Deep Learning for Natural Language Processing

Yufeng Ma

CS 6604 - Digital Libraries
Virginia Polytechnic Institute and State University, Blacksburg, VA
Professor Edward Fox

March 21, 2017

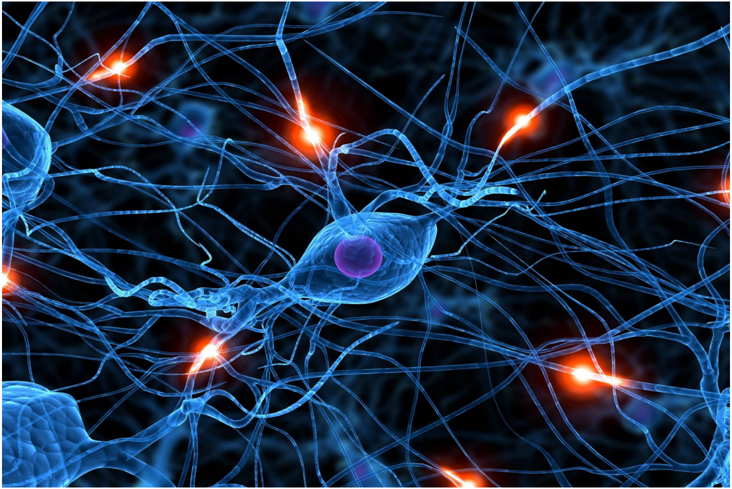
Overview

1. What's Deep Learning?
2. Neural Networks Recap
   - Demystifying Neural Networks
   - Forward/Backward Propagation
   - Gradient Descent & Chain Rule
3. Lego Blocks for Building NLP Deep Nets
   - Word Embedding - word2vec
   - Recurrent Neural Network - LSTM & Bidirectional
   - Recursive Neural Network
   - Convolutional Neural Network
4. Advanced Models
   - Attention Model
   - Seq2seq/End-to-end Learning

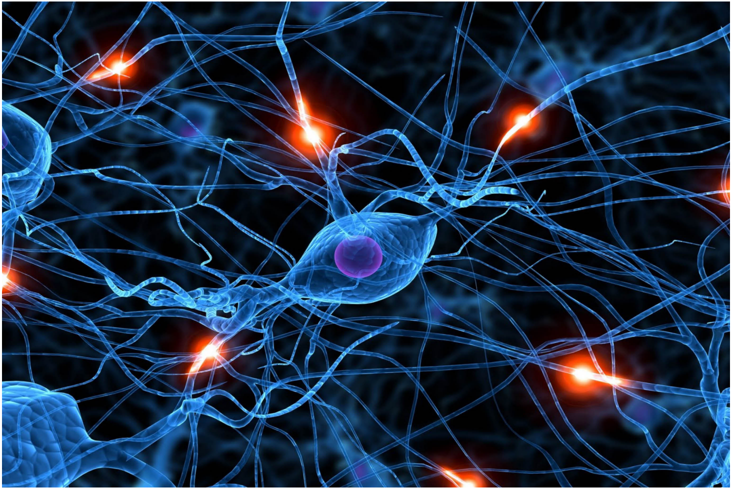
Overview

1. What's Deep Learning?
2. Neural Networks Recap
   - Demystifying Neural Networks
   - Forward/Backward Propagation
   - Gradient Descent & Chain Rule
3. Lego Blocks for Building NLP Deep Nets
   - Word Embedding - word2vec
   - Recurrent Neural Network - LSTM & Bidirectional
   - Recursive Neural Network
   - Convolutional Neural Network
4. Advanced Models
   - Attention Model
   - Seq2seq/End-to-end Learning

What's Deep Learning?

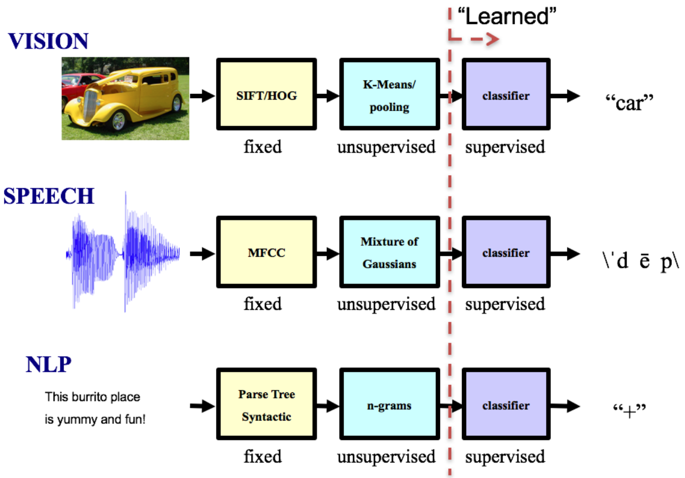


What's Deep Learning?

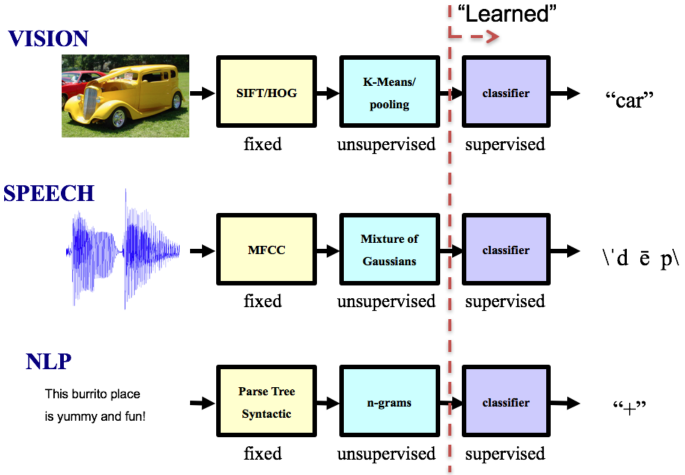


Neural nets try to mimic how humans process information.

Deep vs. Traditional Machine Learning



Deep vs. Traditional Machine Learning



Representation Learning $\Longrightarrow$ Classifier Training

Motivations for Deep Learning

Motivations for Deep Learning

1. Hierarchical Compositionality

- Cascade of non-linear transformations
- Multiple layers of representations

Motivations for Deep Learning

1. Hierarchical Compositionality

- Cascade of non-linear transformations
- Multiple layers of representations

2. Distributed Representations

- No single neuron "encodes" everything
- Groups of neurons work together

Motivations for Deep Learning

1. Hierarchical Compositionality

- Cascade of non-linear transformations
- Multiple layers of representations

2. Distributed Representations

- No single neuron "encodes" everything
- Groups of neurons work together

3. End-to-end Learning

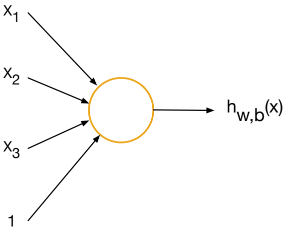
- Learning (goal-driven) representations
- Feature extraction learning

Overview

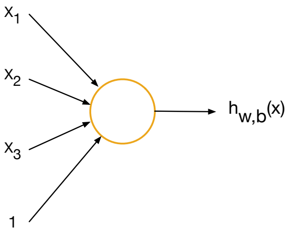
From Logistic Regression to Neural Nets

From Logistic Regression to Neural Nets

- Logistic Regression



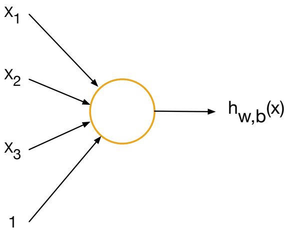
- Logistic Regression



$$x = [x_1, x_2, x_3]$$
$$h_{w,b}(x) = \sigma(w^T x + b)$$
$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

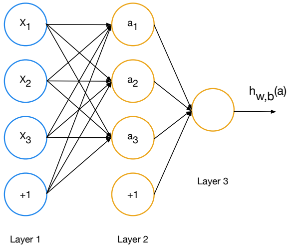
From Logistic Regression to Neural Nets

- Logistic Regression

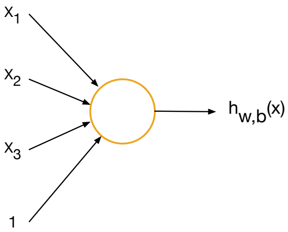


$$x = [x_1, x_2, x_3]$$
$$h_{w,b}(x) = \sigma(w^T x + b)$$
$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

- Neural Networks

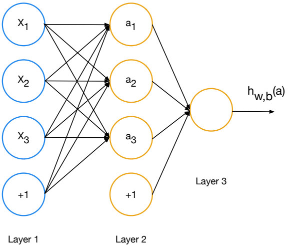


- Logistic Regression



$$x = [x_1, x_2, x_3]$$
$$h_{w,b}(x) = \sigma(w^T x + b)$$
$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

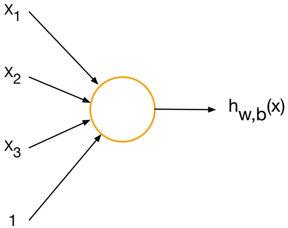
- Neural Networks



$$x = [x_1, x_2, x_3]$$
$$z = W^T x + B$$
$$a = \sigma(z)$$
$$= [\sigma(z_1), \sigma(z_2), \sigma(z_3)]$$

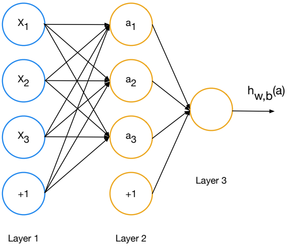
From Logistic Regression to Neural Nets

- Logistic Regression



$$x = [x_1, x_2, x_3]$$
$$h_{w,b}(x) = \sigma(w^T x + b)$$
$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

- Neural Networks



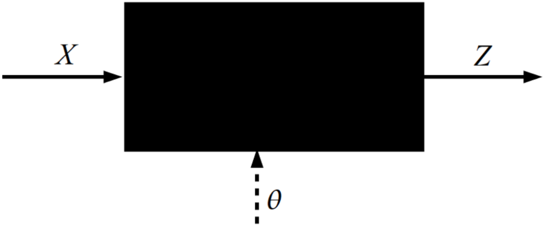
$$\begin{aligned} x &= [x_1, x_2, x_3] \\ z &= W^T x + B \\ a &= \sigma(z) \\ &= [\sigma(z_1), \sigma(z_2), \sigma(z_3)] \end{aligned}$$

Multiple Logistic Regressions

Forward/Backward Propagation

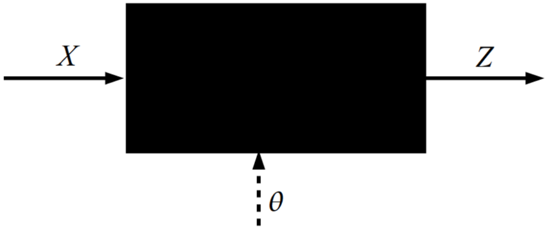
Forward/Backward Propagation

Given input, compute output:

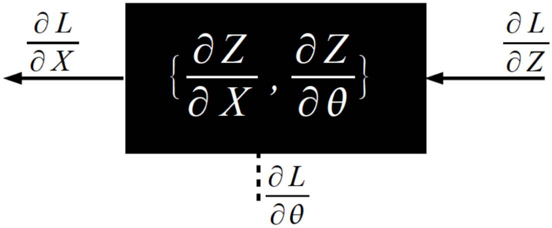


Forward/Backward Propagation

Given input, compute output:



Given ground truth, backpropagate feedbacks:



Gradient Descent

Gradient Descent

- Loss function - measure of error

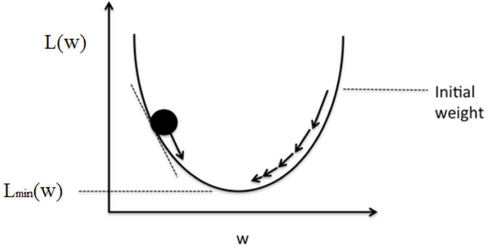
$$L(w) = -\frac{1}{N}\sum_{i=1}^{N} \log p(y^{(i)}|x^{(i)}; w), \text{ Cross Entropy}$$

Gradient Descent

- Loss function - measure of error

$$L(w) = -\frac{1}{N}\sum_{i=1}^{N} \log p(y^{(i)}|x^{(i)}; w), \text{ Cross Entropy}$$

- Optimization strategy



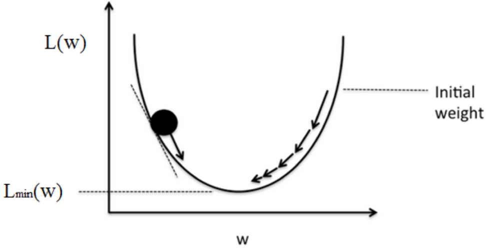
Schematic of gradient descent.

Gradient Descent

- Loss function - measure of error

$$L(w) = -\frac{1}{N}\sum_{i=1}^{N} \log p(y^{(i)}|x^{(i)}; w), \text{ Cross Entropy}$$

- Optimization strategy



Schematic of gradient descent.

$$w = w - \eta \cdot \frac{dL}{dw}, \ \eta \text{ - step size}$$

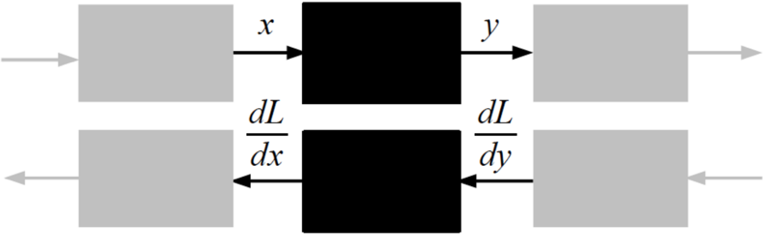
Chain Rule

Chain Rule

How to get the gradients of all the parameters?

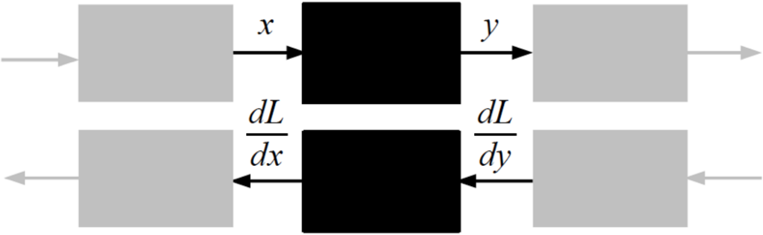
Chain Rule

How to get the gradients of all the parameters?



Chain Rule

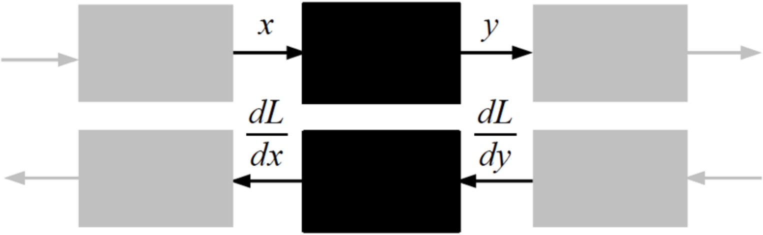
How to get the gradients of all the parameters?



Given $y(x)$ and $\frac{dL}{dy}$, what is $\frac{dL}{dx}$?

Chain Rule

How to get the gradients of all the parameters?



Given $y(x)$ and $\frac{dL}{dy}$, what is $\frac{dL}{dx}$? $\implies \frac{dL}{dx} = \frac{dL}{dy} \cdot \frac{dy}{dx}$

Overview

1. What's Deep Learning?
2. Neural Networks Recap
   - Demystifying Neural Networks
   - Forward/Backward Propagation
   - Gradient Descent & Chain Rule
3. Lego Blocks for Building NLP Deep Nets
   - Word Embedding - word2vec
   - Recurrent Neural Network - LSTM & Bidirectional
   - Recursive Neural Network
   - Convolutional Neural Network
4. Advanced Models
   - Attention Model
   - Seq2seq/End-to-end Learning

Word Embedding - word2vec

Word Embedding - word2vec

"You shall know a word by the company it keeps." (J. R. Firth 1957:11)

Word Embedding - word2vec

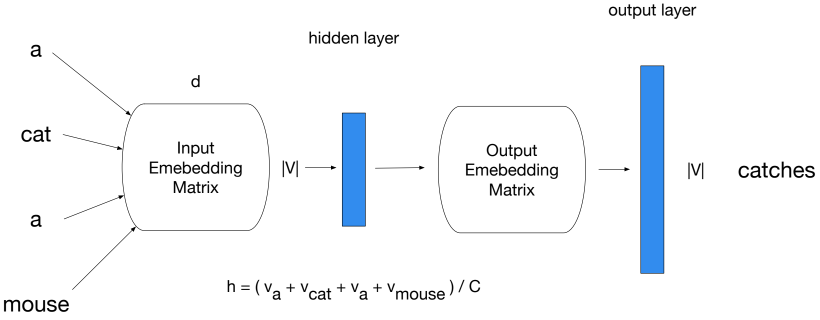
"You shall know a word by the company it keeps." (J. R. Firth 1957:11)

Example sentence: "A cat catches a mouse."

Word Embedding - word2vec

"You shall know a word by the company it keeps." (J. R. Firth 1957:11)

Example sentence: "A cat catches a mouse."



Continuous Bag-of-word (CBOW)

Mikolov, Tomas, et al. "Efficient estimation of word representations in vector space." arXiv, 2013

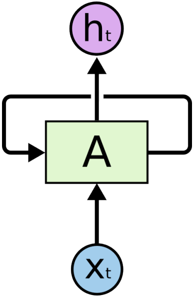
Recurrent Neural Network

Recurrent Neural Network

Tokens are composed together sequentially.

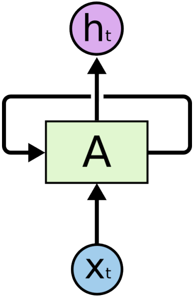
Recurrent Neural Network

Tokens are composed together sequentially.



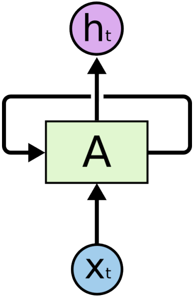
Recurrent Neural Network

Tokens are composed together sequentially.

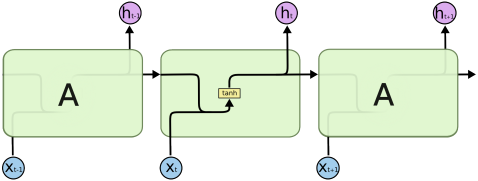


$$a^t = W_x^T x^t + W_h^T h^{t-1}$$
$$h^t = \theta(a^t)$$
$$\theta : \sigma, \text{tanh}, \text{ReLU}, \ldots$$

Tokens are composed together sequentially.



$$a^t = W_x^T x^t + W_h^T h^{t-1}$$
$$h^t = \theta(a^t)$$
$$\theta : \sigma, \mathrm{tanh}, \mathrm{ReLU}, \ldots$$



Unrolled version of vanilla RNN

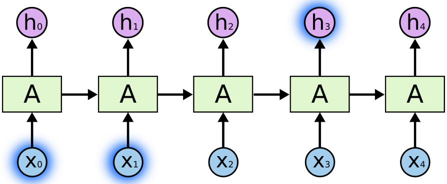
Problem of Long Term Dependencies

Problem of Long Term Dependencies

I come from China, I speak fluent

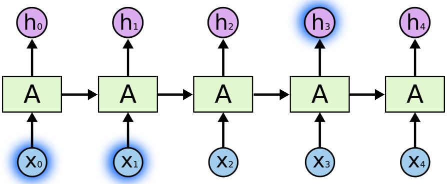
Problem of Long Term Dependencies

I come from China, I speak fluent Chinese.



Problem of Long Term Dependencies

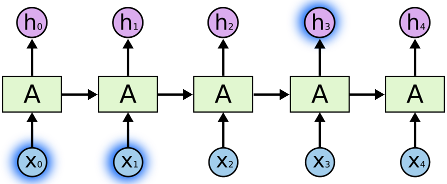
I come from China, I speak fluent Chinese.



I grew up in China . . . .

Problem of Long Term Dependencies

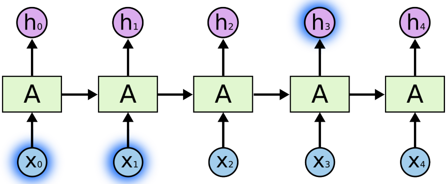
I come from China, I speak fluent Chinese.



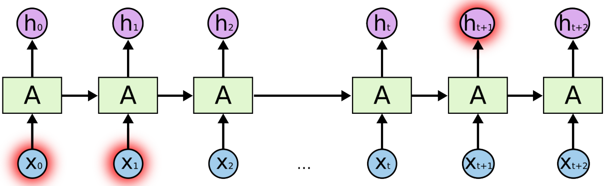
I grew up in China . . . . I speak fluent

Problem of Long Term Dependencies

I come from China, I speak fluent Chinese.

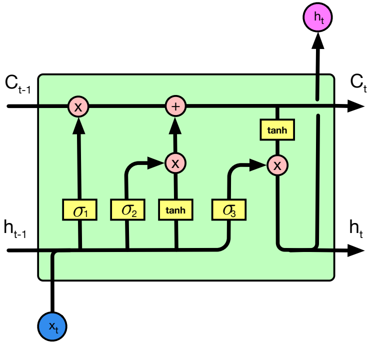


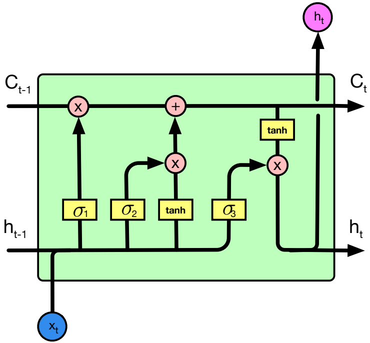
I grew up in China . . . . I speak fluent Chinese.



Long and Short Term Memory networks

Long and Short Term Memory networks



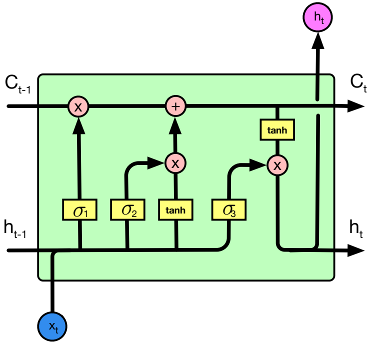


$$m_t = \sigma_1(W_f \cdot [h_{t-1}, x_t] + b_f)$$

$$i_t = \sigma_2(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$o_t = \sigma_3(W_o \cdot [h_{t-1}, x_t] + b_o)$$

Long and Short Term Memory networks



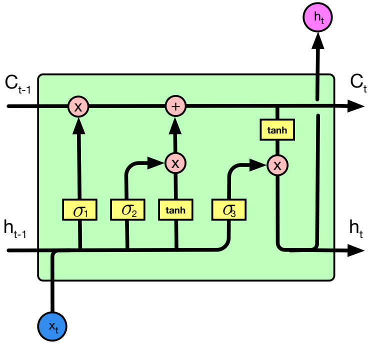
$$m_t = \sigma_1(W_f \cdot [h_{t-1}, x_t] + b_f)$$

$$i_t = \sigma_2(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$o_t = \sigma_3(W_o \cdot [h_{t-1}, x_t] + b_o)$$

$$\tilde{C}_t = \mathsf{tanh}(W_C \cdot [h_{t-1}, x_t] + b_C)$$

Long and Short Term Memory networks



$$m_t = \sigma_1(W_f \cdot [h_{t-1}, x_t] + b_f)$$

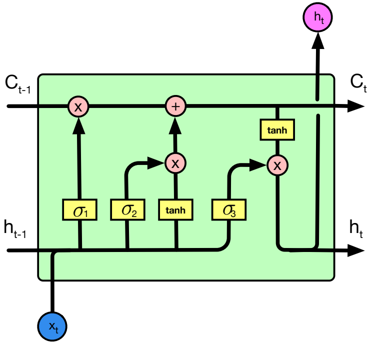
$$i_t = \sigma_2(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$o_t = \sigma_3(W_o \cdot [h_{t-1}, x_t] + b_o)$$

$$\tilde{C}_t = \mathsf{tanh}(W_C \cdot [h_{t-1}, x_t] + b_C)$$

$$C_t = m_t \odot C_{t-1} + i_t \odot \tilde{C}_t$$

Long and Short Term Memory networks



$$m_t = \sigma_1(W_f \cdot [h_{t-1}, x_t] + b_f)$$

$$i_t = \sigma_2(W_i \cdot [h_{t-1}, x_t] + b_i)$$

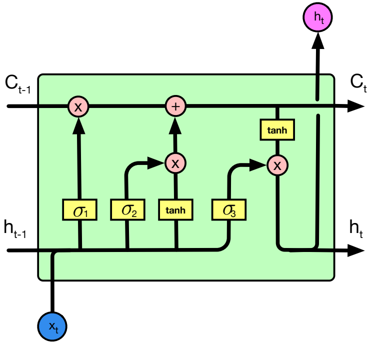
$$o_t = \sigma_3(W_o \cdot [h_{t-1}, x_t] + b_o)$$

$$\tilde{C}_t = \text{tanh}(W_C \cdot [h_{t-1}, x_t] + b_C)$$

$$C_t = m_t \odot C_{t-1} + i_t \odot \tilde{C}_t$$

$$h_t = o_t \odot \text{tanh}(C_t)$$

Long and Short Term Memory networks



$$m_t = \sigma_1(W_f \cdot [h_{t-1}, x_t] + b_f)$$

$$i_t = \sigma_2(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$o_t = \sigma_3(W_o \cdot [h_{t-1}, x_t] + b_o)$$

$$\tilde{C}_t = \mathrm{tanh}(W_C \cdot [h_{t-1}, x_t] + b_C)$$

$$C_t = m_t \odot C_{t-1} + i_t \odot \tilde{C}_t$$

$$h_t = o_t \odot \mathrm{tanh}(C_t)$$

Ideas behind LSTM

- The cell states on top memorizes long term information
- Update strengths are controlled by gates, i.e., σ function

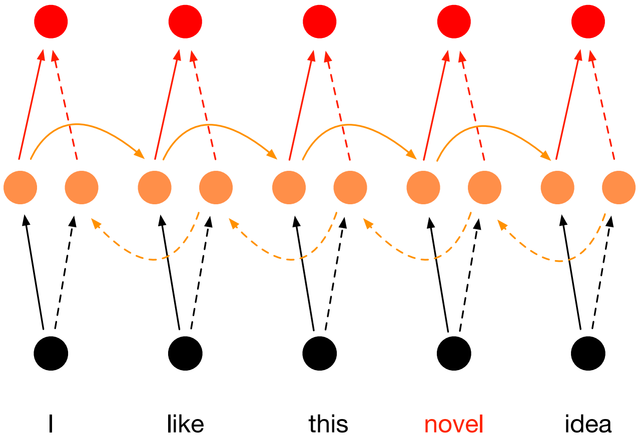
Bidirectional RNN

Bidirectional RNN

The semantics at some step not only depends on previous words, but also future ones.

Bidirectional RNN

The semantics at some step not only depends on previous words, but also future ones.



Overview

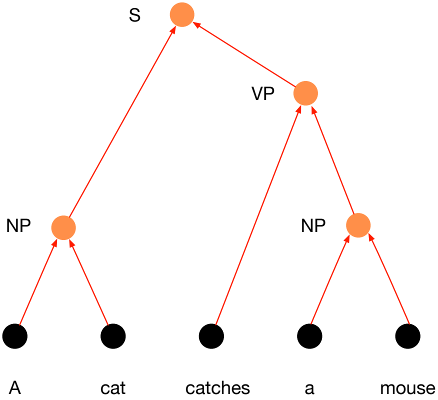
Recursive Neural Network

Recursive Neural Network

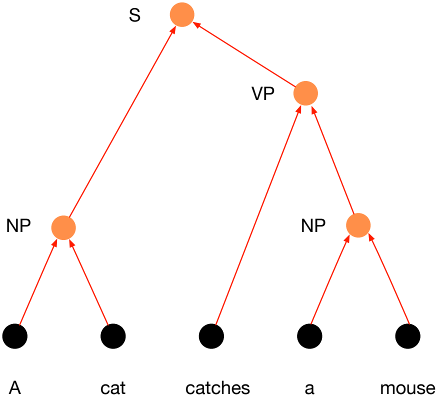
Tokens are composed based on output of dependency parser.

Recursive Neural Network

Tokens are composed based on output of dependency parser.



Tokens are composed based on output of dependency parser.

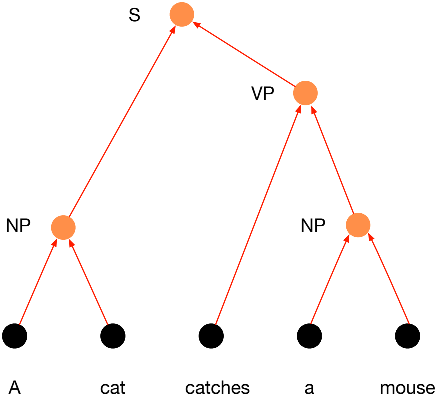


$$p = \tanh(W \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} + b)$$

$$s = U^T p$$

Recursive Neural Network

Tokens are composed based on output of dependency parser.



$$p = \tanh(W \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} + b)$$

$$s = U^T p$$

Training objective:

$$s(x_i, y_i) = \sum s$$

$$W, U = \arg\max \sum_i s(x_i, y_i)$$

Fixed W and U at all nodes

Socher, et al. "Learning continuous phrase representations and syntactic parsing with recursive neural networks." NIPS, 2010.

Overview

Convolution

Convolution

Mathematical Definition

$$(I * K)(\mu) = \int I(t) \cdot K(\mu - t)dt$$

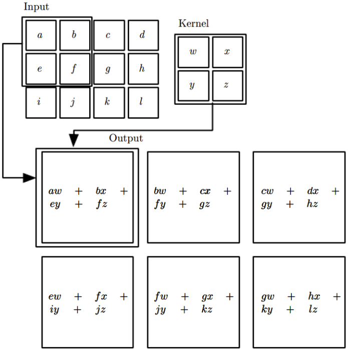
$$(I * K)(\mu) = \sum I(t) \cdot K(\mu - t)$$

Convolution

Mathematical Definition

$$(I * K)(\mu) = \int I(t) \cdot K(\mu - t)dt$$

$$(I * K)(\mu) = \sum I(t) \cdot K(\mu - t)$$

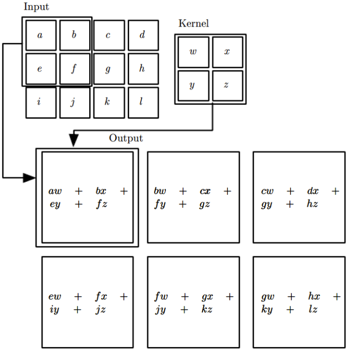


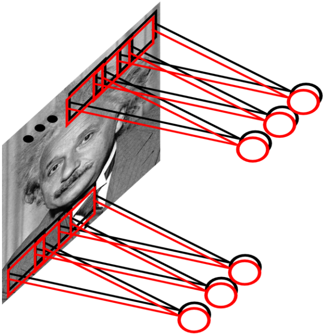
Convolution

Mathematical Definition

$$(I * K)(\mu) = \int I(t) \cdot K(\mu - t)dt$$

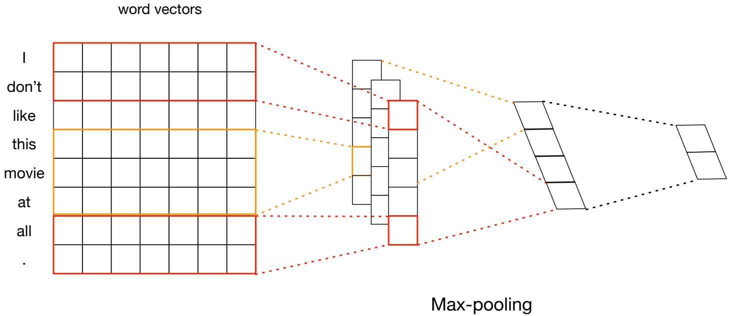
$$(I * K)(\mu) = \sum I(t) \cdot K(\mu - t)$$



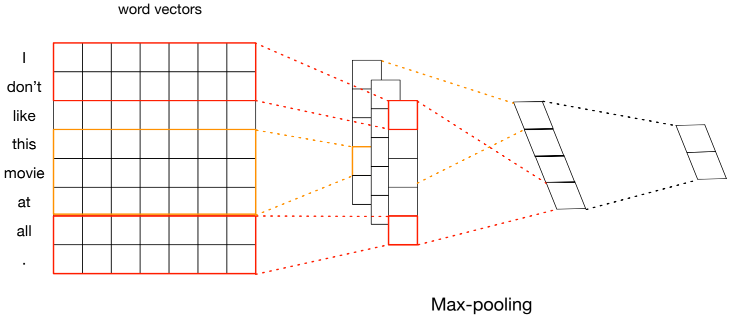


CNN Text Classifier

CNN Text Classifier

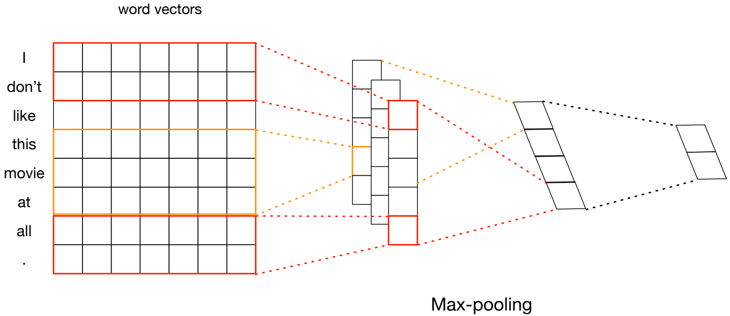


CNN Text Classifier



Motivations

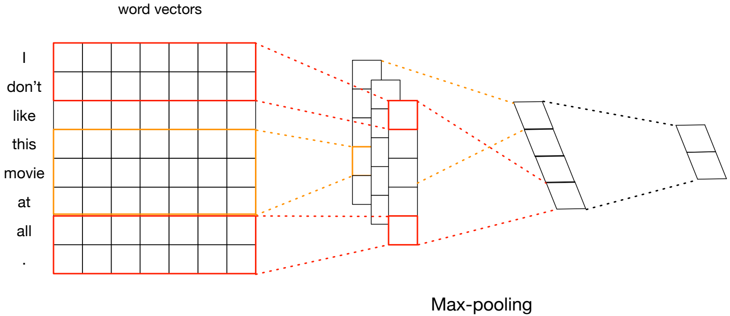
CNN Text Classifier



Motivations

- Convolution with different size learns n-grams.

CNN Text Classifier



Motivations

- Convolution with different size learns n-grams.
- Max-pooling learns the most salient features (phrases).

Kim, Yoon. "Convolutional neural networks for sentence classification." EMNLP, 2014

Overview

1. What's Deep Learning?
2. Neural Networks Recap
   - Demystifying Neural Networks
   - Forward/Backward Propagation
   - Gradient Descent & Chain Rule
3. Lego Blocks for Building NLP Deep Nets
   - Word Embedding - word2vec
   - Recurrent Neural Network - LSTM & Bidirectional
   - Recursive Neural Network
   - Convolutional Neural Network
4. Advanced Models
   - Attention Model
   - Seq2seq/End-to-end Learning

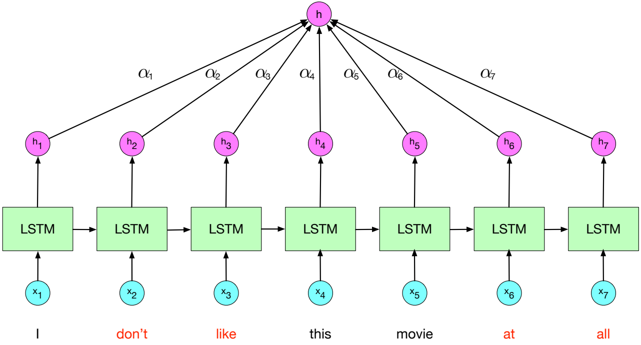
Attention Model

Attention Model

Words at different steps should have different effects.

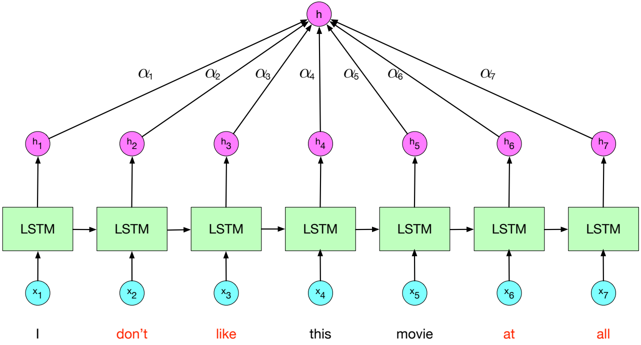
Attention Model

Words at different steps should have different effects.



Attention Model

Words at different steps should have different effects.



$$h_c = [h_1, \ldots, h_7], \; S = \tanh(W h_c)$$

$$\alpha = \mathrm{softmax}(w^T S), \; h = \sum_{i=1}^{7} \alpha_i h_i$$

Seq2seq/End-to-end Learning

Seq2seq/End-to-end Learning

Suppose we want to do machine translation.

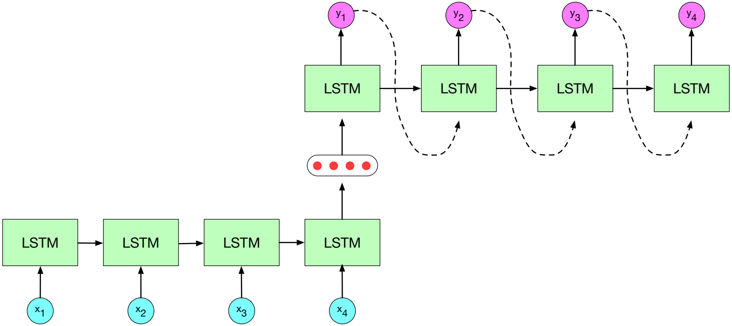
Seq2seq/End-to-end Learning

Suppose we want to do machine translation.

E.g., $(x_1, x_2, \ldots, x_4) \longrightarrow (y_1, y_2 \ldots, y_4)$

Suppose we want to do machine translation.

E.g., $(x_1, x_2, \ldots, x_4) \longrightarrow (y_1, y_2 \ldots, y_4)$



Sutskever, Ilya, et al. "Sequence to sequence learning with neural networks." NIPS, 2014

Questions?

Activity - Get familiar with neural networks

Activity - Get familiar with neural networks

Go to http://playground.tensorflow.org

1. Select dataset and split them into training and test;
2. Choose input features you want to feed into the network;
3. Determine number of layers and neurons in each layer;
4. Tune hyperparameter:
   - Learning rate - start with intermediate ones like 0.03;
   - Nonlinear activation function - ReLU is recommended;
   - Regularization: L1 or L2 norm, and its strength rate;
5. Start training and see how your model fits the data;

Activity - Get familiar with neural networks