Copyright Notice

deeplearning.ai

Sequence to sequence models

Basic models

Sequence to sequence model

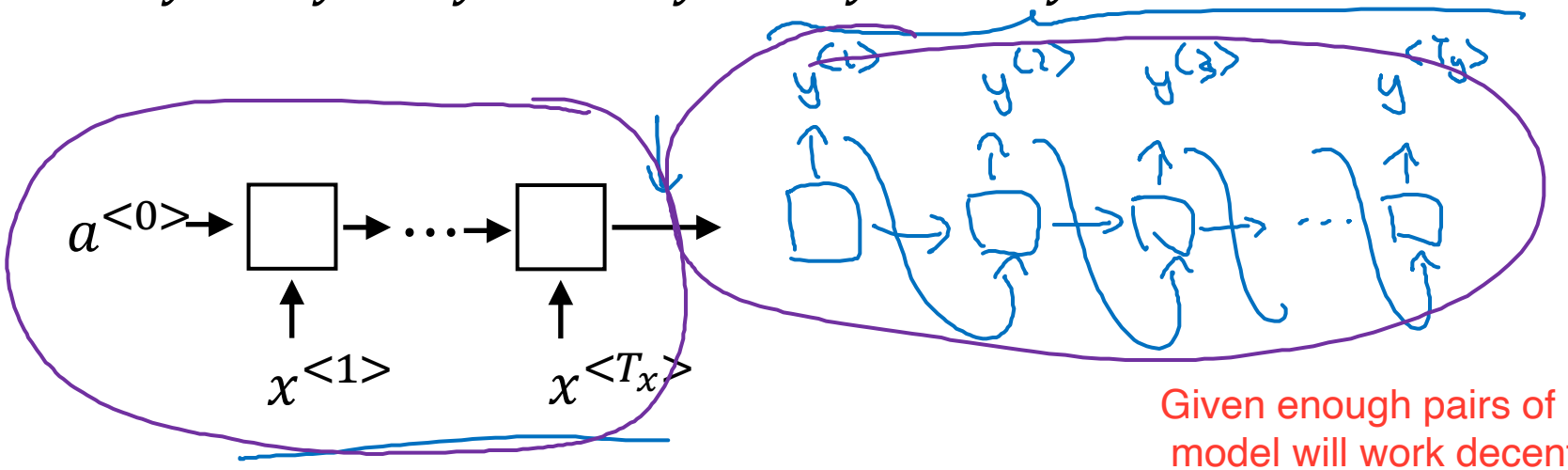
$x^{<1>}$ $x^{<2>}$ $x^{<3>}$ $x^{<4>}$ $x^{<5>}$

Jane visite l'Afrique en septembre

Encoder network is built as a RNN and this could be a Gru or LSTM feeding the input, French words, one word at a time. After ingesting the input sequence the RNN then outputs a vector that represents the input sequence.

→ Jane is visiting Africa in September.

$y^{<1>}$ $y^{<2>}$ $y^{<3>}$ $y^{<4>}$ $y^{<5>}$ $y^{<6>}$



Decoded network takes as input the encoding output by the encoding network shown in blank on the left, and then can be trained to output the translation one word.

Given enough pairs of French and English sentences, this model will work decently well. This model simply uses an encoding network whose job is to find an encoding of the input French sentence, and then use a decoding network in order to generate the corresponding English translation.

This architecture is very similar to image cationing.

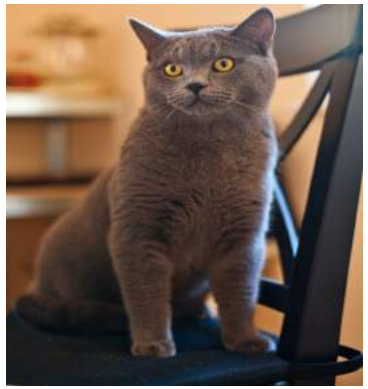
[Sutskever et al., 2014. Sequence to sequence learning with neural networks]

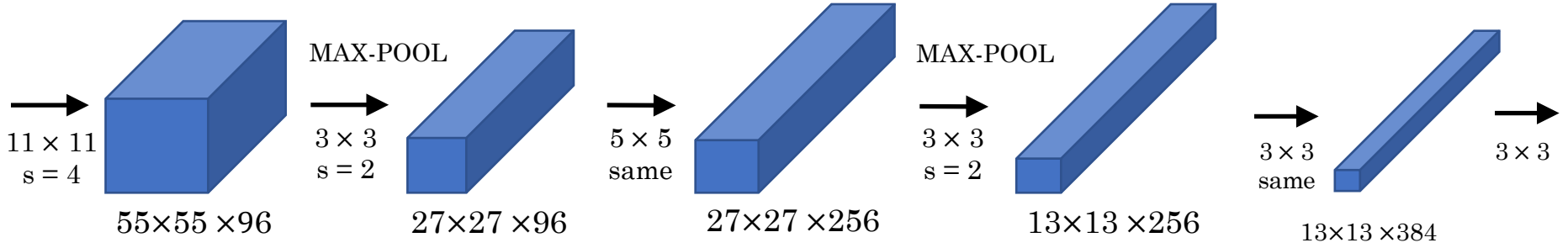
[Cho et al., 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation]

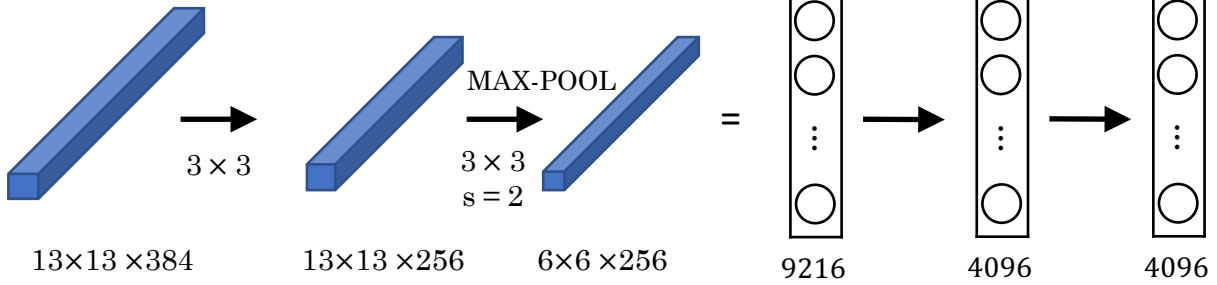
Andrew Ng

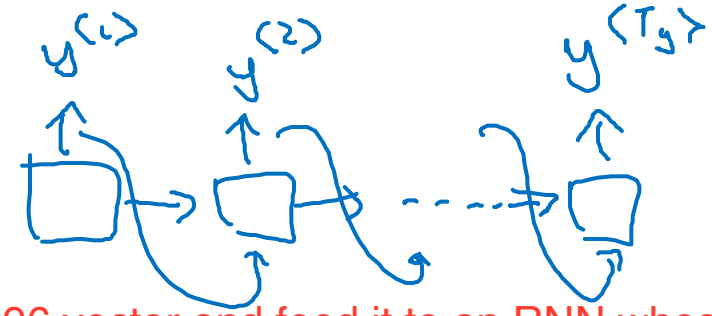
Image captioning

$y^{<1>}$ $y^{<2>}$ $y^{<3>}$ $y^{<4>}$ $y^{<5>}$ $y^{<6>}$
A cat sitting on a chair



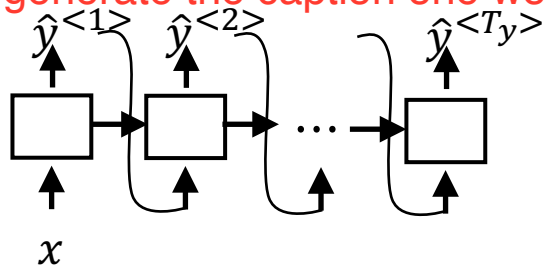






Take this 4,096 vector and feed it to an RNN whose job is to generate the caption one word at a time.

You can input an image into a convolutional network, a pre-trained AlexNEt, and have the network learn and encoding a learner to the features of the input image—a 4,096 dimensional vector that represents the image in this example.



However, there are some differences between how to run/ generate a sequence and how to synthesize novel text using a language model.

[Mao et. al., 2014. Deep captioning with multimodal recurrent neural networks]
[Vinyals et. al., 2014. Show and tell: Neural image caption generator]
[Karpathy and Li, 2015. Deep visual-semantic alignments for generating image descriptions]

deeplearning.ai

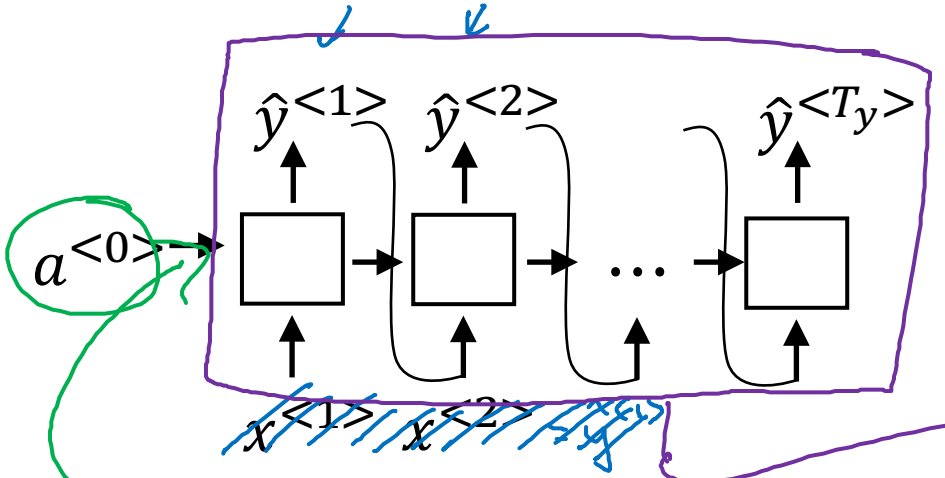
Sequence to sequence models

There are some similarity between the sequence to sequence machine translation model and the language models.
But there are some significant differences as well.

Picking the most likely sentence

Machine translation as building a conditional language model

Language model:

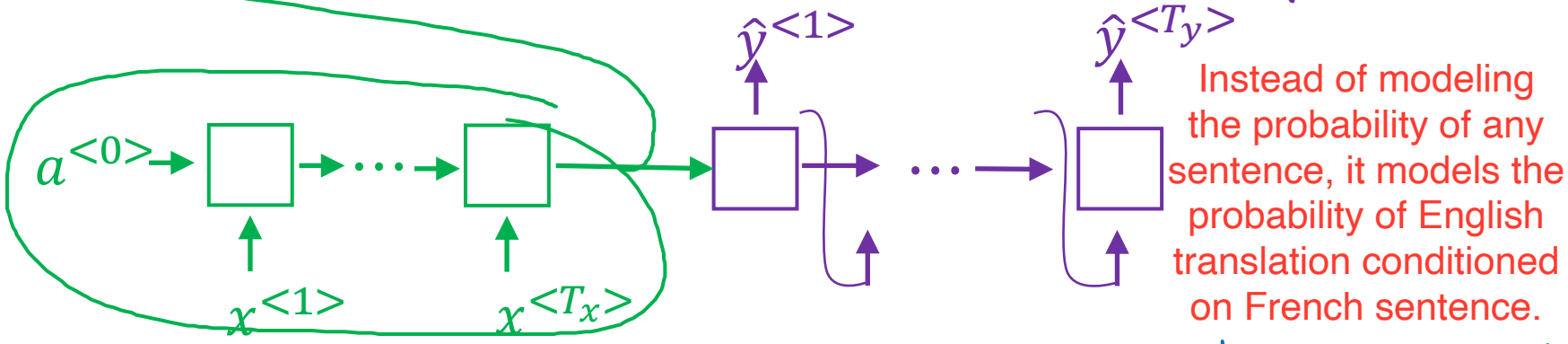


This language model allows you to estimate the probability of a sentence

$P(y^{<1>}, \ldots, y^{<T_y>})$

Instead of always starting along with the vector of all zeros, the machine translation model has an encoded network that figures out some representation for the input sentence.

Machine translation:



Instead of modeling the probability of any sentence, it models the probability of English translation conditioned on French sentence.

Machine translation model takes the input sentence and starts off the decoded network with the representation of the input sentence rather than the representation of all zeros..

"Conditional language model"

$P(y^{<1>}, \ldots, y^{<T_y>} \mid x^{<1>}, \ldots, x^{<T_x>})$

Andrew Ng

Finding the most likely translation

Jane visite l'Afrique en septembre.

$$P(y^{<1>}, \ldots, y^{<T_y>} \mid x)$$

English

French

Given the input—French sentence, the model might tell you what the prob. of difference in corresponding English translation.

→ Jane is visiting Africa in September.

→ Jane is going to be visiting Africa in September.

→ In September, Jane will visit Africa.

→ Her African friend welcomed Jane in September.

You don't want to sample output at random. If you sampled words from the distribution (p of y given x) at random, then you may get a different translation, e.g., the 2nd ~ 4th sentences

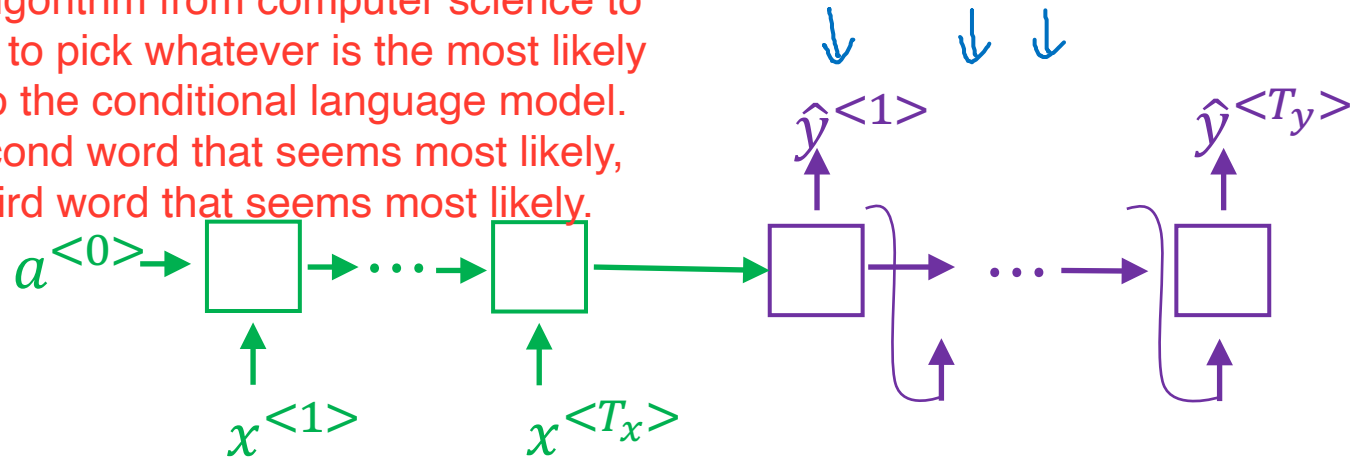
So, instead of trying to sample at random from the distribution, you would like to find the english sentence, y, that maximize the conditional probability.

$$\arg\max_{y^{<1>},\ldots,y^{<T_y>}} P(y^{<1>}, \ldots, y^{<T_y>} \mid x)$$

The most common algorithm for this is "beam search"

Why not a greedy search?

Greedy search is an algorithm from computer science to generate the first word to pick whatever is the most likely first word according to the conditional language model. And then pick the second word that seems most likely, and then pick the third word that seems most likely.



$p(\hat{y}^{<1>} | x)$

10,000 words in a dictionary and 10 word-long translation, then there are a huge space of possible sentences, which is impossible to rate them all. So, people commonly use an approximate search algorithm.

10,000

10

$10{,}000^{10}$

$P(y|x)$

$\arg\max_y P(\hat{y}^{<1>}, \hat{y}^{<2>}, \ldots, \hat{y}^{<T_y>} | x)$

What you really like is to pick the entire sequence of words, y1 through yTy, and maximizes the joint probability of the whole thing.

→ Jane is visiting Africa in September.

→ Jane is going to be visiting Africa in September.

However, it is not always optimal to just pick one word at a time. For example, given "Jane is", "going" can be more common than "visiting".

$P(\text{Jane is going} | x) > P(\text{Jane is visiting} | x)$

Andrew Ng

deeplearning.ai

Sequence to sequence models

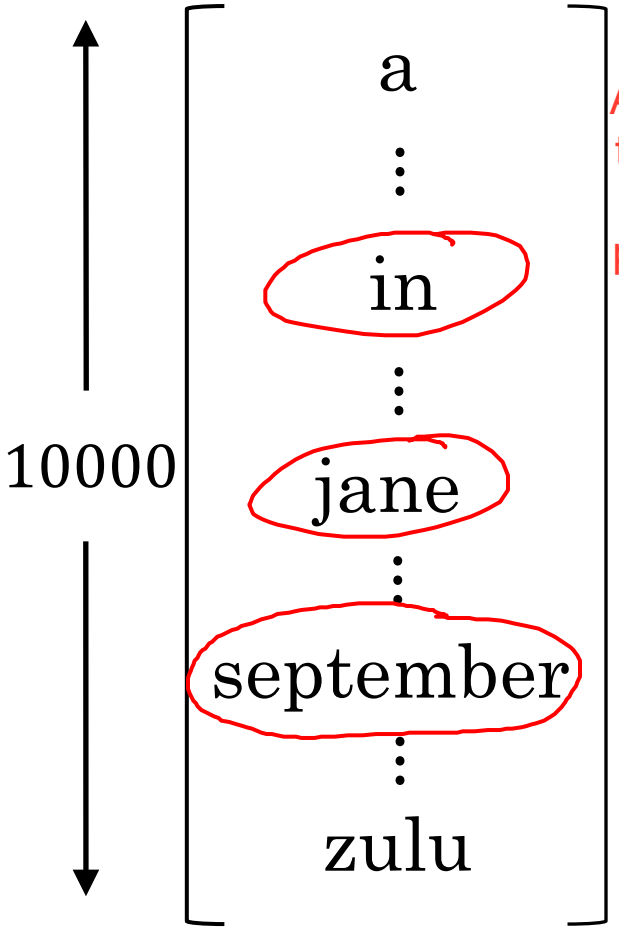
You want to output the best and the most likely Enlish translation rather than a random English translation

Beam search

Beam search algorithm

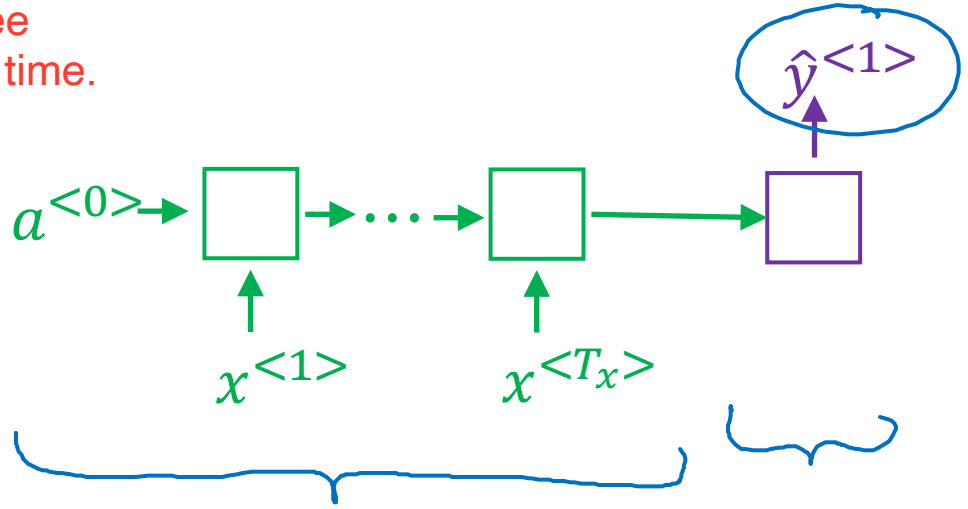
B = 3 (beam width)

Step 1: to pick the first words of the English translation by considering multiple alternatives.



A parameter B means that beam search will consider three possibilities at a time.

Evaluating the possibility over different choices for the first words → $P(y^{<1>} \mid x)$



After running the input French sentence through the encoder network, the first step decodes the network (e.g., a softmax output — overall 10,000 possibilities). Then you would take the 10,000 possible outputs and keep in memory which were the top three.

When B=3, you instantiate 3 copies of the network.

Beam search algorithm

$(B = 3)$

Step 1

Step 2

For each of these three choices, you consider what should be the second word.

To evaluate the probability of the second word, $P(y^{<2>} | x, \text{"in"})$

you use the new network fragments, remembering the decoder first output

$P(y^{<1>}, y^{<2>} | x) = P(y^{<1>} | x) \, P(y^{<2>} | x, y^{<1>})$

$P(y^{<2>} | x, \text{"jane"})$

Of 30,000 possibilities, 3 words would be memorized.

10000

a

in

jane

september

zulu

it is now rejecting 'september' as a candidate for the first word.

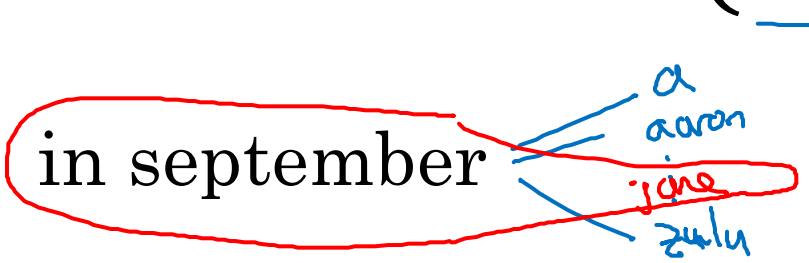
10,000

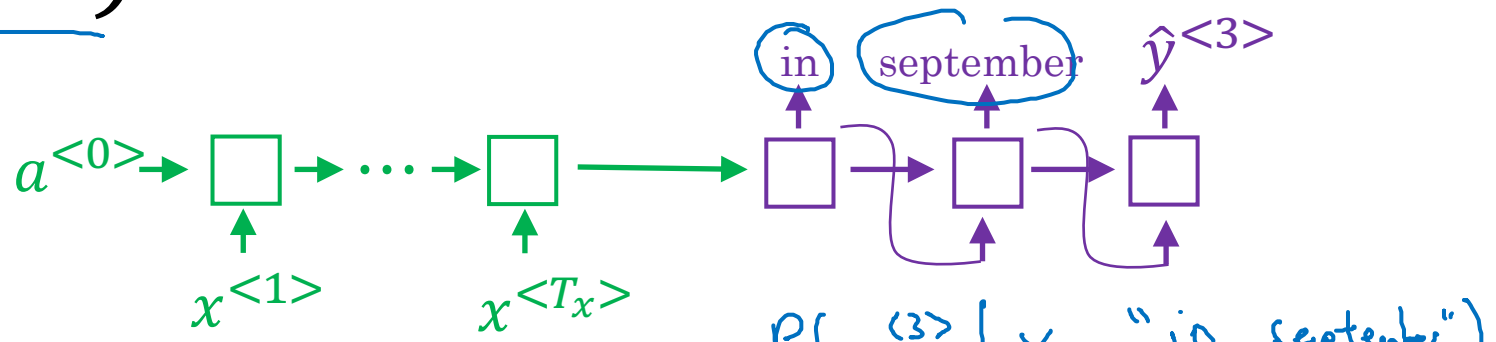
If beam search decides that the most likely choices for the first and second words like here $y^{<1>}, y^{<2>}$

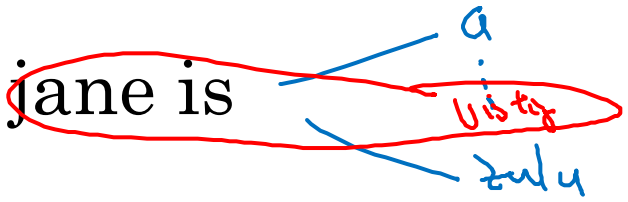
Andrew Ng

Beam search ($B = 3$)

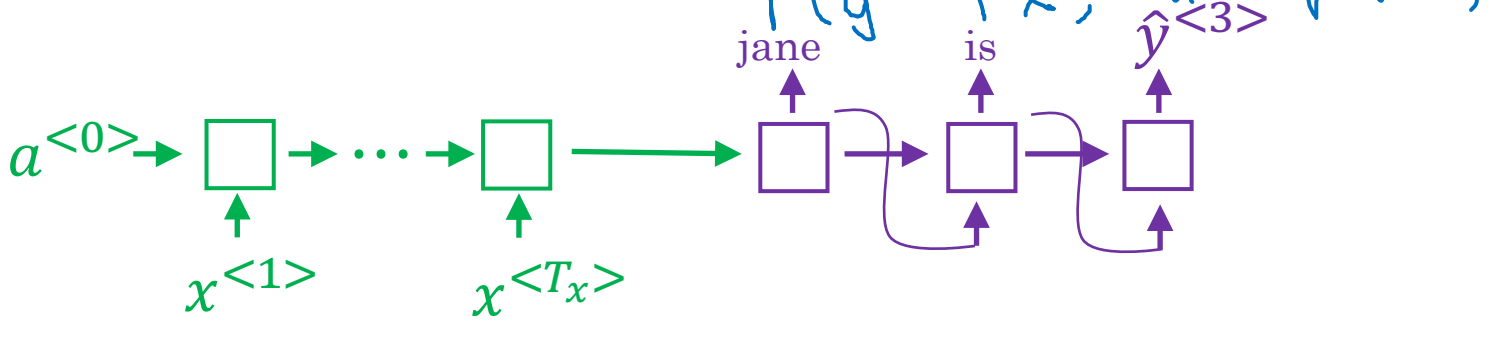
$B = 1$ ⇝ greedy search

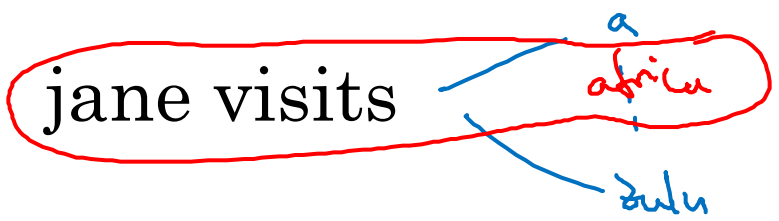


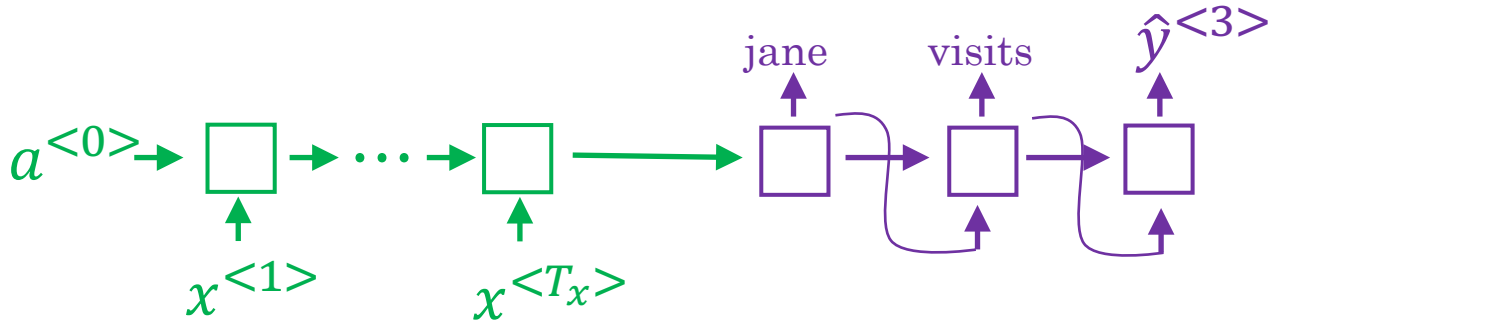




$P(y^{<3>} | x, \text{"in september"})$







$P(y^{<1>}, y^{<2>} | x)$

jane visits africa in september. <EOS>

Andrew Ng

deeplearning.ai

Sequence to sequence models

Refinements to beam search

Length normalization

$$\arg\max_{y} \prod_{t=1}^{T_y} P(y^{<t>} | x, y^{<1>}, \ldots, y^{<t-1>})$$

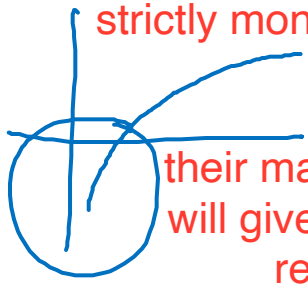
product of probabilities

$P(y^{<1>} \ldots y^{<T_y>} | x) = P(y^{<1>} | x)\, P(y^{<2>} | x, y^{<1>}) \cdots P(y^{<T_y>} | x, y^{<1>}, \ldots, y^{<T_y-1>})$

log

These probabilities are all numbers less than one, and multiplying such a tiny number can result in numerical underfloor (i.e., too small for the floating point of representation in a computer to store accurately).

Because logarithmic function is a strictly monotonically increasing function.



$\log P(y|x)$ ←

their maximization will give the same results

$P(y|x)$ ←

$$\arg\max_{y} \sum_{t=1}^{T_y} \log P(y^{<t>} | x, y^{<1>}, \ldots, y^{<t-1>})$$

By taking logs, you end up with a more numerically stable algorithm that is less prone to numerical rounding errors or unmerical underfloor.

If you have a very long sentence, the prob. of the sentence will be low ecause of multiplying many terms.

This obj. fn. may unnaturally tend to prefer very short translation

$T_y = 1, 2, 3, \ldots, 30.$

$$\frac{1}{T_y^{\alpha}} \sum_{t=1}^{T_y} \log P(y^{<t>} | x, y^{<1>}, \ldots, y^{<t-1>})$$

Normalize this by the number of words in the translation in order to reduce the penalty for putting longer translations.

If alpha is one, it is full normalization.
If alpha is zero, it is no normalization.

$\alpha = 0.7$ $\alpha = 1$ $\alpha = 0$

Andrew Ng

Beam search discussion

Pros and cons of setting beam width to be very large vs. very small.

large B: better result, slower

small B: worse result, faster

How to choose the beam width?

Beam width B?

1 -> 3 -> 10, 100, 1000, -> 3000

If B is very large, you consider a lot of possibilities, and you tend to get a better results because you are consuming a lot of different options.

Unlike exact search algorithms like BFS (Breadth First Search) or DFS (Depth First Search), Beam Search runs faster but is not guaranteed to find exact maximum for $\arg\max_y P(y|x)$.

deeplearning.ai

Sequence to sequence models

Beam search: Approximate search alogorithm, also called a heuristic search algorithm

---

Error analysis can help you focus your time on doing the most useful work for your project.

Error analysis on beam search

What if beam search makes a mistake? Learn how error analysis interacts with beam search and how you can figure out whether it is the beam search algorithm that is causing problems.

Example

→ RNN

→ Beam Search

BT

Jane visite l'Afrique en septembre.

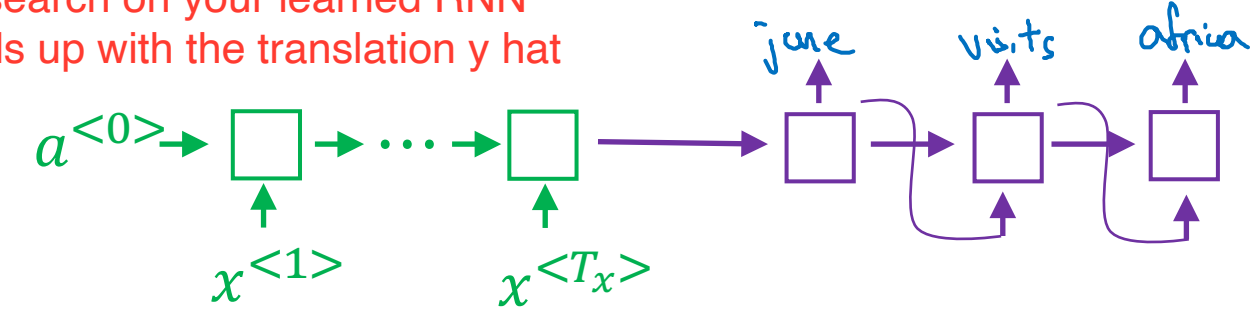
Human: Jane visits Africa in September. (y^*)

In your machine translation development set, the human provided the translation and call this y star.

Algorithm: Jane visited Africa last September. $(\hat{y})$ ←

RNN computes $P(y^*|x) < \leq P(\hat{y}|x)$

You run beam search on your learned RNN model, which ends up with the translation y hat



Your model has two components: 1) RNN model: a neural network model, the sequence to sequence model, which means an encoder and a decoder; 2) Beam search algorithm.

It is always tempting to collect more training data that never hurts. In similar way, it is always tempting to increase the beam width that never hurts. But just as getting more training data set might not get you to the level of performance you want. Likewise, increasing the beam width might not get you where you want to go.

Andrew Ng

Error analysis on beam search

$P(y^*|x)$

$P(\hat{y}|x)$

Human: Jane visits Africa in September. (y^*)

Algorithm: Jane visited Africa last September. ($\hat{y}$)

Case 1: $P(y^*|x) > P(\hat{y}|x)$

P(y star given x) as output by the RNN model is greater than P(y hat given x)

$\arg\max_y P(y|x)$

Beam search chose $\hat{y}$. But y^* attains higher $P(y|x)$.

Since beam search algorithm chose y hat, the way you got y hat was you had an RNN that computes P(y given x)

Conclusion: Beam search is at fault.

Beam search fails to actually give you the value of y that maximizes P(y given x). It chose y hat that has a smaller value.

Then, you may need to increase the beam width.

Case 2: $P(y^*|x) \leq P(\hat{y}|x)$

y^* is a better translation than $\hat{y}$. But RNN predicted $P(y^*|x) < P(\hat{y}|x)$.

Even though y star is better translation in this example, the RNN says P(y star) is less than P(y hat).

saying that y star is less likely the output than y hat.

Conclusion: RNN model is at fault.

Then, you could do a deeper layer of analysis to try to figure out if you want to add regularization, or get more training data, or try a different network architecture, or etc.

Andrew Ng

Error analysis process

Human	Algorithm	$P(y^*\|x)$	$P(\hat{y}\|x)$	At fault?
Jane visits Africa in September.	Jane visited Africa last September.	2×10^{-10}	1×10^{-10}	B
				R
				B
				R
				R
				⋮

Figures out what faction of errors are “due to” beam search vs. RNN model

Andrew Ng

deeplearning.ai

Sequence to sequence models

Bleu score (optional)

BLEU stands for Bilingual Evaluation Understudy

Evaluating machine translation

French: Le chat est sur le tapis.

Reference 1: The cat is on the mat.

Human 1 & 2 translation

Reference 2: There is a cat on the mat.

MT output: the the the the the the the.

Bleu

bilingual evaluation understudy

BLEU score: given Machine generated translation, it allows you to automatically compute a score that measures how god is that machine translation. The intuition is so long as the Machine generated translation is pretty close to any of references by humans, then it will get high BLEU score

BLEU score could be a substitute for having humans evaluate every output of a machine translation system.

Precision: 7 over 7

One way to measur how good the MT output is to look at each the words in the output and see if it appears in the reference: it is called a precision of the MT output.

Modified precision: 2 over 7

A modified precision measure gives each word credit only up to the maximum number of times it appears in the reference sentence. In reference 1, 'the' appears two, In reference 2, 'the' appears one. So, 'the' gets credit up to twice.

However, in the BLEU score, you don't want to just look at isolated words but at pairs of words as well.

[Papineni et. al., 2002. Bleu: A method for automatic evaluation of machine translation]

Bleu score on bigrams

Bigrams: Pairs of words appearing next to each other

Example: Reference 1: The cat is on the mat.

Reference 2: There is a cat on the mat.

MT output: The cat the cat on the mat.

Possible bigrams

Possible bigrams	Count	Count_{clip}
the cat	2	1
cat the	1	0
cat on	1	1
on the	1	1
the mat	1	1

Maximum from references

Modified precision on bigrams.

$$\frac{4}{6}$$

[Papineni et. al., 2002. Bleu: A method for automatic evaluation of machine translation]

Bleu score on unigrams

If the MT output is exactly the same as either Reference 1 or 2, then all of these values P1 and P2 and so on, they will all be equal to 1.

Example: Reference 1: The cat is on the mat.

Reference 2: There is a cat on the mat.

→ MT output: The cat the cat on the mat. ($\hat{y}$)

$p_1, p_2, \ldots = 1.0$

To get a modified precision of 1, you have to have the output exactly be equal to one of the references.

$$p_1 = \frac{\sum_{unigram \in \hat{y}} count_{clip}(unigram)}{\sum_{unigram \in \hat{y}} count(unigram)}$$

$$p_n = \frac{\sum_{ngram \in \hat{y}} count_{clip}(ngram)}{\sum_{ngram \in \hat{y}} count(ngram)}$$

[Papineni et. al., 2002. Bleu: A method for automatic evaluation of machine translation]

Bleu details

p_n = Bleu score on n-grams only

p_1, p_2, p_3, p_4

Combined Bleu score: $\text{BP} \exp\left(\frac{1}{4}\sum_{n=1}^{4} p_n\right)$

BP = brevity penalty

$$\text{BP} = \begin{cases} 1 & \text{if MT\_output\_length} > \text{reference\_output\_length} \\ \exp(1 - \text{MT\_output\_length}/\text{reference\_output\_length}) & \text{otherwise} \end{cases}$$

BLEU score is a useful single real number evaluation metric to use whenever you want your algorithm to generate a piece of text and you want to see whether it has similar meaning as a reference piece of text generated by humans. But, this is not used for speech recognition, because there is usually one ground truth in speech recognition.

For image captionining, and multiple cations for a picture, it could be good.

[Papineni et. al., 2002. Bleu: A method for automatic evaluation of machine translation]

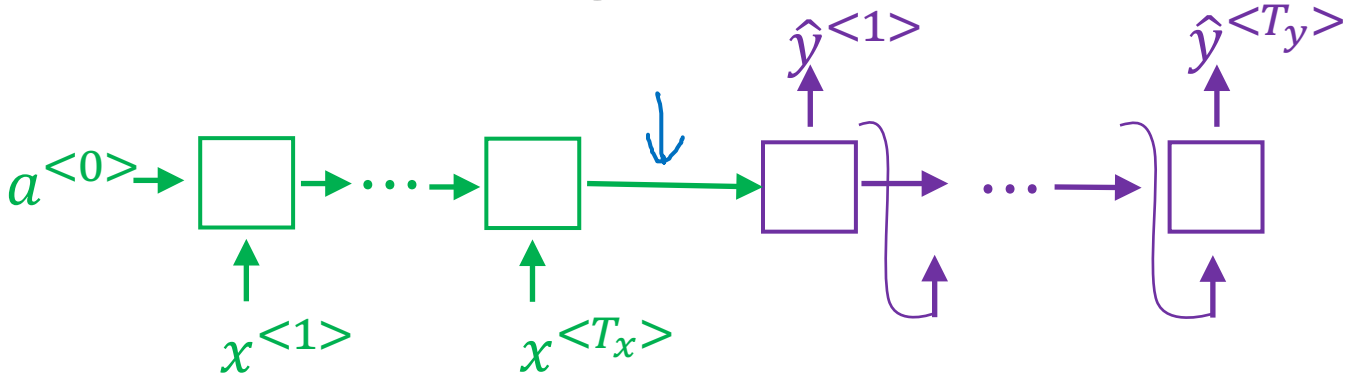
deeplearning.ai

Sequence to sequence models

Attention model: Modification to an Encoder-Decoder architecture for machine translation

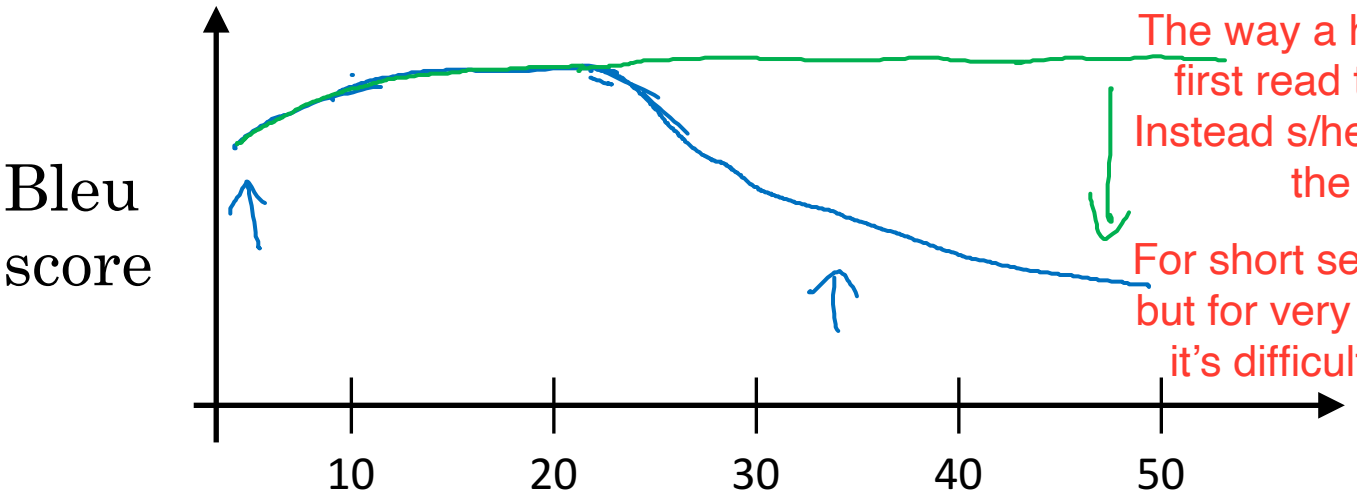
Attention model intuition

The problem of long sequences



Jane s'est rendue en Afrique en septembre dernier, a apprécié la culture et a rencontré beaucoup de gens merveilleux; elle est revenue en parlant comment son voyage était merveilleux, et elle me tente d'y aller aussi.

Jane went to Africa last September, and enjoyed the culture and met many wonderful people; she came back raving about how wonderful her trip was, and is tempting me to go too.



The way a human translator would translate this sentence is not to first read the whole sentence, memorizing the whole sentence. Instead s/he would read the first part of it and then generate part of the translation, and move on, working part by part.

For short sentences, it easily achieve a relatively high BLEU score, but for very long sentences, its performance comes down because it's difficult to get in your network to memorize a long sentence.

Andrew Ng

Like humans, Attention model looks at part of the sentence at a time.

Attention model intuition

Attention model would compute a set of attention weights, (alpha one, 1), (alpha one, 2), (alpha one, 3), which denotes when you are generating the first word, how much you should pay attention to the first piece of information.

S denotes hidden state in the RNN

When you try to generate the first word output, Jane, what part of the input in French sentence you should look at? A few words close by but no need to look away at the end of the sentence.

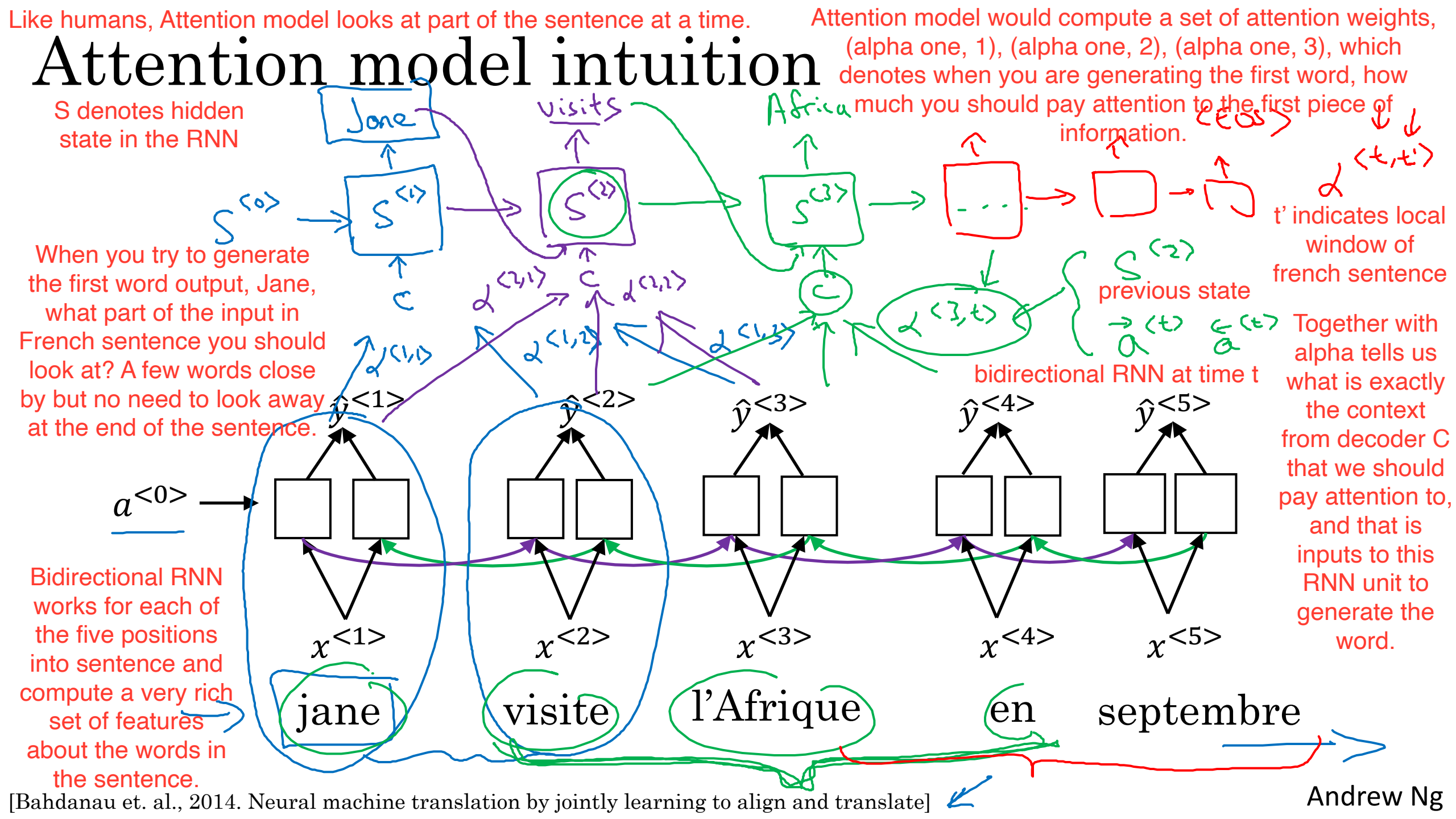
t' indicates local window of french sentence

previous state

bidirectional RNN at time t

Together with alpha tells us what is exactly the context from decoder C that we should pay attention to, and that is inputs to this RNN unit to generate the word.

Bidirectional RNN works for each of the five positions into sentence and compute a very rich set of features about the words in the sentence.



[Bahdanau et. al., 2014. Neural machine translation by jointly learning to align and translate]

Andrew Ng

deeplearning.ai

Sequence to sequence models

Attention model

Attention model

$\alpha^{<t,t'>}$ = amount of "attention" $y^{<t>}$ should pay to $a^{<t'>}$.

Attention weights

single direction RNN

The network looks like a standard RNN sequence with the context vectors as "input".

$$c^{<2>} = \sum_{t'} \alpha^{<2,t'>} a^{<t'>}$$

$$a^{<t'>} = (\overrightarrow{a}^{<t'>}, \overleftarrow{a}^{<t'>})$$

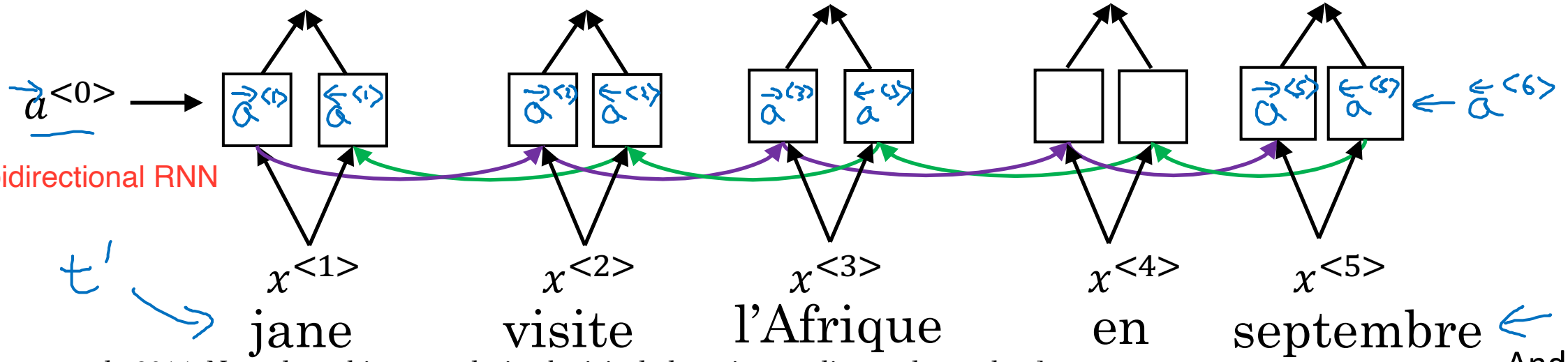
$$\sum_{t'} \alpha^{<1,t'>} = 1$$

non-negative, and sum to one.

Note that the summation is over t'

$$c^{<1>} = \sum_{t'} \alpha^{<1,t'>} a^{<t'>}$$

Context vector



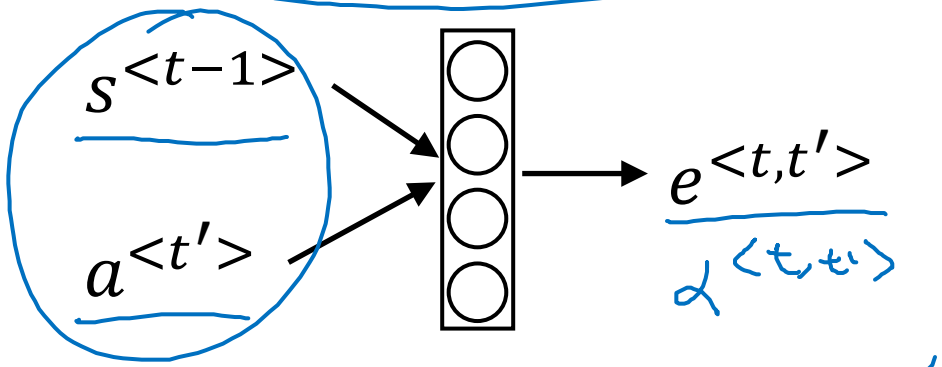
[Bahdanau et. al., 2014. Neural machine translation by jointly learning to align and translate]

Andrew Ng

Computing attention $\alpha^{<t,t'>}$

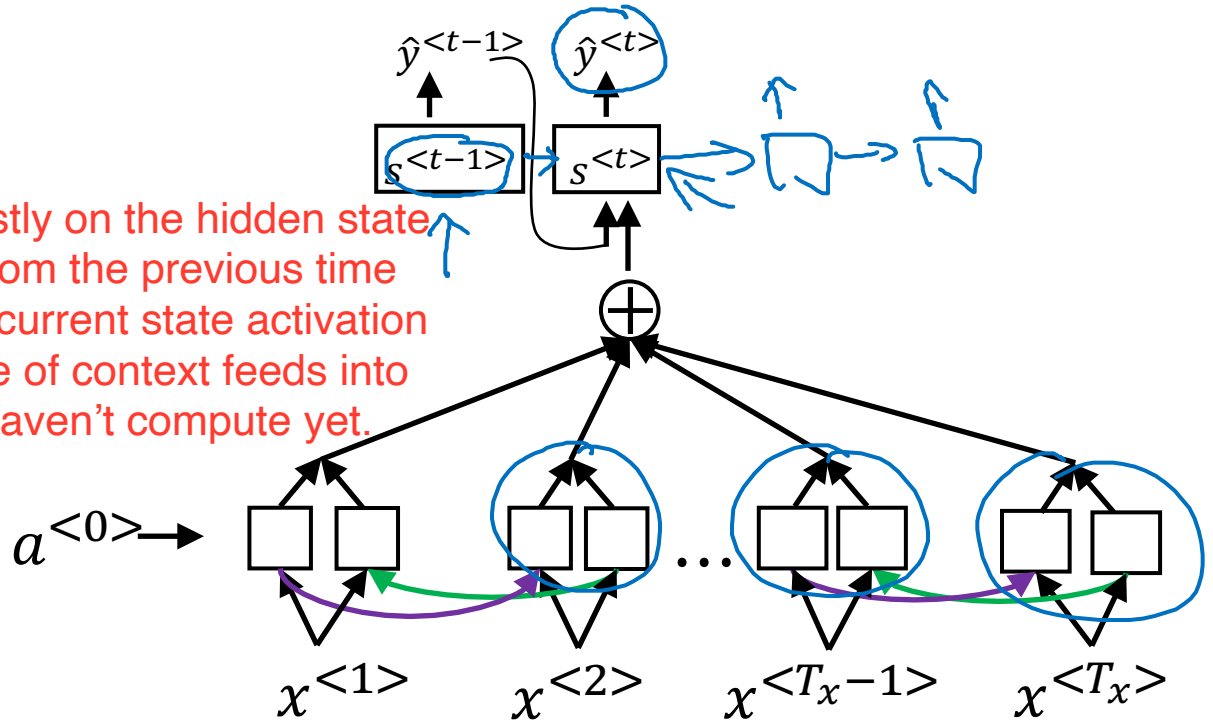
$\alpha^{<t,t'>}$ = amount of attention $y^{<t>}$ should pay to $a^{<t'>}$

$$\alpha^{<t,t'>} = \frac{\exp(e^{<t,t'>})}{\sum_{t'=1}^{T_x} \exp(e^{<t,t'>})}$$



small neural network

Depends mostly on the hidden state activation from the previous time step, not the current state activation yet (because of context feeds into this) you haven't compute yet.



Downside of the algorithm: it takes quadratic time/cost to run

If you have tx words as input and ty words as output, the total number of the attention parameters will be tx times ty

[Bahdanau et. al., 2014. Neural machine translation by jointly learning to align and translate]

[Xu et. al., 2015. Show, attend and tell: Neural image caption generation with visual attention]

Andrew Ng

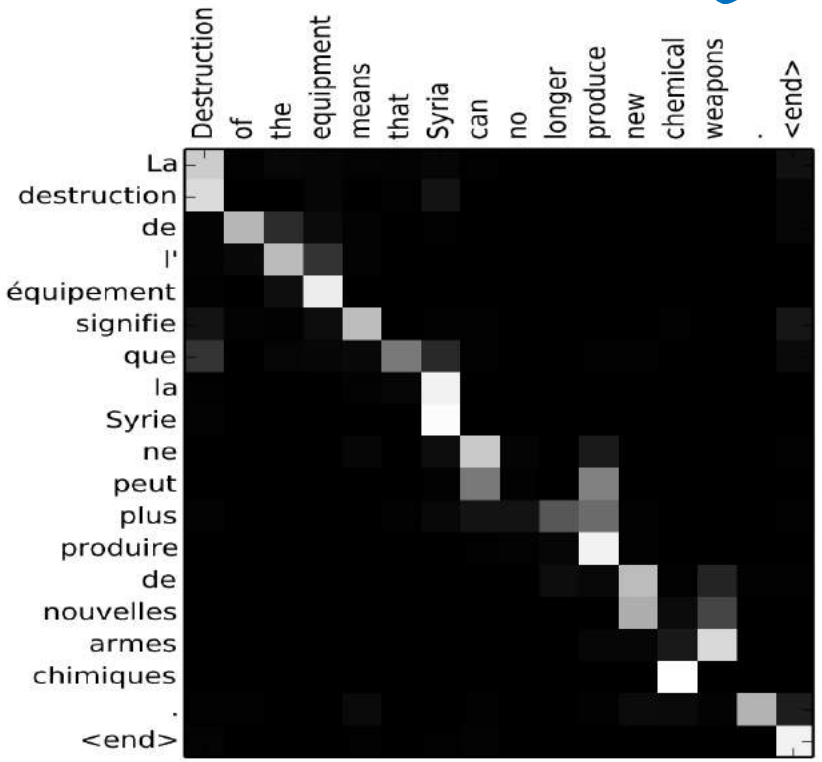
Attention examples

normalized format

July 20th 1969 ⟶ 1969 – 07 – 20

23 April, 1564 ⟶ 1564 – 04 – 23

Visualization of $\alpha^{<t,t'>}$:



different colors are plotted for the magnitude of the different attention weights.

Andrew Ng

deeplearning.ai

Audio data

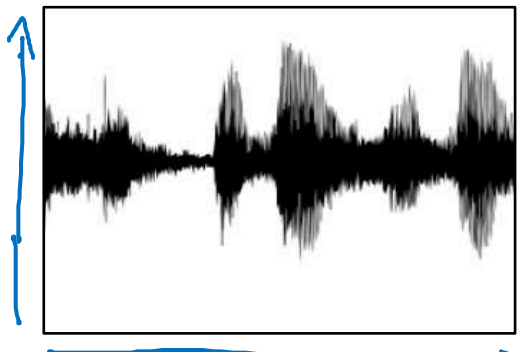
Speech recognition

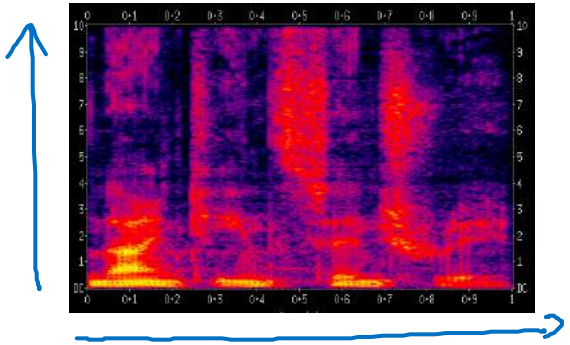
Application of sequence to sequnce model

Speech recognition problem

x

audio clip





human's ear measures different intensity of different frequncies"

y

transcript

"the quick brown fox"

phonemes: de kwik braun

ku", "wu", "ik", "k" sound

300h

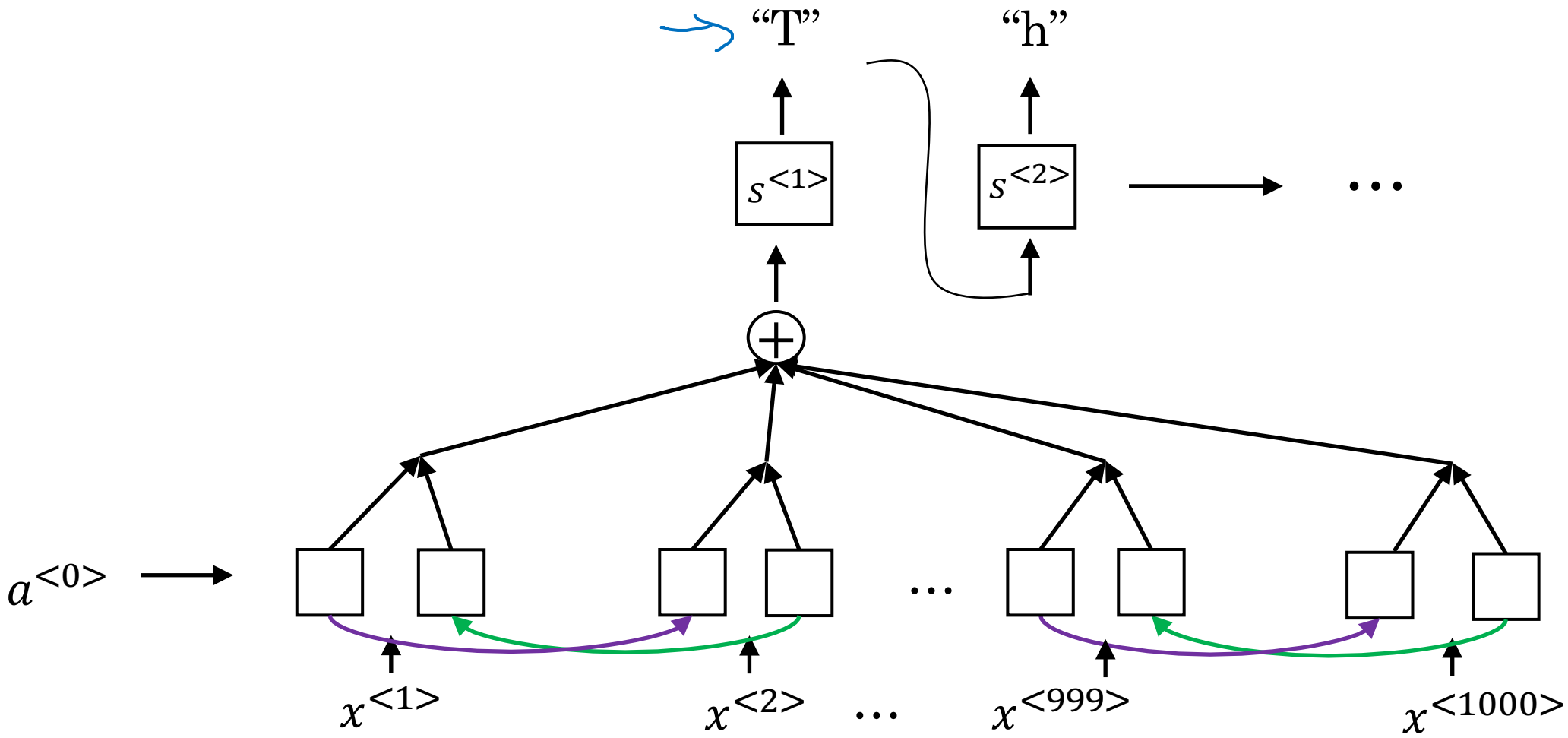
3000h

100,000h

e.g., commercial systems

Andrew Ng

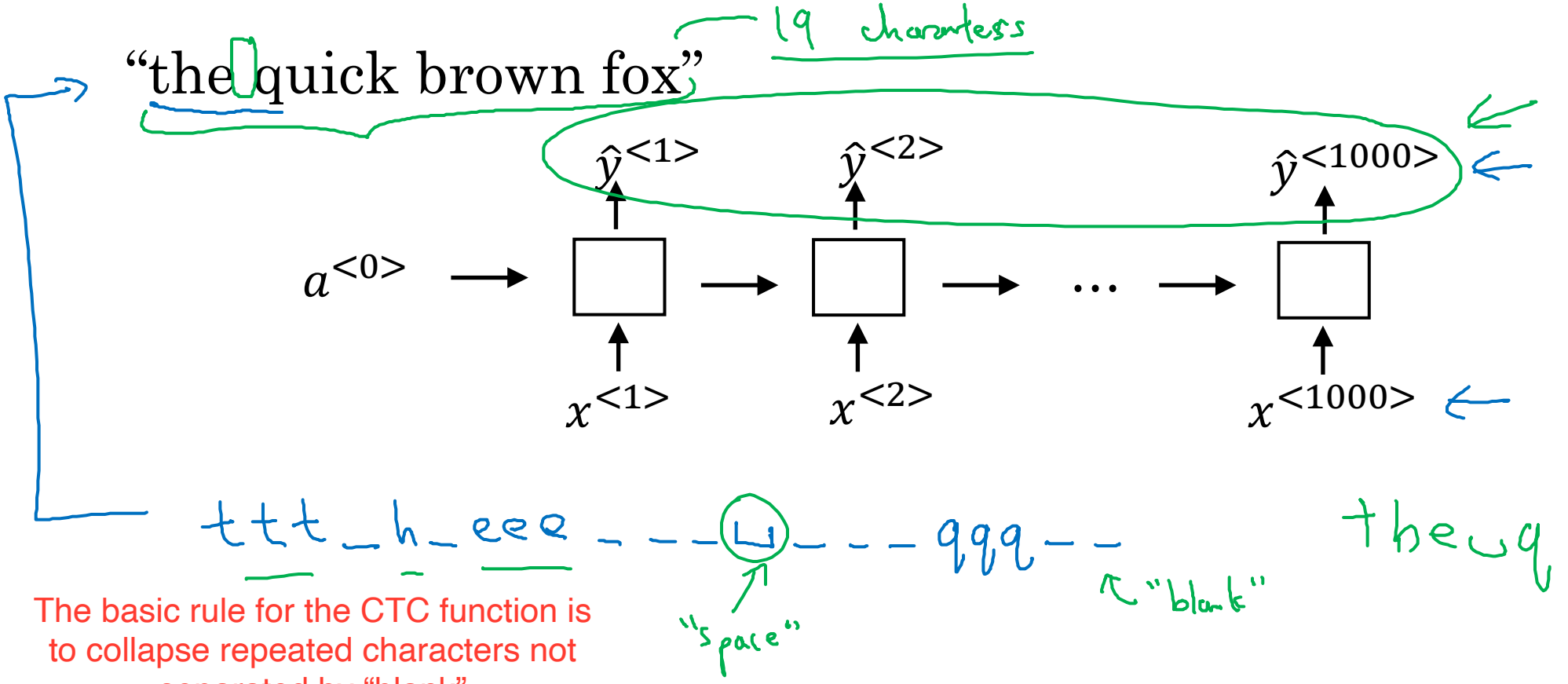
Attention model for speech recognition



CTC cost for speech recognition

In practice, a bidirectional RNN would be used.

(Connectionist temporal classification)



The basic rule for the CTC function is to collapse repeated characters not separated by "blank"

Basic rule: collapse repeated characters not separated by "blank"

[Graves et al., 2006. Connectionist Temporal Classification: Labeling unsegmented sequence data with recurrent neural networks]

deeplearning.ai

Audio data

Trigger word detection

What is trigger word detection?

Amazon Echo
(Alexa)

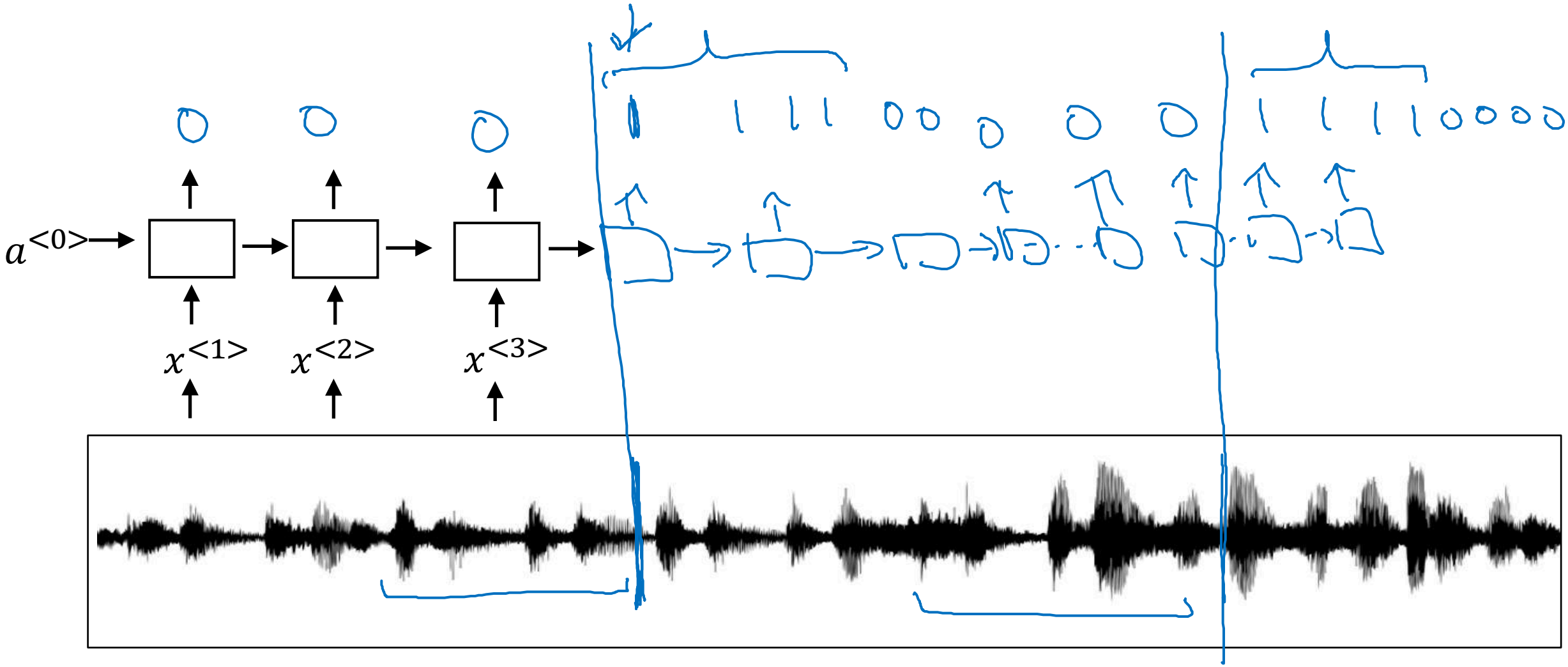
Baidu DuerOS
(xiaodunihao)

Apple Siri
(Hey Siri)

Google Home
(Okay Google)

Trigger word detection algorithm

After saying a trigger word, the output will be coded as one



deeplearning.ai

Conclusion

Summary and thank you

Specialization outline

1. Neural Networks and Deep Learning
2. Improving Deep Neural Networks: Hyperparameter tuning, Regularization and Optimization
3. Structuring Machine Learning Projects
4. Convolutional Neural Networks
5. Sequence Models

Deep learning is a super power

Please buy this from shutterstock and replace in final video.

Andrew Ng

Thank you.

-Andrew Ng