Sentiment Analysis: Neural Methods

Katharina Kann — CSCI/LING5832

In-Class Exercise 2

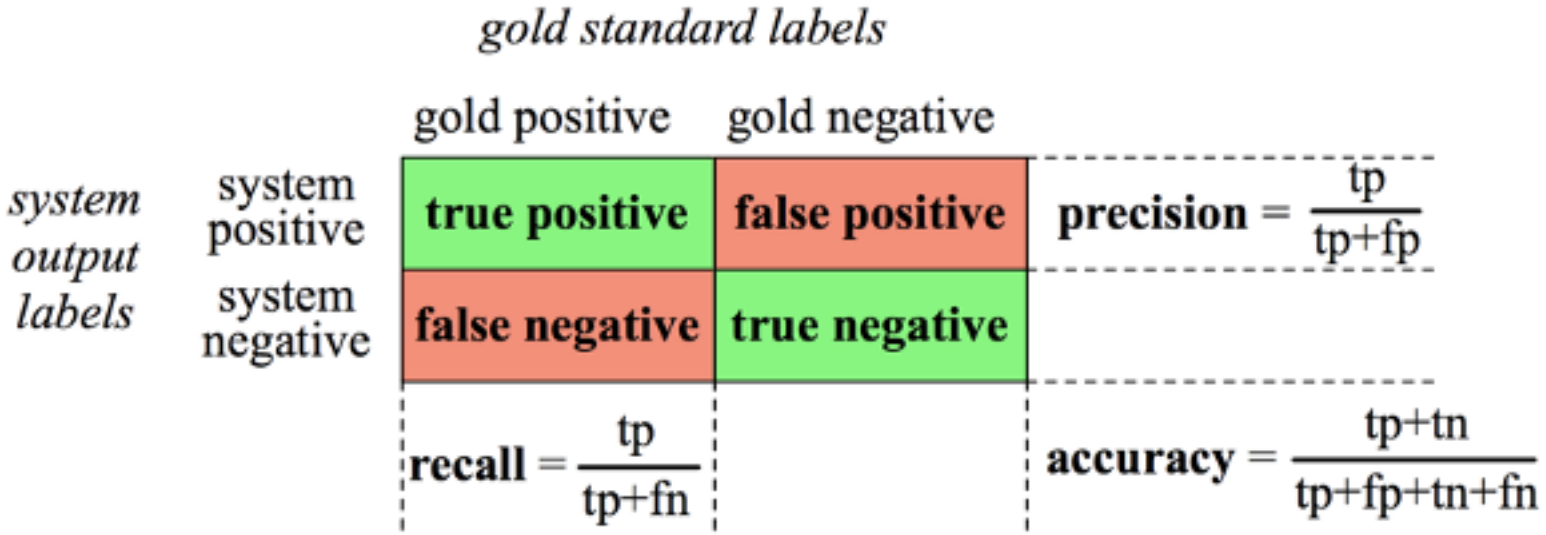
You have data from 100 patients and want to classify who has cancer. 53 of them in fact have cancer, but your system predicts positive for only 45 of them, and negative for 8. For the 47 healthy patients, your system predicts negative for 40, and positive for 7.

In-Class Exercise 2

You have data from 100 patients and want to classify who has cancer. 53 of them in fact have cancer, but your system predicts positive for only 45 of them, and negative for 8. For the 47 healthy patients, your system predicts negative for 40, and positive for 7.

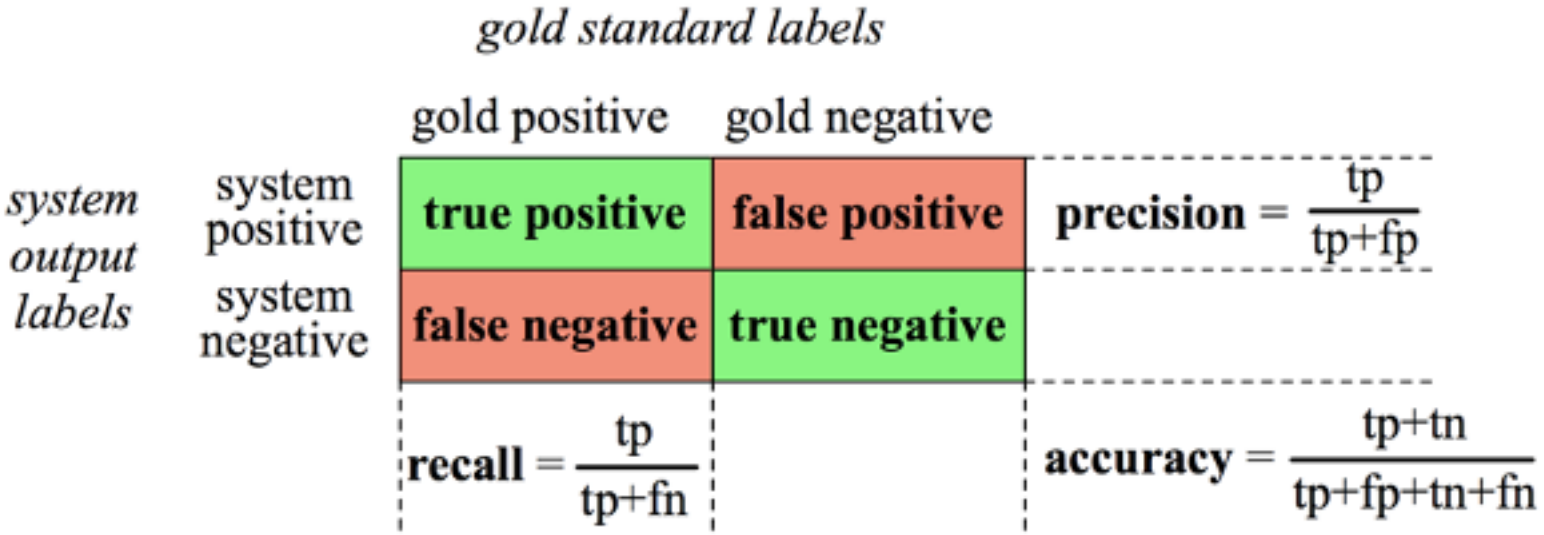
- Compute: precision, recall, and F1 score for your system!

F1 Score



$$F_1 = \frac{2PR}{P+R}$$

F1 Score



What's good for what?

$$F_1 = \frac{2PR}{P+R}$$

Multinomial Logistic Regression

Multi-Class Classification

- Needed for tasks that don't consist of a binary decision
- Many possible examples:
  - Sentiment: positive, negative, neutral
  - Natural language inference: entailment, contradiction, neutral
  - Predicting the next word
  - …

Multinomial Logistic Regression

- Also called **softmax regression** or **maxent classifier**.
- Uses a softmax function for the output!
- This requires weights for each class.

Recap: Softmax

- Use for one-of-many predictions
- Gives you a probability distribution
  - Values are all between 0 and 1
  - Values add up to 1

$$\text{softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}} \quad 1 \leq i \leq D$$

Sentiment Analysis: Neural Methods

The Big Question

What kind of thing is the meaning of a sentence?

The Big Question

~~What kind of thing is the meaning of a sentence?~~

What can you do with a sentence if you know its meaning?

The Big Question

~~What kind of thing is the meaning of a sentence?~~

What can you do with a sentence if you know its meaning?

Judge its sentiment!

Positive or Negative?

- unbelievably disappointing
- Full of zany characters and richly applied satire, and some great plot twists
- this is the greatest screwball comedy ever filmed
- It was pathetic. The worst part about it was the boxing scenes.

Back to Building a Sentiment Classifier

- What we have:
  - About 10k examples of sentence–label pairs.
  - (SST/Rotten Tomatoes)
  - Some intuition for what the labels mean

Back to Building a Sentiment Classifier

- What we have:
  - About 10k examples of sentence–label pairs.
  - (SST/Rotten Tomatoes)
  - Some intuition for what the labels mean
- What we want:
  - A function that can quickly and accurately identify the label (from the seen set) for a new sentence.

Features: Two Views

- The statistical NLP approach:
  - Features should be carefully chosen, and should capture the engineer's intuitions about the kinds of information the model will need. Feature functions can be complex, and can use a variety of external resources.
  - Models should be relatively simple, and should focus on efficiently mapping feature vectors to predictions.

Features: Two Views

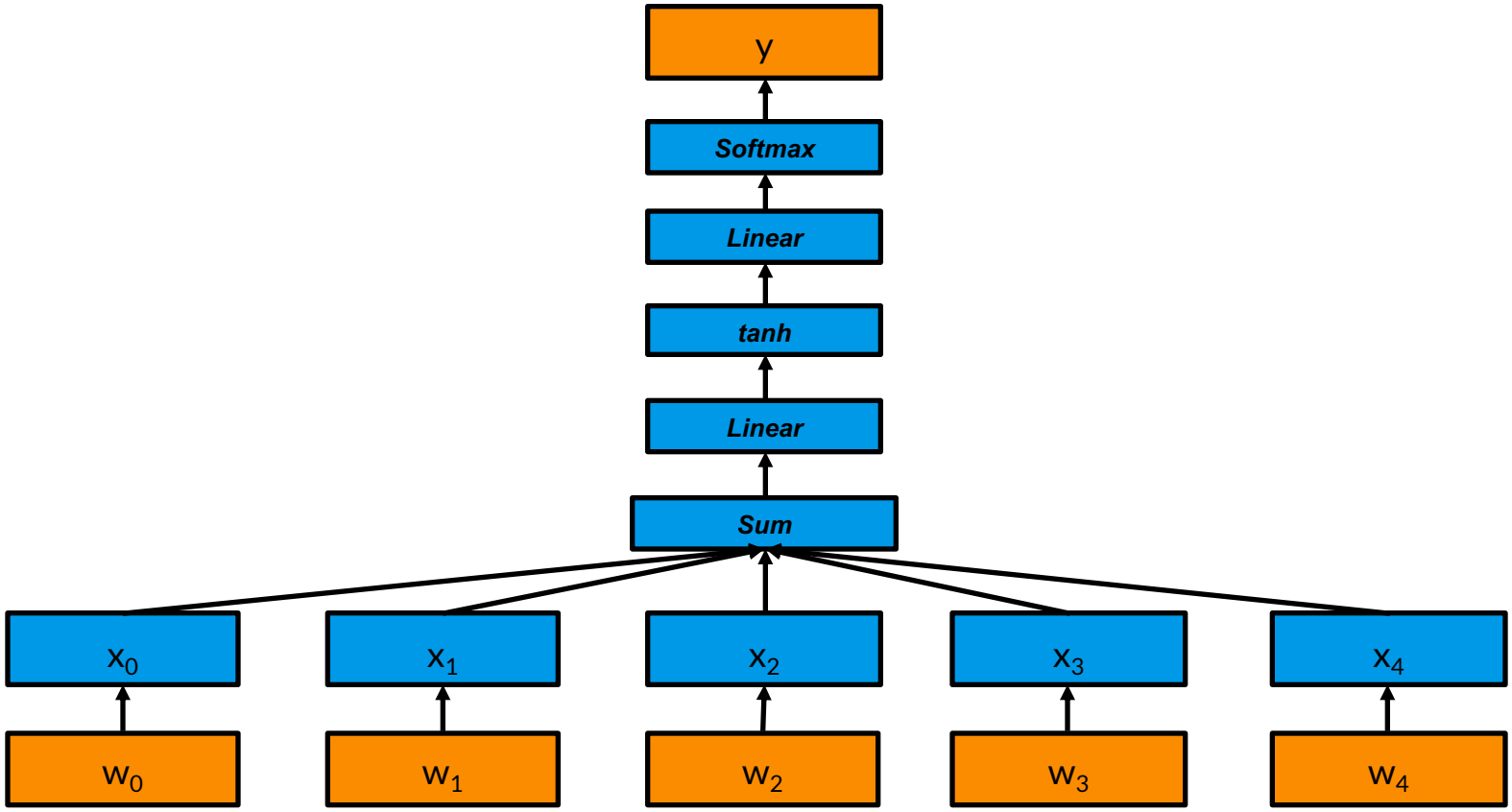
- The statistical NLP approach:
  - Features should be carefully chosen, and should capture the engineer's intuitions about the kinds of information the model will need. Feature functions can be complex, and can use a variety of external resources.
  - Models should be relatively simple, and should focus on efficiently mapping feature vectors to predictions.
- The deep learning approach:
  - Features should be a direct representation of the input, with little or no problem-specific engineering.
  - Models should be complex enough for end-to-end learning: They should learn to build their own feature functions that transform the raw inputs into something more useful.
  - Easier, but requires lots of training data.

Today: Deep Learning for Text Classification

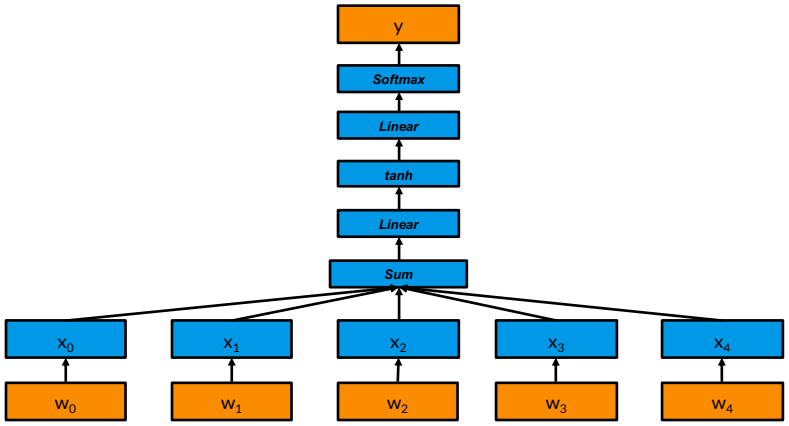
- A high-level survey of deep learning or artificial neural networks for text classification.
  - Feed-forward networks
  - CNNs
  - RNNs

Neural Networks for Text Classification

Feed-Forward Network for Sentence Classification



Feed-Forward Network for Sentence Classification

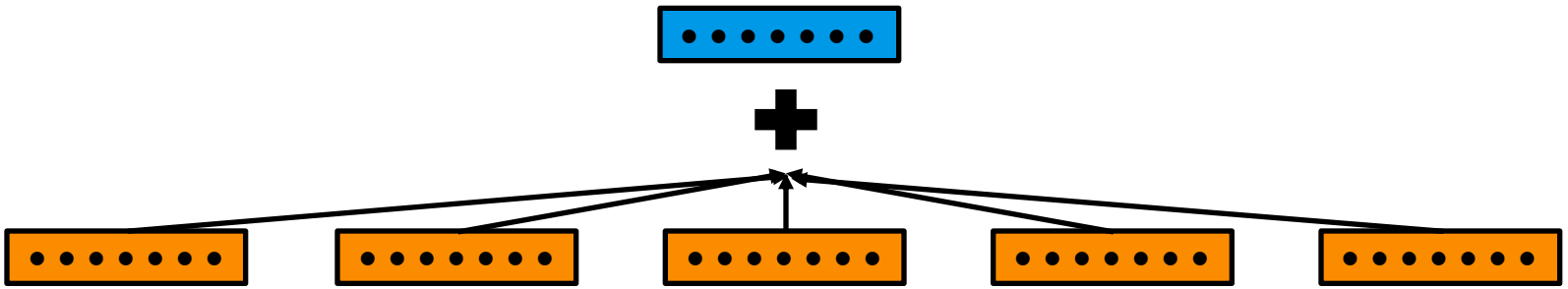


- Average/maximum/minimum can be used instead of sum (e.g., Iyyer et al. (2015)).
- The more hidden layers, the more capacity.

Convolutional Neural Networks (CNNs)

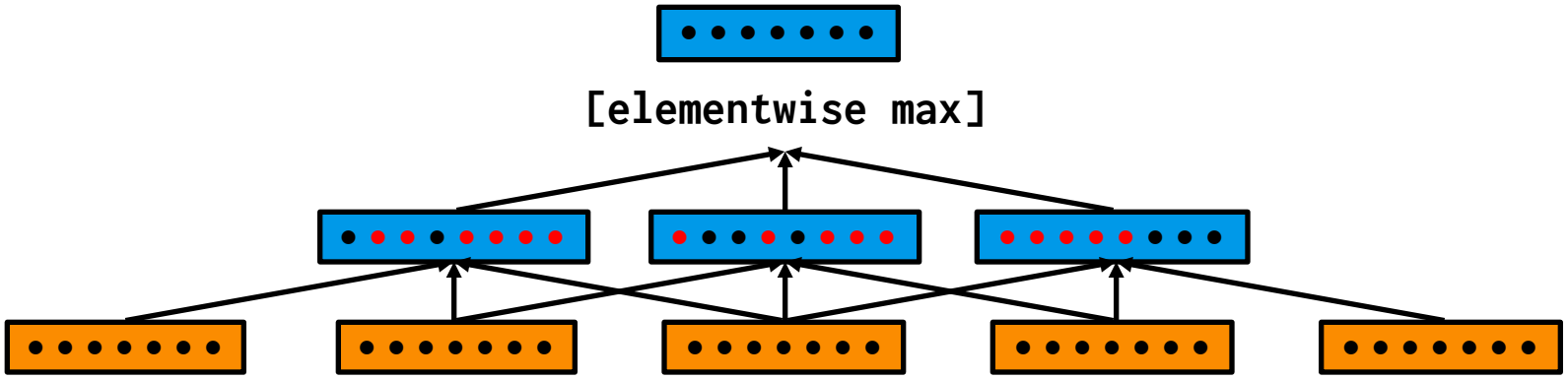
From Bags of Words to Convolutions

$$\vec{h} = \sum_{i=0}^{N} \vec{x}$$



From Bags of Words to Convolutions

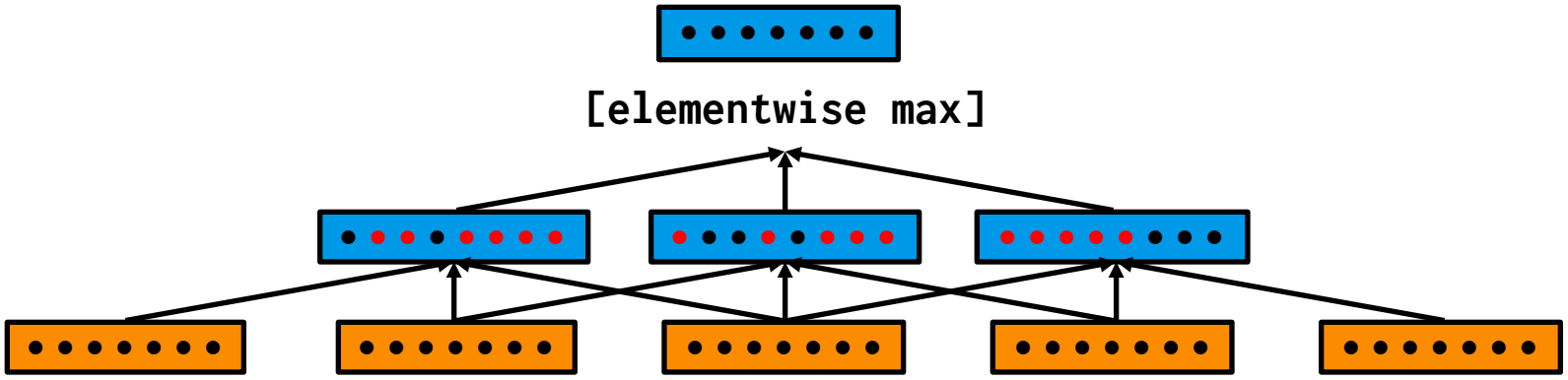
$$\vec{c_i} = \mathbf{W_c} \begin{bmatrix} \vec{x_{i-1}} \\ \vec{x_i} \\ \vec{x_{i+1}} \end{bmatrix} \qquad \vec{h} = \begin{bmatrix} \max_i c_i^0 \\ \max_i c_i^1 \\ \dots \\ \max_i c_i^D \end{bmatrix}$$



From Bags of Words to Convolutions

Max-pooling

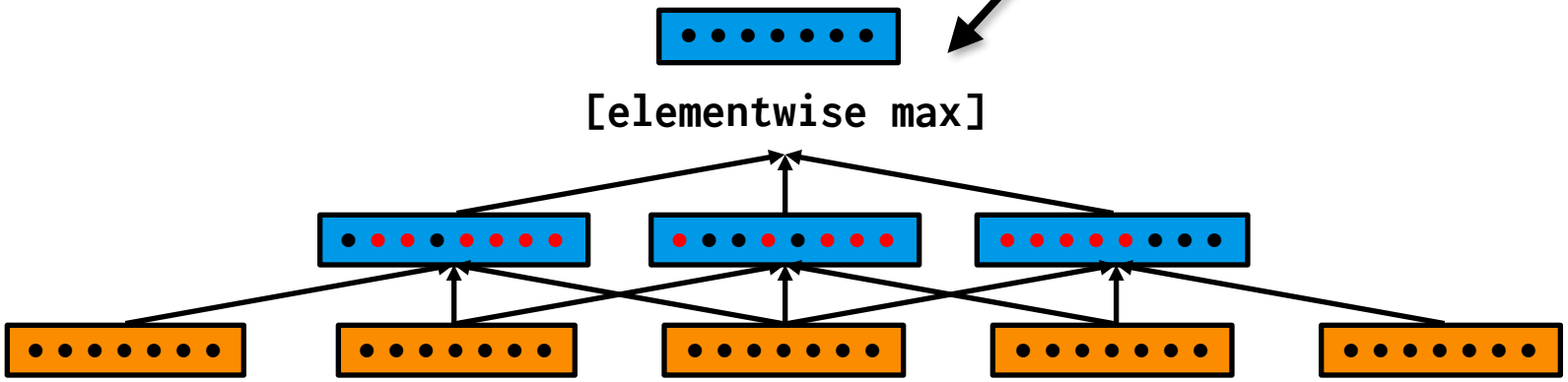
$$\vec{c_i} = \mathbf{W_c} \begin{bmatrix} \vec{x_{i-1}} \\ \vec{x_i} \\ \vec{x_{i+1}} \end{bmatrix} \qquad \vec{h} = \begin{bmatrix} \max_i c_i^0 \\ \max_i c_i^1 \\ \dots \\ \max_i c_i^D \end{bmatrix}$$



From Bags of Words to Convolutions

Max-pooling

$$\vec{c_i} = \mathbf{W_c} \begin{bmatrix} \vec{x_{i-1}} \\ \vec{x_i} \\ \vec{x_{i+1}} \end{bmatrix} \qquad \vec{h} = \begin{bmatrix} \max_i c_i^0 \\ \max_i c_i^1 \\ \dots \\ \max_i c_i^D \end{bmatrix}$$



From Bags of Words to Convolutions

$$\vec{c_i} = \mathbf{W_c} \begin{bmatrix} \vec{x_{i-1}} \\ \vec{x_i} \\ \vec{x_{i+1}} \end{bmatrix} \qquad \vec{h} = \begin{bmatrix} \max_i c_i^0 \\ \max_i c_i^1 \\ \dots \\ \max_i c_i^D \end{bmatrix}$$

- Other options instead of maximum:
  - Minimum
  - Average
- Filter size is a hyperparameter:
  - Something you decide on with the development set.

From Bags of Words to Convolutions

$$\vec{c_i} = \mathbf{W_c} \begin{bmatrix} \vec{x_{i-1}} \\ \vec{x_i} \\ \vec{x_{i+1}} \end{bmatrix} \qquad \vec{h} = \begin{bmatrix} \max_i c_i^0 \\ \max_i c_i^1 \\ \dots \\ \max_i c_i^D \end{bmatrix}$$

- For a window size of 3, input size 100, and hidden size 100:
  - What are the dimensions of the weight matrix of the convolution? (Why?)

From Bags of Words to Convolutions

$$\vec{c_i} = \mathbf{W_c} \begin{bmatrix} \vec{x_{i-1}} \\ \vec{x_i} \\ \vec{x_{i+1}} \end{bmatrix} \qquad \vec{h} = \begin{bmatrix} \max_i c_i^0 \\ \max_i c_i^1 \\ \dots \\ \max_i c_i^D \end{bmatrix}$$

- For a window size of 3, input size 100, and hidden size 100:
  - What are the dimensions of the weight matrix of the convolution? (Why?)
  - 100 x 300

Convolution Example

$$\vec{c_i} = \mathbf{W_c} \begin{bmatrix} \vec{x_{i-1}} \\ \vec{x_i} \\ \vec{x_{i+1}} \end{bmatrix} \qquad \vec{h} = \begin{bmatrix} \max_i c_i^0 \\ \max_i c_i^1 \\ \dots \\ \max_i c_i^D \end{bmatrix}$$

- Our input sentence: *cool*
- Cool = [3, 4, 2], <s> = [3, 2, 1], </s> = [2.5, 1, 1]
- W = [[2, 3, 4, 5, 6, 7]]
- Filter size: 2, hidden dimension: 1

Convolution Example

$$\vec{c_i} = \mathbf{W_c} \begin{bmatrix} \vec{x_{i-1}} \\ \vec{x_i} \\ \vec{x_{i+1}} \end{bmatrix} \qquad \vec{h} = \begin{bmatrix} \max_i c_i^0 \\ \max_i c_i^1 \\ \dots \\ \max_i c_i^D \end{bmatrix}$$

- Our input sentence: *cool*
- Cool = [3, 4, 2], <s> = [3, 2, 1], </s> = [2.5, 1, 1]
- W = [[2, 3, 4, 5, 6, 7]]
- Filter size: 2, hidden dimension: 1

- c1 = [[2, 3, 4, 5, 6, 7]] * [[3], [2], [1], [3], [4], [2]]
  = 6 + 6 + 4 + 15 + 24 + 14 = 69

Convolution Example

$$\vec{c_i} = \mathbf{W_c} \begin{bmatrix} \vec{x_{i-1}} \\ \vec{x_i} \\ \vec{x_{i+1}} \end{bmatrix} \qquad \vec{h} = \begin{bmatrix} \max_i c_i^0 \\ \max_i c_i^1 \\ \dots \\ \max_i c_i^D \end{bmatrix}$$

- Our input sentence: *cool*
- Cool = [3, 4, 2], <s> = [3, 2, 1], </s> = [2.5, 1, 1]
- W = [[2, 3, 4, 5, 6, 7]]
- Filter size: 2, hidden dimension: 1

- c1 = [[2, 3, 4, 5, 6, 7]] * [[3], [2], [1], [3], [4], [2]]
  = 6 + 6 + 4 + 15 + 24 + 14 = 69
- c2 = [[2, 3, 4, 5, 6, 7]] * [[3], [4], [2], [2.5], [1], [1]]
  = 6 + 12 + 8 + 12.5 + 6 + 7 = 51.5

Convolution Example

$$\vec{c_i} = \mathbf{W_c} \begin{bmatrix} \vec{x_{i-1}} \\ \vec{x_i} \\ \vec{x_{i+1}} \end{bmatrix} \qquad \vec{h} = \begin{bmatrix} \max_i c_i^0 \\ \max_i c_i^1 \\ \dots \\ \max_i c_i^D \end{bmatrix}$$

- Our input sentence: *cool*
- Cool = [3, 4, 2], <s> = [3, 2, 1], </s> = [2.5, 1, 1]
- W = [[2, 3, 4, 5, 6, 7]]
- Filter size: 2, hidden dimension: 1

- c1 = [[2, 3, 4, 5, 6, 7]] * [[3], [2], [1], [3], [4], [2]]
  = 6 + 6 + 4 + 15 + 24 + 14 = 69
- c2 = [[2, 3, 4, 5, 6, 7]] * [[3], [4], [2], [2.5], [1], [1]]
  = 6 + 12 + 8 + 12.5 + 6 + 7 = 51.5
- h = max(69, 51.5) = 69

Using a Sentence Representation

$$\vec{c_i} = \mathbf{W_c} \begin{bmatrix} \vec{x_{i-1}} \\ \vec{x_i} \\ \vec{x_{i+1}} \end{bmatrix} \qquad \vec{h} = \begin{bmatrix} \max_i c_i^0 \\ \max_i c_i^1 \\ \dots \\ \max_i c_i^D \end{bmatrix}$$

- What can we do with h?
- Input into our classifier, for example:
  - Logistic regression
  - Feed-forward network
  - …

Using a Sentence Representation

$$\vec{c_i} = \mathbf{W_c} \begin{bmatrix} \vec{x_{i-1}} \\ \vec{x_i} \\ \vec{x_{i+1}} \end{bmatrix} \qquad \vec{h} = \begin{bmatrix} \max_i c_i^0 \\ \max_i c_i^1 \\ \dots \\ \max_i c_i^D \end{bmatrix}$$

- What can we do with h?
- Input into our classifier, for example:
  - Logistic regression
  - Feed-forward network
  - …
- Different filter sizes can be combined.

Using a Sentence Representation

$$\vec{c_i} = \mathbf{W_c} \begin{bmatrix} \vec{x_{i-1}} \\ \vec{x_i} \\ \vec{x_{i+1}} \end{bmatrix} \qquad \vec{h} = \begin{bmatrix} \max_i c_i^0 \\ \max_i c_i^1 \\ \dots \\ \max_i c_i^D \end{bmatrix}$$

- What can we do with h?
- Input into our classifier, for example:
  - Logistic regression
  - Feed-forward network
  - …
- Different filter sizes can be combined.
- Convolutional layers can be stack on top of each other.

In-Class Exercise

$$\vec{c_i} = \mathbf{W_c} \begin{bmatrix} \vec{x_{i-1}} \\ \vec{x_i} \\ \vec{x_{i+1}} \end{bmatrix} \qquad \vec{h} = \begin{bmatrix} \max_i c_i^0 \\ \max_i c_i^1 \\ \dots \\ \max_i c_i^D \end{bmatrix}$$

- Take the sentence: *the dog sleeps*
- Our embeddings are:
  - the = [2, 5]; dog = [4, 4]; sleeps = [6, 6.1]; <s> = [3, 2]; </s> = [4, 2]
- w = [[4, 1, 0, 5], [2, 1, 1.1, 2]]

- **What is the filter size? What is the hidden dimension?**
- **Compute h, using max-pooling!**

Where Does This Help?

Model	20News	Fudan	ACL	SST
BoW + LR	92.81	92.08	46.67	40.86
Bigram + LR	93.12	92.97	47.00	36.24
CNN	94.79	94.04	47.47	46.35

Lai et al. (2015)

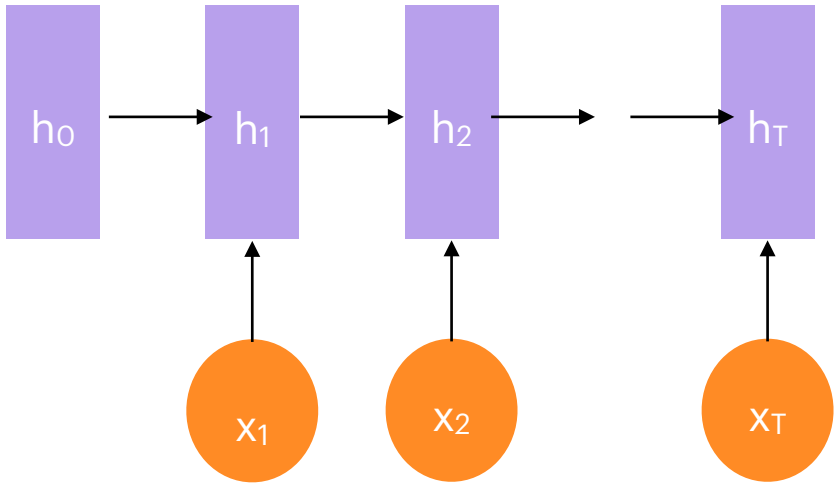
Recurrent Neural Networks (RNNs)

Recurrent Neural Networks

Input: $x_1, x_2, \ldots, x_T$

$$h_i = \text{sigmoid}(W_h h_{i-1} + W_x x_i)$$

$$y = argmax(\text{softmax}(W_y h_T))$$



Recurrent Neural Networks

- Processes a sequence using the same matrix for each input.
- Maintains a representation of the history up to each input.

In-Class Exercise 2

- Take the sentence: *cool*
- Our embeddings are:
  - cool = [0.3, 0, 0.1], <s> = [3, 2, 1], </s> = [2.5, 1, 1]
- w_x = [[2, 3, 1], [4, 2, 1]], w_h = [[8, 1], [2, 5]]
- h0 = [0, 0]

- **What is the hidden dimension?**
- **Compute the last hidden state!**

Problems with Recurrent Neural Networks

- During backpropagation, since RNNs multiply the same matrix over and over, gradients vanish or explode.

Problems with Recurrent Neural Networks

- During backpropagation, since RNNs multiply the same matrix over and over, gradients vanish or explode.
- Cannot capture long-term dependencies (or learn much) with uninformative gradients!

Problems with Recurrent Neural Networks

- During backpropagation, since RNNs multiply the same matrix over and over, gradients vanish or explode.
- Cannot capture long-term dependencies (or learn much) with uninformative gradients!
- History gets forgotten over time.

Problems with Recurrent Neural Networks

The flights the airline was cancelling were full.

Problems with Recurrent Neural Networks

The flights the airline was cancelling were full.

Problems with Recurrent Neural Networks

The flights the airline was cancelling were full.

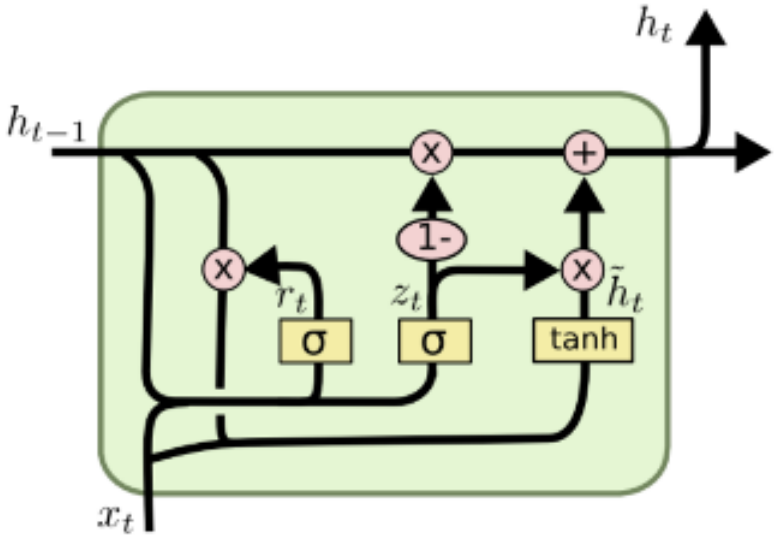
Solutions

- **Gradient Clipping**: Normalize the gradient of a parameter vector when its L2 norm reaches a certain value.

Solutions

- **Gradient Clipping**: Normalize the gradient of a parameter vector when its L2 norm reaches a certain value.
- Explicit **memory cell** with which a model can keep important long-term information.

Gated Recurrent Units (GRUs)



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

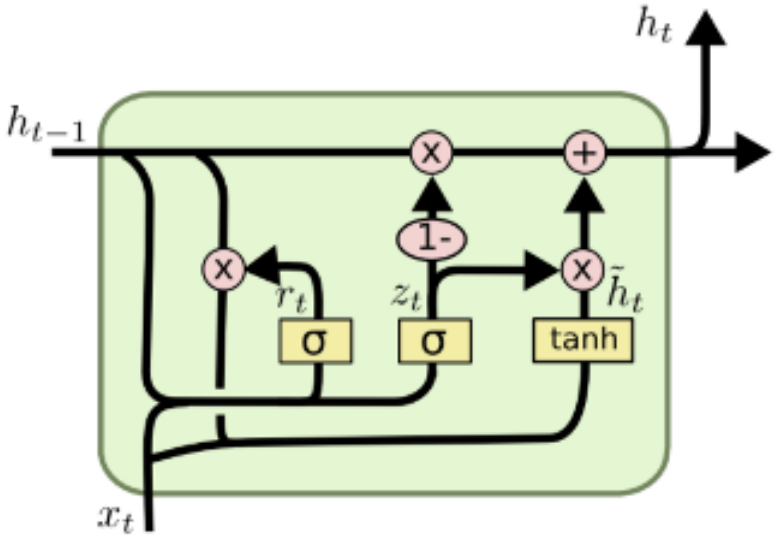
$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Visual Obtained from: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Gated Recurrent Units (GRUs)



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

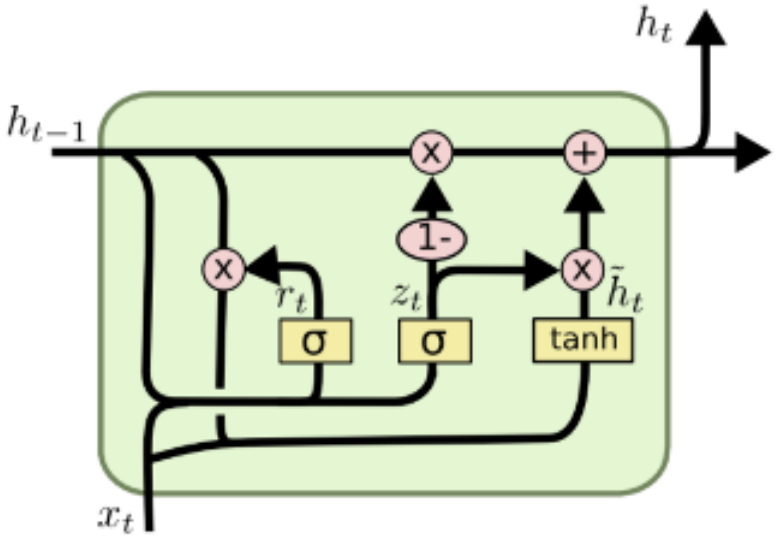
$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Update gate: controls how much of the previous hidden state to update

Visual Obtained from: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Gated Recurrent Units (GRUs)



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

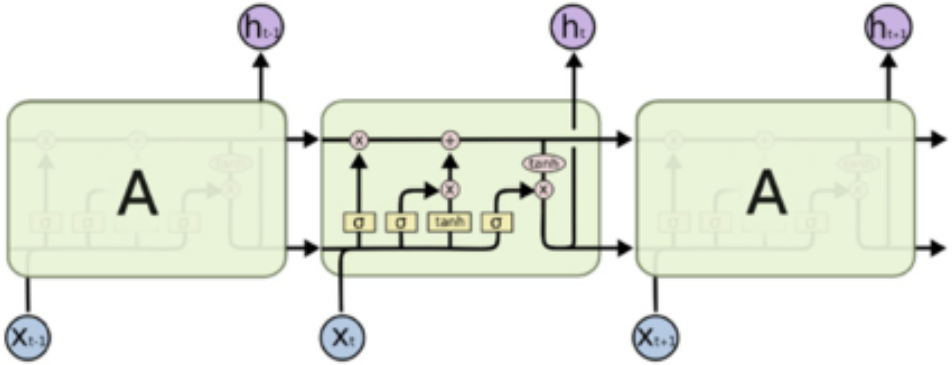
Reset gate: controls how much of the previous hidden state to keep

Visual Obtained from: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Long Short-Term Memory Networks (LSTMs)

- First proposed by Hochreiter and Schmidhuber (1997).
- Similar to the GRU, but some more parameters.
- Performance is usually extremely similar.

Long Short-Term Memory Networks (LSTMs)

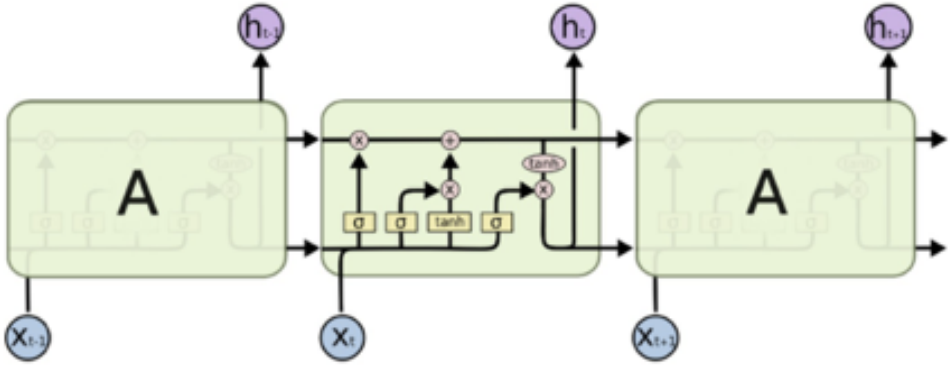


The repeating module in an LSTM contains four interacting layers.

$$
\begin{aligned}
g_t &= tanh(U_g h_{t-1} + W_g x_t) \\
i_t &= \sigma(U_i h_{t-1} + W_i x_t) \\
f_t &= \sigma(U_f h_{t-1} + W_f x_t) \\
o_t &= \sigma(U_o h_{t-1} + W_o x_t) \\
c_t &= f_t \odot c_{t-1} + i_t \odot g_t \\
h_t &= o_t \odot tanh(c_t)
\end{aligned}
$$

Visual Obtained from: http://colah.github.io/posts/2015-08-Understanding-LSTMs/, J&M, Ch.9

Long Short-Term Memory Networks (LSTMs)



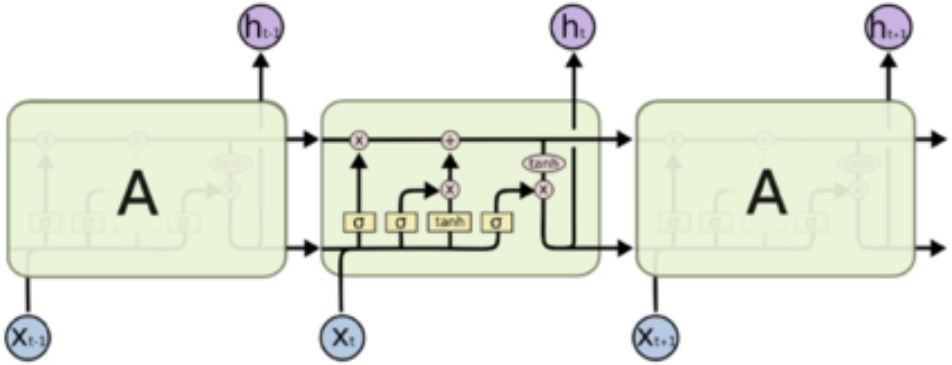
The repeating module in an LSTM contains four interacting layers.

$$
\begin{aligned}
g_t &= tanh(U_g h_{t-1} + W_g x_t) \\
i_t &= \sigma(U_i h_{t-1} + W_i x_t) \\
f_t &= \sigma(U_f h_{t-1} + W_f x_t) \\
o_t &= \sigma(U_o h_{t-1} + W_o x_t) \\
c_t &= f_t \odot c_{t-1} + i_t \odot g_t \\
h_t &= o_t \odot tanh(c_t)
\end{aligned}
$$

Add gate

Visual Obtained from: http://colah.github.io/posts/2015-08-Understanding-LSTMs/, J&M, Ch.9

Long Short-Term Memory Networks (LSTMs)



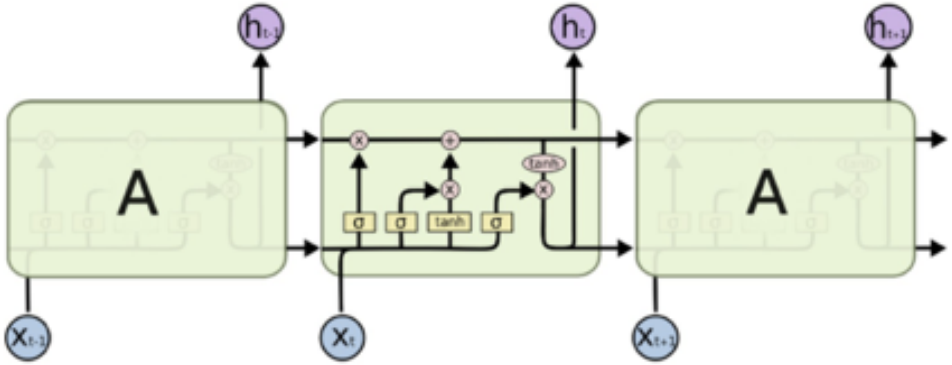
The repeating module in an LSTM contains four interacting layers.

$$
\begin{aligned}
g_t &= tanh(U_g h_{t-1} + W_g x_t) \\
i_t &= \sigma(U_i h_{t-1} + W_i x_t) \\
f_t &= \sigma(U_f h_{t-1} + W_f x_t) \\
o_t &= \sigma(U_o h_{t-1} + W_o x_t) \\
c_t &= f_t \odot c_{t-1} + i_t \odot g_t \\
h_t &= o_t \odot tanh(c_t)
\end{aligned}
$$

Forget gate

Visual Obtained from: http://colah.github.io/posts/2015-08-Understanding-LSTMs/, J&M, Ch.9

Long Short-Term Memory Networks (LSTMs)



The repeating module in an LSTM contains four interacting layers.

$$
\begin{aligned}
g_t &= tanh(U_g h_{t-1} + W_g x_t) \\
i_t &= \sigma(U_i h_{t-1} + W_i x_t) \\
f_t &= \sigma(U_f h_{t-1} + W_f x_t) \\
o_t &= \sigma(U_o h_{t-1} + W_o x_t) \\
c_t &= f_t \odot c_{t-1} + i_t \odot g_t \\
h_t &= o_t \odot tanh(c_t)
\end{aligned}
$$

Output gate

Visual Obtained from: http://colah.github.io/posts/2015-08-Understanding-LSTMs/, J&M, Ch.9

How Do We Get a Sentence Representation?

- Apply the RNN to all words in a sequence.
- Use the last hidden vector as the representation of your sentence.
- The sentence representation is often called *h*.

Bidirectional Recurrent Neural Networks

- Most prominent one: BiLSTM
- Idea: Read the input from both sides.
  - Beginning isn’t forgotten!

Bidirectional Recurrent Neural Networks

- Most prominent one: BiLSTM
- Idea: Read the input from both sides.
  - Beginning isn't forgotten!
- For the final representation:
  - Concatenate the last hidden vectors of the two individual RNNs.
  - (What is the size of the final sentence representation computed by a bidirectional RNN?)

In-Class Exercise 3

- Assume that we want to classify the sentiment of tweets into *positive*, *neutral*, and *negative*.
- Design a neural network that can learn this task, given a large-enough training set.
- Which preprocessing steps do we need?
- Write down all dimensions explicitly. What is your output dimension?
- (Keep your architecture safe for later!)

Wrapping up

- Discussed today:
  - Neural networks for text classification
  - Convolutional neural networks
  - Recurrent neural networks

- Next Monday: Part-of-Speech Tagging and Hidden Markov Models