STRUCTURE OF THIS LECTURE

1. Quick LM Recap
2. Recurrent Neural Nets
3. Evaluation
LANGUAGE MODELING
The goal is to obtain a model to compute:

\[ P(X) = \prod_{i=1}^{I} P(x_i \mid x_1, \ldots, x_{i-1}) \]

What if we could have infinite history, instead of relying on finite n-grams?
AN ALTERNATIVE: FEATURIZED LOG-LINEAR MODELS
AN ALTERNATIVE: FEATURIZED MODELS

Calculate features of the context

Based on the features, calculate probabilities

Optimize feature weights using gradient descent, etc.
EXAMPLE:

Previous words: "giving a"

\[
b = \begin{pmatrix} 3.0 \\ 2.5 \\ -0.2 \\ 0.1 \\ 1.2 \\ \vdots \end{pmatrix}
\]

\[
w_{1,a} = \begin{pmatrix} -6.0 \\ -5.1 \\ 0.2 \\ 0.1 \\ 0.5 \\ \vdots \end{pmatrix}
\]

\[
w_{2,\text{giving}} = \begin{pmatrix} -0.2 \\ -0.3 \\ 1.0 \\ 2.0 \\ -1.2 \\ \vdots \end{pmatrix}
\]

\[
s = \begin{pmatrix} -3.2 \\ -2.9 \\ 1.0 \\ 2.2 \\ 0.6 \\ \vdots \end{pmatrix}
\]

Words we’re predicting

How likely are they?

How likely are they given prev. word is “a”?

How likely are they given 2nd prev. word is “giving”?

Total score
Convert scores into probabilities by taking the exponent and normalizing (softmax)

\[ P(x_i \mid x_{i-n+1}^{i-1}) = \frac{e^{s(x_i \mid x_{i-n+1}^{i-1})}}{\sum \tilde{x}_i e^{s(\tilde{x}_i \mid x_{i-n+1}^{i-1})}} \]

\[ s = \begin{pmatrix} -3.2 \\ -2.9 \\ 1.0 \\ 2.2 \\ 0.6 \\ \vdots \end{pmatrix} \quad \rightarrow \quad p = \begin{pmatrix} 0.002 \\ 0.003 \\ 0.329 \\ 0.444 \\ 0.090 \\ \vdots \end{pmatrix} \]
A COMPUTATION GRAPH VIEW

Each word has a vector of weights for each tag

\[
giving + \text{ lookup2} + \text{ lookup1} + \text{ bias} = \text{ scores} \Downarrow \text{ probs}
\]

Each vector is size of output vocabulary
A NOTE: “LOOKUP”

Lookup can be viewed as “grabbing” a single vector from a big matrix of word embeddings.

Similarly, can be viewed as multiplying with an “one-hot” vector.

The former tends to be faster.
To train, we calculate a “loss function” (a measure of how bad our predictions are), and move the parameters to reduce the loss.

The most common loss function for probabilistic models is “negative log likelihood”
TRAINING A MODEL

Reminder: to train, we calculate a “loss function” (a measure of how bad our predictions are), and move the parameters to reduce the loss.

The most common loss function for probabilistic models is “negative log likelihood”:

If element 3 (or zero-indexed, 2) is the correct answer:

\[
p = \begin{pmatrix} 0.002 \\ 0.003 \\ 0.329 \\ 0.444 \\ 0.090 \\ \ldots \end{pmatrix}
\]

\[-\log 1.112\]
CHOOSING A VOCABULARY
UNKNOWN WORDS

Necessity for UNK words

We won’t have all the words in the world in training data

Larger vocabularies require more memory and computation time

Common ways:

Frequency threshold (usually UNK <= 1)

Rank threshold
A very large number of published documents contain text only. They often look boring, and they are often written in obscure language, using mile-long sentences and cryptic technical terms, using one font only, perhaps even without headings. Such style, or lack of style, might be the one you are strongly expected to follow when writing eg scientific or technical reports, legal documents, or administrative papers. It is natural to think that such documents would benefit from a few illustrative images. (However, just adding illustration might be rather useless, if the text remains obscure and unstructured.)
a very large number of published documents contain text only. they often look boring, and they are often written in obscure language, using mile-long sentences and cryptic technical terms, using one font only, perhaps even without headings. such style, or lack of style, might be the one you are strongly expected to follow when writing eg scientific or technical reports, legal documents, or administrative papers. it is natural to think that such documents would benefit from a few illustrative images. (however, just adding illustration might be rather useless, if the text remains obscure and unstructured.)
a very large number of published documents contain text only. They often look boring, and they are often written in obscure language, using mile-long sentences and cryptic technical terms, using one font only, perhaps even without headings. Such style, or lack of style, might be the one you are strongly expected to follow when writing eg scientific or technical reports, legal documents, or administrative papers. It is natural to think that such documents would benefit from a few illustrative images. (However, just adding illustration might be rather useless, if the text remains obscure and unstructured.)

Find rare words (e.g. with freq<2)
a very large number of published documents contain text only. They often look boring, and they are often written in obscure language, using **UNK** sentences and cryptic technical terms, using one font only, perhaps even without headings. Such style, or lack of style, might be the one you are strongly expected to follow when writing, e.g., scientific or technical reports, legal documents, or **UNK** papers. It is natural to think that such documents would benefit from a few illustrative images. (However, just adding **UNK** might be rather useless, if the text remains obscure and **UNK**.)

Substitute with **UNK**
Important: the vocabulary must be the same over models you compare
Important: the vocabulary must be the same over models you compare

Or more accurately, all models must be able to generate the test set (it’s OK if they can generate more than the test set, but not less)

  e.g. Comparing a character-based model to a word-based model is fair, but not vice-versa
RECURRENT NEURAL NETWORKS
LINEAR MODELS CAN’T LEARN FEATURE COMBINATIONS

farmers eat steak $\rightarrow$ **high**  cows eat steak $\rightarrow$ **low**
farmers eat hay $\rightarrow$ **low**  cows eat hay $\rightarrow$ **high**

These can’t be expressed by linear features

What can we do?
Remember combinations as features (individual scores for “farmers eat”, “cows eat”)
$\rightarrow$ Feature space explosion!
Neural nets
NEURAL LANGUAGE MODELS

(See Bengio et al. 2004)
NEURAL LANGUAGE MODELS

giving a lookup

(See Bengio et al. 2004)
NEURAL LANGUAGE MODELS

(See Bengio et al. 2004)
NEURAL LANGUAGE MODELS

giving a

lookup lookup

tanh(W * h + b_1)

(See Bengio et al. 2004)
NEURAL LANGUAGE MODELS

giving + a
lookup
lookup

$tanh(\ W \cdot h + b) + W = bias \ scores$

(See Bengio et al. 2004)
NEURAL LANGUAGE MODELS

\[
\text{tanh}(W^T h + b) + \text{bias} = \text{scores}
\]

\[
\text{softmax}(\text{scores}) = \text{probs}
\]

(See Bengio et al. 2004)
WHERE IS STRENGTH SHARED?

Word embeddings: Similar input words get similar vectors

(See Bengio et al. 2004)
WHERE IS STRENGTH SHARED?

Giving a lookup

lookup

lookup

tanh($W \cdot h + b_1$)

W

bias = scores

softmax

probs

Similar output words get similar rows in the softmax matrix

Word embeddings: Similar input words get similar vectors

(See Bengio et al. 2004)
WHERE IS STRENGTH SHARED?

Giving a lookup

lookup

lookup

tanh(W*h + b_1) = scores

W

bias = softmax

scores

probs

Similar output words get similar rows in the softmax matrix

Similar contexts get similar hidden states

Word embeddings:
Similar input words get similar vectors

(See Bengio et al. 2004)
WHAT PROBLEMS ARE HANDLED?

Cannot share strength among similar words

she bought a car she bought a bicycle
she purchased a car she purchased a bicycle

→ solved, and similar contexts as well! 😊

Cannot condition on context with intervening words

Dr. Jane Smith Dr. Gertrude Smith

→ solved! 😊

Cannot handle long-distance dependencies

for tennis class he wanted to buy his own racquet
for programming class he wanted to buy his own computer

→ not solved yet 😞
Agreement in number, gender, etc.

He does not have very much confidence in **himself**.
She does not have very much confidence in **herself**.
LONG-DISTANCE DEPENDENCIES IN LANGUAGE

Agreement in number, gender, etc.

He does not have very much confidence in himself.
She does not have very much confidence in herself.

Selectional preference

The reign has lasted as long as the life of the queen.
The rain has lasted as long as the life of the clouds.
What is the referent of “it”? 

The trophy would not fit in the brown suitcase because it was too **big**.

Trophy

The trophy would not fit in the brown suitcase because it was too **small**.

Suitcase

(from Winograd Schema Challenge: http://commonsensereasoning.org/winograd.html)
Tools to “remember” information

**Feed-forward NN**
- context
- lookup
- transform
- predict
- label

**Recurrent NN**
- context
- lookup
- transform
- predict
- label

**Recurrent Neural Networks (Elman 1990)**
What does processing a sequence look like?
TRAINING RNNS

Calculating the loss

RNN → RNN → RNN → RNN
predict → predict → predict → predict
prediction 1 → prediction 2 → prediction 3 → prediction 4
loss 1 → loss 2 → loss 3 → loss 4
label 1 → label 2 → label 3 → label 4
sum → total loss
RNN TRAINING

The unrolled graph is a well-formed (DAG) computation graph—we can run backprop
RNN TRAINING

The unrolled graph is a well-formed (DAG) computation graph—we can run backprop

![Diagram showing sum and total loss](image)
RNN TRAINING

The unrolled graph is a well-formed (DAG) computation graph—we can run backprop
The unrolled graph is a well-formed (DAG) computation graph—we can run backprop

Parameters are tied across time, derivatives are aggregated across all time steps
The unrolled graph is a well-formed (DAG) computation graph—we can run backprop

Parameters are tied across time, derivatives are aggregated across all time steps

This is historically called “backpropagation through time” (BPTT)
Calculating the loss

Parameters are shared! Derivatives are accumulated.
APPLICATIONS OF RNNS
Language modeling is like a tagging task, where each tag is the next word!
VANISHING GRADIENTS
VANISHING GRADIENT

Gradients decrease as they get pushed back

\[
\frac{dl}{dh_0} = \text{tiny} \quad \frac{dl}{dh_1} = \text{small} \quad \frac{dl}{dh_2} = \text{med.} \quad \frac{dl}{dh_3} = \text{large}
\]

Why? “Squashed” by non-linearities or small weights in matrices
Basic idea: make additive connections between time steps

Addition does not modify the gradient, no vanishing

Gates to control the information flow
LSTM STRUCTURE

Most important idea: we want an additive connection between time steps.

- **update u**: what value do we try to add to the memory cell?
- **input i**: how much of the update do we allow to go through?
- **output o**: how much of the cell do we reflect in the next state?
LSTM STRUCTURE

Most important idea: we want an additive connection between time steps.

update $u$: what value do we try to add to the memory cell?
input $i$: how much of the update do we allow to go through?
output $o$: how much of the cell do we reflect in the next state?
LSTM STRUCTURE

Most important idea: we want an additive connection between time steps

- $c_{t-1}$
- $h_{t-1}$
- $x_t$
- $c_t$
- $h_t$

update $u$: what value do we try to add to the memory cell?
input $i$: how much of the update do we allow to go through?
output $o$: how much of the cell do we reflect in the next state?
Most important idea: we want an additive connection between time steps.

- **Update $u$:** what value do we try to add to the memory cell?
- **Input $i$:** how much of the update do we allow to go through?
- **Output $o$:** how much of the cell do we reflect in the next state?
Most important idea: we want an additive connection between time steps.

update $u$: what value do we try to add to the memory cell?
input $i$: how much of the update do we allow to go through?
output $o$: how much of the cell do we reflect in the next state?
Most important idea: we want an additive connection between time steps.

The $c_t$ part captures long context and $h_t$ captures the short term updates.

- **update $u$:** what value do we try to add to the memory cell?
- **input $i$:** how much of the update do we allow to go through?
- **output $o$:** how much of the cell do we reflect in the next state?
Additive connections make single nodes surprisingly interpretable
WHAT CAN LSTMS LEARN? (2)
(SHI ET AL. 2016, RADFORD ET AL. 2017)

Count length of sentence

Sentiment(?)

25 August 2003 League of Extraordinary Gentlemen: Sean Connery is one of the all-time greats and I have been a fan of his since the 1960's. I went to this movie because Sean Connery was the main actor. I had not read reviews or had any prior knowledge of the movie. The movie surprised me quite a bit. The scenery and sights were spectacular, but the plot was unreal to the point of being ridiculous. In my mind this was not one of his better movies it could be the worst. Why he chose to be in this movie is a mystery. For me, going to this movie was a waste of my time. I will continue to go to his movies and add his movies to my video collection, but I can't see wasting money on this movie in my collection.

I found this to be a charming adaptation, very lively and full of fun. With the exception of a couple of major errors, the cast is wonderful. I have to echo some of the earlier comments - Chyna "Billius" is horribly miscast as a teenager. At 27, she's just too old (yes, yep, it DOES snow), and lacks the singing "chops" for Broadway-style music. Vanessa Williams is not strong enough singer and, for a womanizer, she's a virgin. However, she is NOT Latina, and her character definitely isn't. She's also very SHREWISH throughout, which gets tiresome. The girls of Brewster's Millions really sparkle -- with special kudos to Lilliana Zanetti and Chiara Zanetti. I also enjoyed Tyre Dal's performance, though I'm not generally a fan of her work. Finally, the dancing Zorries are a riot, especially the dorky three in the bar. The movie is suitable for the whole family, and I highly recommend it.
Language models produce good representations!

BERT and family