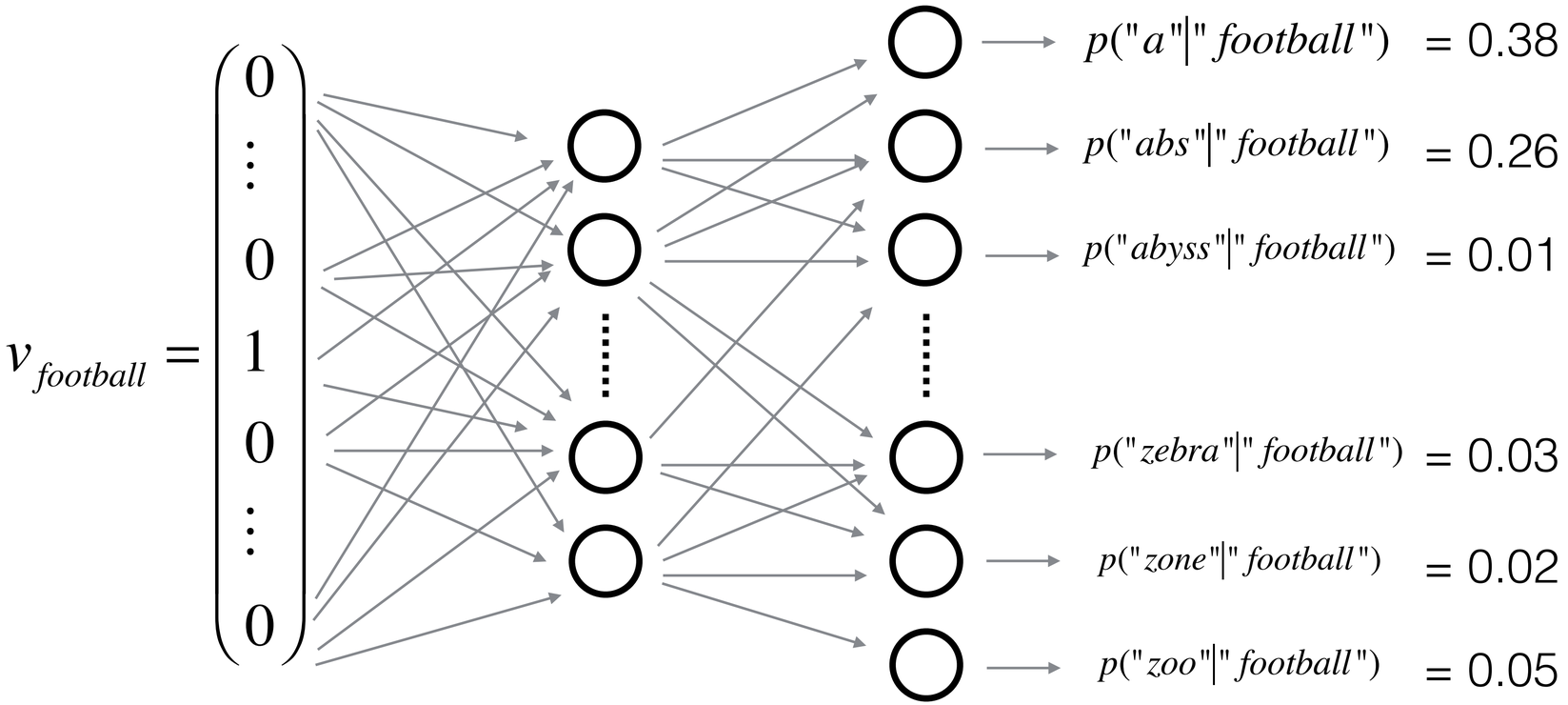
1. Data?

window

Stanford is going to beat Cal next week

target

Target word (x)	Nearby word (y)
Stanford	is
is	Stanford
is	going
going	is
going	to
…	…



one-hot

loss? $L = -\sum y \log(\hat{y})$

$$z^{[1]} = W^{[1]} v + b^{[1]}$$

$shape = (200, |V|)$

$$\begin{pmatrix} \vdots & \vdots & & \vdots & \vdots \\ e_{"a"} & e_{"abs"} & \cdots & e_{"zone"} & e_{"zoo"} \\ \vdots & \vdots & & \vdots & \vdots \end{pmatrix}$$

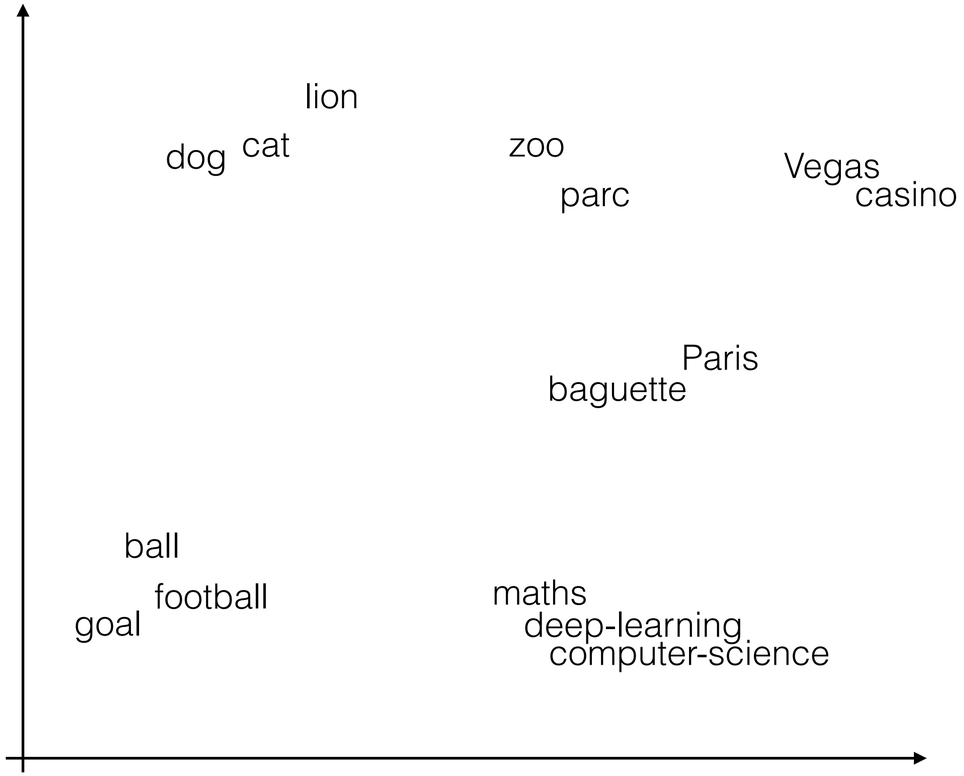
Kian Katanforoosh, Andrew Ng

Word Vector Representation: Embedding matrix

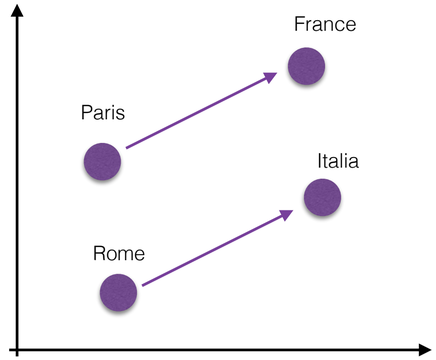
$$W^{[1]} v_{football} = \begin{pmatrix} \vdots & \vdots & & \vdots & \vdots \\ e_{"a"} & e_{"abs"} & \cdots & e_{"zone"} & e_{"zoo"} \\ \vdots & \vdots & & \vdots & \vdots \end{pmatrix} \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} = e_{football}$$

Word Vector Representation: Visualization and Operations

Scatter plot of Word vectors



Operations on vectors



Expression	*Nearest token*
Paris - France + Italy	Rome
bigger - big + cold	colder
sushi - Japan + Germany	bratwurst
Cu - copper + gold	Au
Windows - Microsoft + Google	Android
Montreal Canadiens - Montreal + Toronto	Toronto Maple Leafs

$$man - king = women - x$$

$$x = queen$$

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, Jeffrey Dean: Distributed Representations of Words and Phrases and their Compositionality
Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean: Efficient Estimation of Word Representations in Vector Space
Laurens van der Maaten, Geoffrey Hinton: Visualizing Data using t-SNE

Word Vector Representation: bias

Sexist bias

$$man - king = women - x$$

$$x = queen$$

but also …

man - woman = computer programmer - homemaker

Extreme *she* occupations

1. homemaker	2. nurse	3. receptionist
4. librarian	5. socialite	6. hairdresser
7. nanny	8. bookkeeper	9. stylist
10. housekeeper	11. interior designer	12. guidance counselor

Extreme *he* occupations

1. maestro	2. skipper	3. protege
4. philosopher	5. captain	6. architect
7. financier	8. warrior	9. broadcaster
10. magician	11. figher pilot	12. boss

Figure 1: The most extreme occupations as projected on to the *she*−*he* gender direction on g2vN
Occupations such as *businesswoman*, where gender is suggested by the orthography, were excluded.

Gender stereotype *she-he* analogies.

sewing-carpentry	register-nurse-physician	housewife-shopkeeper
nurse-surgeon	interior designer-architect	softball-baseball
blond-burly	feminism-conservatism	cosmetics-pharmaceuticals
giggle-chuckle	vocalist-guitarist	petite-lanky
sassy-snappy	diva-superstar	charming-affable
volleyball-football	cupcakes-pizzas	hairdresser-barber

Gender appropriate *she-he* analogies.

queen-king	sister-brother	mother-father
waitress-waiter	ovarian cancer-prostate cancer	convent-monastery

Figure 2: **Analogy examples**. Examples of automatically generated analogies for the pair *she-he* using the procedure described in text. For example, the first analogy is interpreted as *she*:*sewing* :: *he*:*carpentry* in the original w2vNEWS embedding. Each automatically generated analogy is evaluated by 10 crowd-workers are to whether or not it reflects gender stereotype. Top: illustrative gender stereotypic analogies automatically generated from w2vNEWS, as rated by at least 5 of the 10 crowd-workers. Bottom: illustrative generated gender-appropriate analogies.

Tolga Bolukbasi et al. : Man is to Computer Programmer as Woman is to Homemaker? Debiasing Word Embeddings

Kian Katanforoosh, Andrew Ng

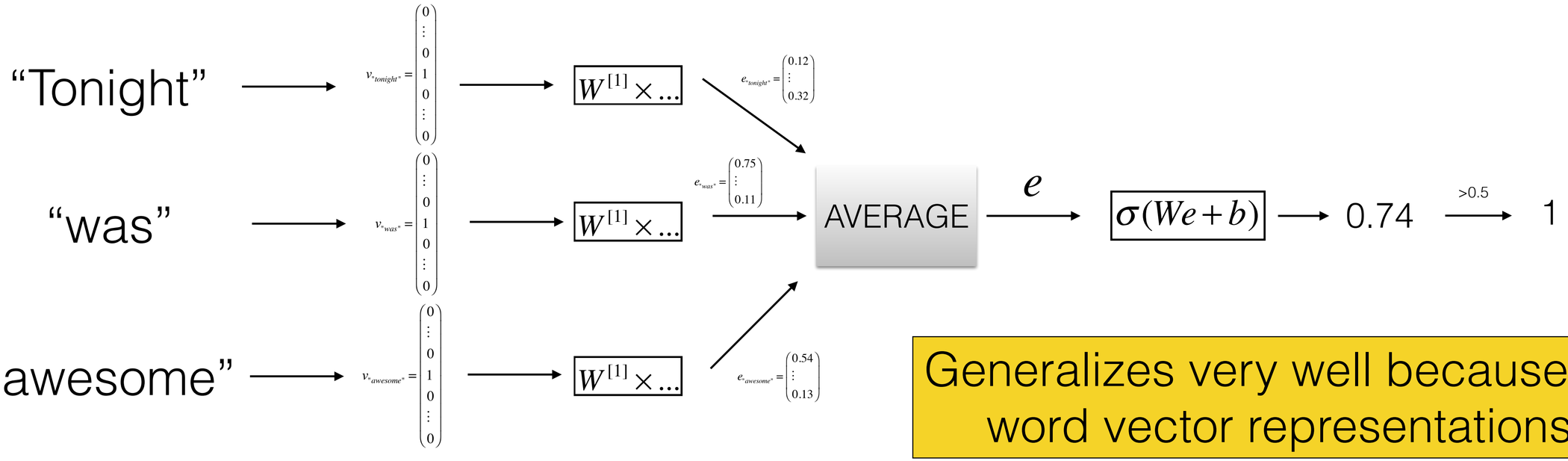
Word Vector Representation: Application

Sentiment analysis on restaurant reviews

1. **Data?** 50 reviews

Review (x)	Label (Negative/
“Tonight was awesome”	1
“Worst entrée ever!”	0
….	….

2. **Architecture?**



Generalizes very well because of word vector representations

3. **Loss?**

$$L = y\log(\hat{y}) + (1-y)\log(1-\hat{y})$$

Kian Katanforoosh, Andrew Ng

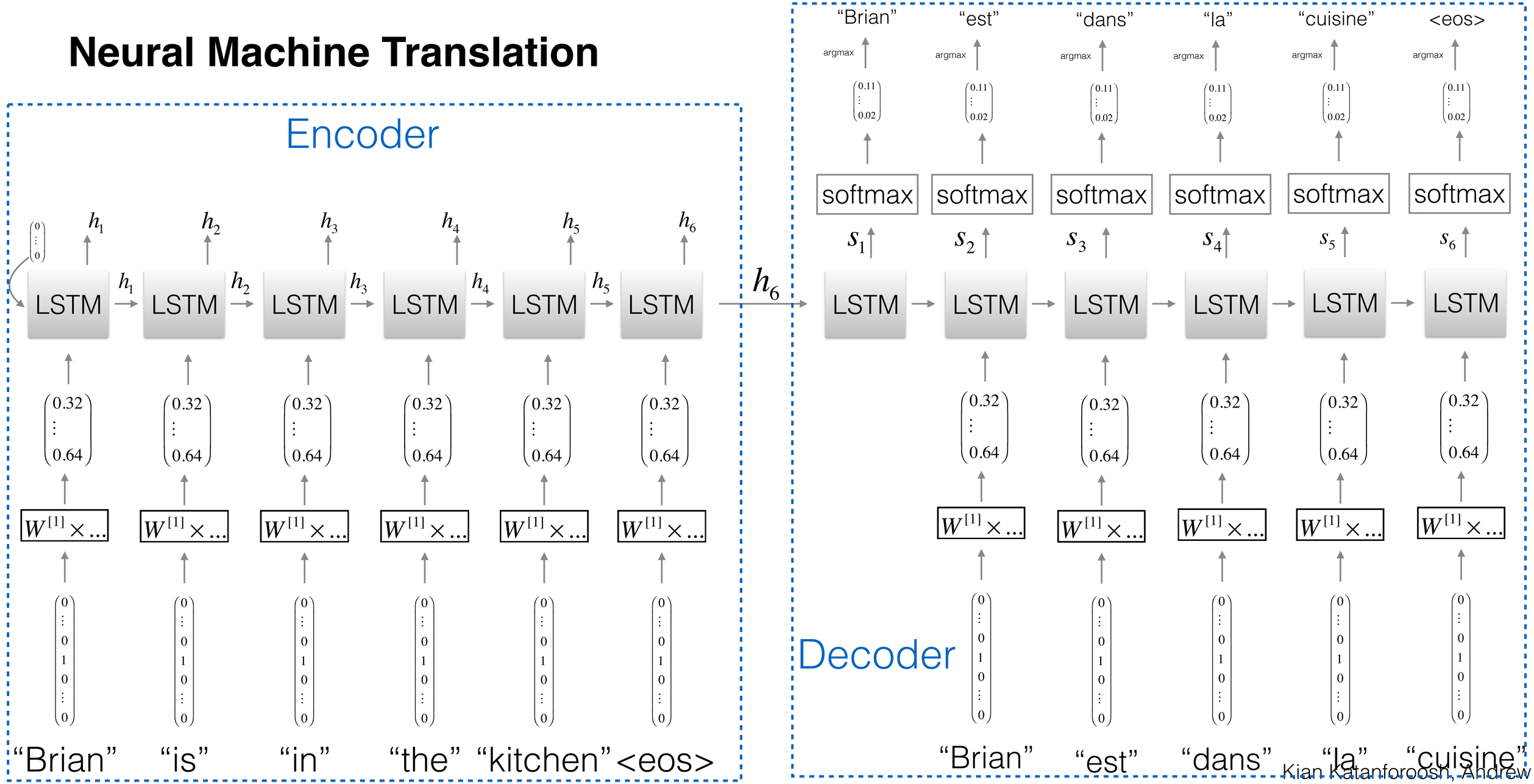
Word Vector Representation: Application

To remember:

- In NLP, Words are often represented by “meaningful” vectors
- These vectors are trained thanks to a Neural Network
- We can do operations on these vectors
- They can be biased, depending on the dataset used to train them
- They have a great generalization power

Attention: Motivation

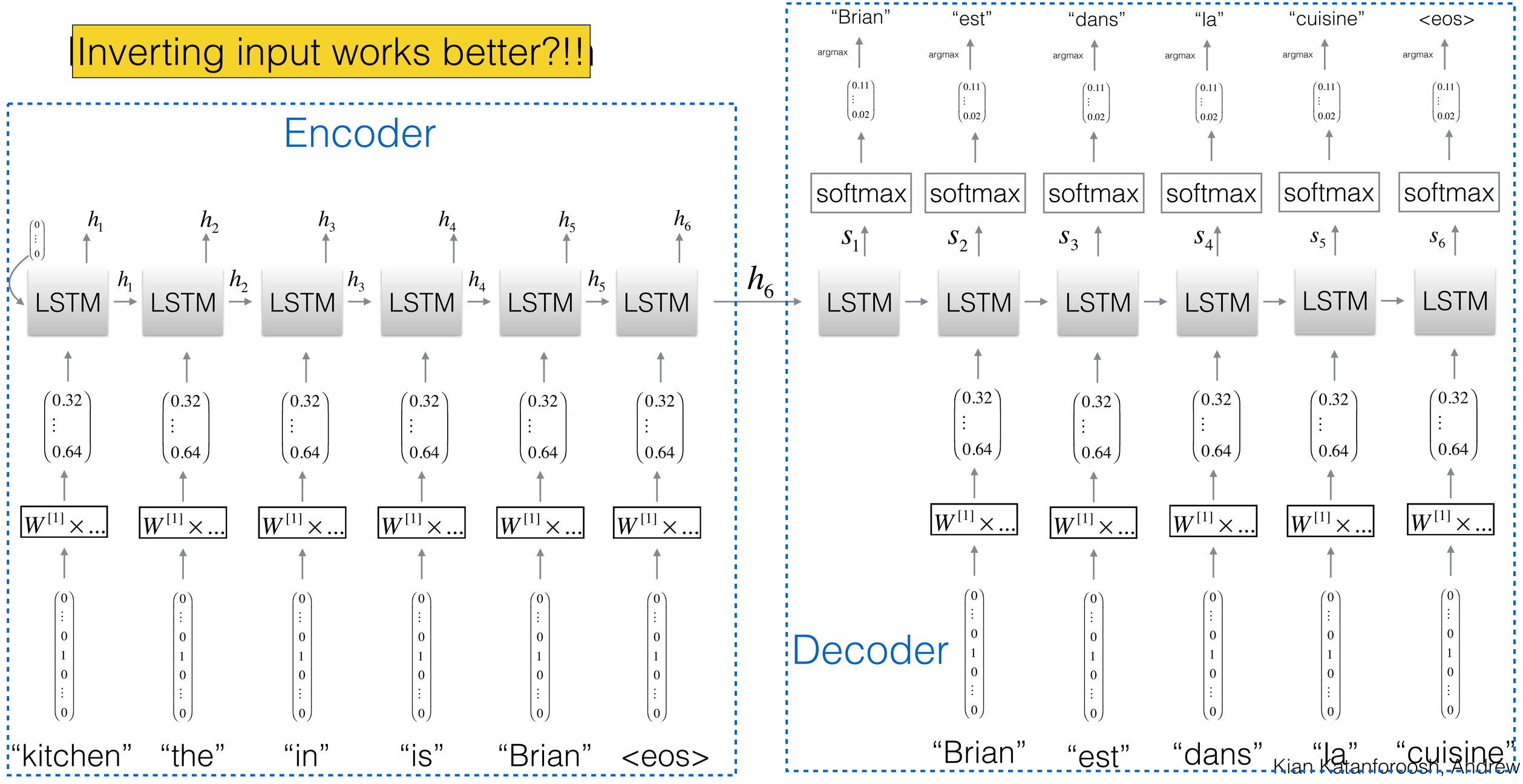
Neural Machine Translation



Kian Katanforoosh, Andrew Ng

Attention: Motivation

Inverting input works better?!!



Kian Katanforoosh, Andrew Ng

Neural Machine Translation with Attention: Problems & Ideas

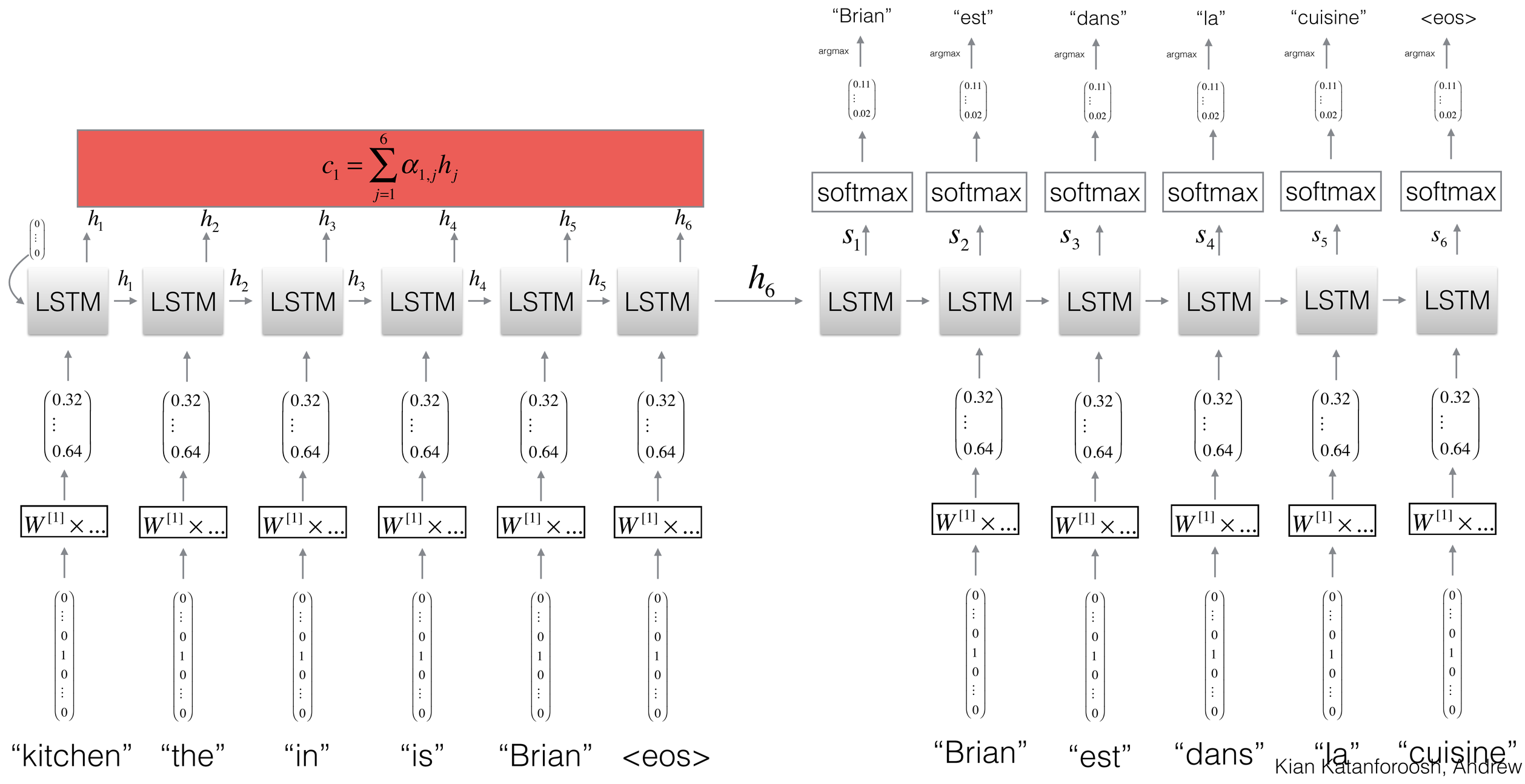
Some problems:

- The encoder encodes all information in the source sentence into a fixed length vector
- While it seems that some specific parts of the source sentence are more useful to predict some parts of the output sentence
- Bad performances on long sentences

Ideas:

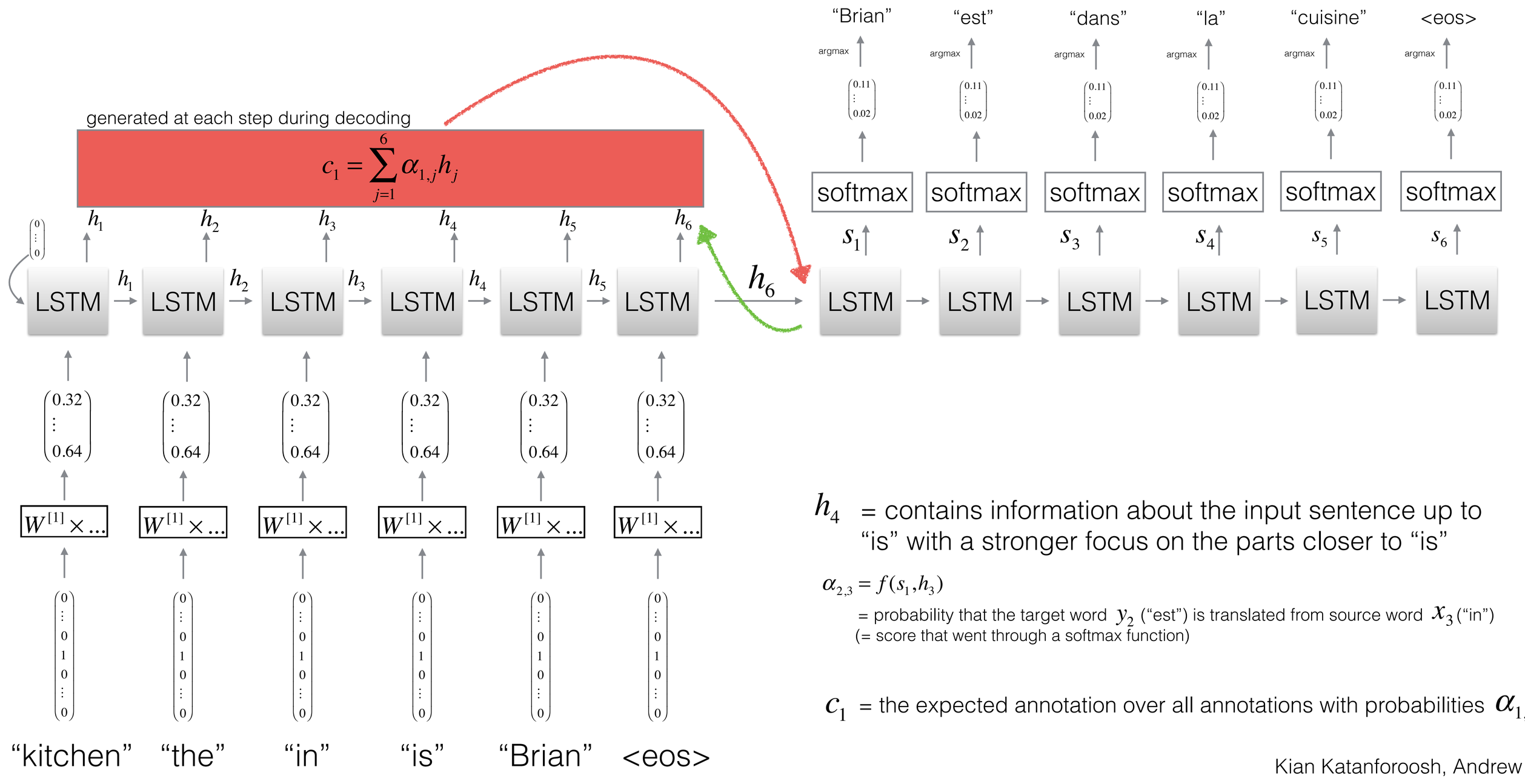
- We'd like to spread the information encoded from the source sentence and selectively retrieve the relevant parts at each prediction of the output sentence
- Why don't we use every hidden state hj from the encoding part?

Attention: Motivation



Kian Katanforoosh, Andrew Ng

Attention: Motivation



h_4 = contains information about the input sentence up to "is" with a stronger focus on the parts closer to "is"

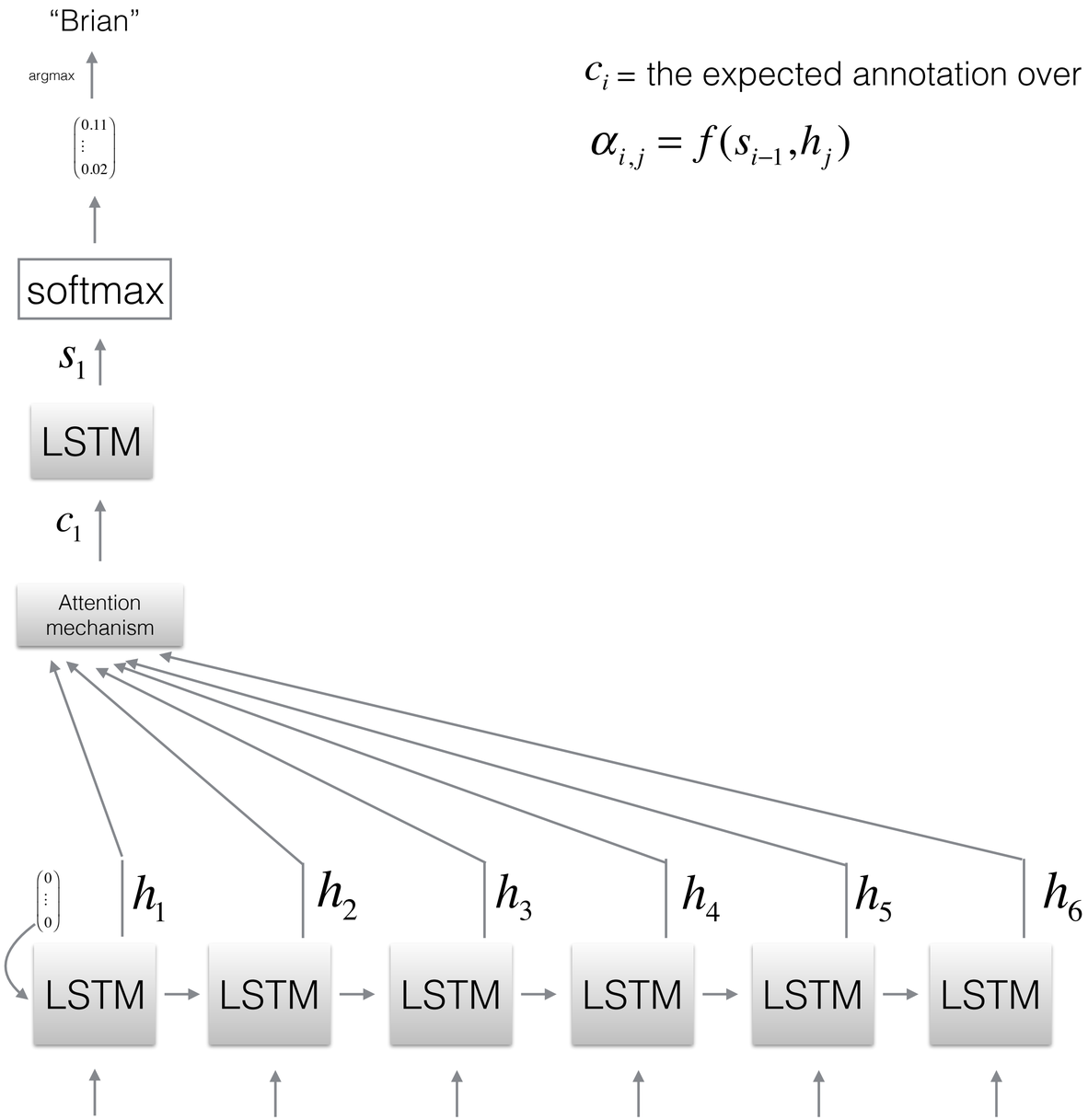
$\alpha_{2,3} = f(s_1, h_3)$

= probability that the target word y_2 ("est") is translated from source word x_3 ("in")
(= score that went through a softmax function)

c_1 = the expected annotation over all annotations with probabilities $\alpha_{1,j}$

Kian Katanforoosh, Andrew Ng

Neural Machine Translation with Attention: Architecture

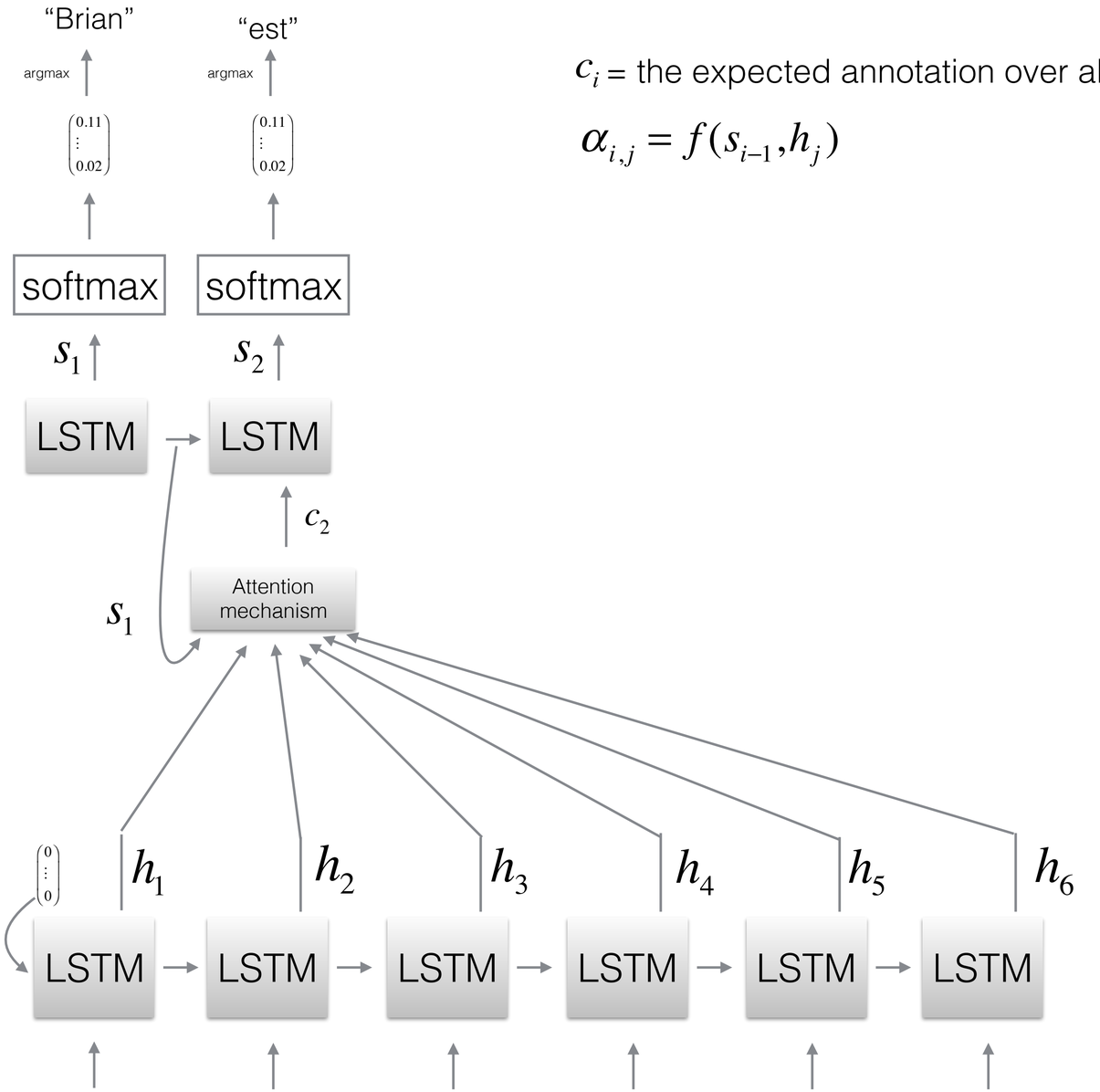


c_i = the expected annotation over all annotations with probabilities $\alpha_{i,j}$

$$\alpha_{i,j} = f(s_{i-1}, h_j)$$

Kian Katanforoosh, Andrew Ng

Neural Machine Translation with Attention: Architecture

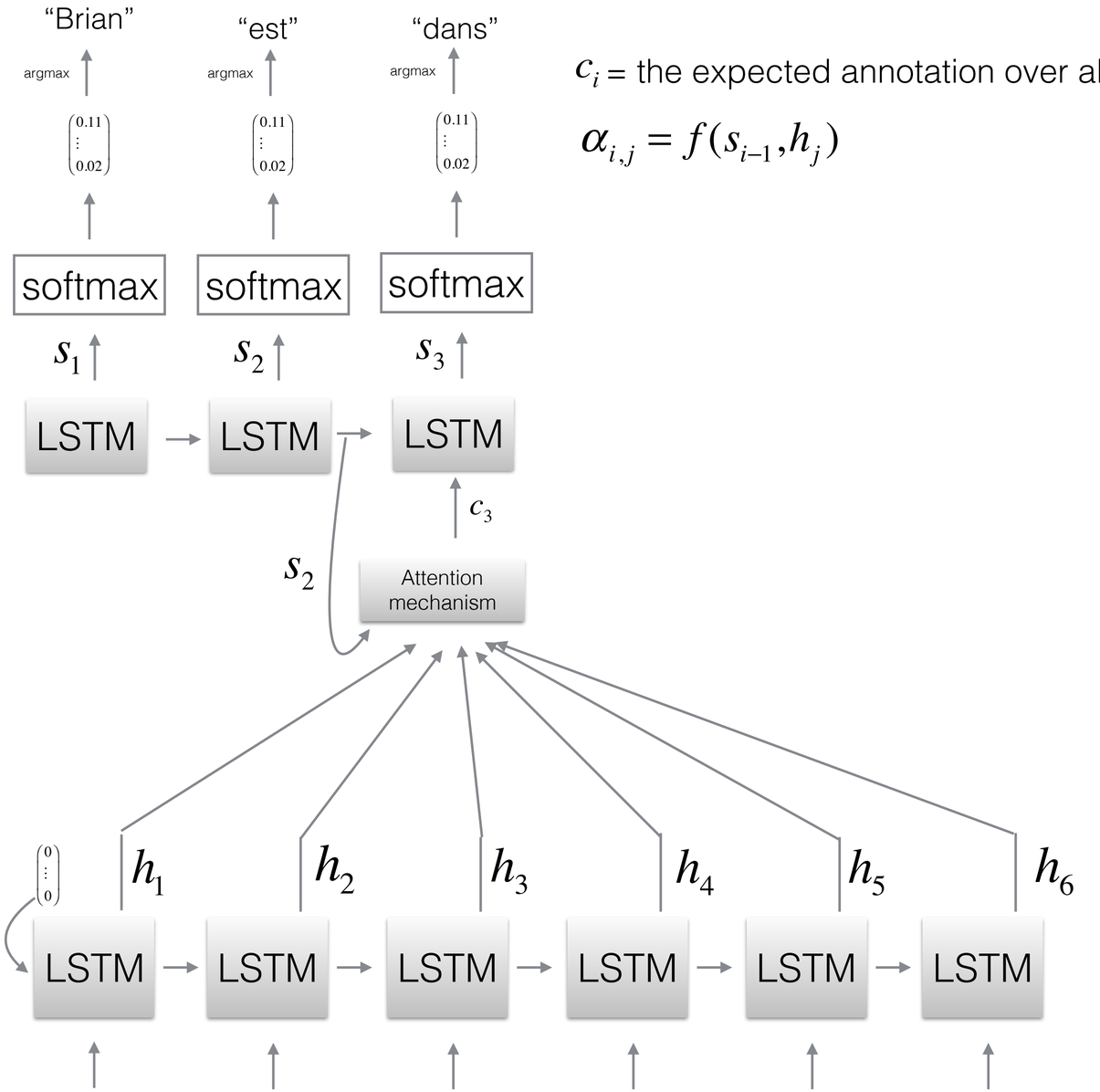


c_i = the expected annotation over all annotations with probabilities $\alpha_{i,j}$

$$\alpha_{i,j} = f(s_{i-1}, h_j)$$

. . . .

Neural Machine Translation with Attention: Architecture



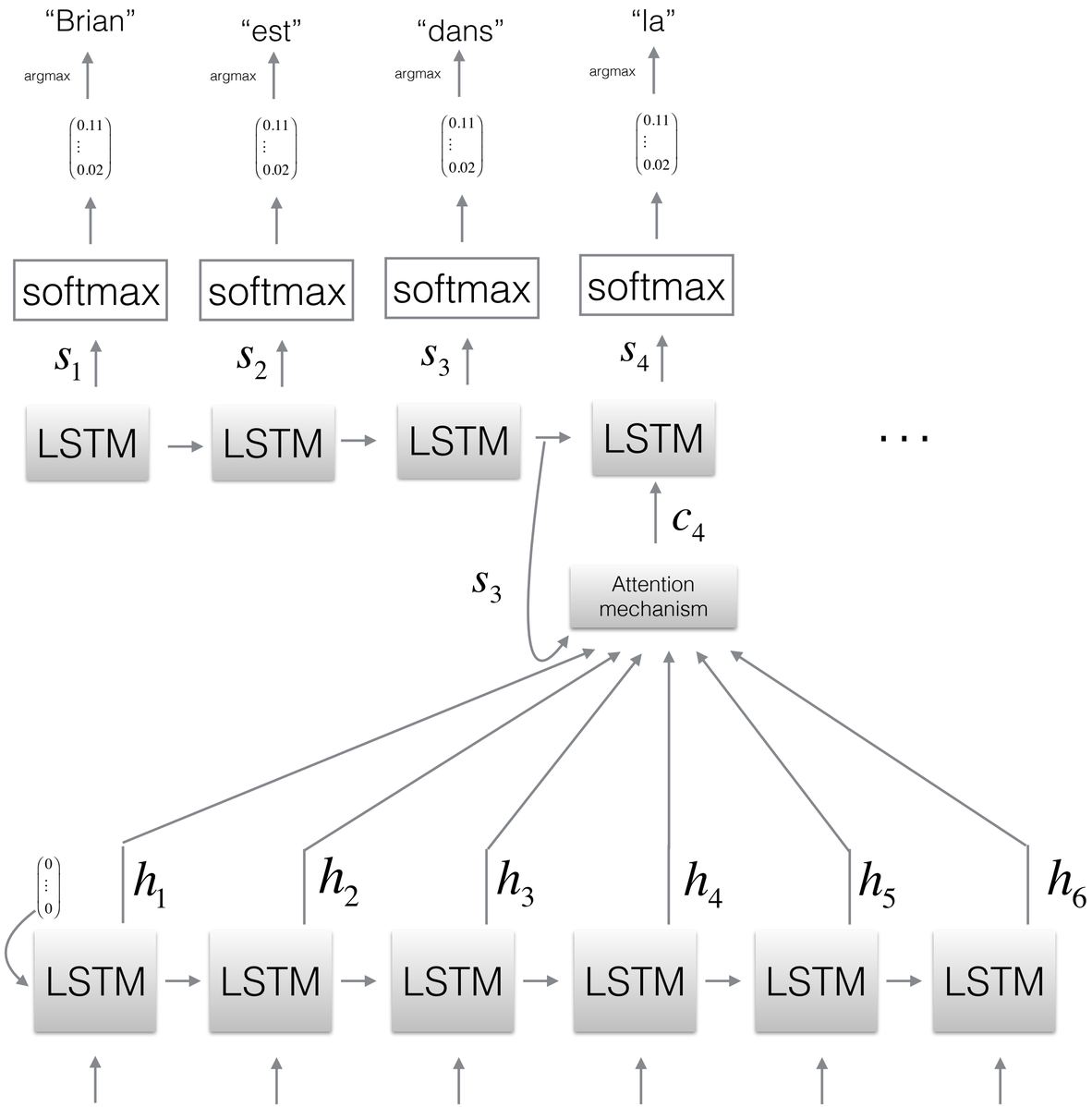
c_i = the expected annotation over all annotations with probabilities $\alpha_{i,j}$

$$\alpha_{i,j} = f(s_{i-1}, h_j)$$

. . . .

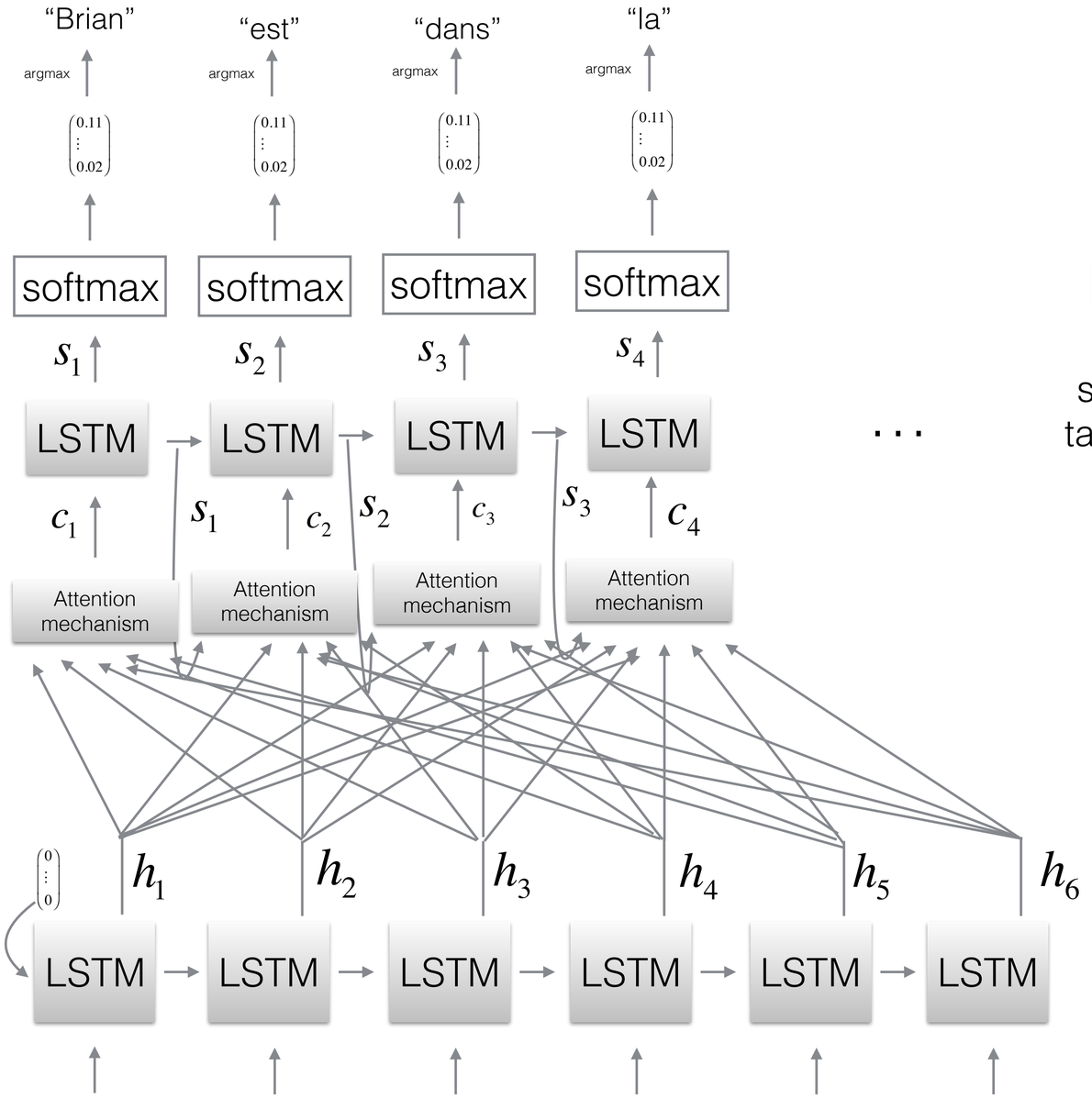
Kian Katanforoosh, Andrew Ng

Neural Machine Translation with Attention: Architecture



Kian Katanforoosh, Andrew Ng

Neural Machine Translation with Attention: Architecture



How to train this?

same as Machine Translation but takes also derivatives with respect to attention parameters

Kian Katanforoosh, Andrew Ng

Neural Machine Translation with Attention: Training

What are the parameters?

Encoder

$$W^{[1]} = EmbeddingMatrix$$

LSMT:

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f)$$
$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i)$$
$$\tilde{C}_t = \tanh(W_C[h_{t-1}, x_t] + b_C)$$
$$C_t = f_t \circ C_{t-1} + i_t \circ \tilde{C}_t$$
$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o)$$
$$h_t = o_t \circ \tanh(C_t)$$

Attention

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j$$

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}$$

$$e_{ij} = v_a^T \tanh(W_a s_{i-1} + U_a h_j)$$

Decoder

LSMT:

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f)$$
$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i)$$
$$\tilde{C}_t = \tanh(W_C[h_{t-1}, x_t] + b_C)$$
$$C_t = f_t \circ C_{t-1} + i_t \circ \tilde{C}_t$$
$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o)$$
$$h_t = o_t \circ \tanh(C_t)$$

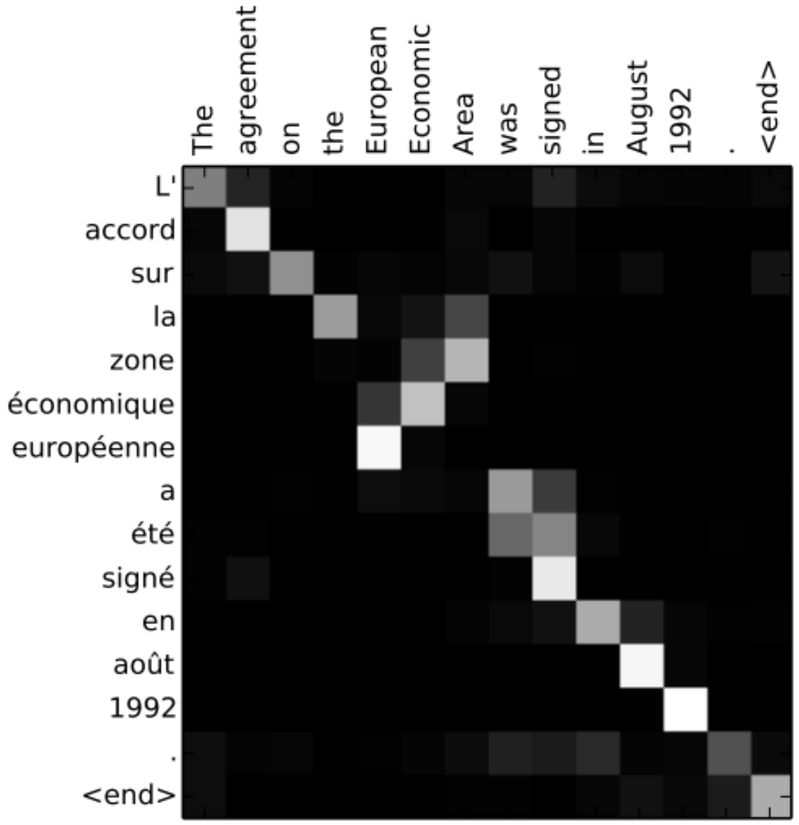
Loss function:

$$L = -\sum_{batch} \left(\log P(y|x;\theta) + \lambda \sum_{i \in patch} (1 - \sum_{t \in output} \alpha_{ti}) \right)$$

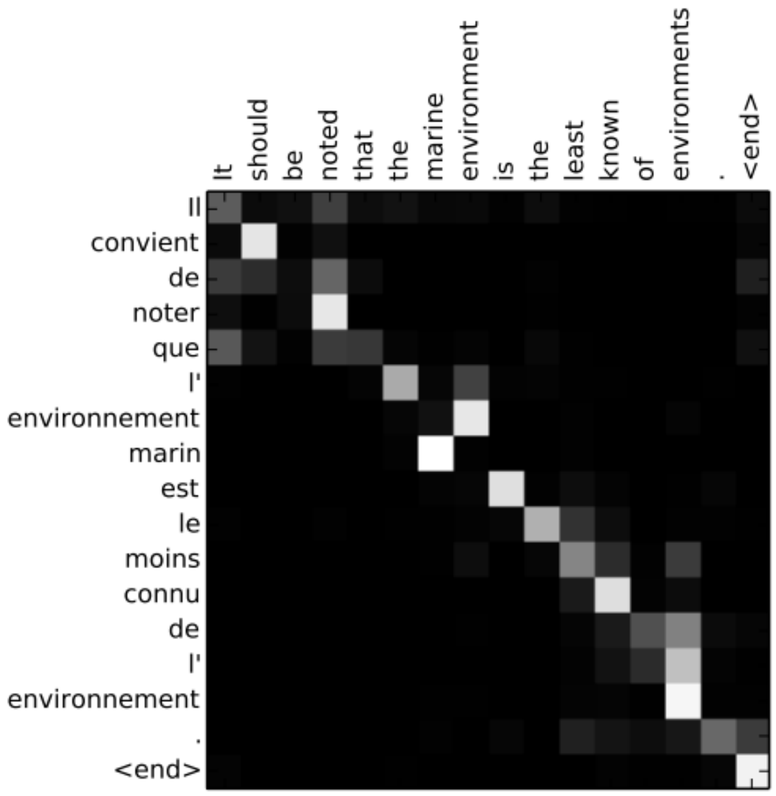
Kian Katanforoosh, Andrew Ng

Neural Machine Translation with Attention: Training

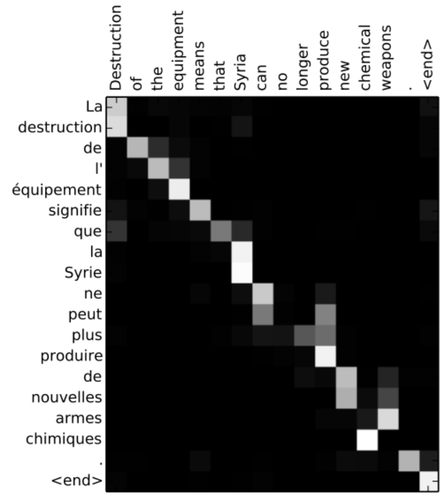
Visualizing Attention



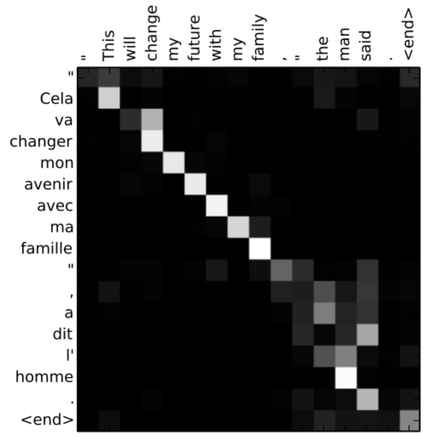
(a)



(b)



(c)



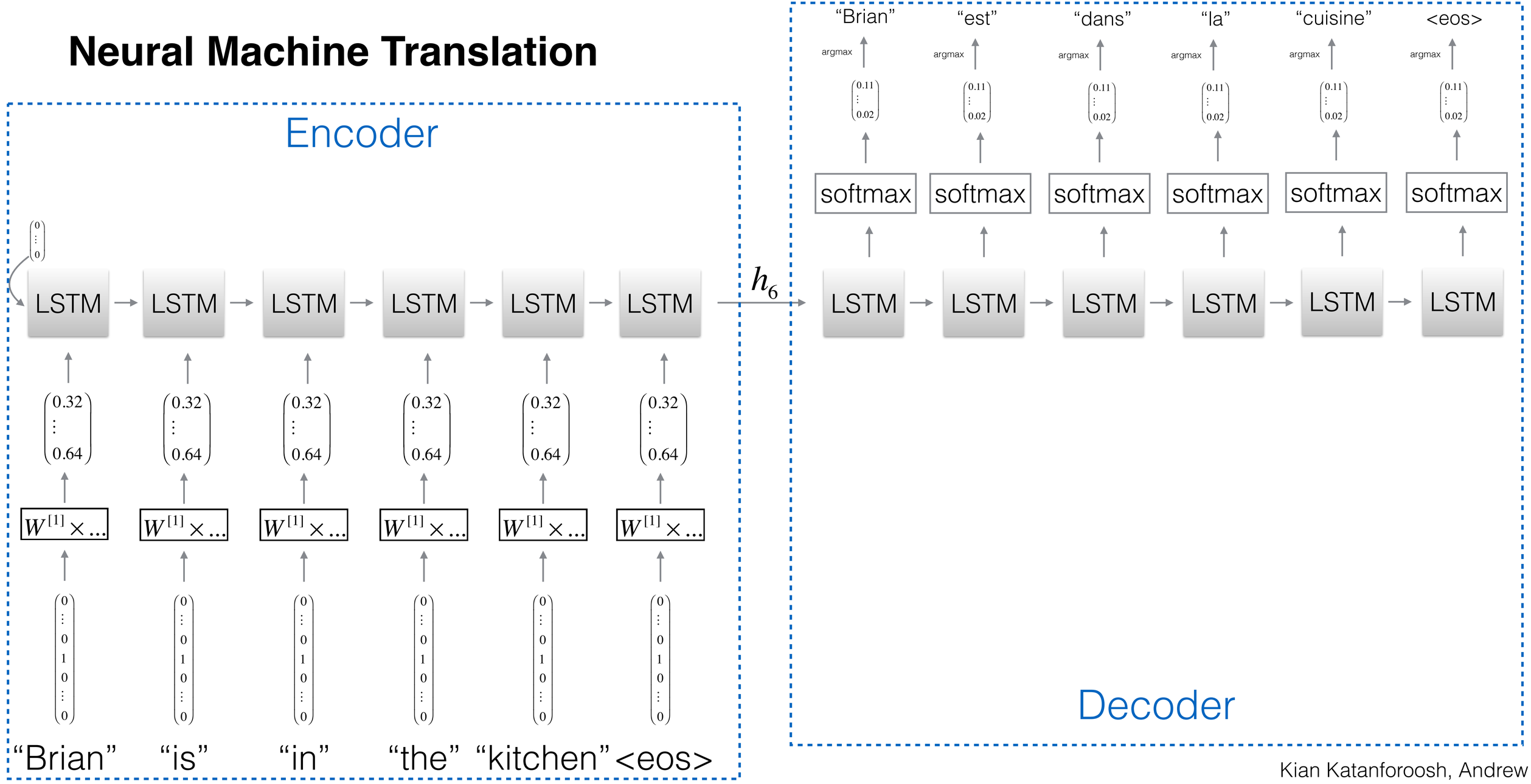
(d)

Figure 3: Four sample alignments found by RNNsearch-50. The x-axis and y-axis of each plot correspond to the words in the source sentence (English) and the generated translation (French), respectively. Each pixel shows the weight α_{ij} of the annotation of the j-th source word for the i-th target word (see Eq. (6)), in grayscale (0: black, 1: white). (a) an arbitrary sentence. (b–d) three randomly selected samples among the sentences without any unknown words and of length between 10 and 20 words from the test set.

Dzmitry Bahdanau, Kyunghyun Cho, Yoshua Bengio : Neural Machine Translation by Jointly Learning to Align and Translate

Image Captioning with Attention

Neural Machine Translation



Kian Katanforoosh, Andrew Ng

Image Captioning with Attention

Image Captioning

Encoder

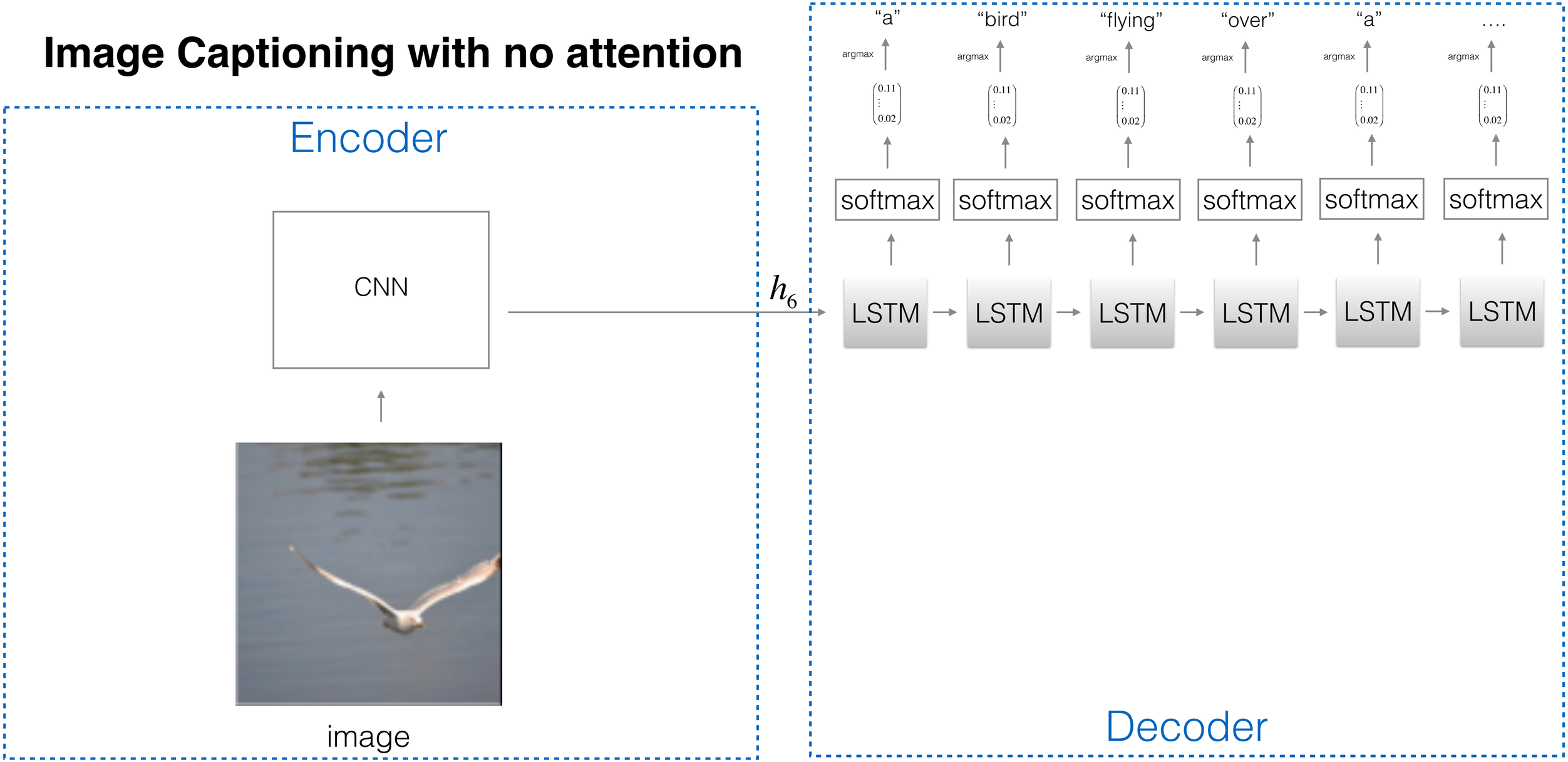
image

Decoder

Kian Katanforoosh, Andrew Ng

Image Captioning with Attention

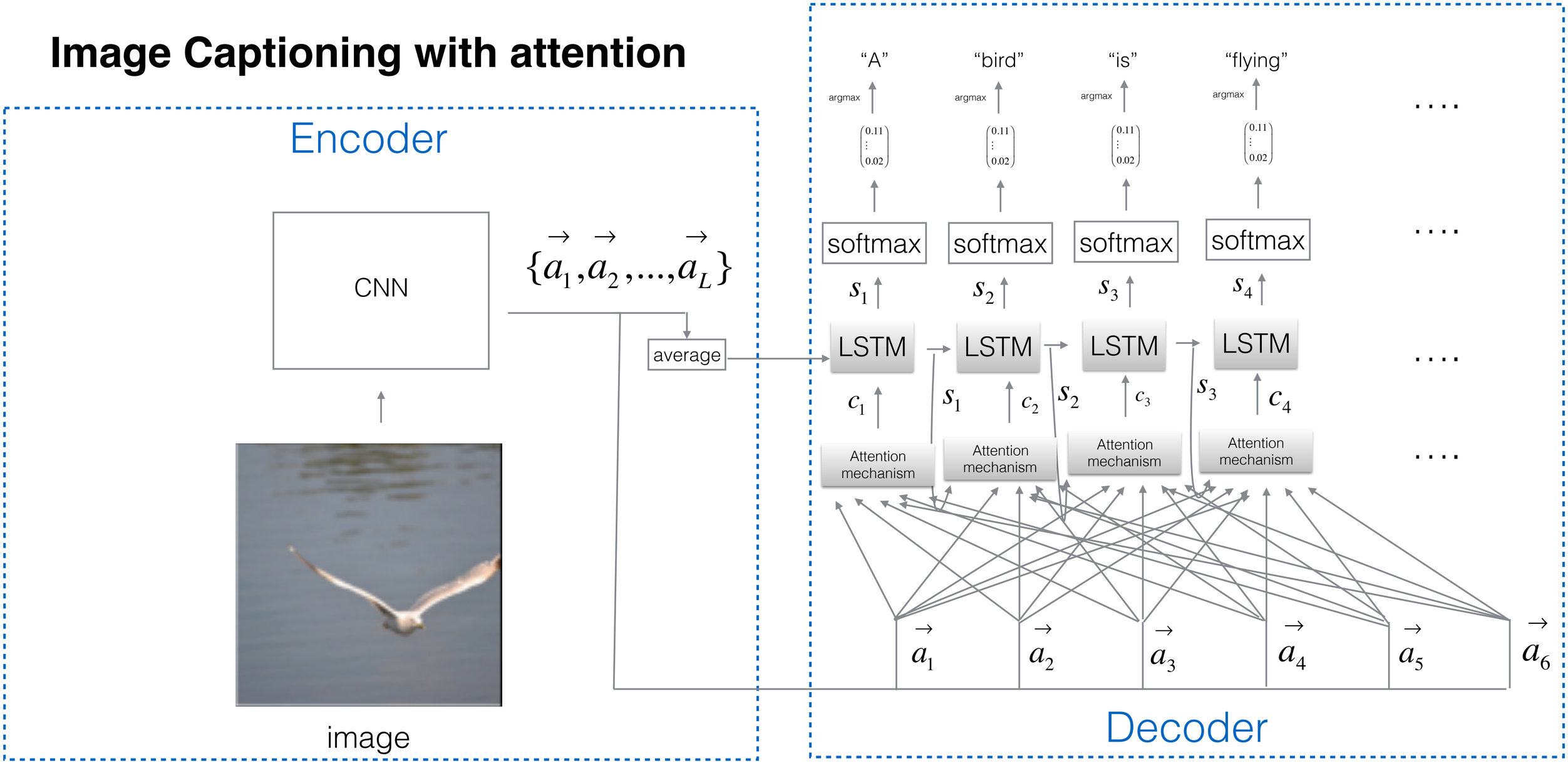
Image Captioning with no attention



Vinyals et al. : Show and Tell: A Neural Image Caption Generator

Image Captioning with Attention

Image Captioning with attention



Vinyals et al. : Show and Tell: A Neural Image Caption Generator
Kelvin Xu et al. : Show, Attend and Tell: Neural Image Caption Generation with Visual Attention