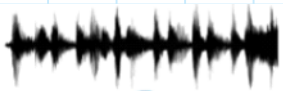
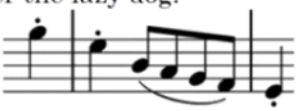
Week 1 Recurrent Neural Networks

Thursday, April 5, 2018 9:26 PM

RNNs

Thursday, April 5, 2018 9:27 PM

Why Sequence models?

Speech recognition		→	"The quick brown fox jumped over the lazy dog."
Music generation	∅	→	
Sentiment classification	"There is nothing to like in this movie."	→	★☆☆☆☆
DNA sequence analysis	AGCCCCTGTGAGGAACTAG	→	AGCCCCTGTGAGGAACTAG
Machine translation	Voulez-vous chanter avec moi?	→	Do you want to sing with me?
Video activity recognition		→	Running
Name entity recognition	Yesterday, Harry Potter met Hermione Granger.	→	Yesterday, Harry Potter met Hermione Granger.

Screen clipping taken: 4/5/2018 9:34 PM

Screen clipping taken: 4/19/2018 1:23 PM

Notation

Screen clipping taken: 4/19/2018 1:22 PM

motivating example: Named entity recognition

$x^{<1>}$ $x^{<2>}$ $x^{<3>}$ ... $x^{<t>}$ ... $x^{<9>}$

x: Harry Potter and Hermione Granger ...

y: 1 1 0 1 1 0 0 0 0

$y^{<1>}$ $y^{<2>}$ $y^{<3>}$ ... $y^{<t>}$ ... $y^{<9>}$

$X^{(i)<t>}$ - t^{th} value in example i

$T_x^{(i)}$ - length of train ex. i

How do we represent words?

Vocab

a	1
aaron	2
⋮	
and	367
⋮	
harry	4075
⋮	
potter	6830
⋮	
Zulu	10^4

represent w/ one-hot [recall: $\begin{bmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix}^{m}_{n} \rightarrow [m]$]

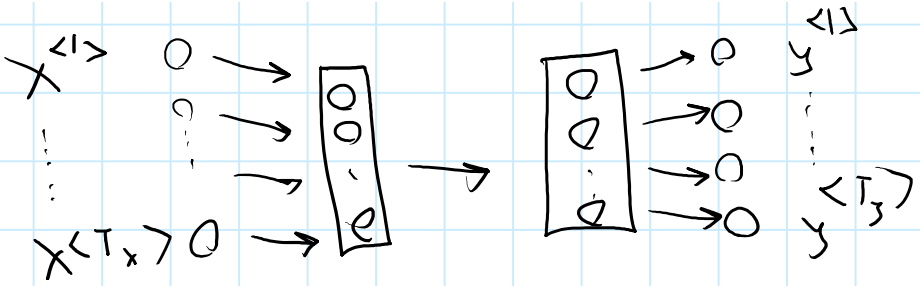
vector for $x^{<t>}$

Words not in vocab denoted <unk>

Recurrent Neural Network Model

Why not use a standard NN?

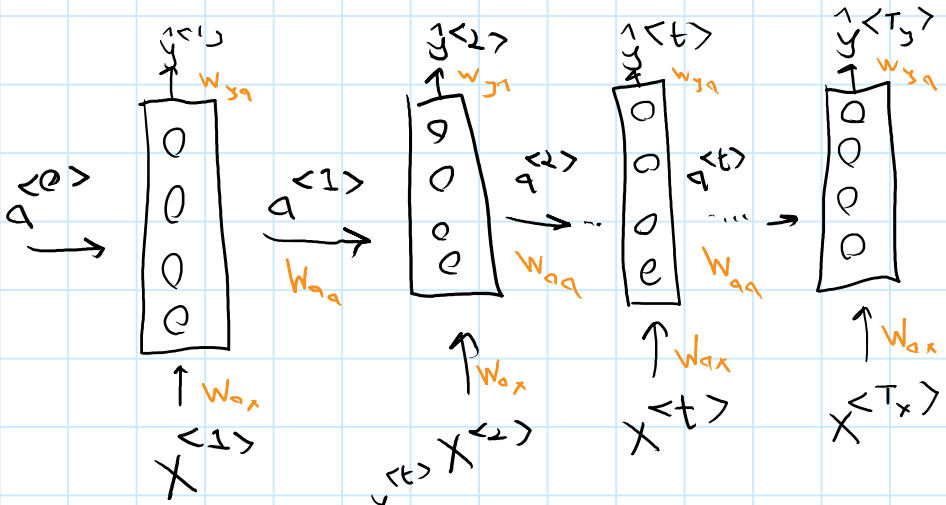
$x^{<1>}$  $y^{<1>}$



Problems:

- Inputs & outputs can be different lengths
- Doesn't share features learned across diff positions of text

What is an RNN?



other notation

Can't get info on later words this way, issue case:

"He said, 'Teddy Roosevelt was a great President'"

He said "Teddy bears are on sale."

Solve later with Bidirectional RNN (BRNN)

$$a^{\langle 0 \rangle} = \vec{0}; \quad a^{\langle 1 \rangle} = g(W_{aa} a^{\langle 0 \rangle} + W_{ax} X^{\langle 1 \rangle} + b_a) \leftarrow \text{tanh/ReLU (vanishing gradient)}$$

$$\hat{y}^{\langle 1 \rangle} = g(W_{ya} a^{\langle 1 \rangle} + b_y) \leftarrow \text{sigmoid / softmax}$$

$$a^{\langle t \rangle} = g(W_{aa} a^{\langle t-1 \rangle} + W_{ax} X^{\langle t \rangle} + b_a)$$

$$y^{\langle t \rangle} = g(W_{ya} a^{\langle t \rangle} + b_y)$$

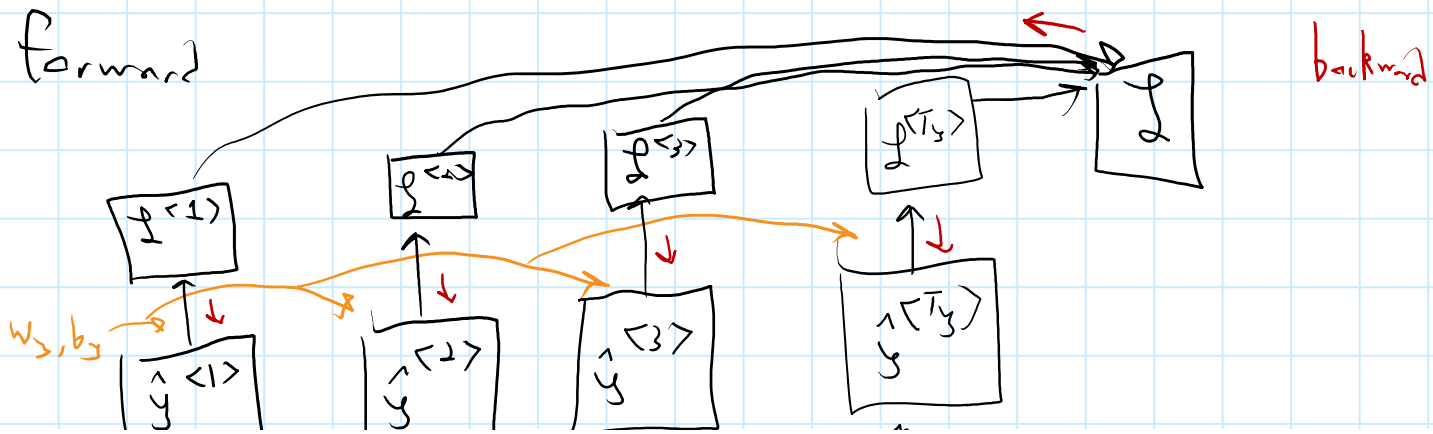
Shorthand notation

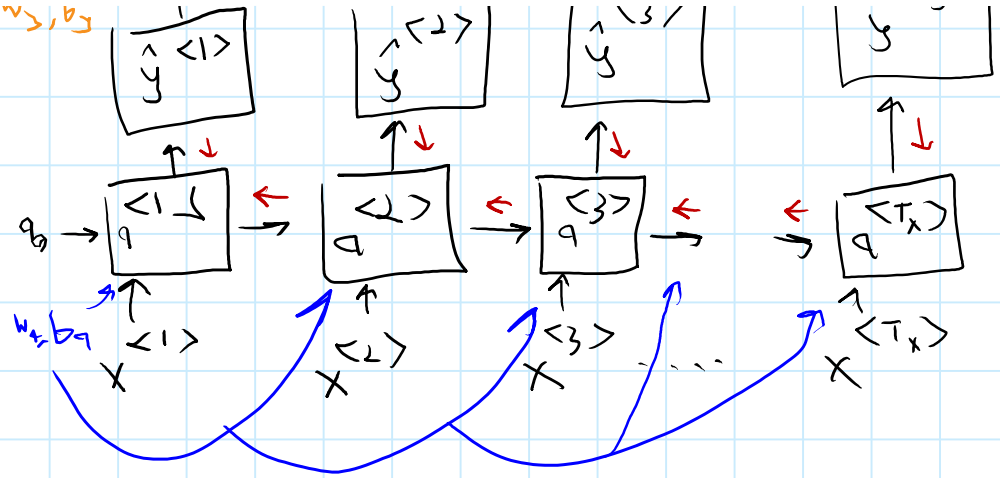
$$W_a = [W_{aa} \; W_{ax}]$$

$$a^{\langle t \rangle} = g(W_a [a^{\langle t-1 \rangle}, x^{\langle t \rangle}] + b_a)$$

$$\hat{y}^{\langle t \rangle} = g(W_y a^{\langle t \rangle} + b_y)$$

Back propagation through time





$$\mathcal{L}^{\langle t \rangle}(\hat{y}^{\langle t \rangle}, y^{\langle t \rangle}) = -y^{\langle t \rangle} \log(\hat{y}^{\langle t \rangle}) - (1 - y^{\langle t \rangle}) \log(1 - \hat{y}^{\langle t \rangle})$$

$$\mathcal{L}(\hat{y}, y) = \sum_{t=1}^{T_y} \mathcal{L}^{\langle t \rangle}(\hat{y}^{\langle t \rangle}, y^{\langle t \rangle})$$

Different types of RNNs

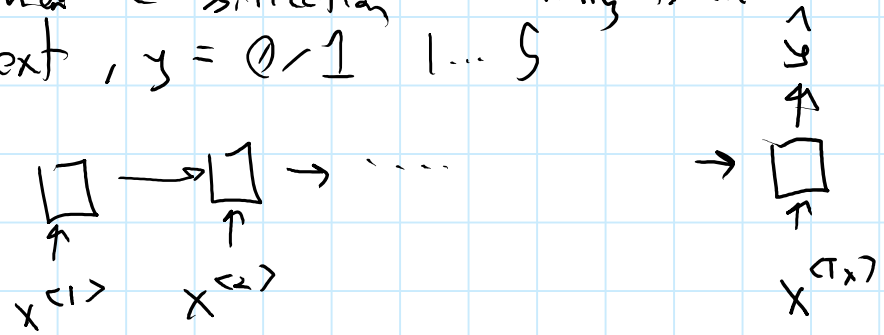
[Read "The unreasonable effectiveness of RNNs" – Andrej Karpathy]

What if x & y are different sizes? (see pic from L1)

Examples so far $T_x = T_y$, "many-to-many"

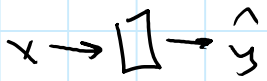
Sentiment classification "many-to-one"

x = text, y = 0/1 | ... | 5

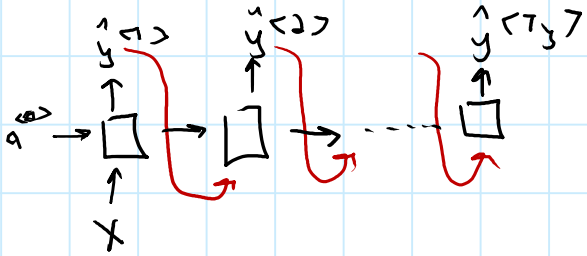


There is nothing to like in this movie

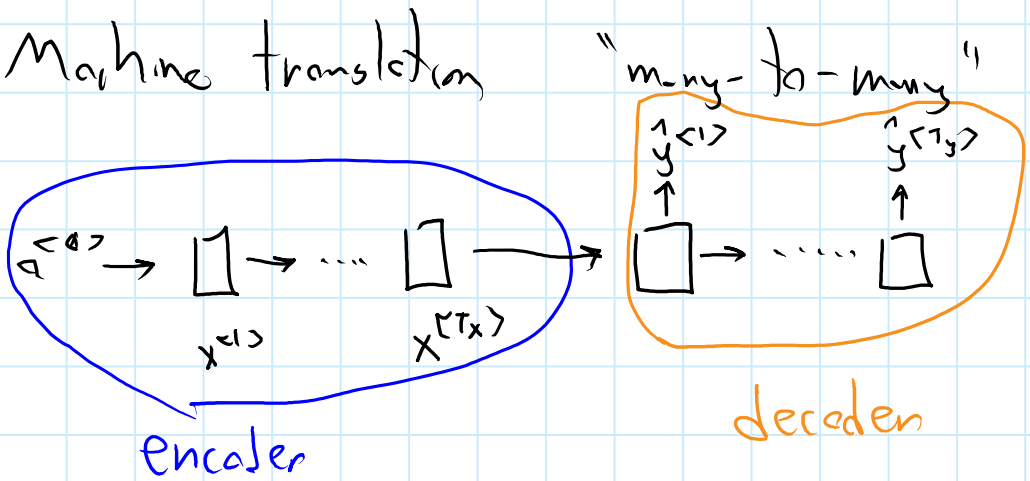
"One-to-one" like a standard NN from first 2 courses



Music Generation "One-to-many"



Machine translation "many-to-many"



Language Model & Sequence Generation

Speech recognition

Decide on homonyms pear/pair

P(sentence) = output

$P(y^{<1>}, y^{<2>}, \dots, y^{<T_y>})$

Training set: large corpus of english text

"Tokenize" sentence

End of Sentence token

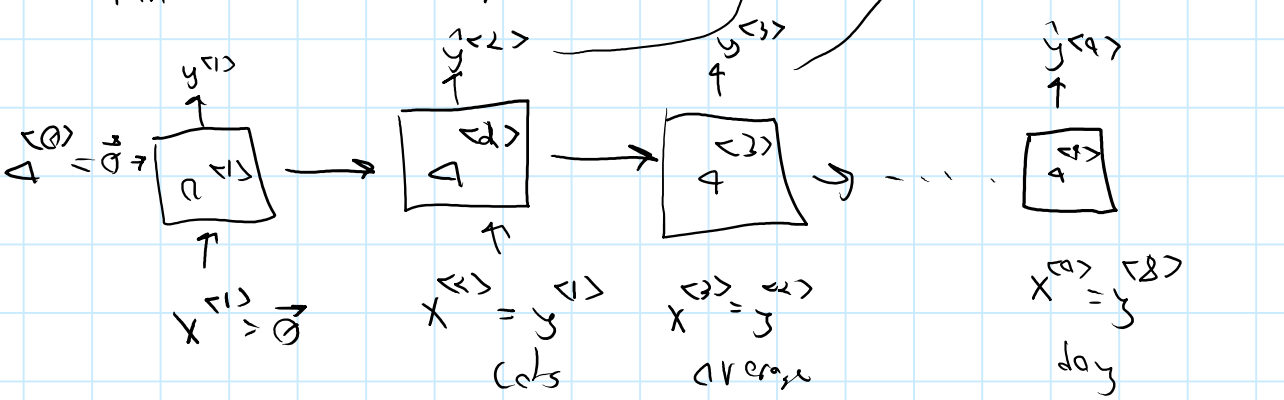
Cats average 15 hours of sleep a day. <EOS>

$y^{<1>}$ $y^{<2>}$ ... $y^{<8>}$ $y^{<9>}$

$P(\text{average} \mid \text{cats})$

$P(\text{—} \mid \text{"cats average"})$

RNN $P(y^{<1>}) = [\text{dict}1, \text{dict}2, \ldots]$



$$\mathcal{L}(\hat{y}^{<t>}, y^{<t>}) = -\sum_i y_i^{<t>} \log \hat{y}_i^{<t>} \quad ; \quad \mathcal{L} = \sum_t \mathcal{L}(\hat{y}^{<t>}, y^{<t>})$$

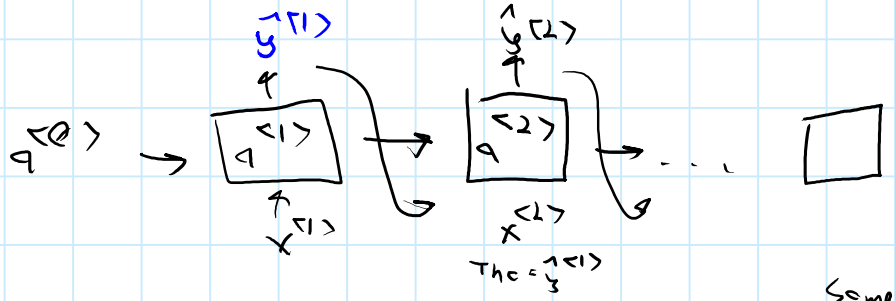
$$P(y^{<1>}, y^{<2>}, y^{<3>}) = P(y^{<1>}) \, P(y^{<2>} \mid y^{<1>}) \, P(y^{<3>} \mid y^{<1>}, y^{<2>})$$

Sampling Novel Sequences

Good way to check an trained network

Sample a sequence from a trained RNN, gen sentence

Sample a sequence from a trained RNN, gen sentence



Sample until <EOS> or word limit

$\hat{y}^{<1>}$ = np.random.choice([P(a), P(aaron)...])

Can also do this on character level

pro's: no <UNK> (good for specialized vocabulary)

con: much longer sequences, computationally expensive

Vanishing Gradients With RNNs

Ex The cat, which already ate ..., was full
" cats ... were "

Basic RNN not good at detecting long term dependencies, this is due to vanishing gradients

Values are mainly affected by other values near them

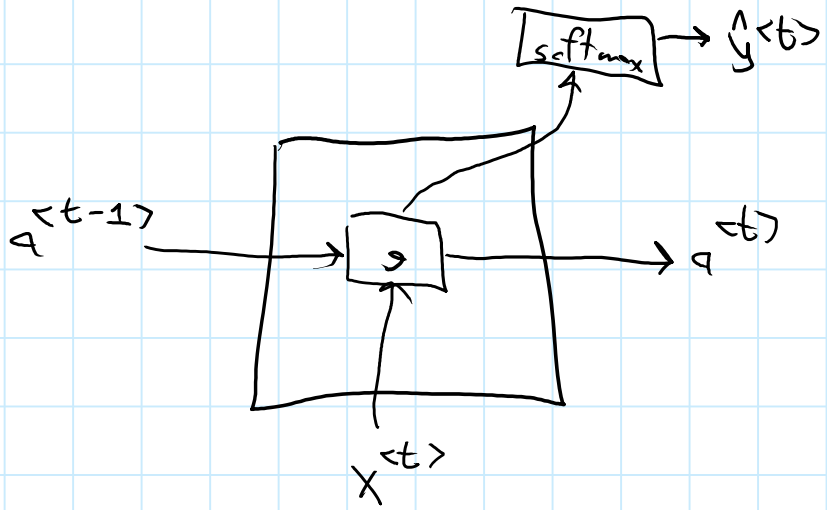
What if you get exploding gradients?

Use gradient clipping (rescale gradient vector if > threshold)

Gated Recurrent Unit (GRU)

RNN:

$$a^{<t>} = g(W_a [a^{<t-1>}, x^{<t>}] + b_a)$$



GRU (simplified)

c = memory cell

$c^{<t>} = a^{<t>}$

The [cat] ..... [was] full

At every time step, consider overwriting $c^{<t>}$ w/ $c^{<t>}$ tilde ($\tilde{c}^{<t>}$)

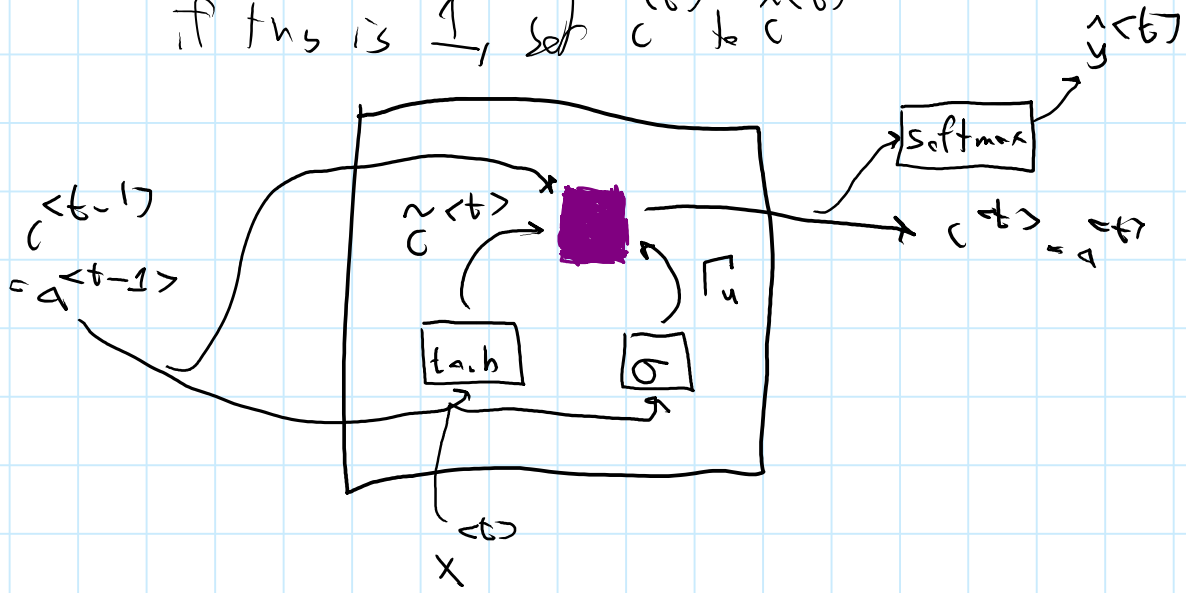
$$\tilde{c}^{<t>} = \tanh(W_c [c^{<t-1>}, x^{<t>}] + b_c)$$

Gate: $\Gamma_u = \sigma(W_u [c^{<t-1>}, x^{<t>}] + b_u)$

in this example, $c^{<t>}$ { 0 plural, 1 singular

$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + (1 - \Gamma_u) * c^{<t-1>}$$

if this is 1, set $c^{<t>}$ to $\tilde{c}^{<t>}$



Full GRU

$$\tilde{c}^{<t>} = \tanh(W_c[\Gamma_r * c^{<t-1>}, x^{<t>}] + b_c)$$

$$\Gamma_r = \sigma(W_r[c^{<t-1>}, x^{<t>}] + b_r)$$

Long Short Term Memory (LSTM)

$$\tilde{c}^{<t>} = \tanh(W_c[a^{<t-1>}, x^{<t>}] + b_c)$$

$$\Gamma = \sigma(W_{\cdot}[a^{<t-1>}, x^{<t>}] + b_{\cdot})$$

$$\Gamma_u = \sigma(W_u [a^{<t-1>}, x^{<t>}] + b_u)$$

"forget" $$\Gamma_f = \sigma(W_f [a^{<t-1>}, x^{<t>}] + b_f) \sim (1 - \Gamma_u)$$

"output" $$\Gamma_o = \sigma(W_o [a^{<t-1>}, x^{<t>}] + b_o)$$

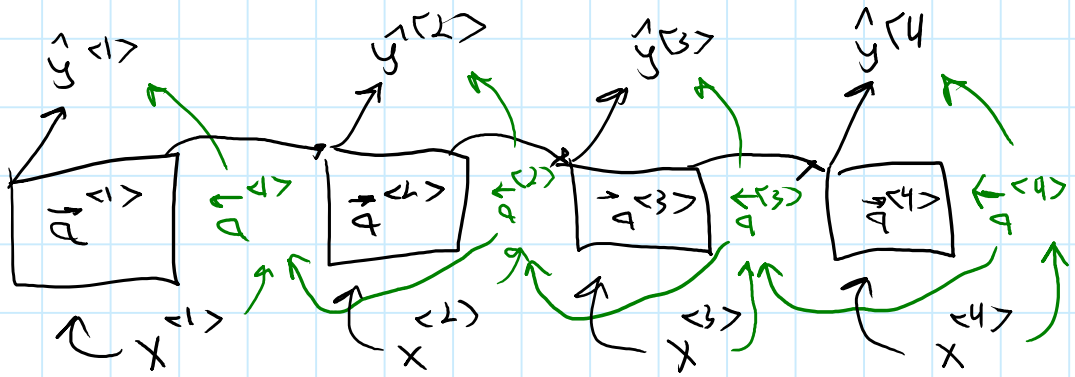
$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + \Gamma_f * c^{<t-1>}$$, can add old + new value

$$a^{<t>} = \Gamma_o * c^{<t>}$$

"peephole connection" $\sigma(W_- [a^{<t-1>}, c^{<t-1>}] + b_-)$

Bidirectional RNNs

Recall Teddy sentence issue

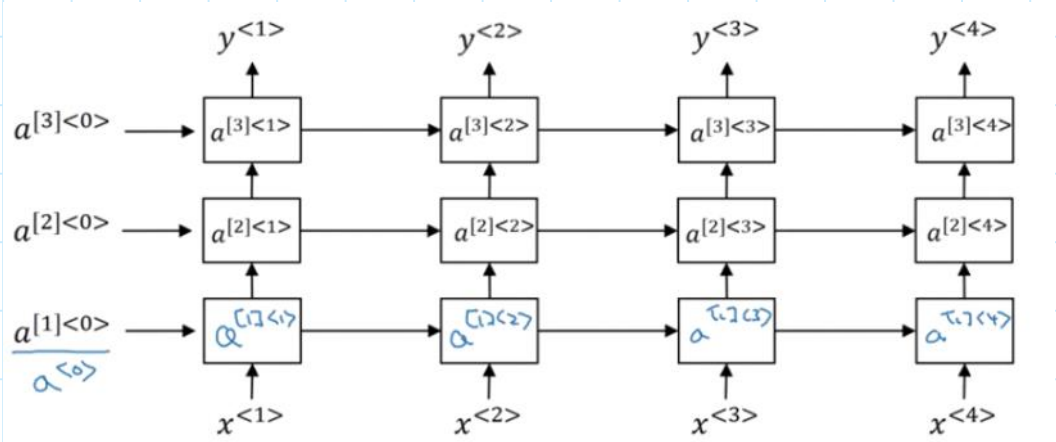


Still forward propagation! Acyclic graph

$$\hat{y}^{<t>} = g(W_y[\vec{a}^{<t>}, \overleftarrow{a}^{<t>}] + b_y)$$

Disadvantage: need whole sentence before computing (e.g. bad for real time speech processing)

Deep RNNs



$$a_4^{[2]\langle 3\rangle} = g\left(W_4^{[2]}\begin{bmatrix} a_1^{[2]\langle 2\rangle} \\ a_4^{[1]\langle 3\rangle} \end{bmatrix} + b_4^{[2]}\right)$$

Week 2 NLP and Word Embeddings

Tuesday, April 24, 2018 10:53 AM

Intro to Word Embeddings

Tuesday, April 24, 2018 10:53 AM

Helps understand analogies

Screen clipping taken: 4/24/2018 3:30 PM

Word Representation

previously:

$V = [a, aaron, \ldots, zulu, \langle UNK \rangle]$ Onehot vectors

Downside: Each word self-contained, can't generalize across words

e.g. I want a glass of orange juice

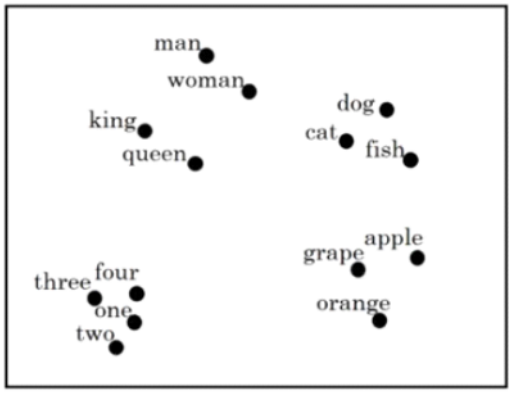
" " " " " apple ——— ? doesn't recognize link

This arises b/c inner product between one-hot vectors is 0, Euclidean distance always the same

Featurized Representation

index	(5391)	(9853)	(4914)	(7157)	(456)	(6257)
word	Man	Woman	King	Queen	Apple	Orange
Gender	-1	1	-0.95	0.97	0.00	0.01
Royal	0.01	0.02	0.93	0.95	-0.01	0.00
Age	0.03	0.02	0.7	0.69	0.03	-0.02
Food	0.04	0.01	0.02	0.01	0.95	0.97
⋮						

(embedding) e_{5391} e_{9853}



t-SNE

300D

↓

2D (PCA?)

Using Word Embeddings

Use example

Sally Johnson is an orange farmer

Robert Lin is a durian cultivator

1) Train on 1B~100B words
   (or download pretrained embedding online)
2) Xfer embedding to new task w/ smaller set
3) (Optional) Continue to fine tune word embeddings (if 2 has large set)

Relation to Face Encoding

$f(x^{(i)}) \approx e_i$

Properties of Word Embeddings

Analogies

index	(5391)	(9853)	(4914)	(7157)	(456)	(6257)
word	Man	Woman	King	Queen	Apple	Orange
Gender	-1	1	-0.95	0.97	0.00	0.01
Royal	0.01	0.02	0.93	0.95	-0.01	0.00
Age	0.03	0.02	0.7	0.69	0.03	-0.02
Food	0.04	0.01	0.02	0.01	0.95	0.97
	e_{man}	e_{woman}	e_{King}	e_{queen}		

Can it learn man is to woman as king is to queen?

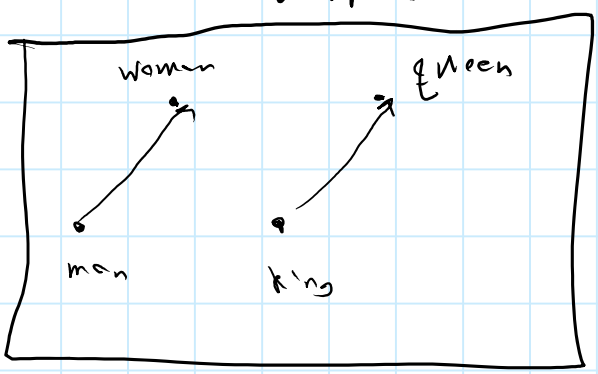
$$e_{man} - e_{woman} = \begin{bmatrix} -2 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad e_{King} - e_{queen} = \begin{bmatrix} -2 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

[Algorithm: for e_w in e: $e_{man} - e_{woman} \approx e_{King} - e_w$]

Dr. No: Find word in w:

$$\arg\max_w \; sim(e_w, e_{King} - e_{man} + e_{woman})$$

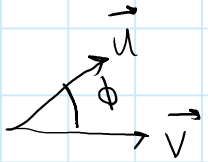
in 300D space



(won't hold after t-SNE mapping)

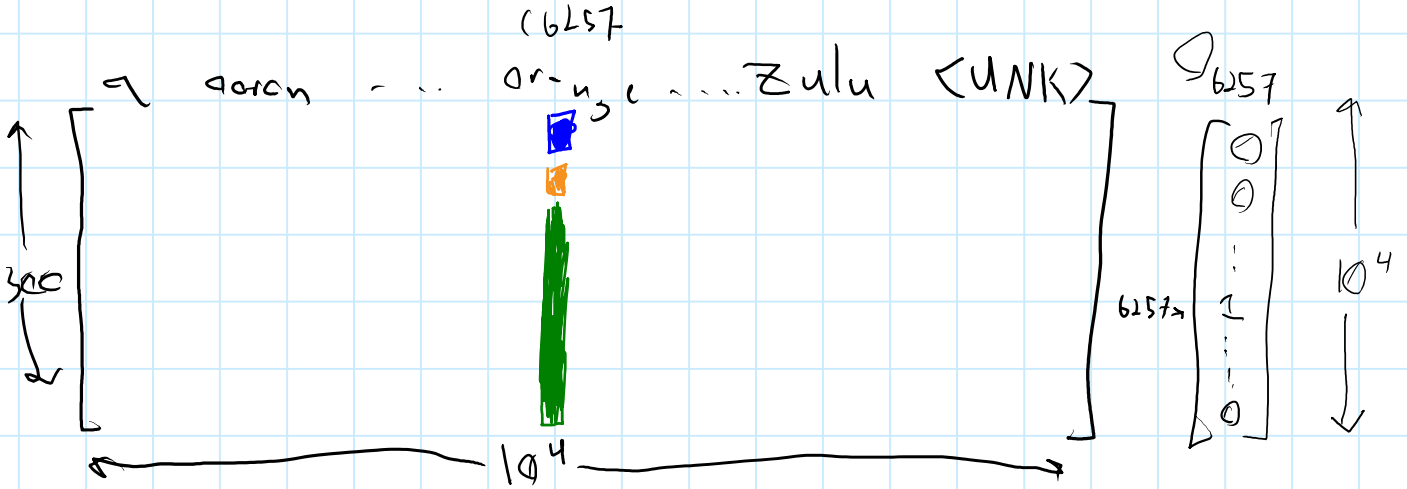
Cosine similarity (inner product)

$$sim(u, v) = \frac{u^T v}{\|u\|_2 \|v\|_2} \equiv$$



[Could also use - Euclidean distance (dissimilarity) $\|u - v\|^2$]

Embedding Matrix



$E \cdot O_{6257} = [\,] \uparrow 300 \downarrow = e_{6257}$

$E \cdot O_j = e_j$

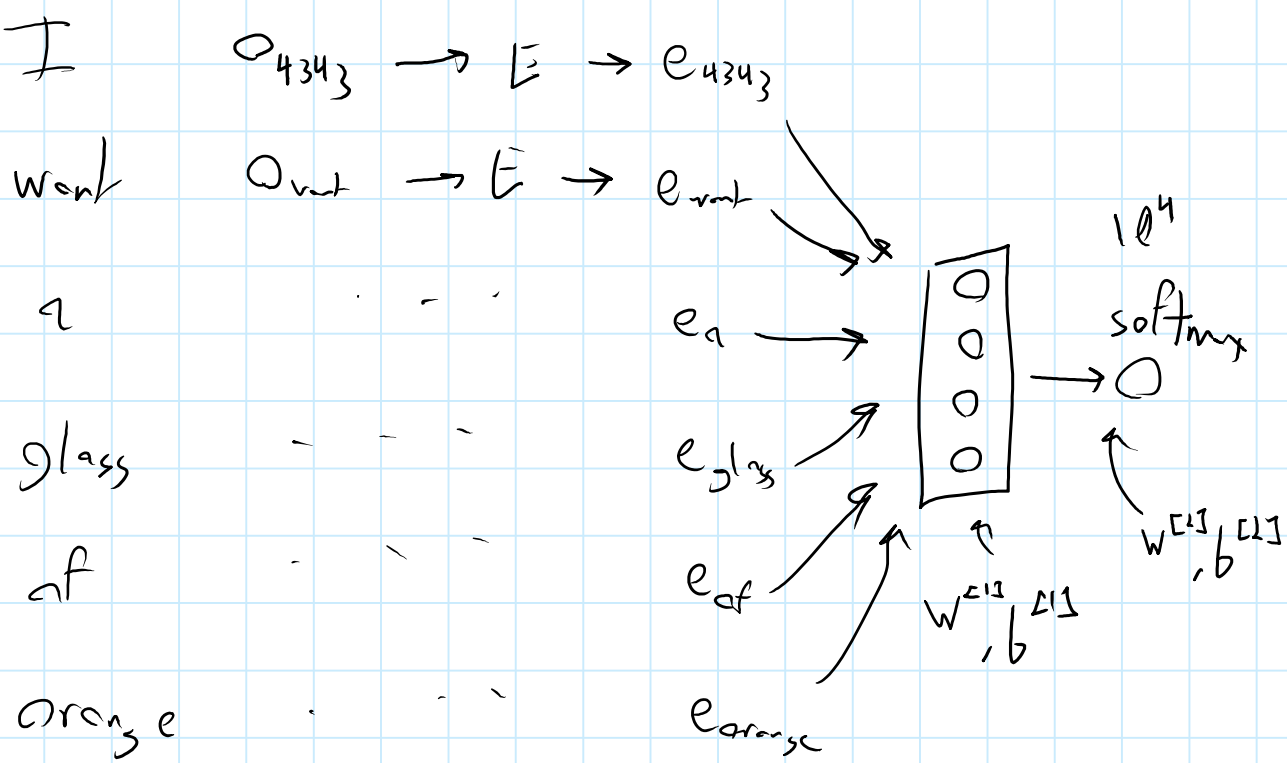
In practice, this isn't very efficient, use specialized function

Learning Word Embeddings

Wednesday, April 25, 2018 11:02 AM

Algorithms have gotten simpler over time

Neural Language Model



often use fixed history (4 words prior to prediction)

Other context/target pairs

I want a glass of orange juice to go along with my cereal

context target

last 4 words
4 words on left + right
last one word

orange ?

- last one word
- Nearby one word (skip gram)
- Good for learning embeddings

orange ?

glass ?

Word 2 Vec

Skip-gram

Create context-target pairs for supervised learning

I want a glass of orange juice to go along with my cereal

Context	Target
orange	juice
orange	glass
orange	my

Model

Vocab size = 10,000 k

Context c ("orange") → Target t ("juice")

6257 → 4834

$O_c \rightarrow E \rightarrow e_c \rightarrow \text{softmax} \rightarrow \hat{y}$

$\text{softmax}(t|c) = \frac{e^{\theta_t^T e_c}}{\sum_{j=1}^{10000} e^{\theta_j^T e_c}}$ θ_t = parameter associated w/ output t

Softmax: $p(t/c) = \frac{e^{\theta_t^T e_c}}{\sum_{i=1}^{10,000} e^{\theta_t^T e_c}}$ θ_t = parameter associated w/ output t

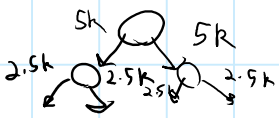
Loss $\mathcal{L}(\hat{y}, y) = -\sum_{i=1}^{10,000} y_i \log(\hat{y}_i)$

y_i: one-hot; $\hat{y}_i$: 10k dim vector output by softmax

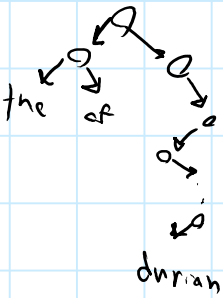
Problems

computational speed, vocab size due to summing

Solutions: hierarchical softmax (split data set)



in practice, not symmetric (common words near top)



How to sample context c?

Don't want commonly occurring words dominating (the, of, a, and, etc.)

Sampling not truly random

Negative Sampling

Make skip-gram more efficient

Given a pair, are they target-context pairs?

First row generated by neighbor sampling, next random dictionary words

Context	Word	Target?
orange	juice	1
orange	king	0
"	book	0
"	the	0
"	of	0

k random dictionary words, labelled 0

k ← 5-20 for small data sets
2-5 for larger data sets

Model target context or word

$$P(y=1 \mid c, t) = \sigma(\theta_t^T e_c)$$

NN

orange

$O_{6257} \rightarrow E \rightarrow e_{6257}$ → 10,000 binary classifiers for V[:] (than k+1 per iteration)

Selecting Negative Examples

Best from paper: $P(w_i) = \frac{f(w_i)^{3/4}}{\sum_{j=1}^{10{,}000} f(w_j)^{3/4}}$

In between uniform & representing

GloVe Word Vectors

Used less than word2vec + skip-gram

Global Vectors for Word representation

X_{ij} = # times i appears in context of j

(i = t, j = c)

if context is symmetric, $X_{ij} = X_{ji}$

minimize $\sum_{i=1}^{10{,}000} \sum_{j=1}^{10{,}000} f(X_{ij}) (\theta_i^T e_j + b_i + b'_j - \log(X_{ij}))^2$

(θ_i ↔ t, e_j ↔ c)

weighting, = 0 if $X_{ij} = 0$

(this is for $\log 0 = 0$ in this context)

θ_i & e_i are symmetric

$e_w^{(final)} = \frac{e_w + \theta_w}{2}$

A note on the featurization view of a word

Can't guarantee individual components are interpretable

$\theta_i^T e_i$ [A is invertible]

$(A\theta_i)^T (A^{-T} e_j) = \theta_i^T A^T A^{-T} e_j$

axes may not be well aligned

Applications Using Word Embeddings

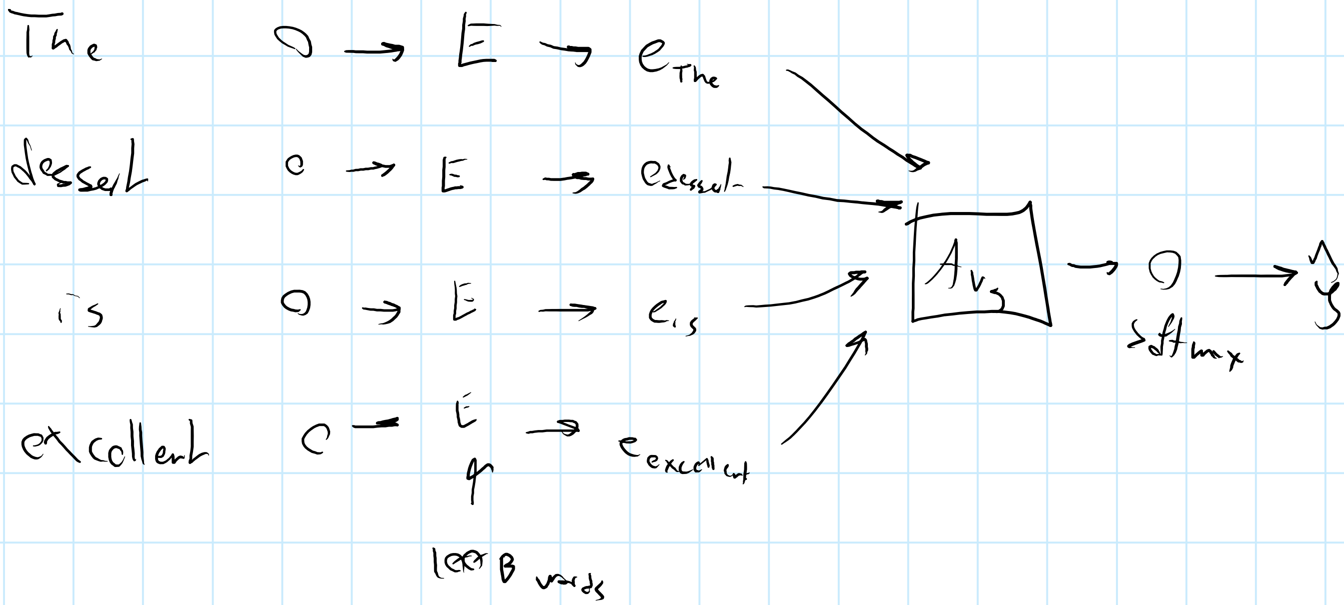
Wednesday, April 25, 2018 2:49 PM

Sentiment Classification

Sentence about restaurant → star rating

Training sets <10k words → 100k words

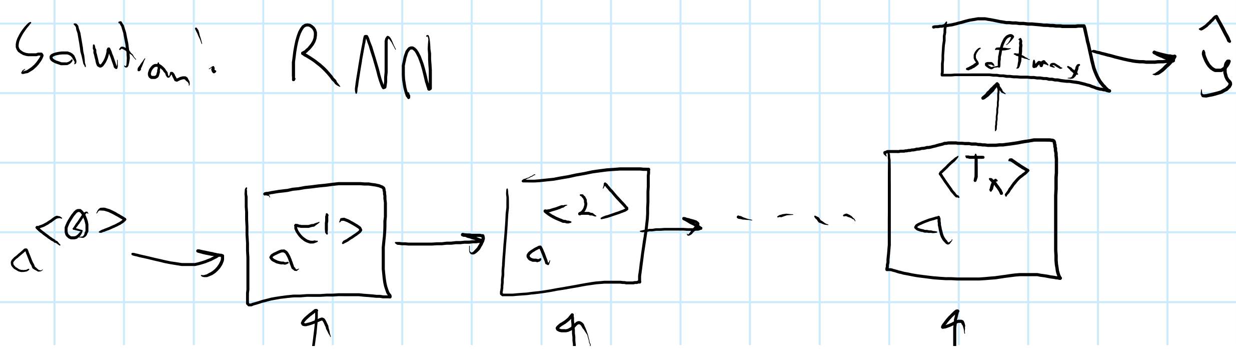
Ex:



Issue: ignores order

"Lacking in good service" marked high

Solution: RNN



a → a a

e_1 e_2 e_3

Debiasing Word Embeddings

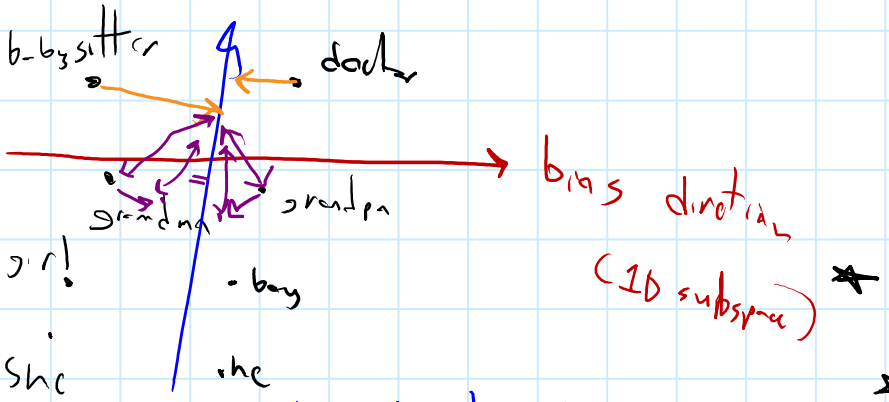
(Discriminatory in the ethical, not statistical sense)

Man : Woman as King : Queen [Bolukbasi 2016]

Man : Programmer as Woman : Homemaker

Father : Doctor as Mother : Nurse

Addressing bias in word embeddings



*found using PCA by SVD

1) 1D bias direction

$e_{he} - e_{she}$

$e_{male} - e_{female}$

etc.

2) Neutralize!

Project every non-definitional word to eliminate bias

(

grandma, grandpa would be definitional

3) Equalize pairs

grandma ⟷ grandpa

boy ⟷ girl

Week 3 Sequence Models and Attention Mechanism

Thursday, April 26, 2018 1:22 PM

Various Sequence to Sequence Architectures

Thursday, April 26, 2018 1:24 PM

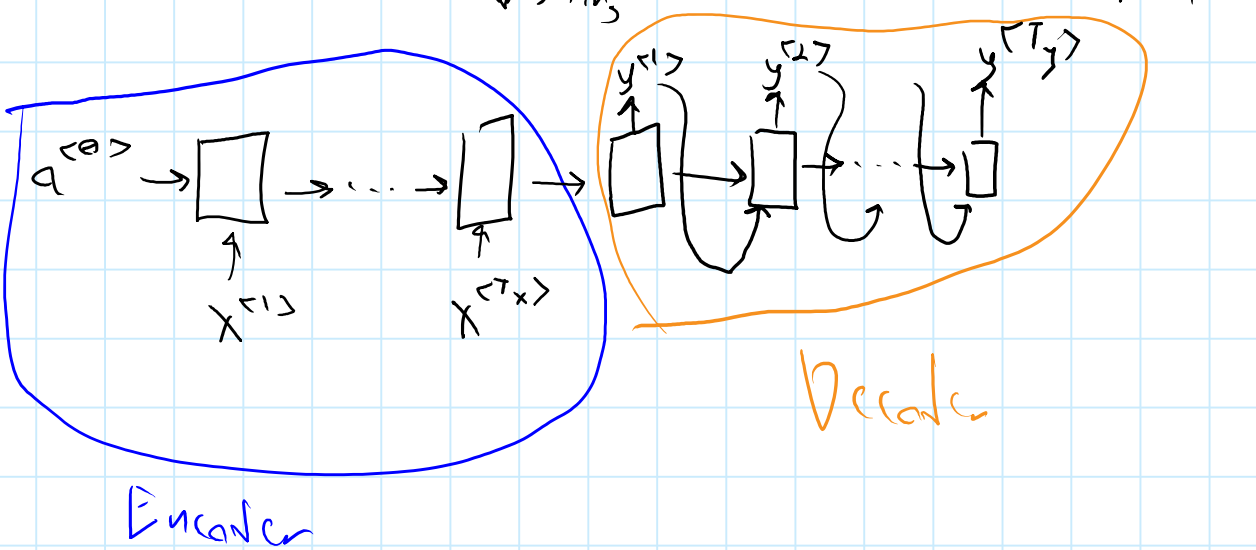
Screen clipping taken: 4/26/2018 2:37 PM

Basic Models

Sequence-to-sequence model

Jane visite l'Afrique en septembre

→ Jane is visiting Africa in September

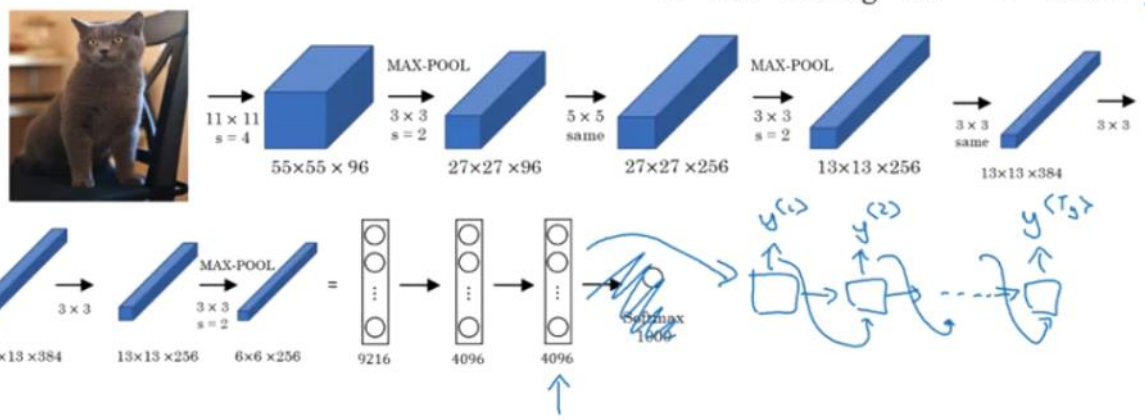


Encoder

Decoder

Image captioning

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



[Mao et. al., 2014. Deep captioning with multimodal recurrent neural networks]
[Vinyals et. al., 2014. Show and tell: Neural image caption generator]

Screen clipping taken: 4/27/2018 8:10 AM

Picking the Most Likely Sentence

Decoder network resembles language model

Machine translation "conditional language model"

Where Language Model computes $P(y^{<1>}, y^{<2>}, \ldots, y^{<T_y>})$

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

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

↑ English

Do not want random sampling! $\arg\max_{y^{<1>},\ldots,y^{<T_y>}} P(y^{<1>},\ldots,y^{<T_y>}|x)$

Why not use greedy search? [most likely from position $y^{<1>}$]

Jane is < going ... better $P(y^{<3>}|x)$
< visiting ... better sentence

Also computationally inefficient, better to use approximate

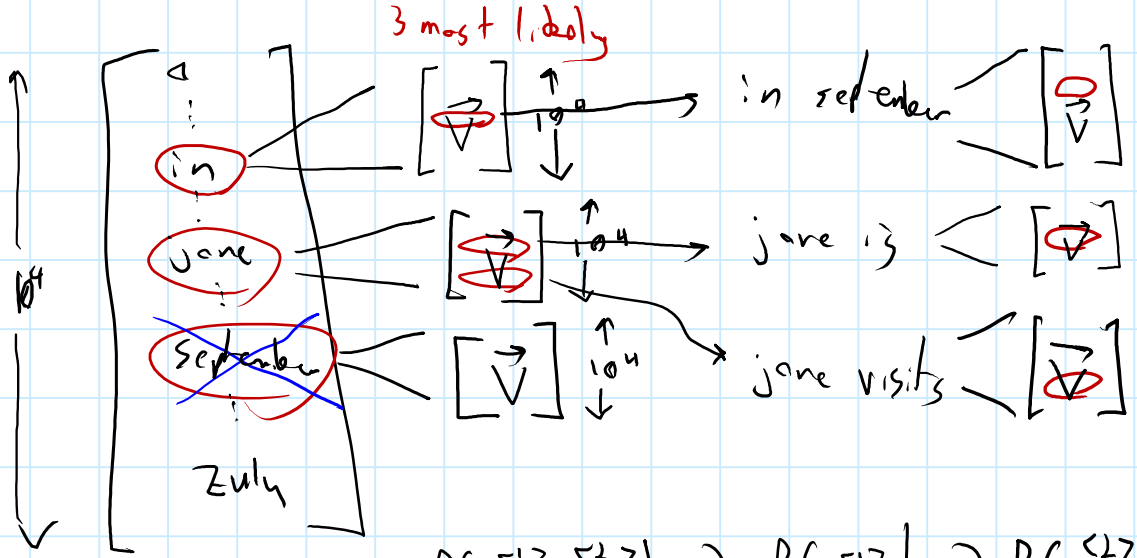
Beam Search

Parameter B = beam width

Picks B most likely possibilities for $P(y^{<1>}|x)$
keeps in memory

ex $B=3$

Step 1 Step 2 Step 3



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

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

Every step runs model 3 copies of network

Refinements to Beam Search

Length Normalization

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

Small enough to run into underflow

Soln: $$\arg\max_{y} \sum_{y=1}^{T_y} \log\left(P(y^{<t>} \mid x, y^{<1>}, \ldots, y^{<t-1>})\right)$$

[Same value will maximize both since log monotonically increases]

Problem: Shorter sentences are preferred by nature of algorithm

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

How to choose B?

large: better result, slower, more memory

small: worse, faster, less memory

prod systems: $10-100$; research: $10^3 - 3 \times 10^3$

Grows ~ logarithmic

Error Analysis in Beam Search

Beam search doesn't ALWAYS find best result,

how do we decide if it's an RNN issue or Beam Search?

Ex: Algorithm applies past tense erroneously ($\hat{y}$)

Plug in part of human translated (y^*)

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

Beam search chose $\hat{y}$, tune Beam Search

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

Likely RNN at fault (maybe tune length in BS)

Try multiple $(\hat{y}, y^*)$ pairs & determine BS: RNN fail rate

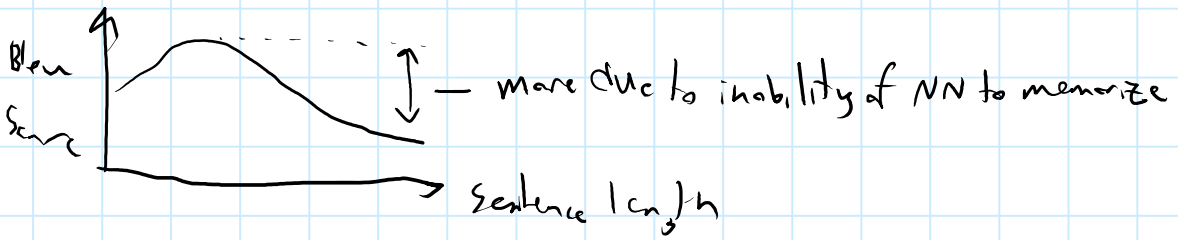
---

Bleu Score (optional) put later, as my interest in translation isn't THAT high

Attention Model Intuition

Issue with long sequences!

People do not memorize an entire sentence & then decode it



Bidirectional RNN on sentence $a^{<0>} \rightarrow \rightarrow \rightarrow$

another RNN $s^{<0>} \rightarrow \rightarrow \rightarrow$

$s^{<i>}$ should weight $x^{<i>}$ more heavily than $x^{<i \pm \text{large}>}$

Use weights α, $\alpha^{<i,t>}$

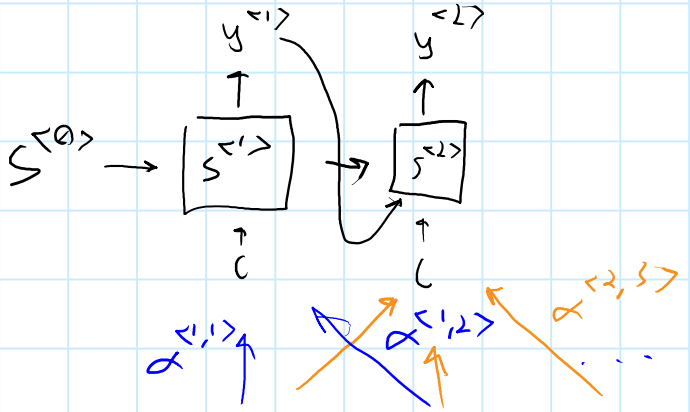
$s' \leftarrow x$

$\alpha^{<i,t>}$ will depend on $s^{<i>}$, $\overrightarrow{a}^{<t>}$, $\overleftarrow{a}^{<t>}$

Attention Model

Attention Model

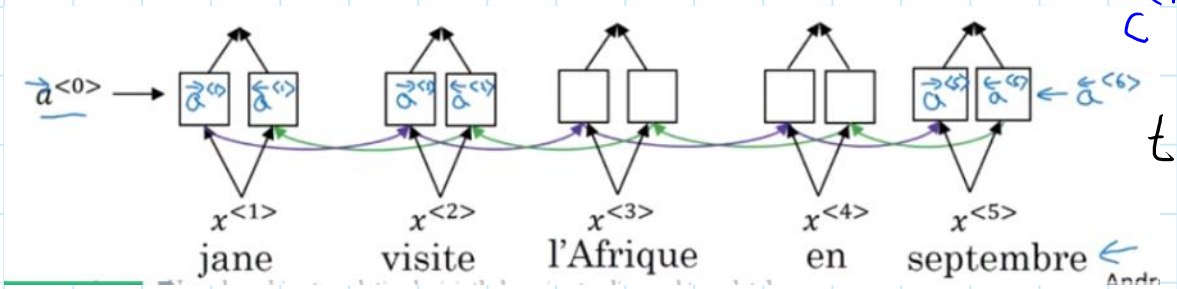
(Typically GRU/LSTM)



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

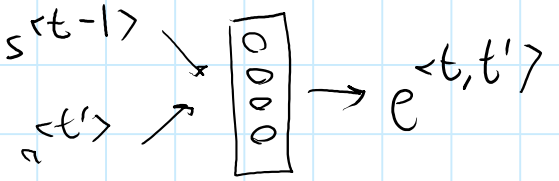
$\sum_{t'} \alpha^{<1,t'>} = 1$

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



t'

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



$\rightarrow e^{<t,t'>}$ intuition: previous hidden state should influence $\alpha^{<t,t'>}$

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

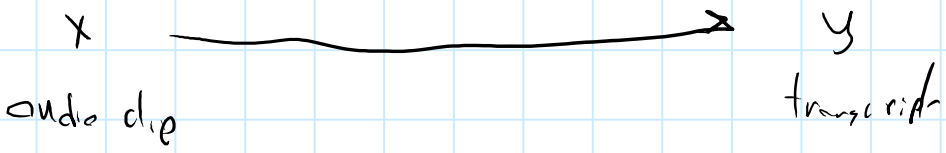
ensures α components Σ to 1

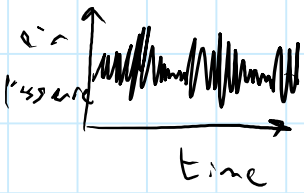
Downside: Quadratic cost, but sentences are short so typically acceptible

[This can also be used for image captioning]

Speech Recognition

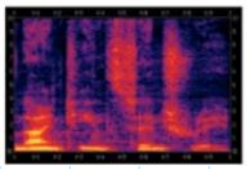
Friday, April 27, 2018 8:23 AM





↓

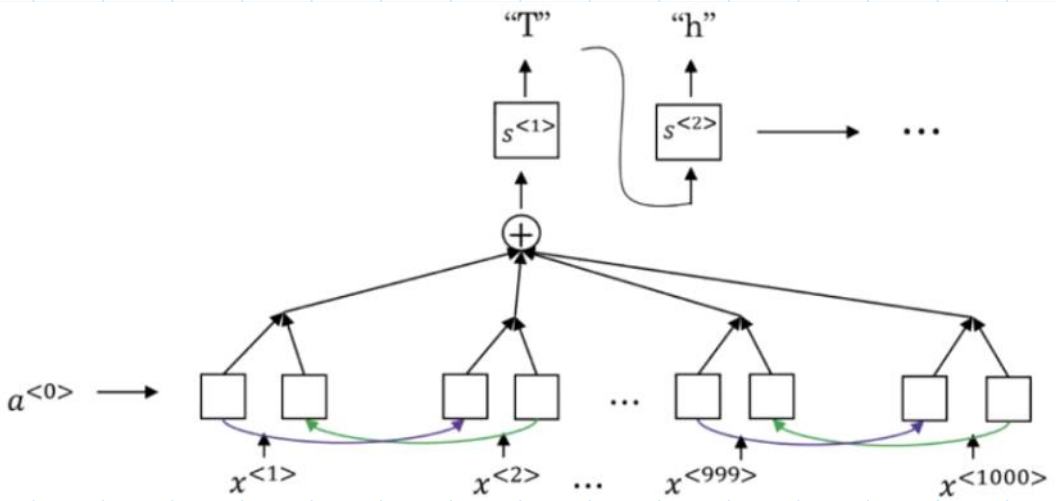
heatmap of
frequency vs time



'2018 1:19 PM

Screen clipping taken: 4/27/2018 1:20 PM

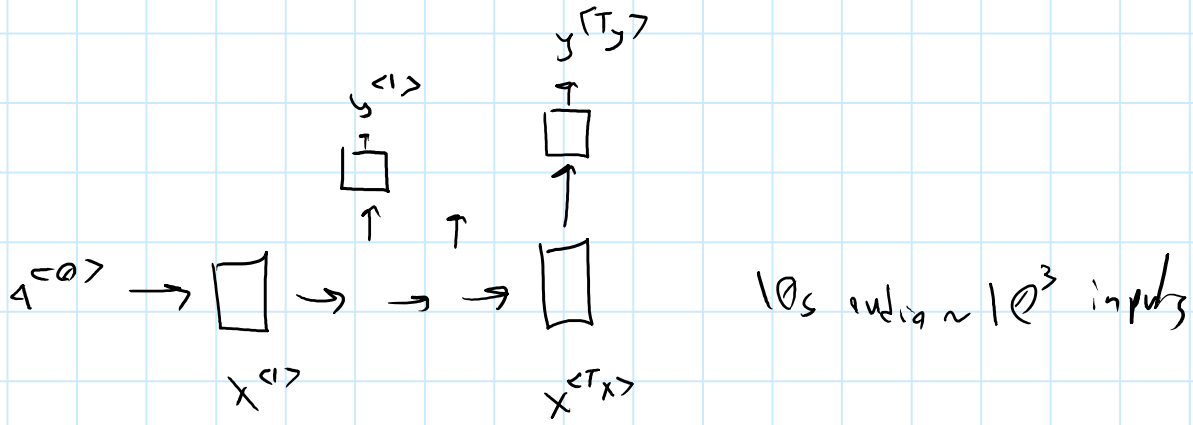
Attention model possible



CTC cost for speech recognition

(connectionist temporal classification)

B: RNN typically used



10s outputs ~ 10^3 inputs

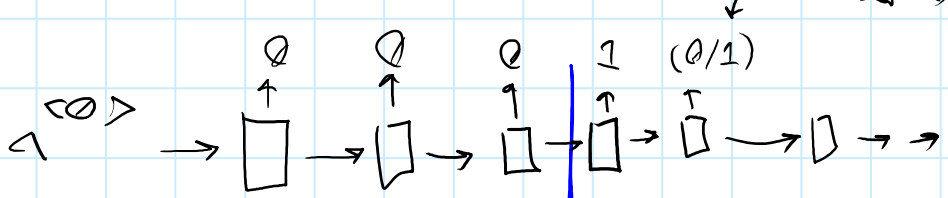
"the quick brown fox" 19 characters from 10^3 inputs

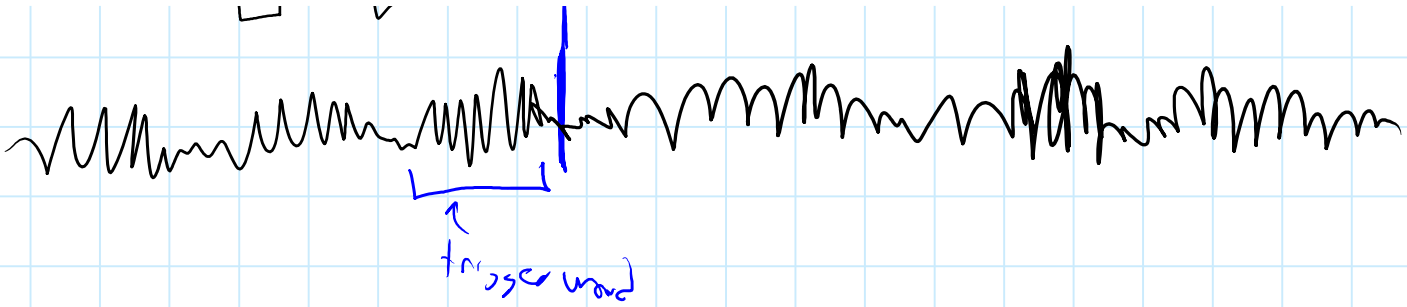
y: ttttttt_h_eeee____u____q__

Collapse repeated characters not separated by "blank" (_)

Trigger Word Detection

Algorithm not yet set in stone




frozen word