Language Models

Dan Klein
UC Berkeley

1

Language Models

2

Language Models

3

Acoustic Confusions

the station signs are in deep in english	-14732
the stations signs are in deep in english	-14735
the station signs are in deep into english	-14739
the station 's signs are in deep in english	-14740
the station signs are in deep in the english	-14741
the station signs are indeed in english	-14757
the station 's signs are indeed in english	-14760
the station signs are indians in english	-14790

4

Noisy Channel Model: ASR

- We want to predict a sentence given acoustics:

$$w^* = \arg\max_w P(w|a)$$

- The noisy-channel approach:

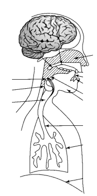
$$w^* = \arg\max_w P(w|a)$$

$$= \arg\max_w P(a|w)P(w)/P(a)$$

$$\propto \arg\max_w P(a|w)P(w)$$

Acoustic model: score fit between sounds and words

Language model: score plausibility of word sequences



5

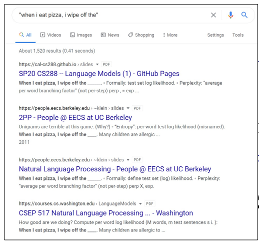
Noisy Channel Model: Translation

"Also knowing nothing official about, but having guessed and inferred considerable about, the powerful new mechanized methods in cryptography—methods which I believe succeed even when one does not know what language has been coded—one naturally wonders if the problem of translation could conceivably be treated as a problem in cryptography. When I look at an article in Russian, I say: 'This is really written in English, but it has been coded in some strange symbols. I will now proceed to decode.' "

Warren Weaver (1947)

6

Perplexity



ctor" (not per-step)

grease 0.5
sauce 0.4
dust 0.05
....
mice 0.0001
....
the 1e-100

3516 wipe off the excess
1034 wipe off the dust
547 wipe off the sweat
518 wipe off the mouthpiece
...
120 wipe off the grease
0 wipe off the sauce
0 wipe off the mice

28048 wipe off the *

7

N-Gram Models

8

N-Gram Models

- Use chain rule to generate words left-to-right

$$P(w_1 \dots w_n) = \prod_i P(w_i | w_1 \dots w_{i-1})$$

- Can't condition atomically on the entire left context

P(??? | The computer I had put into the machine room on the fifth floor just)

- N-gram models make a Markov assumption

$$P(w_1 \dots w_n) = \prod_i P(w_i | w_{i-k} \dots w_{i-1})$$

$$P(\text{please close the door}) = P(\text{please}|\text{START})P(\text{close}|\text{please}) \dots P(\text{STOP}|door)$$

9

Empirical N-Grams

- Use statistics from data (examples here from Google N-Grams)

Training Counts

198015222 the first
194623024 the same
168504105 the following
158562063 the world
...
14112454 the door

23135851162 the *

$$\hat{P}(\text{door}|\text{the}) = \frac{14112454}{23135851162} = 0.0006$$

- This is the maximum likelihood estimate, which needs modification
- N-gram models use such counts to compute probabilities on demand

10

Increasing N-Gram Order

- Higher orders capture more correlations

Bigram Model

198015222 the first
194623024 the same
168504105 the following
158562063 the world
...
14112454 the door

23135851162 the *

Trigram Model

197302 close the window
191125 close the door
152500 close the gap
116451 close the thread
87298 close the deal

3785230 close the *

P(door | the) = 0.0006

P(door | close the) = 0.05

11

Increasing N-Gram Order

Unigram

- To him swallowed confess hear both. Which. Of save on trail for are ay device and rote life have
- Every enter now severally so, let
- Hill he late speaks; or! a more to leg less first you enter
- Are where exeunt and sighs have rise excellency took of.. Sleep knave we. near; vile like

12

What's in an N-Gram?

- Just about every local correlation!
  - Word class restrictions: "will have been ___"
  - Morphology: "she ___", "they ___"
  - Semantic class restrictions: "danced a ___"
  - Idioms: "add insult to ___"
  - World knowledge: "ice caps have ___"
  - Pop culture: "the empire strikes ___"
- But not the long-distance ones
  - "The computer which I had put into the machine room on the fifth floor just ___."

13

Linguistic Pain

- The N-Gram assumption hurts your inner linguist
  - There are many linguistic arguments that language isn't regular
    - Long-distance dependencies
    - Recursive structure
  - At the core of the early hesitance in linguistics about statistical methods
- Answers
  - N-grams only model local correlations… but they get them all
  - As N increases, they catch even more correlations
  - N-gram models scale well -- much more easily than combinatorially-structured LMs
  - Can build LMs from structured models, eg grammars (though people generally don't)

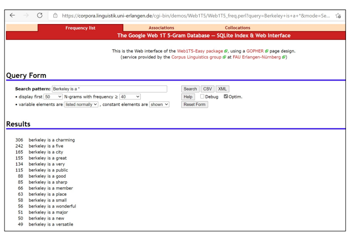
14

Structured Language Models

- Bigram model:
  - [texaco, rose, one, in, this, issue, is, pursuing, growth, in, a, boiler, house, said, mr., gurria, mexico, 's, motion, control, proposal, without, permission, from, five, hundred, fifty, five, yen]
  - [outside, new, car, parking, lot, of, the, agreement, reached]
  - [this, would, be, a, record, november]
- PCFG model:
  - [This, quarter, 's, surprisingly, independent, attack, paid, off, the, risk, involving, IRS, leaders, and, transportation, prices, .]
  - [It, could, be, announced, sometime, .]
  - [Mr., Toseland, believes, the, average, defense, economy, is, drafted, from, slightly, more, than, 12, stocks, .]

15

N-Grams on the Web



16

N-Gram Models: Challenges

17

Sparsity

Please close the first door on the left.

3380 please close the door
1601 please close the window
1164 please close the new
1159 please close the gate
…
0 please close the first

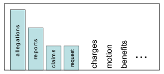
13951 please close the *

18

Smoothing

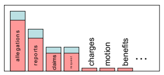
- We often want to make estimates from sparse statistics:

P(w | denied the)
3 allegations
2 reports
1 claims
1 request
7 total



- Smoothing flattens spiky distributions so they generalize better:

P(w | denied the)
2.5 allegations
1.5 reports
0.5 claims
0.5 request
2 other
7 total



- Very important all over NLP, but easy to do badly

19

Back-off

Please close the first door on the left.

4-Gram	3-Gram	2-Gram
3380 please close the door 1601 please close the window 1164 please close the new 1159 please close the gate … 0 please close the first ----------------- 13951 please close the *	197302 close the window 191125 close the door 152500 close the gap 116451 close the thread … 8662 close the first ----------------- 3785230 close the *	198015222 the first 194623024 the same 168504105 the following 158562063 the world … … ----------------- 23135851162 the *
0.0	0.002	0.009

Specific but Sparse ⟷ Dense but General

$$\lambda \hat{P}(w|w_{-1}, w_{-2}) + \lambda' \hat{P}(w|w_{-1}) + \lambda'' \hat{P}(w)$$

20

Discounting

- Observation: N-grams occur more in training data than they will later

Empirical Bigram Counts (Church and Gale, 91)

Count in 22M Words	Future c* (Next 22M)
1	
2	
3	
4	
5	

- Absolute discounting: reduce counts by a small constant, redistribute "shaved" mass to a model of new events

$$P_{\text{ad}}(w|w') = \frac{c(w', w) - d}{c(w')} + \alpha(w')\hat{P}(w)$$

21

Fertility

- Shannon game: "There was an unexpected _____"

delay? Francisco?

- Context fertility: number of distinct context types that a word occurs in
  - What is the fertility of "delay"?
  - What is the fertility of "Francisco"?
  - Which is more likely in an arbitrary new context?
- Kneser-Ney smoothing: new events proportional to context fertility, not frequency

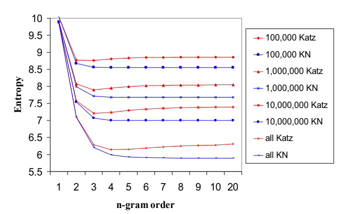
[Kneser & Ney, 1995]

$$P(w) \propto |\{w' : c(w', w) > 0\}|$$

- Can be derived as inference in a hierarchical Pitman-Yor process [Teh, 2006]

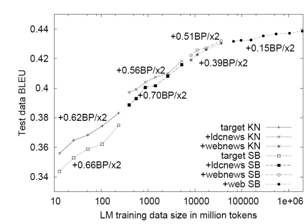
22

Better Methods?



23

More Data?



[Brants et al, 2007]

24

Storage

...	
searching for the best	192593
searching for the right	45805
searching for the cheapest	44965
searching for the perfect	43959
searching for the truth	23165
searching for the “	19086
searching for the most	15512
searching for the latest	12670
searching for the next	10120
searching for the lowest	10080
searching for the name	8402
searching for the finest	8171
...	

Google N-grams
- 14 million < 2^{24} words
- 2 billion < 2^{31} 5-grams
- 770 000 < 2^{20} unique counts
- 4 billion n-grams total

25

Storage

- For 5+-gram models, need to store between 100M and 10B context-word-count triples

(a) Context-Encoding			(b) Context Deltas			(c) Bits Required		
w	c	val	Δw	Δc	val	$\|\Delta w\|$	$\|\Delta c\|$	$\|val\|$
1933	15176585	3	1933	15176585	3	24	40	3
1933	15176587	2	+0	+2	1	2	3	3
1933	15176593	1	+0	+5	1	2	3	3
1933	15176613	8	+0	+40	8	2	9	6
1933	15179801	1	+0	+188	1	2	12	3
1935	15176585	298	+2	15176585	298	4	36	15
1935	15176589	1	+0	+4	1	2	6	3

- Make it fit in memory by *delta encoding* scheme: store deltas instead of values and use variable-length encoding

Pauls and Klein (2011), Heafield (2011)

Slide: Greg Durrett

26

Graveyard of Correlations

- Skip-grams
- Cluster models
- Topic variables
- Cache models
- Structural zeros
- Dependency models
- Maximum entropy models
- Subword models
- …

27

Entirely Unseen Words

- What about totally unseen words?
- Classical real world option: systems are actually closed vocabulary
  - ASR systems will only propose words that are in their pronunciation dictionary
  - MT systems will only propose words that are in their phrase tables (modulo special models for numbers, etc)
- Classical theoretical option: build open vocabulary LMs
  - Models over character sequences rather than word sequences
  - N-Grams: back-off needs to go down into a “generate new word” model
  - Typically if you need this, a high-order character model will do
- Modern approach: syllable-sized subword units (more later)

30

Neural LMs: Preview

31

A GPT2 Continuation

- **The computer I had put into the machine room on the fifth floor just just** outside our landing was taken by a lot of people. It was going to be recovered from outside the machine room as soon as we could, but after the machine room was shut down, we had to open all of the windows and doors to save ourselves, and the computer would have been lost. Luckily we are prepared for this. We had five steel doors on each side of the landing to hold any rain and wind that might have hit, so we weren't affected by the storm.

  The situation was really grim for days. A few days before, the Apache at Qush Tefah had been shot. This was the most damaged machine in the Armies Corps. The command was only meant to repair Qush

32

A GPT2 Continuation

- **I shall go seek the traitor Gloucester**, when he hath declared his vengeance. Cunegund Shall I back her to kill my cousin in my stead? Witbane Far first shall the brother take his Queen's soul and that in the flames. Clotilda For to slay the King of Scotland with, the blood of my cousin, lie true; And she shall have the right to my feather." Sefton leapt into the rushes, and stole along in the sunlight: the small figure flashed like lightning in the west. In the likeness of a snare he had laid to catch the rushes and made of them a snares, a road to flee from his pursuers; but he now came to an oak where the branches were wreathed in an oak-

33

Neural LMs: Three Key Ideas

- Word embeddings
  - Different words are not entirely unrelated events
  - Words can be more and less similar, in complex ways
- Partially factored representations
  - Multiple semi-independent processes happen in parallel in language
  - It's too expensive to track language in an unfactored way, and too inaccurate to assume everything of interest is independent
- Long distance dependencies
  - Information can be relevant without being local
  - Different notions of locality are important at different times

34

Words: Clusterings and Embeddings

35

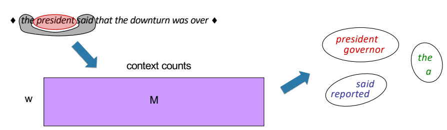
Stuffing Words into Vector Spaces?

Cartoon: Greg Durrett

36

Distributional Similarity

- Key idea in clustering and embedding methods: characterize a word by the words it occurs with (cf Harris' distributional hypothesis, 1954)
- "You can tell a word by the company it keeps." [Firth, 1957]
- Harris / Chomsky divide in linguistic methodology



37

Clusterings

38

Clusterings

- Automatic (Finch and Chater 92, Shuetze 93, many others)

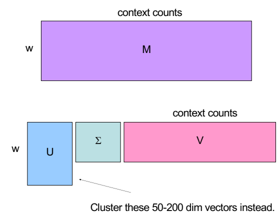
word	nearest neighbors
accompanied	submitted banned financed developed authorized headed canceled awarded barred
almost	virtually merely formally fully quite officially just nearly only less
causing	reflecting forcing providing creating producing becoming carrying particularly
classes	elections courses payments losses computers performances violations levels pictures
directors	professionals investigations materials competitors agreements papers transactions
goal	mood roof eye image tool song pool scene gap voice
japanese	chinese iraqi american western arab foreign european federal soviet indian
represent	reveal attend deliver reflect choose contain impose manage establish retain
think	believe wish know realize wonder assume feel say mean bet
york	angeles francisco sox rouge kong diego zone vegas inning layer
on	through in at over into with from for by across
must	might would could cannot will should can may does helps
they	we you i he she nobody who it everybody there

- Manual (e.g. thesauri, WordNet)

39

"Vector Space" Methods

- Treat words as points in R^n (eg Shuetze, 93)
  - Form matrix of co-occurrence counts
  - SVD or similar to reduce rank (cf LSA)
  - Cluster projections
  - People worried about things like: log of counts, U vs UΣ
- This is actually more of an embedding method (but we didn't want that in 1993)



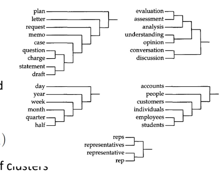
40

Models: Brown Clustering

- Classic model-based clustering (Brown et al, 92)
  - Each word starts in its own cluster
  - Each cluster has co-occurrence stats
  - Greedily merge clusters based on a mutual information criterion
  - Equivalent to optimizing a class-based bigram LM.

$$P(w_i|w_{i-1}) = P(c_i|c_{i-1})P(w_i|c_i)$$

  - Produces a dendrogram (hierarchy) of clusters



41

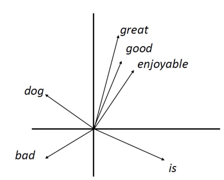
Embeddings

Most slides from Greg Durrett

42

Embeddings

- Embeddings map discrete words (eg |V| = 50k) to continuous vectors (eg d = 100)
- Why do we care about embeddings?
  - Neural methods want them
  - Nuanced similarity possible; generalize across words
- We hope embeddings will have structure that exposes word correlations (and thereby meanings)

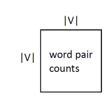


43

Embedding Models

- Idea: compute a representation of each word from co-occurring words

the dog bit the man



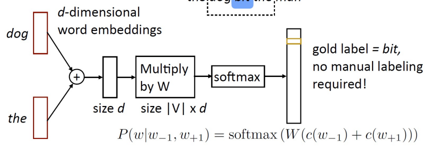
Token-Level *Type-Level*

- We'll build up several ideas that can be mixed-and-matched and which frequently get used in other contexts

44

word2vec: Continuous Bag-of-Words

- Predict word from context *the dog* **bit** *the man*



$P(w|w_{-1}, w_{+1}) = \text{softmax}\left(W(c(w_{-1}) + c(w_{+1}))\right)$

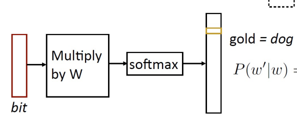
- Parameters: d x |V| (one d-length context vector per voc word), |V| x d output parameters (W)

Mikolov et al. (2013)

45

word2vec: Skip-Grams

- Predict one word of context from word *the dog* **bit** *the man*



$P(w'|w) = \text{softmax}(We(w))$

- Another training example: *bit* -> *the*
- Parameters: d x |V| vectors, |V| x d output parameters (W) (also usable as vectors!)

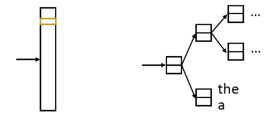
Mikolov et al. (2013)

46

word2vec: Hierarchical Softmax

$P(w|w_{-1}, w_{+1}) = \text{softmax}\left(W(c(w_{-1}) + c(w_{+1}))\right)$ $\quad P(w'|w) = \text{softmax}(We(w))$

- Matmul + softmax over |V| is very slow to compute for CBOW and SG



- Huffman encode vocabulary, use binary classifiers to decide which branch to take
- log(|V|) binary decisions
- Standard softmax: [|V| x d] x d
- Hierarchical softmax: log(|V|) dot products of size d, |V| x d parameters

Mikolov et al. (2013)

47

word2vec: Negative Sampling

- Take (word, context) pairs and classify them as "real" or not. Create random negative examples by sampling from unigram distribution

(*bit, the*) => +1
(*bit, cat*) => -1
(*bit, a*) => -1
(*bit, fish*) => -1

$$P(y=1|w,c) = \frac{e^{w \cdot c}}{e^{w \cdot c} + 1}$$

words in similar contexts select for similar c vectors

- d x |V| vectors, d x |V| context vectors (same # of params as before)
- Objective = $\log P(y=1|w,c) + \frac{1}{k}\sum_{i=1}^{n} \log P(y=0|w_i, c)$ sampled

Mikolov et al. (2013)

48

fastText: Character-Level Models

- Same as SGNS, but break words down into n-grams with n = 3 to 6

where:
3-grams: <wh, whe, her, ere, re>
4-grams: <whe, wher, here, ere>,
5-grams: <wher, where, here>,
6-grams: <where, where>

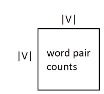
- Replace $w \cdot c$ in skip-gram computation with $\left(\sum_{g \in \text{ngrams}} w_g \cdot c\right)$
- Advantages?

Bojanowski et al. (2017)

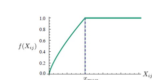
49

GloVe

- Idea: Fit co-occurrence matrix directly (weighted least squares)



$$J = \sum_{i,j=1}^{V} f(X_{ij})\left(w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij}\right)^2$$



- Type-level computations (so constant in data size)
- Currently the most common word embedding method

Pennington et al, 2014

50

Bottleneck vs Co-occurrence

- Two main views of inducing word structure
  - Co-occurrence: model which words occur in similar contexts
  - Bottleneck: model latent structure that mediates between words and their behaviors
- These turn out to be closely related!

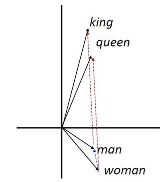
51

Language Models

52

Structure of Embedding Spaces

- How can you fit 50K words into a 64-dimensional hypercube?
- Orthogonality: Can each axis have a global "meaning" (number, gender, animacy, etc)?
- Global structure: Can embeddings have algebraic structure (eg king – man + woman = queen)?



53

Bias in Embeddings

- Embeddings can capture biases in the data! (Bolukbasi et al 16)

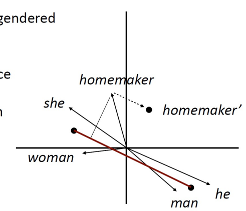
$$\overrightarrow{\text{man}} - \overrightarrow{\text{woman}} \approx \overrightarrow{\text{king}} - \overrightarrow{\text{queen}}$$

- Debiasing methods (as in Bolukbasi et al 16) are an active area of research

54

Debiasing?

- Identify gender subspace with gendered words
- Project words onto this subspace
- Subtract those projections from the original word



Bolukbasi et al. (2016)

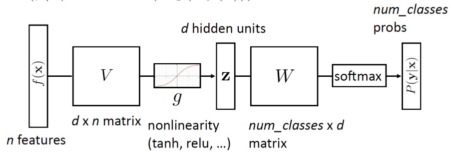
55

Neural Language Models

56

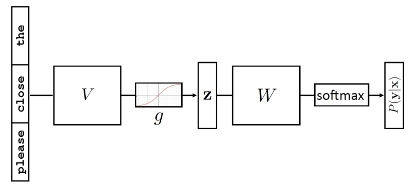
Reminder: Feedforward Neural Nets

$P(\mathbf{y}|\mathbf{x}) = \mathrm{softmax}(W g(V f(\mathbf{x})))$



57

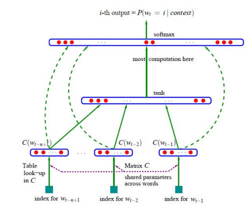
A Feedforward N-Gram Model?



58

Early Neural Language Models

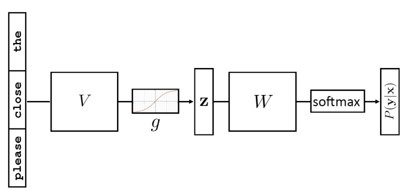
- Fixed-order feed-forward neural LMs
  - Eg Bengio et al 03
  - Allow generalization across contexts in more nuanced ways than prefixing
  - Allow different kinds of pooling in different contexts
  - Much more expensive to train



Bengio et al 03

59

Using Word Embeddings?



60

Using Word Embeddings

- Approach 1: learn embeddings as parameters from your data
  - Often works pretty well
- Approach 2: initialize using GloVe, keep fixed
  - Faster because no need to update these parameters
- Approach 3: initialize using GloVe, fine-tune
  - Works best for some tasks

61

Limitations of Fixed-Window NN LMs?

- What have we gained over N-Grams LMs?
- What have we lost?
- What have we not changed?

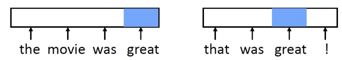
62

Recurrent NNs

63

RNNs

- Feedforward NNs can't handle variable length input: each position in the feature vector has fixed semantics



- These don't look related (*great* is in two different orthogonal subspaces)
- Instead, we need to:

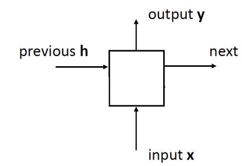
1) Process each word in a uniform way

2) ...while still exploiting the context that that token occurs in

64

General RNN Approach

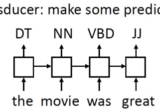
- Cell that takes some input **x**, has some hidden state **h**, and updates that hidden state and produces output **y** (all vector-valued)



65

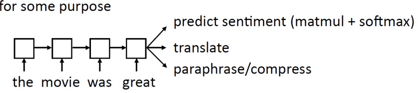
RNN Uses

- Transducer: make some prediction for each element in a sequence



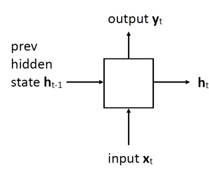
output **y** = score for each tag, then softmax

- Acceptor/encoder: encode a sequence into a fixed-sized vector and use that for some purpose



66

Basic RNNs



$$\mathbf{h}_t = \tanh(W\mathbf{x}_t + V\mathbf{h}_{t-1} + \mathbf{b}_h)$$

- Updates hidden state based on input and current hidden state

$$\mathbf{y}_t = \tanh(U\mathbf{h_t} + \mathbf{b}_y)$$

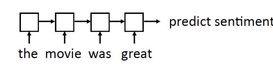
- Computes output from hidden state

- Long history! (invented in the late 1980s)

Elman (1990)

67

Training RNNs



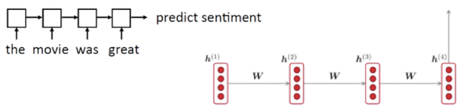
- "Backpropagation through time": build the network as one big computation graph, some parameters are shared
- RNN potentially needs to learn how to "remember" information for a long time!

it was my favorite movie of 2016, though it wasn't without problems -> +

- "Correct" parameter update is to do a better job of remembering the sentiment of *favorite*

68

Problem: Vanishing Gradients



- Contribution of earlier inputs decreases if matrices are contractive (first eigenvalue < 1), non-linearities are squashing, etc
- Gradients can be viewed as a measure of the effect of the past on the future
- That's a problem for optimization but also means that information naturally decays quickly, so model will tend to capture local information

Next slides adapted from Abigail See / Stanford

69

Core Issue: Information Decay

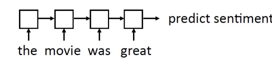
- The main problem is that *it's too difficult for the RNN to learn to preserve information over many timesteps.*
- In a vanilla RNN, the hidden state is constantly being rewritten

$$\boldsymbol{h}^{(t)} = \sigma\left(\boldsymbol{W}_h \boldsymbol{h}^{(t-1)} + \boldsymbol{W}_x \boldsymbol{x}^{(t)} + \boldsymbol{b}\right)$$

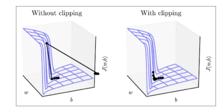
- How about a RNN with separate memory?

70

Problem: Exploding Gradients

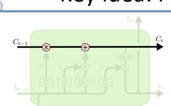


- Gradients can also be too large
  - Leads to overshooting / jumping around the parameter space
  - Common solution: gradient clipping

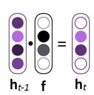


71

Key Idea: Propagated State



Cell State



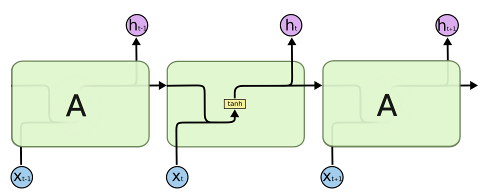
Gating

- Information decays in RNNs because it gets multiplied each time step
- Idea: have a channel called the *cell state* that by default just gets propagated (the "conveyer belt")
- Gates make explicit decisions about what to add / forget from this channel

Image: https://colah.github.io/posts/2015-08-Understanding-LSTMs/

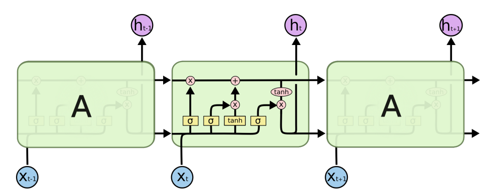
72

RNNs

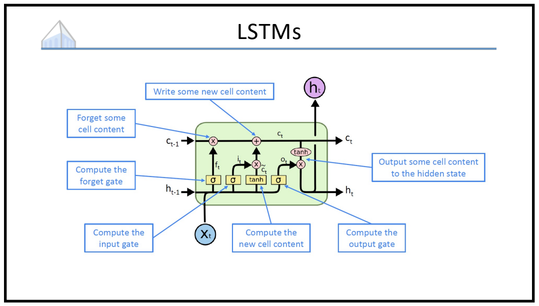


73

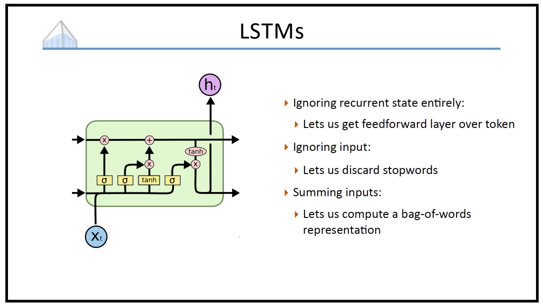
LSTMs



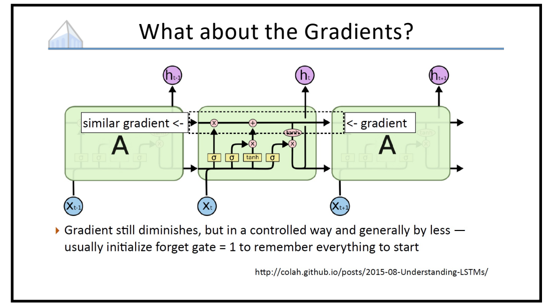
74



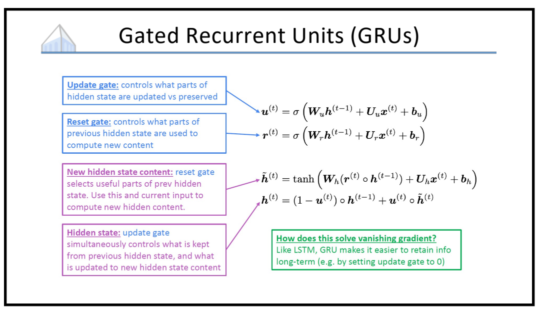
75



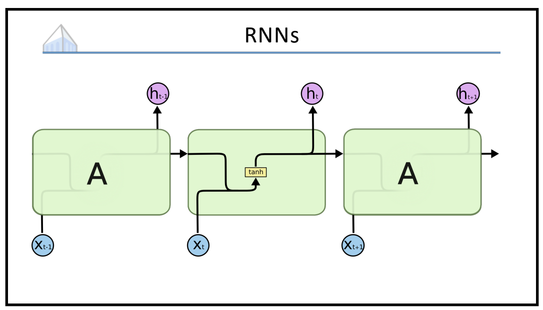
76



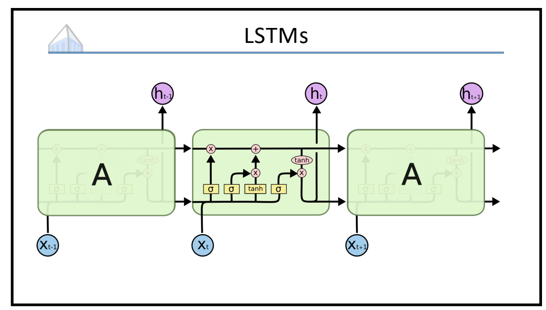
77



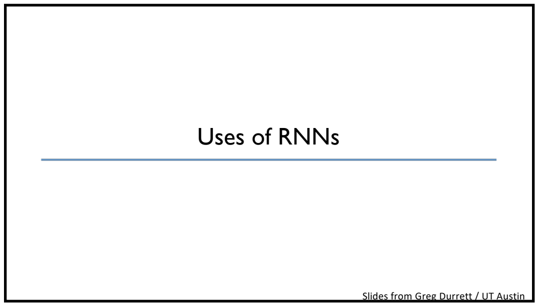
78



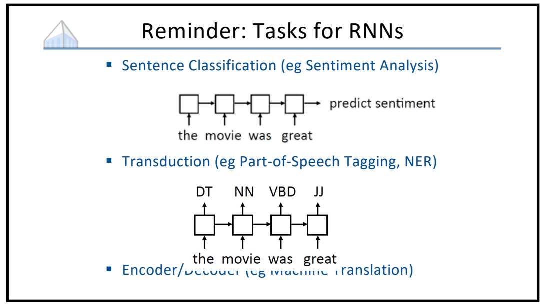
80



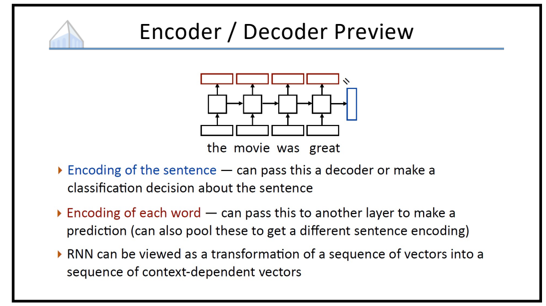
81



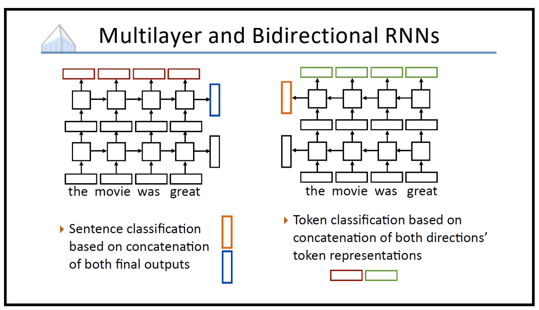
82



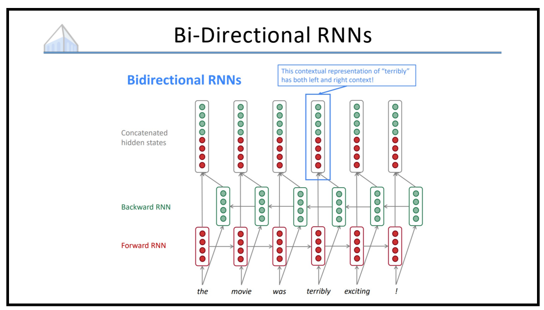
83



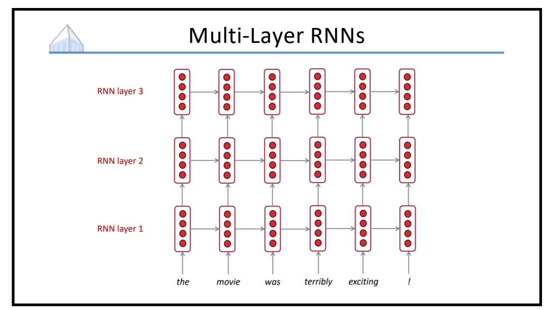
84



85

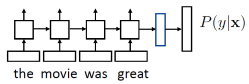


86



87

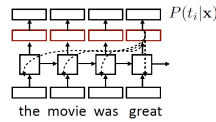
Training for Sentential Tasks



- Loss = negative log likelihood of probability of gold label (or use SVM or other loss)
- Backpropagate through entire network
- Example: sentiment analysis

88

Training for Transduction Tasks



- Loss = negative log likelihood of probability of gold predictions, summed over the tags
- Loss terms filter back through network
- Example: language modeling (predict next word given context)

89

Example Sentential Task: NL Inference

Premise		Hypothesis
A boy plays in the snow	*entails*	A boy is outside
A man inspects the uniform of a figure	*contradicts*	The man is sleeping
An older and younger man smiling	*neutral*	Two men are smiling and laughing at cats playing

- Long history of this task: "Recognizing Textual Entailment" challenge in 2006 (Dagan, Glickman, Magnini)
- Early datasets: small (hundreds of pairs), very ambitious (lots of world knowledge, temporal reasoning, etc.)

90

SNLI Dataset

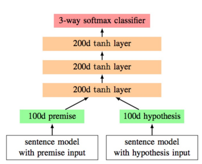
- Show people captions for (unseen) images and solicit entailed / neural / contradictory statements
- >500,000 sentence pairs
- Encode each sentence and process

100D LSTM: 78% accuracy

300D LSTM: 80% accuracy
(Bowman et al., 2016)

300D BiLSTM: 83% accuracy
(Liu et al., 2016)

- Later: better models for this



Bowman et al. (2015)

91

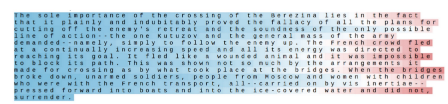
Visualizing RNNs

Slides from Greg Durrett / UT Austin

92

LSTMs Can Model Length

- Train *character* LSTM language model (predict next character based on history) over two datasets: War and Peace and Linux kernel source code
- Visualize activations of specific cells (components of **c**) to understand them
- Counter: know when to generate \n

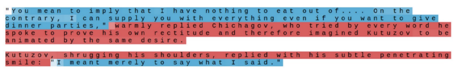


Karpathy et al. (2015)

93

LSTMs Can Model Long-Term Bits

- Train *character* LSTM language model (predict next character based on history) over two datasets: War and Peace and Linux kernel source code
- Visualize activations of specific cells to see what they track
- Binary switch: tells us if we're in a quote or not

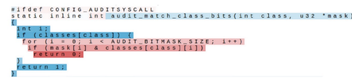


Karpathy et al. (2015)

94

LSTMs Can Model Stack Depth

- Train *character* LSTM language model (predict next character based on history) over two datasets: War and Peace and Linux kernel source code
- Visualize activations of specific cells to see what they track
- Stack: activation based on indentation

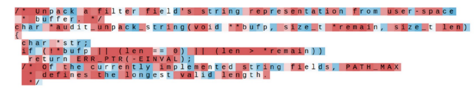


Karpathy et al. (2015)

95

LSTMs Can Be Completely Inscrutable

- Train *character* LSTM language model (predict next character based on history) over two datasets: War and Peace and Linux kernel source code
- Visualize activations of specific cells to see what they track
- Uninterpretable: probably doing double-duty, or only makes sense in the context of another activation



Karpathy et al. (2015)

96