P9120 - Statistical Learning and Data Mining

Lecture 8 - Recurrent Neural Networks (RNN)

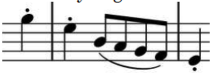
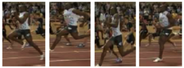
Min Qian

Department of Biostatistics, Columbia University

October 24th, 2024

Outline

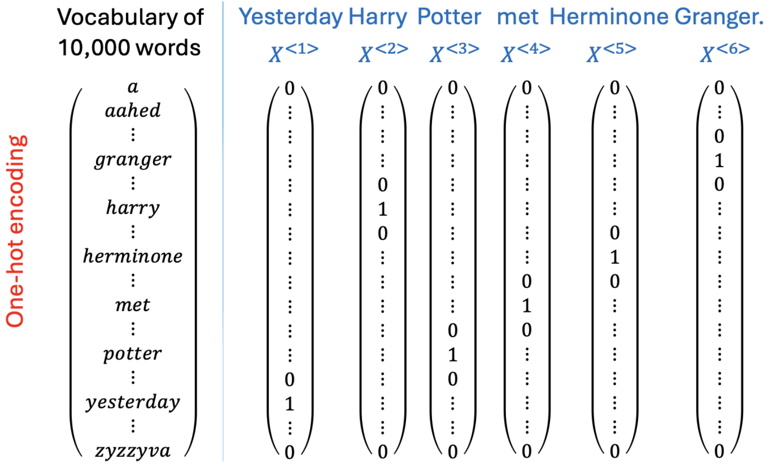
Examples of sequence data

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.

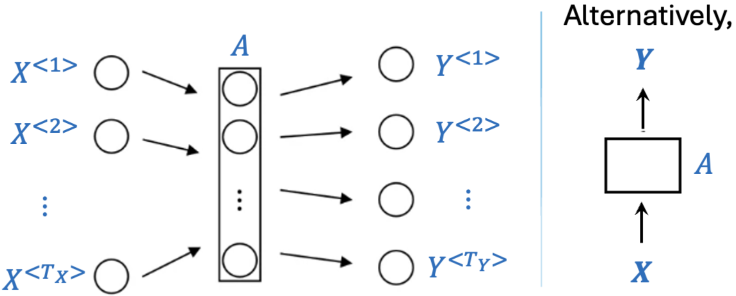
Andrew Ng

Name Entity Recognition

	Yesterday	Harry	Potter	met	Herminone	Granger.
Input X:	$X^{<1>}$	$X^{<2>}$	$\dots$	$X^{<t>}$	$\dots$	$X^{<T_\mathbf{x}>}$
	0	1	1	0	1	1
Output Y:	$Y^{<1>}$	$Y^{<2>}$	$\dots$	$Y^{<t>}$	$\dots$	$Y^{<T_\mathbf{Y}>}$

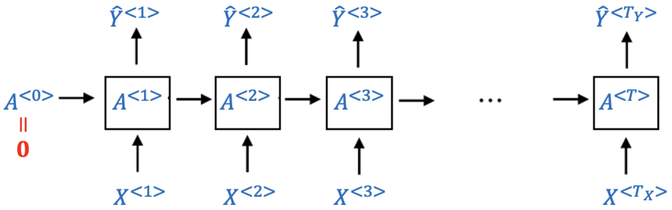


Why Not Standard NN



- Input/output can be of different lengths for different data points.
- Does not consider the sequential structure.

Scheme of Basic RNN (Many-to-Many, Same Length)



$X^{<t>} \in \mathbb{R}^p$
$A^{<t>} \in \mathbb{R}^K$
$g^{[1]}$ is tanh/ReLu

From $< t-1 >$ to $< t >$:

$$\hat{Y}^{<t>} = g^{[2]}\left(W_{ya}A^{<t>} + b_y\right)$$

$$A^{<t-1>} \longrightarrow \quad A^{<t>} = g^{[1]}(W_{aa}A^{<t-1>} + W_{ax}X^{<t>} + b_a)$$

$$X^{<t>}$$

Loss function:

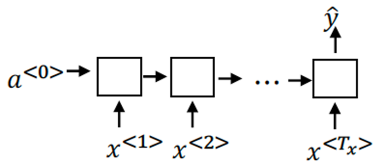
$$l(Y, \hat{Y}) = \sum_{t=1}^{T_Y} l^{<t>}\left(Y^{<t>}, \hat{Y}^{<t>}\right)$$

Parameters are learned using "backpropagation through time"

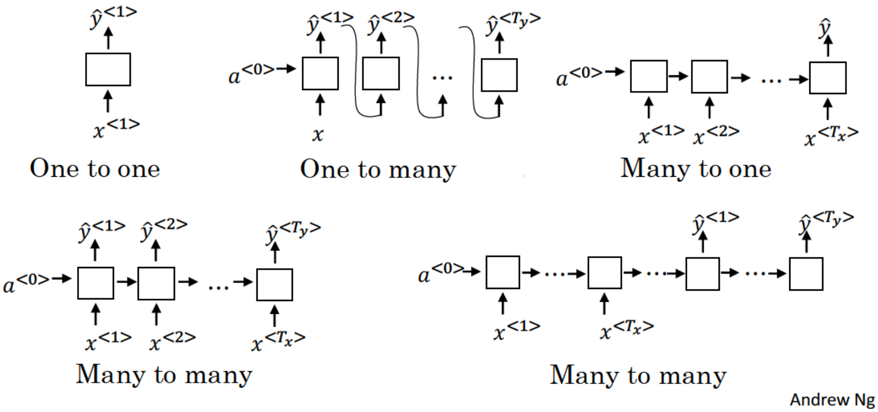
Sentiment Classification: IMDb ratings (Many-to-One)

- Input: review
  This has to be one of the worst films of the 1990s. When my friends & I were watching this film (being the target audience it was aimed at) we just sat & watched the first half an hour with our jaws touching the floor at how bad it really was. The rest of the time, everyone else in the theater just started talking to each other, leaving or generally crying into their popcorn . . .

- Output: sentiment of the review (positive/negative, rating 1-5).



Summary of RNN types



Language Models

A language model is a probabilistic model of a natural language used to predict and generate plausible texts.

Speech recognition example:

- Sentence 1: She will bring flowers.
- Sentence 2: She will bring flours.

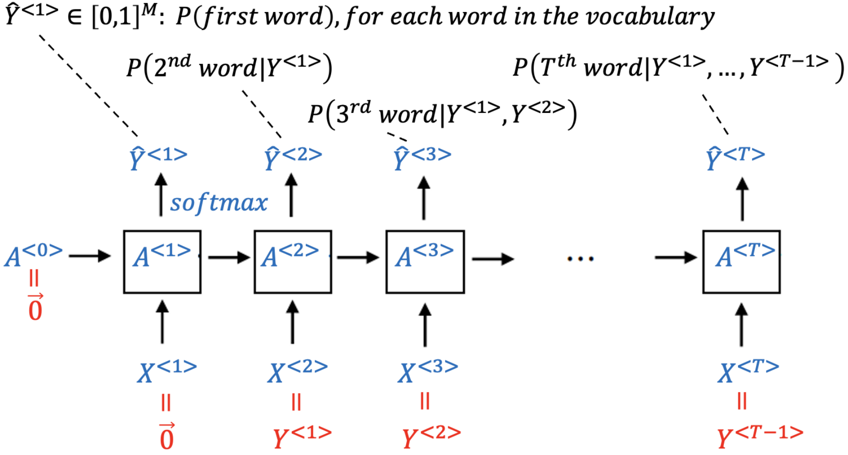
$P(sentence) = ?$

Training set: a large set (corpus) of English text.

She	will	bring	flowers	⟨EOS⟩
$Y^{<1>}$	$Y^{<2>}$	$Y^{<3>}$	$Y^{<4>}$	$Y^{<5>}$

- Can also add punctuation into the vocabulary dictionary.
- Add ⟨UNK⟩(unknown) into the vocabulary dictionary.

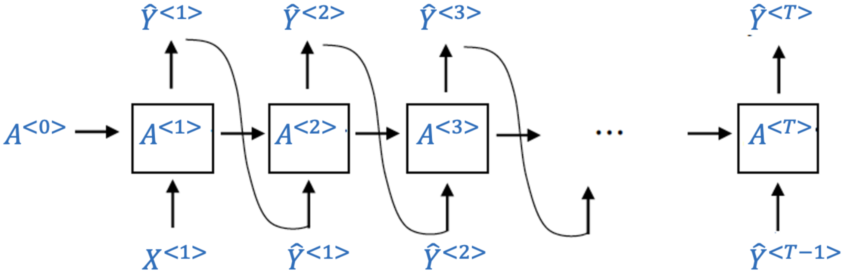
RNN model



Loss function: $L = \sum_{t=1}^{T} l(Y^{<t>}, \widehat{Y}^{<t>})$, where

$$l(Y^{<t>}, \widehat{Y}^{<t>}) = -\sum_{m=1}^{M} Y_m^{<t>}, \log(\widehat{Y}_m^{<t>})$$

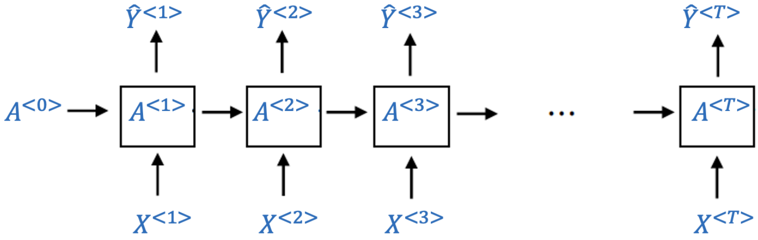
Sampling a Sequence from a Trained RNN



- First, randomly sample a word from the distribution of $\widehat{Y}^{<1>}$.
- For $t = 2, 3, \ldots,$, randomly sample a word using conditional distribution of $\widehat{Y}^{<t>}|$previously sampled $\widehat{Y}^{<1>}, \ldots, \widehat{Y}^{<t-1>}$.
- End the sentence if ⟨EOS⟩is sampled.
- If ⟨UNK⟩is sampled, can reject and resample.

Long Term Dependency in RNN

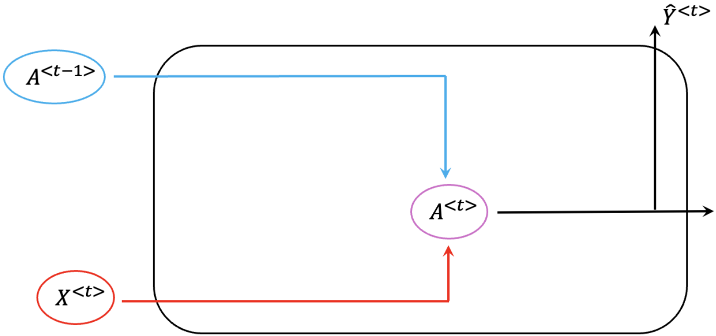
The dog, after playing in the park for a long time ..., sleeps soundly.
The dogs, after playing in the park for a long time ..., sleep soundly.



- RNN is a very deep NN.
- $\widehat{Y}^{<t>}$ mainly depends on near past information.
- Exploding gradients: gradient clipping. rescale gradient vectors if a gradient is above a threshold.
- Vanishing gradients?

Gated Recurrent Unit (Simplified)

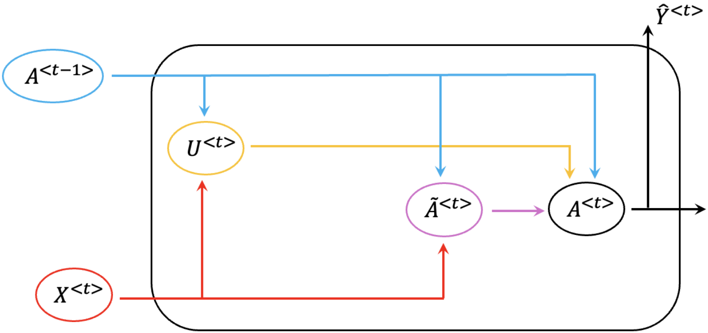
The dog, after playing in the park for a long time ..., sleeps soundly.



- Update gate: $U^{<t>} = \text{sigmoid}(w_{UA}A^{<t-1>} + w_{UX}X^{<t>} + b_U)$
- Current memory: $\tilde{A}^{<t>} = \tanh\left(w_{AA}A^{<t-1>} + w_{AX}X^{<t>} + b_A\right)$
- Output: $A^{<t>} = U^{<t>} \odot \tilde{A}^{<t>} + (1 - U^{<t>}) \odot A^{<t-1>}$

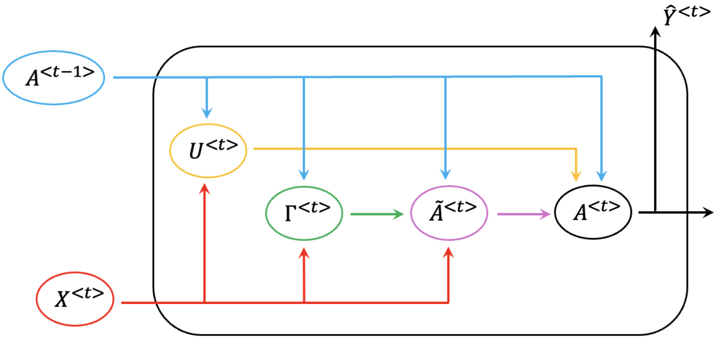
Gated Recurrent Unit (Simplified)

The dog, after playing in the park for a long time ..., sleeps soundly.



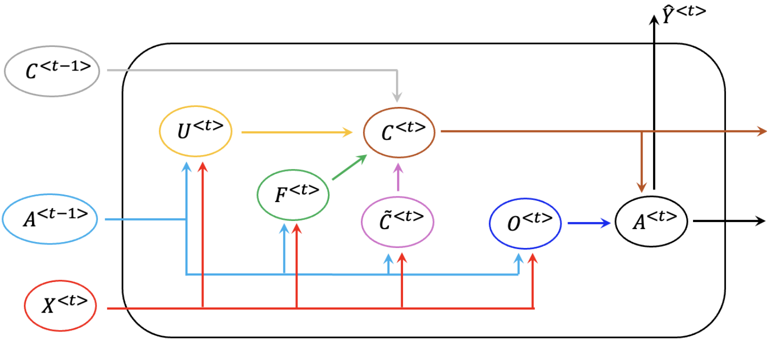
- Update gate: $U^{<t>} = \text{sigmoid}(w_{UA}A^{<t-1>} + w_{UX}X^{<t>} + b_U)$
- Current memory: $\tilde{A}^{<t>} = \tanh\left(w_{AA}A^{<t-1>} + w_{AX}X^{<t>} + b_A\right)$
- Output: $A^{<t>} = U^{<t>} \odot \tilde{A}^{<t>} + (1 - U^{<t>}) \odot A^{<t-1>}$

Full GRU (2014)



- Update gate: $U^{<t>} = \text{sigmoid}(w_{UA}A^{<t-1>} + w_{UX}X^{<t>} + b_U)$
- Reset gate: $\Gamma^{<t>} = \text{sigmoid}(w_{\Gamma A}A^{<t-1>} + w_{\Gamma X}X^{<t>} + b_\Gamma)$
- Current memory:
  $\tilde{A}^{<t>} = \tanh\big(w_{AA}(\Gamma^{<t>} \odot A^{<t-1>}) + w_{AX}X^{<t>} + b_A\big)$
- Output: $A^{<t>} = U^{<t>} \odot \tilde{A}^{<t>} + (1 - U^{<t>}) \odot A^{<t-1>}$

Long Short-Term Memory RNN (LSTM, 1997)



- Update gate: $U^{<t>} = \text{sigmoid}(w_{UA}A^{<t-1>} + w_{UX}X^{<t>} + b_U)$
- Forget gate: $F^{<t>} = \text{sigmoid}(w_{FA}A^{<t-1>} + w_{FX}X^{<t>} + b_F)$
- Current memory: $\widetilde{C}^{<t>} = \tanh(w_{CA}A^{<t-1>} + w_{CX}X^{<t>} + b_C)$
- Memory cell: $C^{<t>} = U^{<t>} \odot \widetilde{C}^{<t>} + F^{<t>} \odot C^{<t-1>}$
- Output gate: $O^{<t>} = \text{sigmoid}(w_{OA}A^{<t-1>} + w_{OX}X^{<t>} + b_O)$
- Output: $A^{<t>} = O^{<t>} \odot \tanh(C^{<t>})$

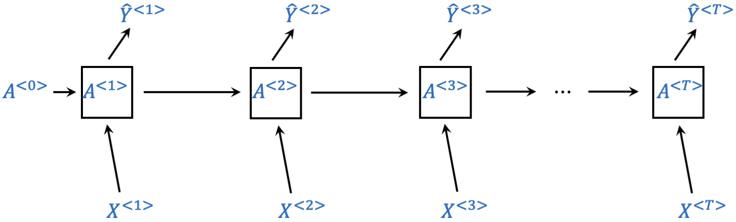
GRU vs LSTM

- Both GRU and LSTM are designed to catch long-term dependency.
- LSTM was invented much earlier than GRU.
- GRU is a simpler model than LSTM. Computationally faster.
- GRU gains more momentum in recent years. It often just works as well as LSTM.

Bi-directional RNN

Name entity recognition example:

1. I like apple, especially iphone.
2. I like apple. It is a healthy fruit.

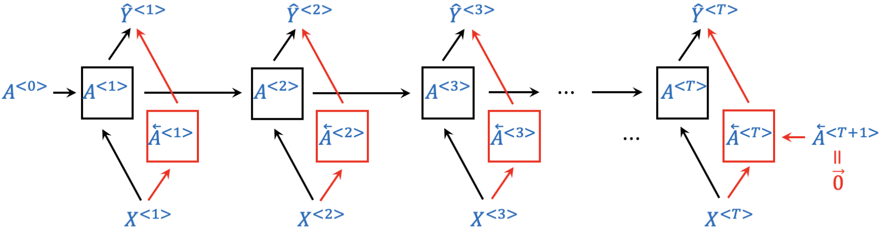


- BRNN + LSTM is commonly used for NLP problem.
- Need entire sequence of data for prediction.

Bi-directional RNN

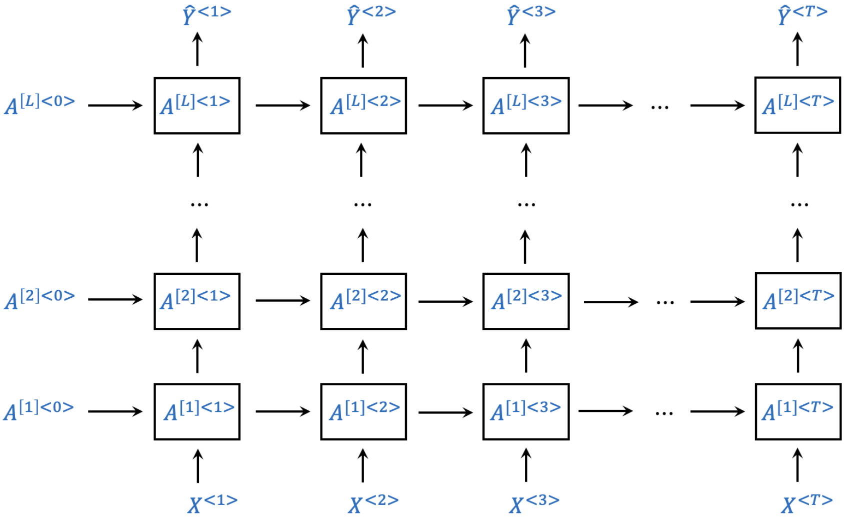
Name entity recognition example:

1. I like apple, especially iphone.
2. I like apple. It is a healthy fruit.



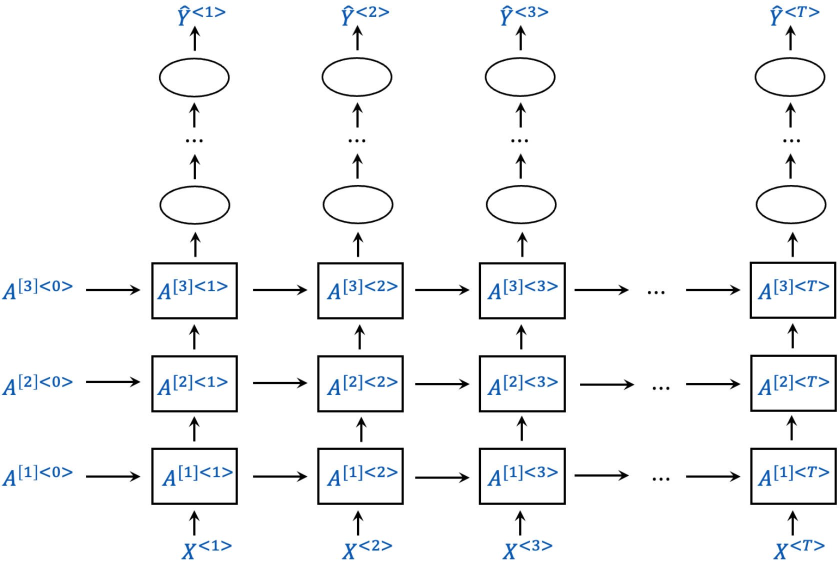
- BRNN + LSTM is commonly used for NLP problem.
- Need entire sequence of data for prediction.

Deep RNN



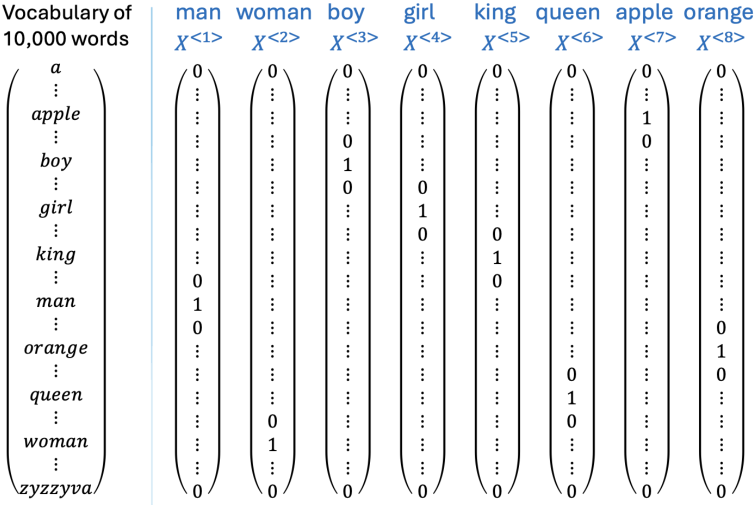
$\hat{Y}^{<1>}$
$\hat{Y}^{<2>}$
$\hat{Y}^{<3>}$
$\hat{Y}^{<T>}$
$A^{[L]<0>}$
$A^{[L]<1>}$
$A^{[L]<2>}$
$A^{[L]<3>}$
…
$A^{[L]<T>}$
…
$A^{[2]<0>}$
$A^{[2]<1>}$
$A^{[2]<2>}$
$A^{[2]<3>}$
$A^{[2]<T>}$
$A^{[1]<0>}$
$A^{[1]<1>}$
$A^{[1]<2>}$
$A^{[1]<3>}$
$A^{[1]<T>}$
$X^{<1>}$
$X^{<2>}$
$X^{<3>}$
$X^{<T>}$

Most Commonly used Deep RNN



Representing Words: One-hot Encoding

Each word is represented by a one-hot-encoded vector (i.e., dummy variable) with $9,999$ zeros and a single 1 in some position.



Word Embedding: Featurized Representation

- The dimension of one-hot-encoding is very high.
- One-hot-encoding does not capture the similarity/correlation among words.

Representing each word (10,000-*dim* vector) by a K-*dim* (e.g., 300) vector.

K rows

Features	man	woman	boy	girl	king	queen	apple	orange
Masculinity	0.97	0.05	0.7	0.1	0.95	0.02	0.5	0.45
Age	0.75	0.7	0.25	0.23	0.6	0.58	0.49	0.51
Royalty	0.05	0.05	0.01	0.03	0.96	0.95	0.53	0.48
Edible	0.01	0.01	0.02	0.01	0.02	0.01	0.99	0.90
⋮								
⋮								
Color								
Verb								
Size								

Word Embedding: Embedding Matrix

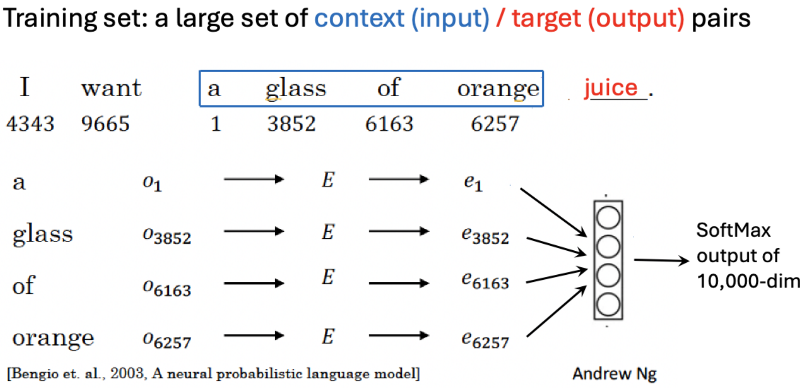
Matrix $\mathbf{E} \in \mathbb{R}^{K \times 10000}$: maps a $10,000$-dim vector to an K-dim vector.

10,000 columns

	a	...	apple	...	...	orange	...	zyzzyva
	0.1125	...	0.5	...	...	0.45	...	0.9295
	0.7606	...	0.49	...	...	0.51	...	0.6156
	0.5513	...	0.53	...	...	0.48	...	0.7775
	0.6219	...	0.99	...	...	0.90	...	0.235
K rows	⋮			⋮				⋮
	⋮			⋮				⋮
	⋮			⋮				⋮
	⋮			⋮				⋮
	⋮			⋮				⋮
	0.8586	...	0.15	...	...	0.13	...	0.9003

embedded vector $e_{word} = \mathbf{E} \bullet$ one-hot-vector o_{word}

Learn Embedding Matrix using NN

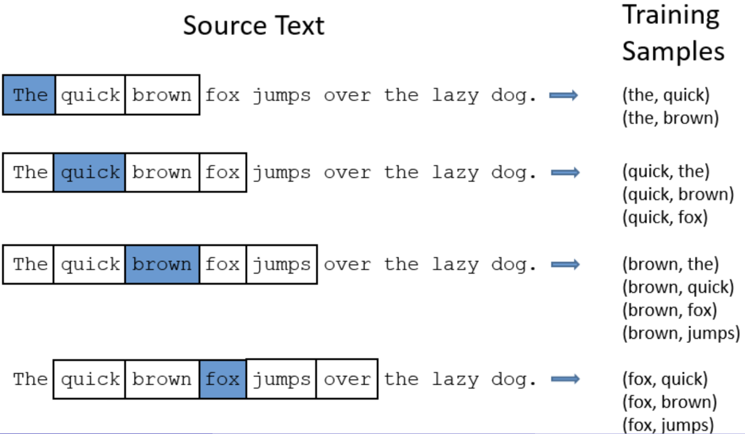


Various choices of context:

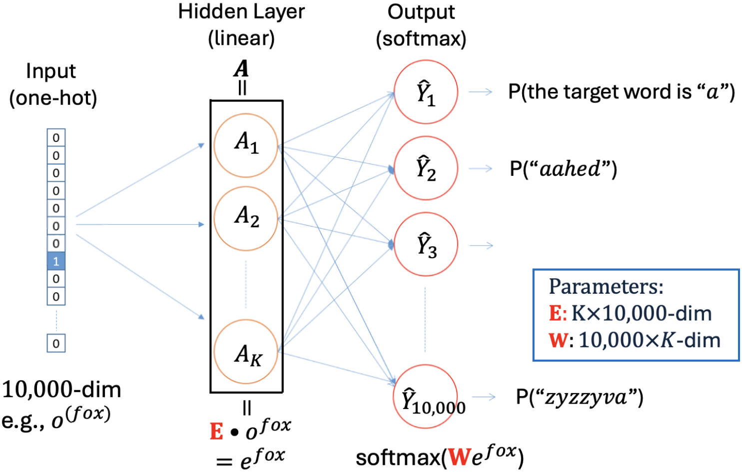
- Previous 4 words, previous 1 word, ...
- 4 words on left and 4 on right, e.g.,
  I want a glass of orange juice to pair with my meal.
- Nearby 1 word

Word2Vec: Skip-Grams Training Data

- Idea: the words that appear in the same context (near each other) should have similar word vectors.
- Use an input word to predict surrounding words.
- Random pick a context (input) word and randomly pick a target (output) word within $\pm x$ (e.g., 5) window of the context word.



The Skip-Gram Model



- The softmax step is very expensive to calculate
- Hierarchical classification is often used.
- Context words are sampled to balance common and less common words.

Negative Sampling

e.g., The quick brown fox jumps over the lazy dog.

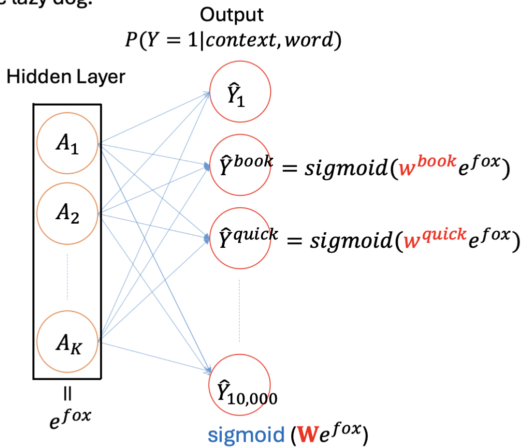
	context	word	target?
0	fox	quick	1
1	fox	book	0
2	fox	school	0
⋮	fox	phone	0
⋮	fox	pencil	0
J	fox	juice	0

Loss function:

$$\sum_{j=0}^{J}\left[-Y^{(j)}\log\hat{Y}^{(j)}-(1-Y^{(j)})\log(1-\hat{Y}^{(j)})\right]$$

$$=-\log\hat{Y}^{(0)}-\sum_{j=1}^{J}\log(1-\hat{Y}^{(j)})$$

Only involves rows in **W** for the $J+1$ words.



- For each positive example, randomly choose J negative words. $J = 5-20$ for smaller datasets, and $J = 2-5$ for larger datasets.
- sample negative target word w_i w.p. $\frac{P(w_i)^{3/4}}{\sum_{j=1}^{10000} P(w_j)^{3/4}}$.

Global Vectors for Word Representation (GloVe)

- For each pair of words i and j in the vocabulary,
  X_{ij} = frequency that pair (i, j) is a pair of (context, target).
- Parameters:
  $\mathbf{E} \in \mathbb{R}^{K\times 10{,}000}$: each column e_i is a word vector.
  $\mathbf{W} \in \mathbb{R}^{K\times 10{,}000}$: each column w_j is another word vector.
- Estimate parameters $\{(e_i, w_j, b_{e,i}, b_{w,j}) : i, j = 1, \ldots, 10000\}$:

$$\min \sum_{i=1}^{10{,}000} \sum_{j=1}^{10{,}000} f(X_{ij})(e_i^\mathsf{T} w_j + b_{e,i} + b_{w,j} - \log X_{ij})^2,$$

  - $f(0) = 0$, by default $0 \log 0 = 0$.
  - $f(x)$ is non-decreasing
  - $f(x)$ is not too large for very large values of x.
- Final embedding vector for each word: $(\hat{e}_i + \hat{w}_i)/2$.

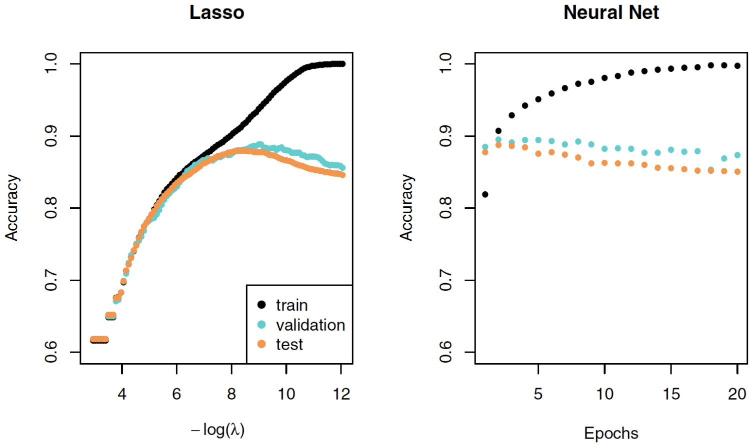
Sentiment Classification: IMDb ratings

- Input: review
  This has to be one of the worst films of the 1990s. When my friends & I were watching this film (being the target audience it was aimed at) we just sat & watched the first half an hour with our jaws touching the floor at how bad it really was. The rest of the time, everyone else in the theater just started talking to each other, leaving or generally crying into their popcorn . . .

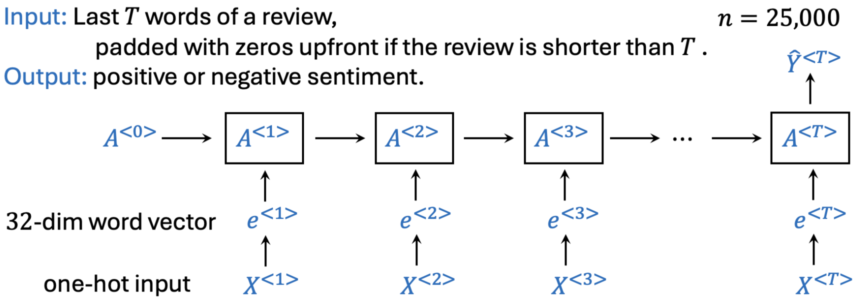
- Response: sentiment of the review (positive/negative)

Bag-of-words Model (ignore sequence of data)

- Consider a vocabulary of M (say 10,000) words.
- For each review, the input is a binary vector of length $10,000$ indicating whether a word is present or not.
- $n = 25,000$ for training and testing sets.



IMDb ratings: "Entry Level" RNN models



- Fit Simple RNN with $K = 32$ hidden units, train embedding matrix. The test set accuracy is 76%.
- With LSTM RNN, the test set accuracy is increased to 87%.
- As of now, the leading RNN configurations report accuracy above 95% on the IMDb data.

Summary

- RNN is a flexible tool for prediction with sequence input and/or output.
- GRU and LSTM are designed to catch long-term dependency.
- BRNN is designed to learn dependency of later information.
- Word embedding: map one-hot-vector to a dense vector.
  - learn using your training data
  - learn from a large text corpus.
  - use pre-trained embedding.

Reminder:

- Quiz for lecture 8 is due at 9pm on Monday Oct. 28th.
- Hw #2 is due at 9pm on Saturday Oct. 26th.