



# Deep Learning for NLP

---

Jay Urbain, PhD

Professor, Electrical Engineering and Computer  
Science Department

Milwaukee School of Engineering

[urbain@msoe.edu](mailto:urbain@msoe.edu)

July 2018

# TOPICS: Classification

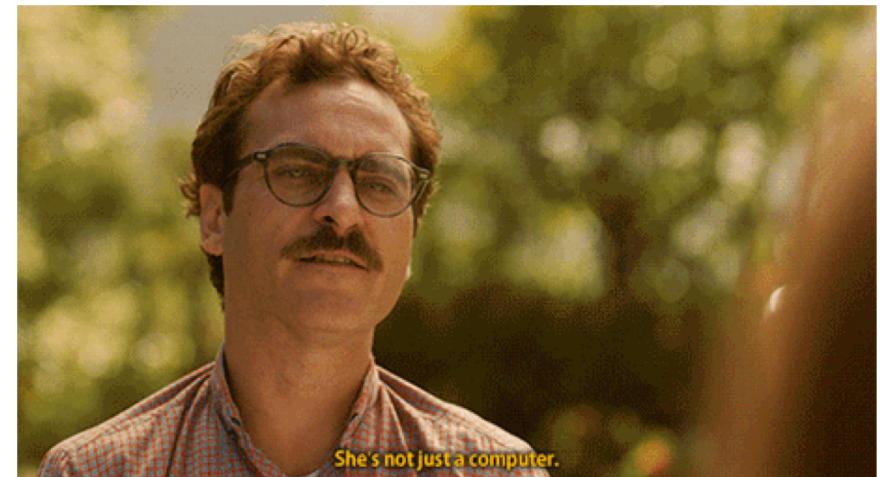
- What is Natural Language Processing
- Why is NLP hard?
- NLP Vocabulary
- Text Classification
  - Word embeddings
  - Network architectures: MLP, CNN, RNN
  - TensorFlow, Keras, Pandas

# **WHAT IS NATURAL LANGUAGE PROCESSING (NLP)?**

# WHAT IS A NATURAL LANGUAGE PROCESSING (NLP)?

Natural language processing is a field at the intersection of:

- Computer Science
- Artificial Intelligence
- Linguistics

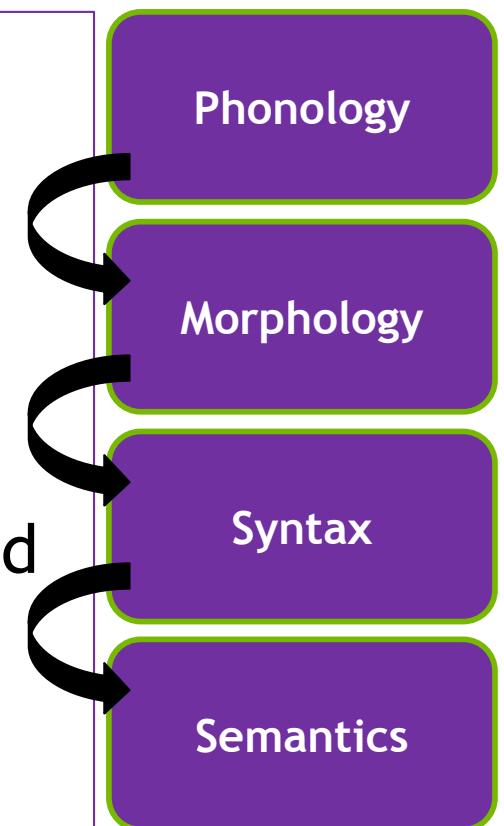


**GOAL:** for computers to process or “understand” natural language in order to perform tasks that are useful such as question answering

- Full understanding for now is unsolved

# LEVELS OF NLP

- Phonology - how do we pronounce words
- Morphology - how words are formed
- Syntax - how the sentences are constructed
- Semantics - what does all of this means



# NLP APPLICATIONS

- Keyword search
- Text classification:
- Named entity recognition:
- Machine translation
- Question answering, chatbot dialog systems
- Much more...



## NLP Service

<https://cis.ctsi.mcw.edu/nlp/>

Input text:

Jay Urbain is an elderly caucasian male suffering from illusions of grandeur and low back pain. Patient has a family history of CAD and DM. Prescribed meloxicam, and venti americano.

Data Format

Parsed results:

SENTENCE: Jay Urbain is an elderly caucasian male suffering from illusions of grandeur and low back pain .  
NNP NNP VBZ DT JJ JJ NN VBG IN NNS IN NN CC JJ NN NN  
|=====| |=====| |=====| |=====|  
Event Disorder Event Finding  
C0020903 C0030193  
|=====|  
Finding C0004604  
|=====|  
Finding C0024031

SENTENCE: Patient has a family history of CAD and DM.  
NN VBZ DT NN NN IN NN CC NN  
|=====| |=====|  
Finding Disorder  
C0262926 C1956346  
|=====|  
Finding C0241889

TLINKS: history CONTAINS CAD

SENTENCE: Prescribed meloxicam, and venti americano.  
VBN NN CC JJ NN  
|=====| |=====|  
Drug Event  
C0083381

Full Processed in 0.20900 secs

Email Us: [jay.urbain@gmail.com](mailto:jay.urbain@gmail.com)

# Question Answering

## Machine Reading for Question Answering

Experiments with deep encoder-decoder networks with memory and attention for question answering of medical records text. Select a question for the sample record, or supply a question. You may also provide your own passage of text from any source. Another passage sample from the WSJ is provided below. Modeled after the [SQuAD](#) data set. Training requires 50,000 to 400,000 training epochs using an AWS nVidia K80 or Tesla GPU instance. *Please note: this is a work in progress!*

### Passage

COLONOSCOPY Procedure Note Date of Procedure: [1\_21\_2016] Primary Physician: [PERSON] [PERSON] [PERSON], MD Attending Physician: [PERSON] [PERSON], MD Fellow:None Indications: Colon cancer screening for family history of colon cancer Colon cancer screening for family history of polyps Previous COLONOSCOPY: Yes. Date: [8\_16\_2007] Medications Administered: Agents given by the anesthesia service during MAC Procedure Details: The patient was placed in the left lateral position and monitored continuously with ECG tracing, pulse oximetry monitoring and direct observations. Medications were administered incrementally over the course of the procedure to achieve an adequate level of moderate sedation. After anorectal examination was performed, the Olympus CF H180 was inserted into the rectum and advanced under direct vision to the terminal ileum. The procedure was considered not difficult. During withdrawal examination, the final quality of the prep was good. Bowel Prep Scale Right Colon: Grade 2 (minor amount of residual staining, small fragments of stool, and/or opaque liquid, but mucosa of colon segment is

### Sample questions

How long should NSAIDs be avoided?  
What is the Histopathologic Diagnosis?  
Was the procedure difficult?  
What were the indications for colon cancer screening?  
How was the patient monitored?  
What was retroflexed evaluation used for?  
What are the indications of colon cancer?  
What are the dietary recommendations?  
How many polyps were found?  
What was the transverse colon bowel prep scale?

### Question

How long should NSAIDs be avoided?

### Answer

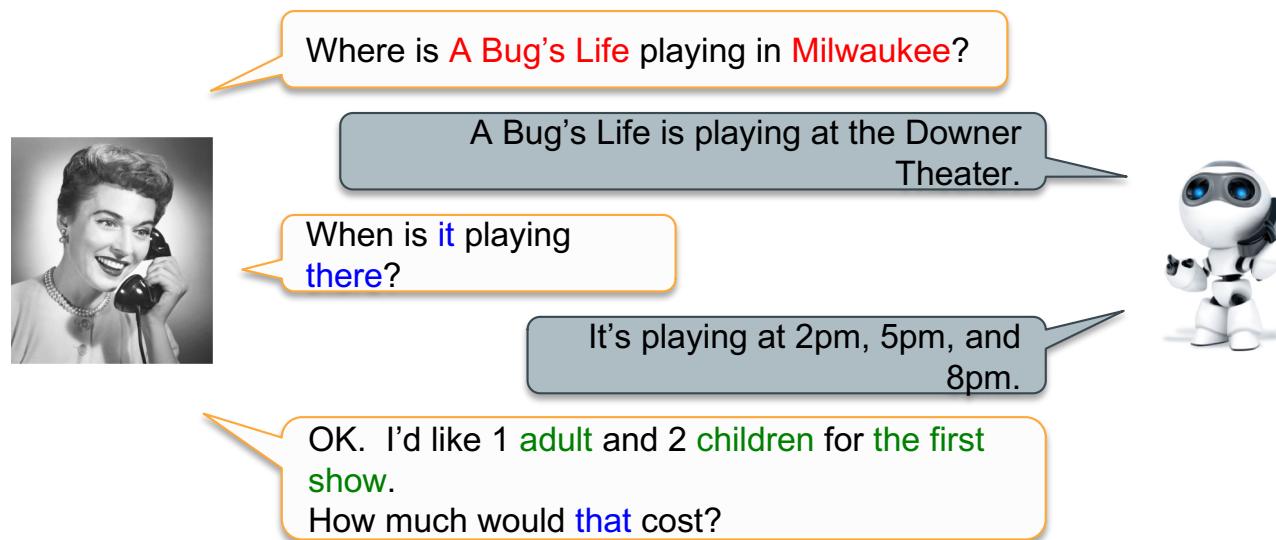
14 days

[Get Answer](#)

<http://ec2-34-201-103-221.compute-1.amazonaws.com:8080/>

**WHY IS NLP HARD?**  
... and why Deep Learning may help

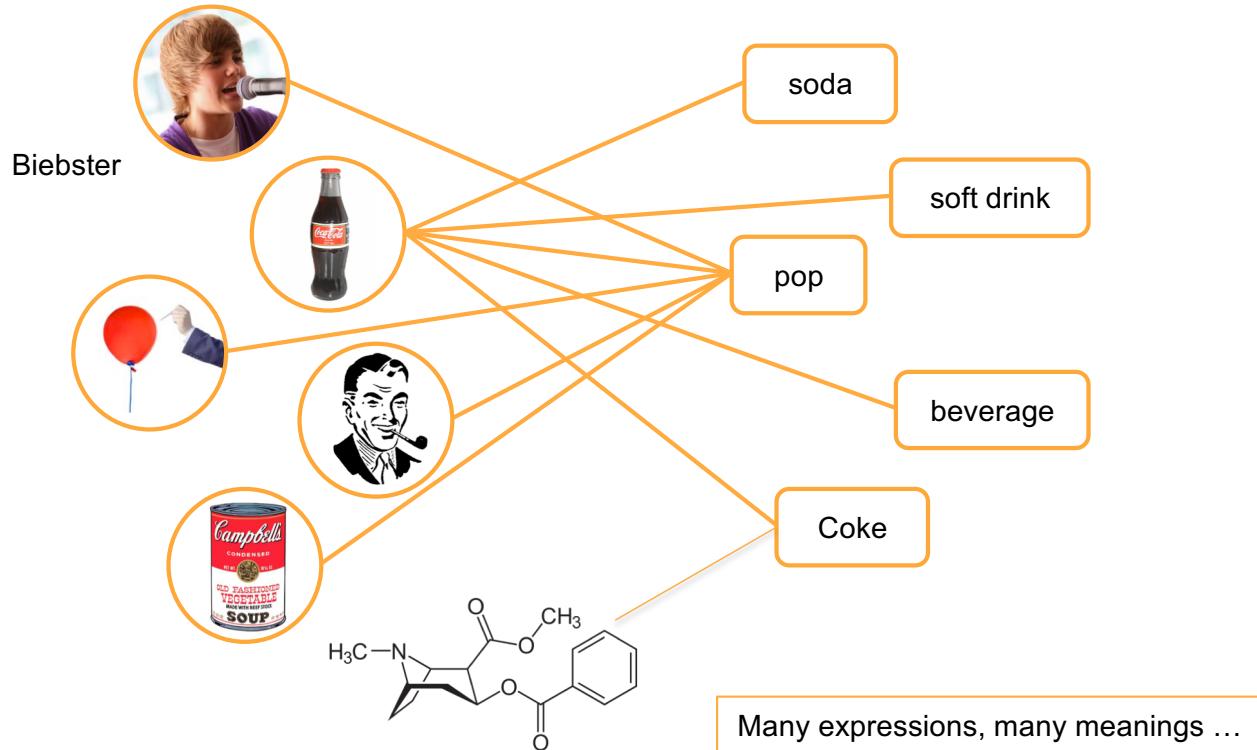
# Language: the ultimate UI



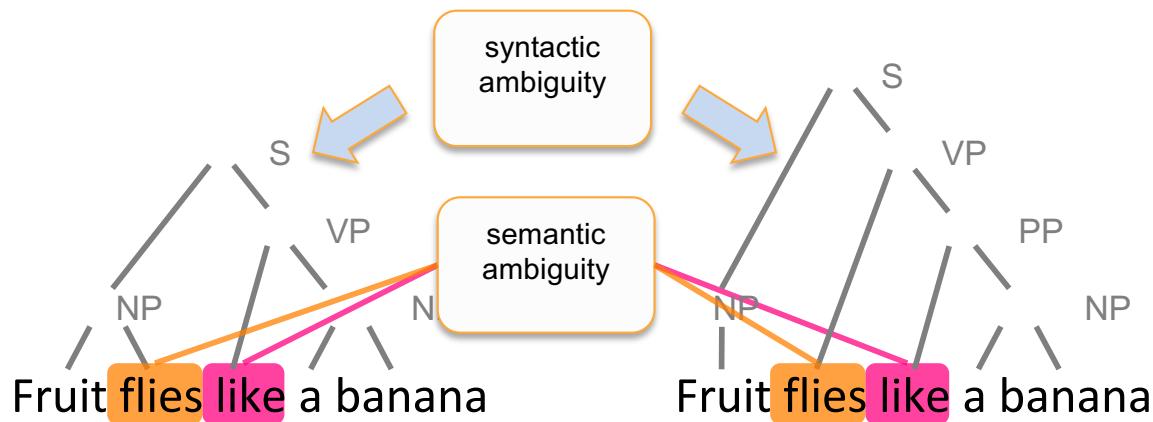
But we need **domain knowledge**, **discourse knowledge**, **world knowledge**

# Meanings and expressions

Big obstacle: relation between meanings and expressions is not one-to-one



# Syntactic & semantic ambiguity

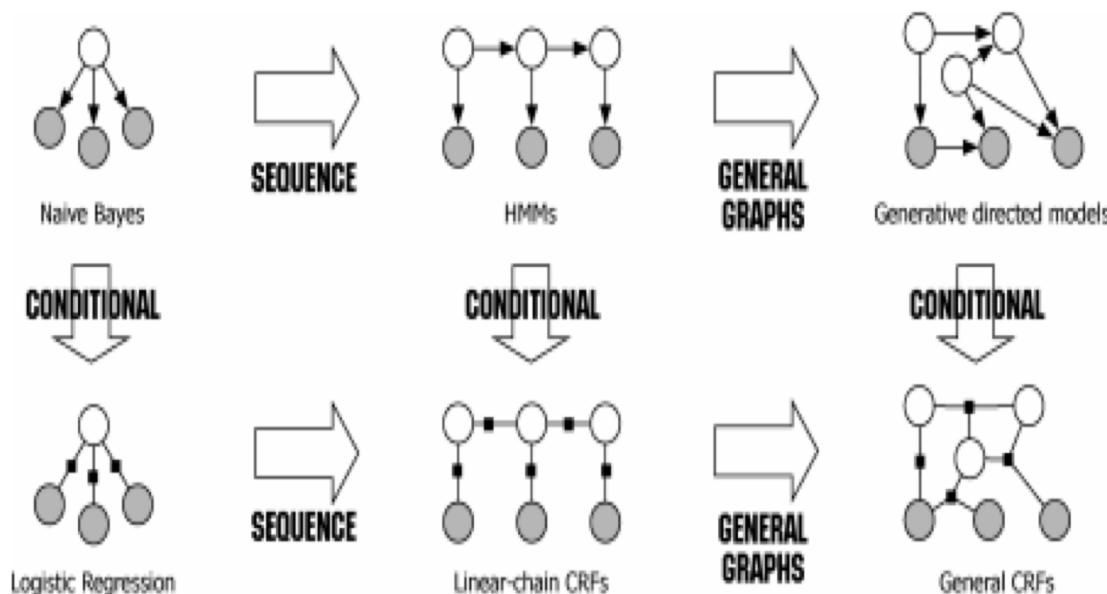


[“like” is a remarkable word: it can be used as a noun, verb, adverb, adjective, preposition, particle, conjunction, or interjection.]

photos from [worth1000.com](http://worth1000.com)

## Traditional NLP use “shallow” statistical machine learning models: HMM, MEMM, CRF

- <http://homepages.inf.ed.ac.uk/csutton/publications/crftut-fnt.pdf>



## Traditional Models: HMM, MEMM, CRF

- <http://homepages.inf.ed.ac.uk/csutton/publications/crftrt-fnt.pdf>
- Linear chain CRF for sequence class, Named entity CRF features

**Definition 2.2.** Let  $Y, X$  be random vectors,  $\theta = \{\theta_k\} \in \mathbb{R}^K$  be a parameter vector, and  $\mathcal{F} = \{f_k(y, y', \mathbf{x}_t)\}_{k=1}^K$  be a set of real-valued feature functions. Then a *linear-chain conditional random field* is a distribution  $p(\mathbf{y}|\mathbf{x})$  that takes the form:

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{t=1}^T \exp \left\{ \sum_{k=1}^K \theta_k f_k(y_t, y_{t-1}, \mathbf{x}_t) \right\}, \quad (2.18)$$

where  $Z(\mathbf{x})$  is an input-dependent normalization function

$$Z(\mathbf{x}) = \sum_{\mathbf{y}} \prod_{t=1}^T \exp \left\{ \sum_{k=1}^K \theta_k f_k(y_t, y_{t-1}, \mathbf{x}_t) \right\}. \quad (2.19)$$

|                |   |                              |
|----------------|---|------------------------------|
| $W=v$          | $w_t = v$   | $\forall v \in \mathcal{V}$  |
| $T=j$          | part-of-speech tag for $w_t$ is $j$ (as determined by an automatic tagger)                | $\forall \text{POS tags } j$ |
| $P=I-j$        | $w_t$ is part of a phrase with syntactic type $j$ (as determined by an automatic chunker) |                              |
| Capitalized    | $w_t$ matches [A-Z] [a-z] +   |                              |
| Allcaps        | $w_t$ matches [A-Z] [A-Z] +   |                              |
| EndsInDot      | $w_t$ matches [^\.] + .*  |                              |
|                | $w_t$ contains a dash   |                              |
| Acro           | $w_t$ matches [A-Z] [a-z] + [A-Z] + [a-z]   |                              |
| Stopword       | $w_t$ matches [A-Z] [A-Z \. ] * \. [A-Z \. ] *  |                              |
| CountryCapital | $w_t$ appears in a hand-built list of stop words  |                              |
| :              | many other lexicons and regular expressions   |                              |
|                | $q_k(\mathbf{x}, t + \delta)$ for all $k$ and $\delta \in [-1, 1]$                        |                              |

Custom solutions for each domain, and for each application.  
 Difficulty capturing long-range dependencies, context, and semantics

# NLP: With deep Learning, reality is getting closer to vision



# NLP VOCABULARY

# VOCABULARY

- *token* - A unit of text, typically a word. Could also be:
  - Group of words like "New York"
  - Sub-word like "mega" in "megabyte"
  - Letter like "m"
- *tokenizing* - Each token can be represented as a distinct number
- *document* - Sequence of *tokens*
  - Could be whole book or a tweet
  - Classify each *document* as having a positive or a negative sentiment
- *corpus* - A set of *documents*
  - We'll use a *corpus* containing movie reviews since each *document* is paired with a rating indicating *sentiment*

# VOCABULARY

- *n-grams* - adjoining sequence of  $n$  tokens (words)
  - Sometimes referred to as shingles
- *sentiment analysis* - evaluate mood of n-grams
  - Determine the attitude / emotion of text
- *tf-idf* = term frequency \* inverse document frequency
  - Numerical statistic intended to reflect significance of a word to a document in a corpus
  - Popular term-weighting schemes
  - TF increases each time a word appears in a document
  - IDF decreases each time a word appears in a corpus

# TEXT CLASSIFICATION

# TEXT CLASSIFICATION

- We will be training several neural network models for text classification.
- In text classification, documents or segments of text are assigned to different *categories*.
- For example, categories could be positive or negative sentiment, different authors, different topics, or different writing styles.
- Interesting problems like ambiguity arise. Examples:
  - "To sanction" can mean "to permit" or "to punish."
  - Berkshire Hathaway vs. Anne Hathaway
- Capturing context at different levels of representation is key. DL helps us here!

# WINDOW TEXT CLASSIFICATION

Classify **Paris** in the context of this sentence with window of length 2:

$$\begin{array}{ccccccc} \dots & \text{museums} & \text{in} & \text{Paris} & \text{are} & \text{amazing} & \dots \\ & \bullet\bullet\bullet & \bullet\bullet\bullet & \bullet\bullet\bullet & \bullet\bullet\bullet & \bullet\bullet\bullet & \\ X_{\text{window}} = [ & x_{\text{museums}} & x_{\text{in}} & x_{\text{Paris}} & x_{\text{are}} & x_{\text{amazing}} ]^T \end{array}$$

# EMBEDDINGS

# WORKING WITH TEXT

- Neural networks don't take raw text as input. Only numeric tensors.
- ***Vectorizing text*** is the process of transforming text into numeric tensors. This can be done in multiple ways:
  - Segment text into words, and transform each word into a vector.
  - Segment text into characters, and transform each character into a vector.
  - Extract n-grams of words or characters, and transform each n-gram into a vector.
- Apply categorical transformation:
  - One-hot encoding
  - Word embedding

# ONE-HOST ENCODING

- Words are categorical, they have no *ordinal* relationship. So they can't just be numerically encoded.
- For each word:
  - Numerically encode each word: {red:0, green:1, blue:2}
  - Generate vector of zeros the length of all possible words/categories
  - Set the index value for the word in the vector

| red, | green, | blue |
|------|--------|------|
| 1,   | 0,     | 0    |
| 0,   | 1,     | 0    |
| 0,   | 0,     | 1    |

# WORD EMBEDDINGS

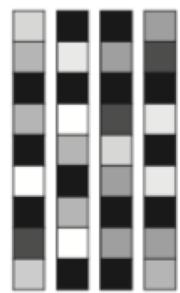
- One-hot results in sparse vector representation (mostly zeros, hi dimensions).
- Word embeddings provide dense vector representations.
- Idea is to “embed” word into a lower-dimensionality space.
- The dimensions of this space are typically defined by word context, i.e., semantically similar words are embedded near each other.
- Popular algorithms:
  - Point-wise mutual information (PMI)
  - Word2Vec: Skip-gram or CBOW
  - GLoVE - Global Vectors for Word Representation

# WORD EMBEDDINGS

- One-hot results in sparse vector representation (mostly zeros, hi dimensions).
- Word embeddings provide dense vector representations.
- Idea is to “embed” words into a lower-dimensionality space.
- The dimensions of this space are typically defined by word context, i.e., semantically similar words are embedded near each other.
- Popular algorithms:
  - Point-wise mutual information (PMI)
  - Word2Vec: Skip-gram or CBOW
  - GLoVE



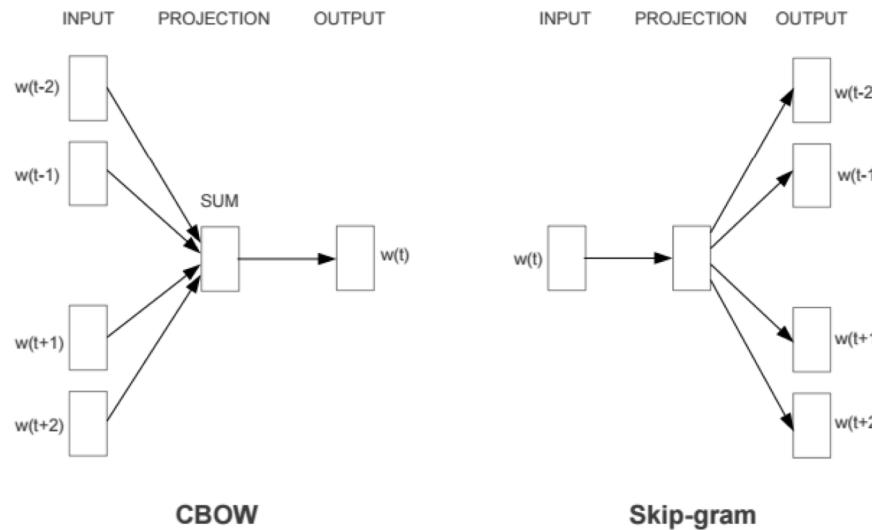
One-hot word vectors:  
- Sparse  
- High-dimensional  
- Hardcoded



Word **embeddings**:  
- Dense  
- Lower-dimensional  
- Learned from data

# Embeddings: Word2Vec

- CBOW model predict missing word (focus word) using context (surrounding words).
- Skip gram model predicts context based on the word in focus.
- Context is a fixed number of words to the left and right of the word in focus.
- Maximize average log probability of context words co-occurring with focus words.



# USING WORD EMBEDDINGS

1. Learn an Embedding
  - Use KERAS embedding layer
  
2. Reuse an Embedding
  - Download re-trained word vectors

# START SENTIMENT CLASSIFICATION LAB

- Got to text classification folder
- Open SentimentClassification.ipynb notebook

# TENSORFLOW, KERAS, PANDAS

# WHAT IS TENSORFLOW?

Created by Google, tensorflow.org

- “Open source software library for machine intelligence”
  - Available on GitHub
- Flexibility—express your computation as a data flow graph
  - If you can express it in TF syntax you can run it
- Portability—CPUs and GPUs, workstation, server, mobile
- Language options—Python, C++
- Performance—Tuned for performance on CPUs and GPUs

# KERAS

- Developed by Francois Chollet, Google Research Engineer
- Modular neural network written in Python
- Runs on TensorFlow, CNTK, and Theano (no longer supported)
- Keras library allows for easy and fast prototyping
- Runs on GPUs and CPUs
- Now included with latest TensorFlow backend

- ##### Keras
- The core data structure of Keras is a model, a way to organize layers. The simplest type of model is the Sequential model, a linear stack of layers. For more complex architectures, you should use the Keras functional API, which allows you to build arbitrary graphs of layers.
- Here is the Sequential model:
  - `````
  - `from keras.models import Sequential`
  - `model = Sequential()`
  - `````
- Stacking layers by using .add():
  - `````
  - `from keras.layers import Dense`
  - `model.add(Dense(units=64, activation='relu', input_dim=100))`

# PANDAS

- Open-source, BSD-licensed project
- Fast and efficient DataFrame object for data manipulation with integrated indexing
  - DataFrames used in this lab
  - Ideal for neural networks
- Contains tools for reading and writing data between in-memory data structures and different formats such as:
  - CSV and text files - used in this lab
  - Microsoft Excel
  - SQL databases
  - HDF5

# NETWORK ARCHITECTURE - MLP

# MULTI-LAYER PERCEPTRON

- MLP = Multi-layer Perceptron
  - Traditional *fully connected feed-forward network*
  - Also called *dense network*
  - Fixed size input

# **LAB DISCUSSION / OVERVIEW**

# LAB OVERVIEW

- Perform text classification for sentiment analysis
  - Classify each document into one of two different sentiments:
    - "positive" or "negative"
- Components of lab include:
  - Pandas
  - TensorFlow and Keras
  - MLP, CNN, RNN/LSTM

# LAB PROCESS

Data Preparation:

1. Load libraries, data and visualize / evaluate data
2. Divide dataset between Training, Validation and Test sets
3. Implement TF-IDF using Keras
4. Create matrix for labels

# LAB PROCESS

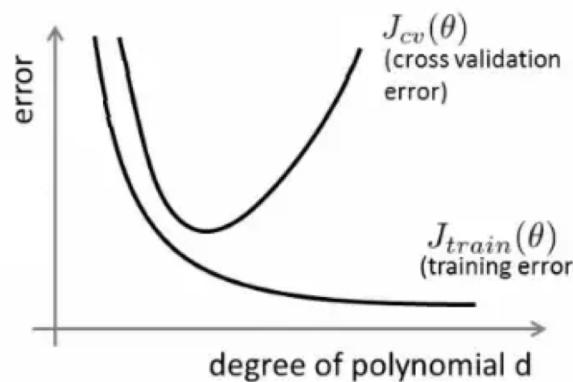
Build and evaluate the 3 network architectures:

1. Design simple neural network
  2. Visual network architecture
  3. Train model
  4. Compare test, validation and training accuracy
  5. Attempt to improve the model
- 
- Complete the Simple Multi-Layer Perceptron Model portion of the lab. Stop when you reach One-Dimensional Convolutional Neural Network Model, we'll regroup.

# **TRAINING TERMINOLOGY**

# Overfitting

- Overfitting refers to a model that fits the training data too well.
- Your model begins to “memorize” the training data and becomes less likely to generalize to unseen test data.
- Validation error  $\gg$  training error
- Correct by either simplifying model, generalizing model, or getting additional data.

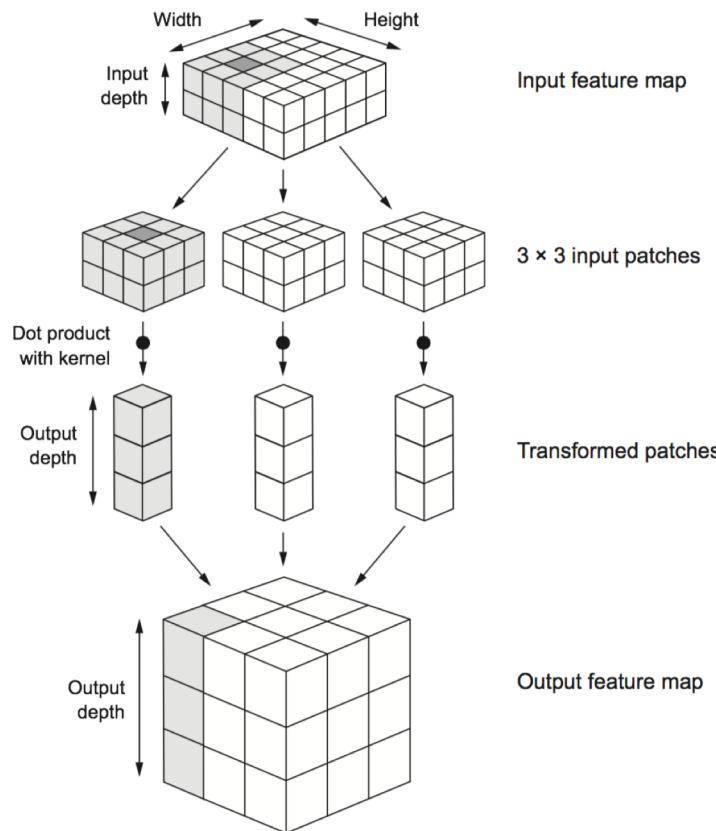


# Learning terminology

- Learning rate -- weighting of  $dE/dW$
- Decay rate - rate to reduce learning rate
- Decay steps: number of steps to execute before changing learning rate
- Num epochs - number of times to cycle through the input data
- Batch size: number of samples that going to be propagated through the network

# NETWORK ARCHITECTURE - CNNs

# Convolution Neural Network (CNN)



- A CNN is made up of Layers: convolution, pooling, fully connected.
- Each layer transforms an input 3D volume to an output 3D volume.
- Learns local parameters in input space using a stack of (e.g., 3x3) convolution filters
- The patterns the filters learn are translation invariant.
- Learn spatial hierarchies of patterns by adding conv and pooling layers. E.g., edges, shapes, motifs.

FRANÇOIS CHOLLET, 2018

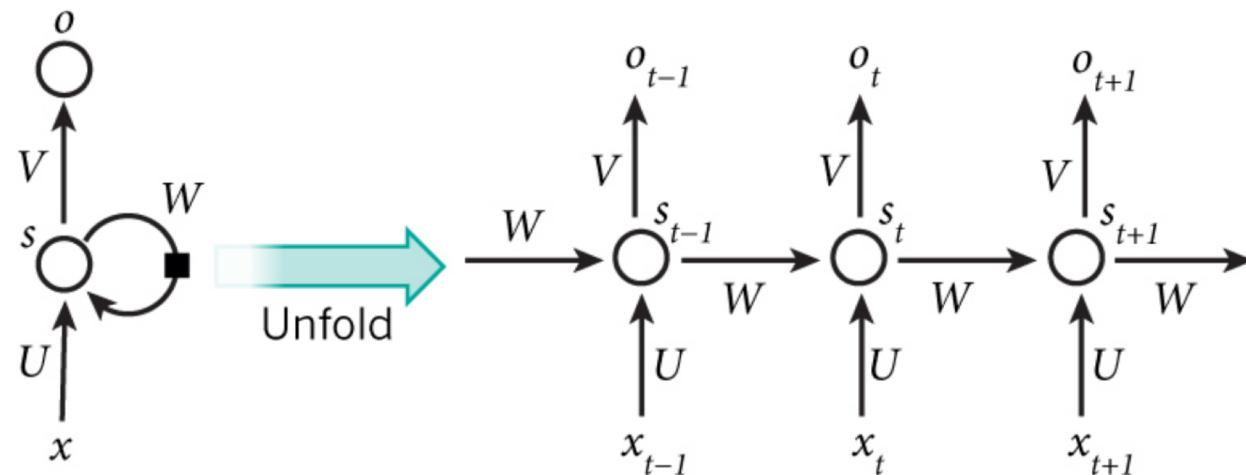
# NETWORK ARCHITECTURE - RNNS / LSTMS

# RECURRENT NEURAL NETWORK

- RNN = Recurrent Neural Network
  - Similar to traditional feed-forward network
  - RNNs include previous output state
  - Limited to looking back only a few steps due to vanishing gradient
    - Errors are backpropagated through time
    - Inputs from previous time steps get exponentially down weighted and are eventually driven to zero

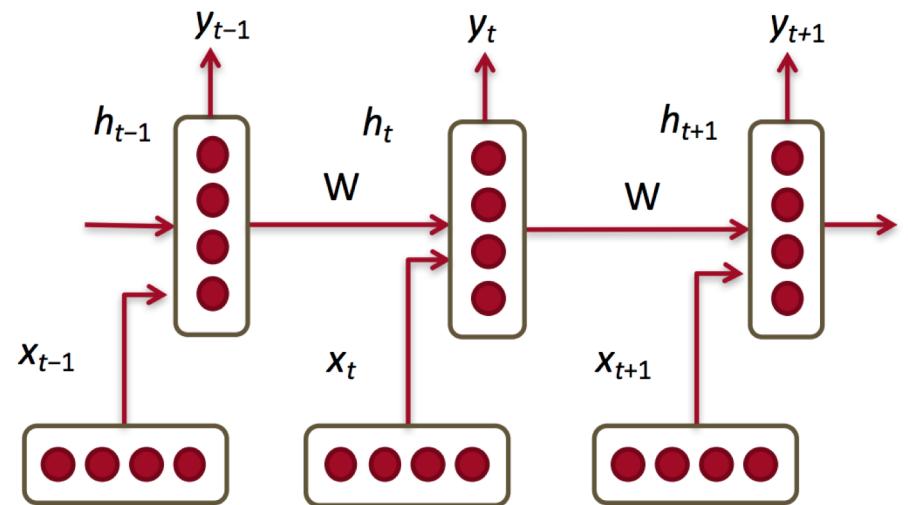
# Recurrent Neural Network (RNN)

- RNNs have connections between units along a sequence.
- RNN's use the hidden state from the last time step and the input at the current step to make predictions.
- Allow RNN's can capture dynamic temporal behavior for a time sequence.



# RECURRENT LAYERS

- RNNs share the weights between the time steps
- Condition the neural network on all previous words
  - (and all document in the end)
- RAM requirement only scales with number of timesteps

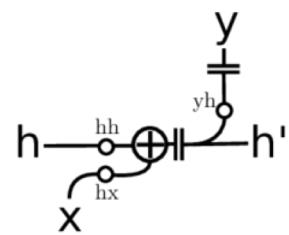


# LONG SHORT TERM MEMORY

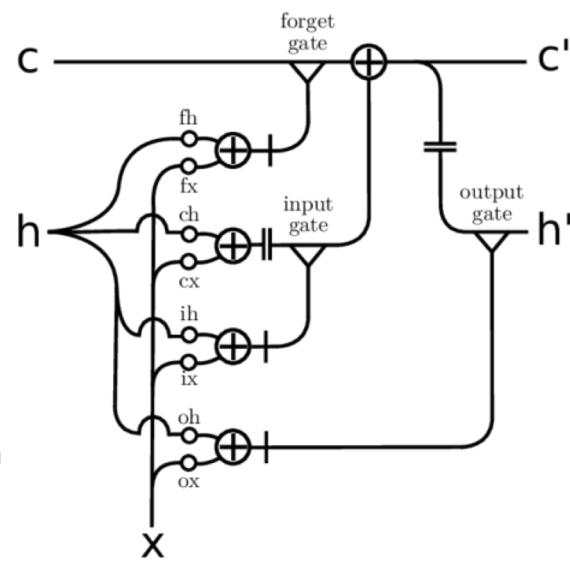
- LSTM = Long Short Term Memory
  - Variant of RNN
  - No vanishing gradient problem
  - LSTMs can learn “very deep” tasks that require memories of words N words ago

# VISUAL COMPARISON - RNN / LSTM

- $\oplus$  addition
- $+$  sigmoid
- $\text{tanh}$
- $\circ$  matrix mult.
- $\diagdown$  gating



RNN



LSTM

# Long Short Term Memory (LSTM)

- RNN's have trouble with long-term dependencies.
- Long chain of differentiation when back propagating can cause vanishing gradients
- Compose RNN with LSTM cells for "remembering" cell state over arbitrary time intervals
- Forget gate - what to forget: 
$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$
- Input gate - what to save: 
$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$
- Update cell state: 
$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$
- Output: 
$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$
$$h_t = o_t * \tanh(C_t)$$

# **LAB DISCUSSION / OVERVIEW**

# LAB PROCESS

Build and evaluate the One-Dimensional Convolutional Neural network architectures

1. Design simple neural network
  2. Visual network architecture
  3. Train model
  4. Compare test, validation and training accuracy
  5. Attempt to improve the model
- 
- If you have time, complete the RNN model. This can be a take-home exercise.

# References

- Neural Responding Machine for Short-Text Conversation (2015-03)
- A Neural Conversational Model (2015-06)
- A Neural Network Approach to Context-Sensitive Generation of Conversational Responses (2015-06)
- The Ubuntu Dialogue Corpus: A Large Dataset for Research in Unstructured Multi-Turn Dialogue Systems (2015-06)
- Building End-To-End Dialogue Systems Using Generative Hierarchical Neural Network Models (2015-07)
- A Diversity-Promoting Objective Function for Neural Conversation Models (2015-10)
- Attention with Intention for a Neural Network Conversation Model (2015-10)
- Improved Deep Learning Baselines for Ubuntu Corpus Dialogs (2015-10)
- A Survey of Available Corpora for Building Data-Driven Dialogue Systems (2015-12)
- Incorporating Copying Mechanism in Sequence-to-Sequence Learning (2016-03)
- A Persona-Based Neural Conversation Model (2016-03)
- How NOT To Evaluate Your Dialogue System: An Empirical Study of Unsupervised Evaluation Metrics for Dialogue Response Generation (2016-03)

# Learning resources

Stanford class on deep learning for  
NLP.<http://cs224d.stanford.edu/syllabus.html>

Hinton's Coursera Course. Get it right from the horse's mouth. He explains things well.

<https://www.coursera.org/course/neuralnets>

Online textbook in preparation for deep learning from  
Yoshua Bengio and friends. Clear and understandable.  
<http://www.iro.umontreal.ca/~bengioy/dlbook/>

TensorFlow tutorials.  
<https://www.tensorflow.org/versions/r0.8/tutorials/index.html>

# Additional Sources

<https://cis.ctsi.mcw.edu/deid/>

<https://cis.ctsi.mcw.edu/nlp/>

<https://github.com/jayurbain/machine-learning>

<https://github.com/jayurbain/ctsi-mcw-deid-service>

<https://github.com/jayurbain/ctsi-mcw-deid-service>