CSCE 771: Computer Processing of Natural Language
Lecture 10: Machine Learning Basics (Review),
Language Models, Word Representations

PROF. BIPLAV SRIVASTAVA, AI INSTITUTE

19TH SEPTEMBER, 2024

Carolinian Creed: "I will practice personal and academic integrity."

Organization of Lecture 10

- Opening Segment
  - Recent NLP-related talks
  - Review of Lecture 9
  - Announcements
- Main Lecture
- Concluding Segment
  - About Next Lecture – Lecture 11

Main Section
- ML Basic – Review
  - ML– Supervised
  - ML - Unsupervised
  - Neural Networks
- Language models
  - Language models and prob. parsing connections
  - RNN
  - AutoEncoders

Announcements

GUEST LECTURES ON LANGUAGE MODELS

Sep 24 (Tu)	Language Model – PyTorch, BERT, {Resume data, two tasks} **– Guest Lecture**
Sep 26 (Th)	Language Model – Finetuning, Mamba - **Guest Lecture**
Oct 1 (Tu)	Language model – comparing arch, finetuning - **Guest Lecture**
Oct 3 (Th)	Language model – comparison of results, discussion, ongoing trends– **Guest Lecture**

Main Lecture

Machine Learning – Insights from Data

- Descriptive analysis
  - Describe a past phenomenon
  - **Methods**: classification, clustering, dimensionality reduction, anomaly detection, neural methods
- Predictive analysis
  - Predict about a new situation
  - **Methods**: time-series, neural networks
- Prescriptive analysis
  - What an agent should do
  - **Methods**: simulation, reinforcement learning, reasoning

- New areas
  - Counterfactual analysis
  - Causal Inferencing
  - Scenario planning

Machine Learning – Label Based View

- Label available – Supervised Learning
  - Example: Classification

- Label unavailable – Unsupervised Learning
  - Example: Clustering

Common Textual Data Processing Steps for ML

- Input: strings / documents/ corpus
- Processing steps (task dependent / optional - *)
  - Parsing
  - Word pre-processing
    - Tokenization – getting tokens for processing
    - Normalization* - making into canonical form
    - Case folding* – handling cases
    - Lemmatization* – handling variants (shallow)
    - Stemming* – handling variants (deep)
  - Semantic parsing – representations for reasoning with meaning *
  - Embedding – creating vector representation*

Classification

Classifier Method Types

- Individual methods
  - Decision Tree
  - Naïve Bayes
- Ensemble
  - Bagging: Aggregate classifiers (“bootstrap aggregation” => bagging)
    - Random Forest
    - Samples are chosen with replacement (bootstrapping), and combined (aggregated) by taking their average
  - Gradient Boosting: aggregate to turn weak learners into strong learners
    - Boosters (aggregators) turn weak learners into strong learners by focusing on where the individual weak models (decision trees, linear regressors) went wrong
    - Gradient Boosting
    - XGBoost: “eXtreme Gradient Boosting.”

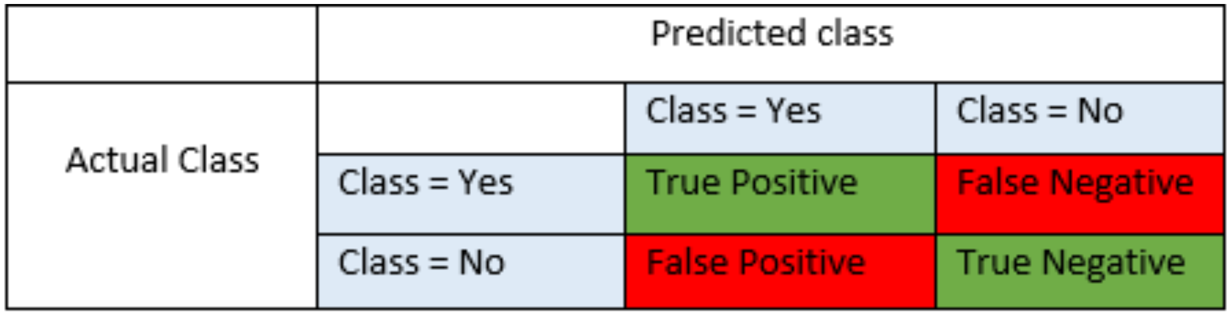
Source:
- Data Mining: Concepts and Techniques, by Jiawei Han and Micheline Kamber
- https://towardsdatascience.com/getting-started-with-xgboost-in-scikit-learn-f69f5f470a97

ML - Supervised

- By Example:
  - https://github.com/biplav-s/course-nl/blob/master/l9-ml-review/Classification%20-%20Fake%20news.ipynb

- Fake news dataset

Metrics: Accuracy, Precision, Recall



		Predicted class	
Actual Class		Class = Yes	Class = No
	Class = Yes	True Positive	False Negative
	Class = No	False Positive	True Negative

Accuracy =
(TP+TN)/
(TP+FP+FN+TN)

Precision =
(TP)/
(TP+FP)

Recall =
(TP)/
(TP+FN)

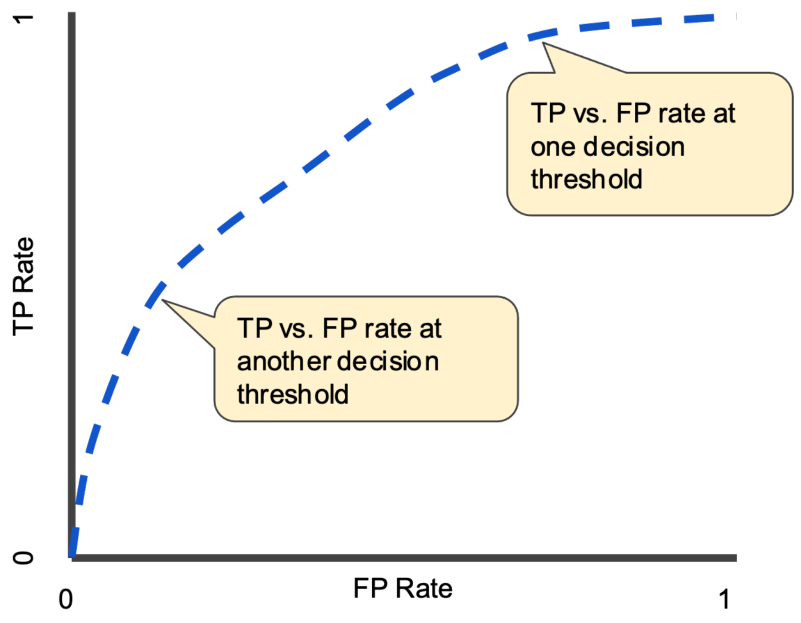
F1 Score: *Harmonic Mean*

1/F1 = 1/Precision + 1/Recall

F1 = 2*(Recall * Precision) /
(Recall + Precision)

ROC – Receiver Operating Characteristic curve

An ROC curve plots TPR vs. FPR at different classification thresholds



True Positive Rate = Recall =
(TP)/
(TP+FN)

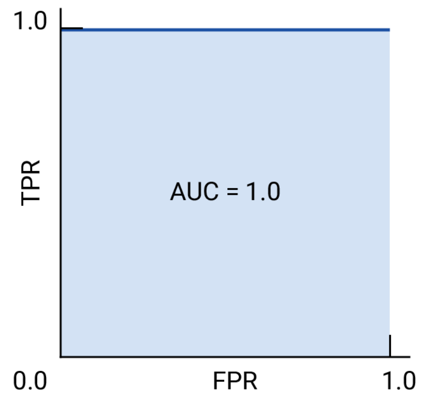
False Positive Rate =
(FP)/
(FP+TN)

	Predicted class	
Actual Class		Class = Yes	Class = No
	Class = Yes	True Positive	False Negative
	Class = No	False Positive	True Negative

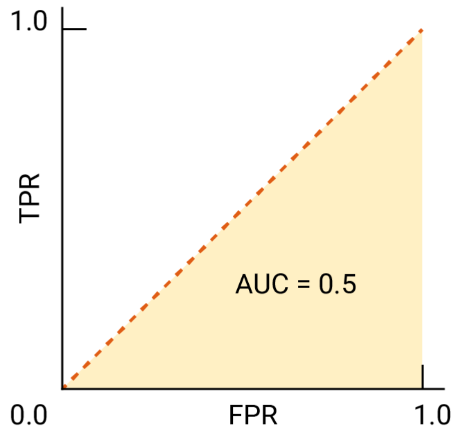
Source: https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc

AUC/ ROC Examples

ROC and AUC of a perfect system



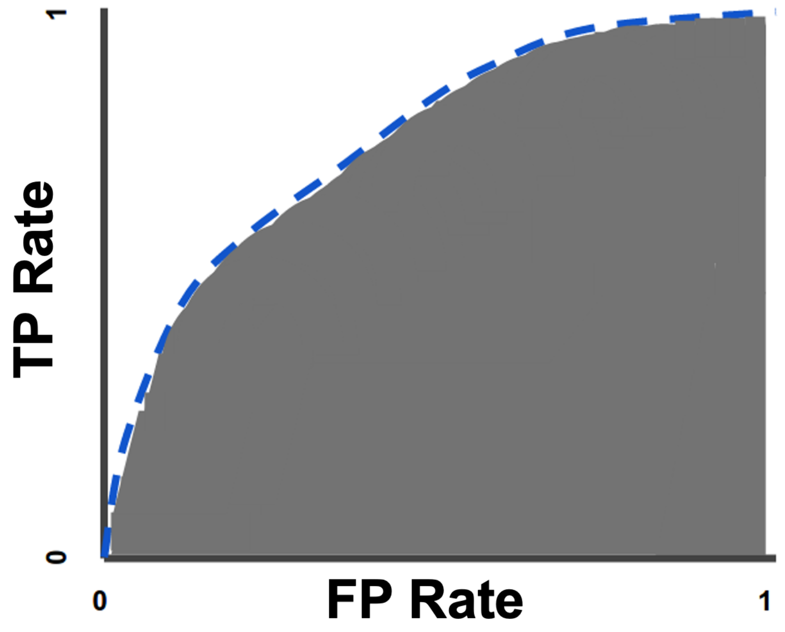
ROC and AUC of completely random guesses



The AUC is 0.5, representing a 50% probability of correctly ranking a random positive and negative example

Source: https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc

AUC – Area Under the ROC Curve



- Aggregate measure of performance across all possible classification thresholds.
- Interpretation: probability that the model ranks a random positive example more highly than a random negative example

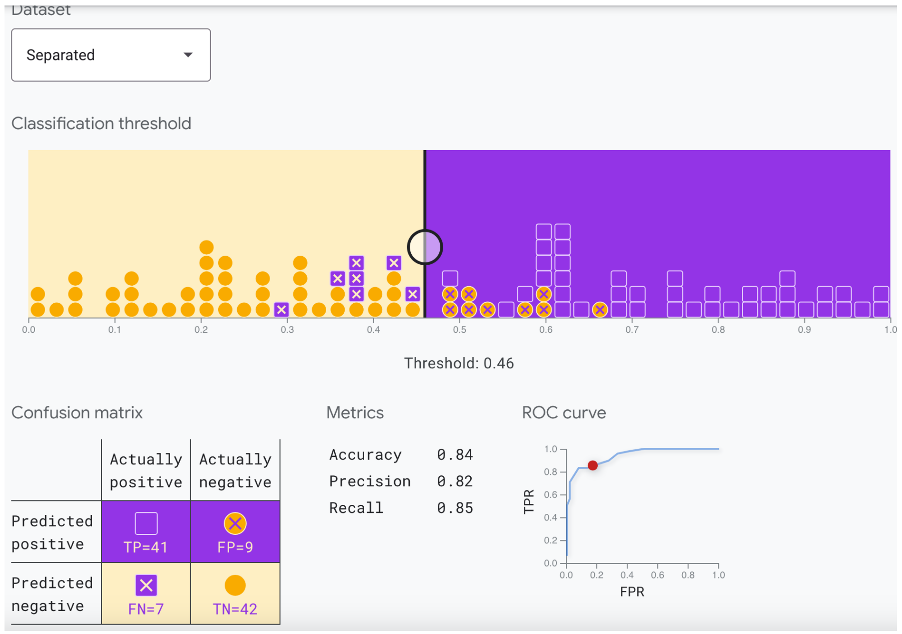
Not helpful when the **cost** of false negatives vs. false positives are asymmetric

Source: https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc

Exercise and References

- Google: https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc
  - Take quiz

- Blogs: https://blog.exsilio.com/all/accuracy-precision-recall-f1-score-interpretation-of-performance-measures/



Dataset: Separated

Classification threshold

Threshold: 0.46

Confusion matrix

	Actually positive	Actually negative
Predicted positive	TP=41	FP=9
Predicted negative	FN=7	TN=42

Metrics

Accuracy	0.84
Precision	0.82
Recall	0.85

ROC curve

Image credit: https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc

Clustering

Unsupervised Machine Learning

- Group data into clusters/ classes without supervision
  - Limited supervision

- What is a good cluster ?
  - Samples within a cluster should be “**near**” to each other (**cohesiveness**)
  - Samples in a cluster should be “**far**” from other samples in other clusters. (**distinctiveness**)

Clustering for Data Understanding and Applications

- Biology: taxonomy of living things: kingdom, phylum, class, order, family, genus and species
- Information retrieval: document clustering
- Land use: Identification of areas of similar land use in an earth observation database
- Marketing: Help marketers discover distinct groups in their customer bases, and then use this knowledge to develop targeted marketing programs
- City-planning: Identifying groups of houses according to their house type, value, and geographical location
- Earth-quake studies: Observed earth quake epicenters should be clustered along continent faults
- Climate: understanding earth climate, find patterns of atmospheric and ocean
- Economic Science: market resarch

Content: Jiawei Han, Micheline Kamber and Jian Pei
Data Mining: Concepts and Techniques, 3rd ed.

Clustering as a Preprocessing Tool (Utility)

- Summarization:
  - Preprocessing for regression, PCA, classification, and association analysis
- Compression:
  - Image processing: vector quantization
- Finding K-nearest Neighbors
  - Localizing search to one or a small number of clusters
- Outlier detection
  - Outliers are often viewed as those “far away” from any cluster

Content: Jiawei Han, Micheline Kamber and Jian Pei Data Mining: Concepts and Techniques, 3rd ed.

Considerations for a Clustering Algorithm

- Need a distance measure for ***far*** and ***near***
- Be able to explain what a cluster means
- Handle different types of attributes: numeric, categorical (nominal, ordinal), binary
- Detect different shapes of clusters
- Handle noisy data
- Scale
  - Size
  - Dimensions

Major Clustering Approaches (I)

Partitioning approach:

- Construct various partitions and then evaluate them by some criterion, e.g., minimizing the sum of square errors
- Typical methods: **k-means**, k-medoids, CLARANS

Hierarchical approach:

- Create a hierarchical decomposition of the set of data (or objects) using some criterion
- Typical methods: Diana, Agnes, **BIRCH**, CAMELEON

Density-based approach:

- Based on connectivity and density functions
- Typical methods: **DBSACN**, OPTICS, DenClue

Grid-based approach:

- based on a multiple-level granularity structure
- Typical methods: STING, WaveCluster, CLIQUE

Content: Jiawei Han, Micheline Kamber and Jian Pei Data Mining: Concepts and Techniques, 3rd ed.

Major Clustering Approaches (II)

Model-based:

- A model is hypothesized for each of the clusters and tries to find the best fit of that model to each other
- Typical methods: **EM**, SOM, COBWEB

Frequent pattern-based:

- Based on the analysis of frequent patterns
- Typical methods: p-Cluster

User-guided or constraint-based:

- Clustering by considering user-specified or application-specific constraints
- Typical methods: COD (obstacles), constrained clustering

Link-based clustering:

- Objects are often linked together in various ways
- Massive links can be used to cluster objects: **SimRank**, LinkClus

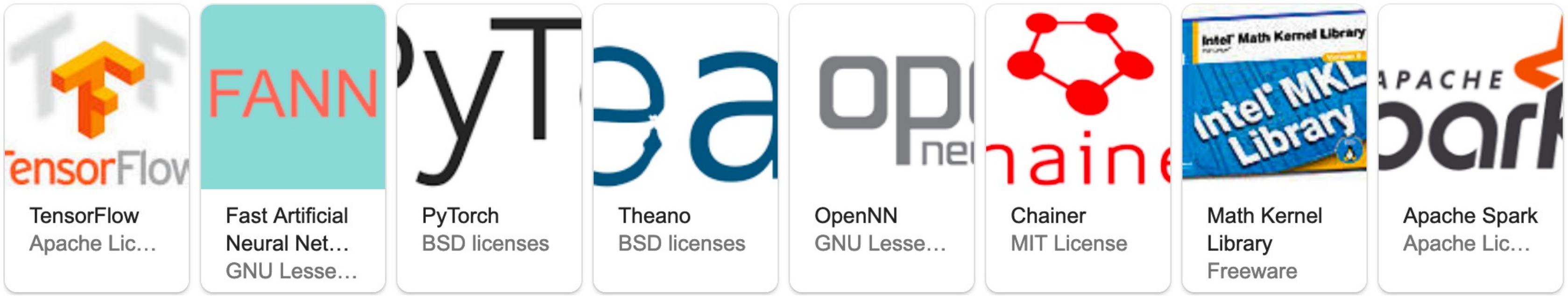
Content: Jiawei Han, Micheline Kamber and Jian Pei
Data Mining: Concepts and Techniques, 3rd ed.

ML - UnSupervised

- Example 1
  - https://github.com/biplav-s/course-nl-f22/tree/main/sample-code/l10-ml-review
  - Two mysterious datasets

- Example 2
  - https://github.com/biplav-s/course-nl/blob/master/l9-ml-review/Clustering%20-%20Fake%20news%20Illustration.ipynb
  - Data
    - Bank agreement dataset
    - Fake news dataset

Neural Network Methods



TensorFlow
Apache Lic…

Fast Artificial Neural Net…
GNU Lesse…

PyTorch
BSD licenses

Theano
BSD licenses

OpenNN
GNU Lesse…

Chainer
MIT License

Math Kernel Library
Freeware

Apache Spark
Apache Lic…

Logistic Regression in a Slide

Function estimate (linear)
W: weight, b: bias

$$f(X_j) = X_j W + b$$

Update Weight

$$W^* = W - \eta \frac{dL}{dW}$$

Error Term (mean squared error)

$$MSE = \frac{1}{n} \sum_{j=1}^{n} \left[f(X_{j\cdot}) - y_j \right]^2$$

Common Code Pattern

```
y = tf.matmul(x, W) + b
loss = tf.reduce_mean(tf.square(y - y_label))
```

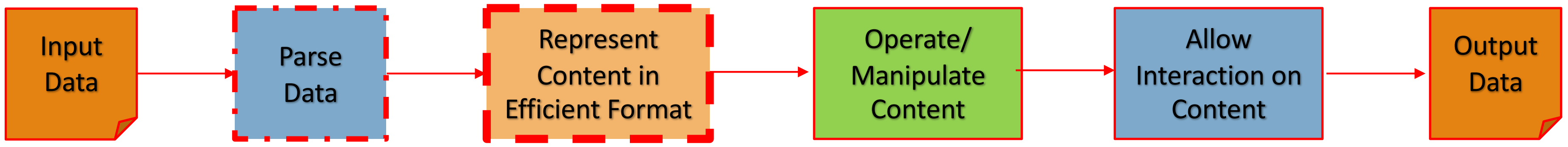
Keras and TensorFlow

- By Example:
  - https://github.com/biplav-s/course-nl/blob/master/l9-ml-review/Basic%20TensorFlow%20and%20Keras.ipynb
- TensorFlow's NMIST tutorial
  - https://www.tensorflow.org/tutorials/quickstart/beginner
- More examples
  - Number Addition by sequence learning: https://keras.io/examples/nlp/addition_rnn/
  - AutoEncoder: https://machinelearningmastery.com/lstm-autoencoders/

Discussion

- Platforms/ Tools
  - Periodic changes
  - Interoperation of models
- Data
- Compute resources
  - Development and test
  - Production

Language Model


Input Data
Parse Data
Represent Content in Efficient Format
Operate/ Manipulate Content
Allow Interaction on Content
Output Data

Connection: PCFG and Language Model

The probability of an ambiguous sentence S is the sum of the probabilities of all the parse trees for the sentence:

$$P(S) = \sum_{T\, s.t.\, S=\text{yield}(T)} P(T,S)$$

$$= \sum_{T\, s.t.\, S=\text{yield}(T)} P(T)$$

But a PCFG also assigns a probability to the substrings of a sentence

From Jurafsky & Martin

Language Model

Problem:
Given a sentence fragment, predict what word(s) come next

Applications:
- Spelling correction
- speech recognition
- machine translation,
- …

Language Model:
estimate probability of substrings of a sentence

$$P(w_i | w_1, w_2, \ldots, w_{i-1}) = \frac{P(w_1, w_2, \ldots, w_{i-1}, w_i)}{P(w_1, w_2, \ldots, w_{i-1})}$$

Bigram approximation

$$P(w_i | w_1, w_2, \ldots, w_{i-1}) \approx \frac{P(w_{i-1}, w_i)}{P(w_{i-1})}$$

From Jurafsky & Martin

Language Model

Markovify library
https://github.com/jsvine/markovify

Language Model:
estimate probability of substrings of a sentence

$$P(w_i | w_1, w_2, ..., w_{i-1}) = \frac{P(w_1, w_2, ..., w_{i-1}, w_i)}{P(w_1, w_2, ..., w_{i-1})}$$

See code samples with Markovify library on Github
- *Prepare data – two datasets shown*
- *Try generator:*
  - *https://github.com/biplav-s/course-nl/blob/master/l7-language/code/TryMarkovifyLangModel.ipynb*

Preq-requisites for Understanding Advanced Language Models

- Advanced language models need pre-requisites to understand
  - BERT, Transformers, GPT-2, GPT-3, GPT-4
  - More architectures: Mamba, ..

- Understand word representation
- Understand context representation
- Understand machine learning/ neural methods

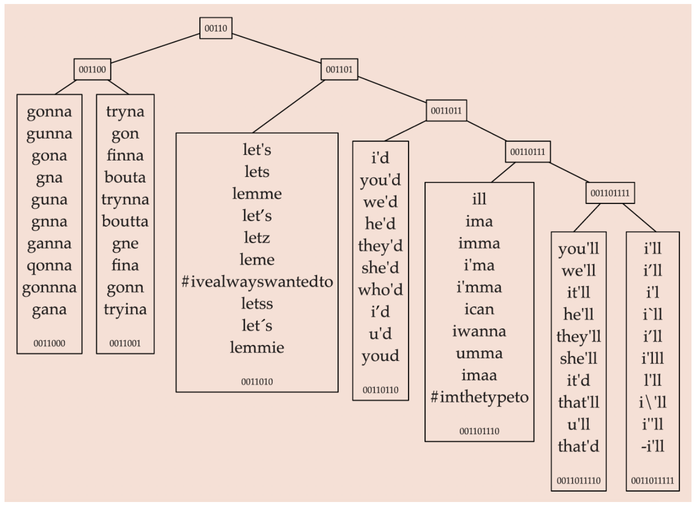
Commentary: http://jalammar.github.io/illustrated-gpt2/

Contextual Word Embeddings

- Words as discrete
- Words with distributional assumptions:
  - Context: given a word, its nearby words or sequences of words
  - Words used in similar ways are likely to have related meanings; i.e., words used in the same (similar) context have related meanings
    - No claim about meaning except relative similarity v/s dis-similarity of words

Contextual Representation by Clustering

- Cluster words by context
- Compare with words in a manually-created taxonomy, e.g., Wordnet



00110
001100
001101
0011011
00110111
001101111
gonna gunna gona gna guna gnna ganna qonna gonnna gana
0011000
tryna gon finna bouta trynna boutta gne fina gonn tryina
0011001
let's lets lemme let's letz leme #ivealwayswantedto letss let´s lemmie
0011010
i'd you'd we'd he'd they'd she'd who'd i'd u'd youd
00110110
ill ima imma i'ma i'mma ican iwanna umma imaa #imthetypeto
001101110
you'll we'll it'll he'll they'll she'll it'd that'll u'll that'd
0011011110
i'll i'll i'l i`ll i'll i'lll I'll i\'ll i"ll -i'll
0011011111

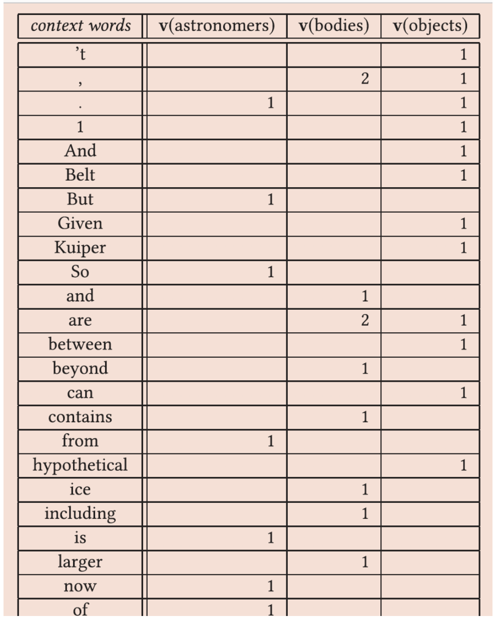
The 10 most frequent words in clusters in the section of the hierarchy with prefix bit string 00110.

Owoputi, O., O'Connor, B., Dyer, C., Gimpel, K., Schneider, N., and Smith, N.A. Improved part-ofspeech tagging for online conversational text with word clusters. In Proceedings of 2013 NAACL.

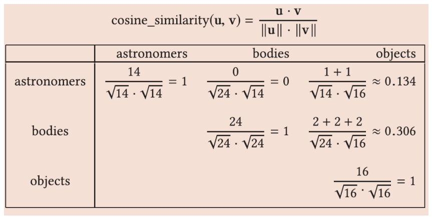
Credit:
Contextual Word Representations: Putting Words into Computers", by Noah Smith, CACM June 2020

Contextual Representation by Dimensionality Reduction

- Creating word vectors in which each dimension corresponds to the frequency the word type occurred in some context.
- Strategy 1: select contexts
  - Examples
    - Custom methods
    - TF-IDF
  - Approach
    - Use words
      - Words in the neighborhood
      - Words of specific types
    - Build vectors
    - Use vector operations to derive meaning



context words	**v**(astronomers)	**v**(bodies)	**v**(objects)
’t			1
,		2	1
.	1		1
1			1
And			1
Belt			1
But	1		
Given			1
Kuiper			1
So	1		
and		1	
are		2	1
between			1
beyond		1	
can			1
contains		1	
from	1		
hypothetical			1
ice		1	
including		1	
is	1		
larger		1	
now	1		
of	1		



$$\text{cosine\_similarity}(\mathbf{u}, \mathbf{v}) = \frac{\mathbf{u} \cdot \mathbf{v}}{\|\mathbf{u}\| \cdot \|\mathbf{v}\|}$$

	astronomers	bodies	objects
astronomers	$\frac{14}{\sqrt{14} \cdot \sqrt{14}} = 1$	$\frac{0}{\sqrt{24} \cdot \sqrt{14}} = 0$	$\frac{1+1}{\sqrt{14} \cdot \sqrt{16}} \approx 0.134$
bodies		$\frac{24}{\sqrt{24} \cdot \sqrt{24}} = 1$	$\frac{2+2+2}{\sqrt{24} \cdot \sqrt{16}} \approx 0.306$
objects			$\frac{16}{\sqrt{16} \cdot \sqrt{16}} = 1$

Bodies and **objects** are **most similar** (0.306) than

- **Bodies** and **astronomers** (0)
- **Objects** and **astronomers** (0.134)

Credit:
Contextual Word Representations: Putting Words into Computers”, by Noah Smith, CACM June 2020

Contextual Representation by Dimensionality Reduction - 1

- Strategy 2: learn contexts from documents. Vector size is given as input

- Train a neural network to learn vector representation
  - value placed in each dimension of each word type's vector is a parameter that will be optimized
  - Selection of parameter values is done using iterative algorithms / gradient descent
  - **Hope** is that ***different senses*** in which a word is used will be captured through the learning procedure as long as the dataset is large enough to represent all senses. Paper quotes: 30 meanings of **get**

- **Optionally**: Sometime task specific inputs are given during pre-processing, processing or post-processing

Disadvantage: individual dimensions are no longer interpretable

Contextual Representation by Dimensionality Reduction -2

- Strategy 2: learn contexts from documents. Vector size is given as input

Sometime task specific inputs are given during pre-processing, processing or post-processing

- Pre-processing
  - Vector initialization by pre-training. Called **finetuning**
- Processing
  - **Knowledge-infusion** (emerging area)
- Post-processing
  - Adjust output vectors so that word types that are related in reference taxonomy (like WordNet) are closer to each other in vector space. Called **retrofitting**.

Credit:
Contextual Word Representations: Putting Words into Computers", by Noah Smith, CACM June 2020

Recap: Language Model

Problem:
Given a sentence fragment, predict what word(s) come next

Applications:
- Spelling correction
- speech recognition
- machine translation,
- …

Language Model:
estimate probability of substrings of a sentence

$$P(w_i|w_1, w_2, \ldots, w_{i-1}) = \frac{P(w_1, w_2, \ldots, w_{i-1}, w_i)}{P(w_1, w_2, \ldots, w_{i-1})}$$

Bigram approximation

$$P(w_i|w_1, w_2, \ldots, w_{i-1}) \approx \frac{P(w_{i-1}, w_i)}{P(w_{i-1})}$$

From Jurafsky & Martin

Recap: Logistic Regression in a Slide

Function estimate (linear)
W: weight, b: bias

$$f(X_j) = X_j W + b$$

Update Weight

$$W^* = W - \eta \frac{dL}{dW}$$

Error Term (mean squared error)

$$MSE = \frac{1}{n} \sum_{j=1}^{n} \left[f(X_{j\cdot}) - y_j \right]^2$$

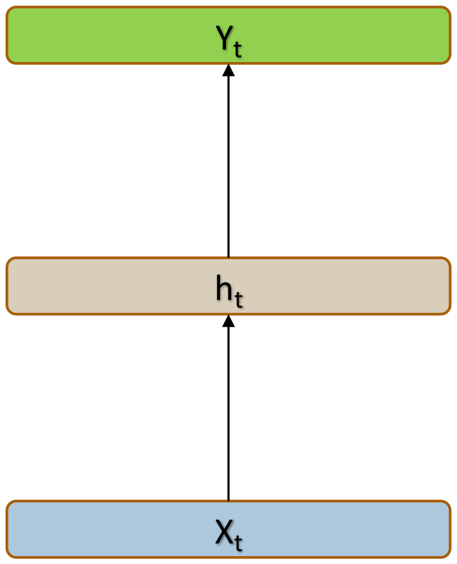
Common Code Pattern

```
y = tf.matmul(x, W) + b
loss = tf.reduce_mean(tf.square(y - y_label))
```

(Feed forward) NN

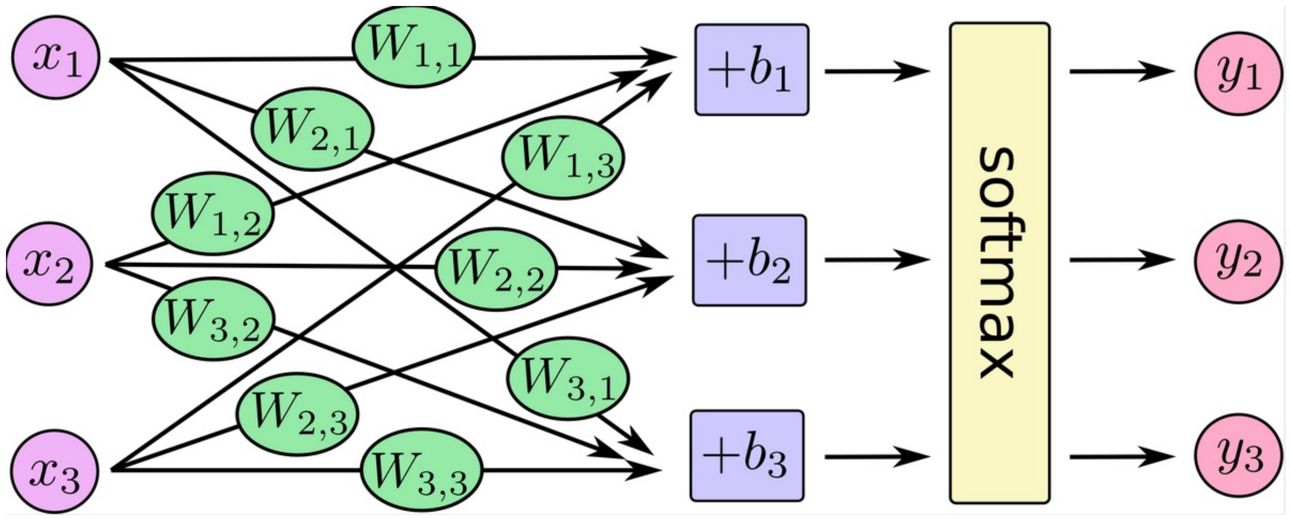
Propagation

$$f(X_j) = X_j W + b$$



Intuitive Description: https://jalammar.github.io/visual-interactive-guide-basics-neural-networks/, https://jalammar.github.io/feedforward-neural-networks-visual-interactive

Using (Feed forward) NN



Softmax

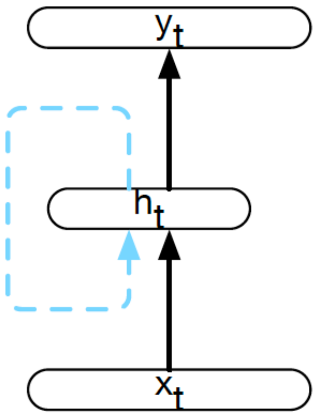
$$f(x)= \frac{1}{1+e^{-x}}$$

Source; see also : https://jalammar.github.io/visual-interactive-guide-basics-neural-networks/, https://jalammar.github.io/feedforward-neural-networks-visual-interactive

RNN - Recurrent Neural Networks

- **Recurrence**: A *recurrence* relation is an equation that defines a sequence based on a rule that gives the next term as a *function* of the previous term(s). [https://mathinsight.org/definition/recurrence_relation]

- Simple Recurrent NN or Elman Network

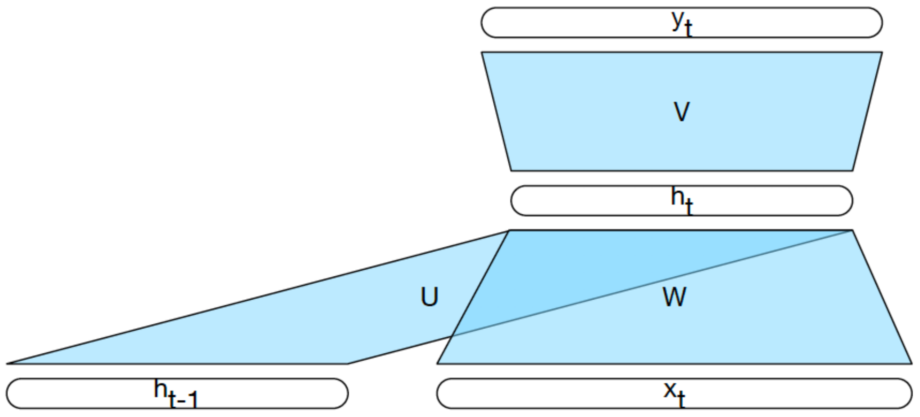


Source: Jurafsky and Martin

RNN

Recurrence unrolled

U, W, V are Weights to be learned



$$
\begin{aligned}
h_t &= g(Uh_{t-1} + Wx_t) \\
y_t &= f(Vh_t)
\end{aligned}
$$

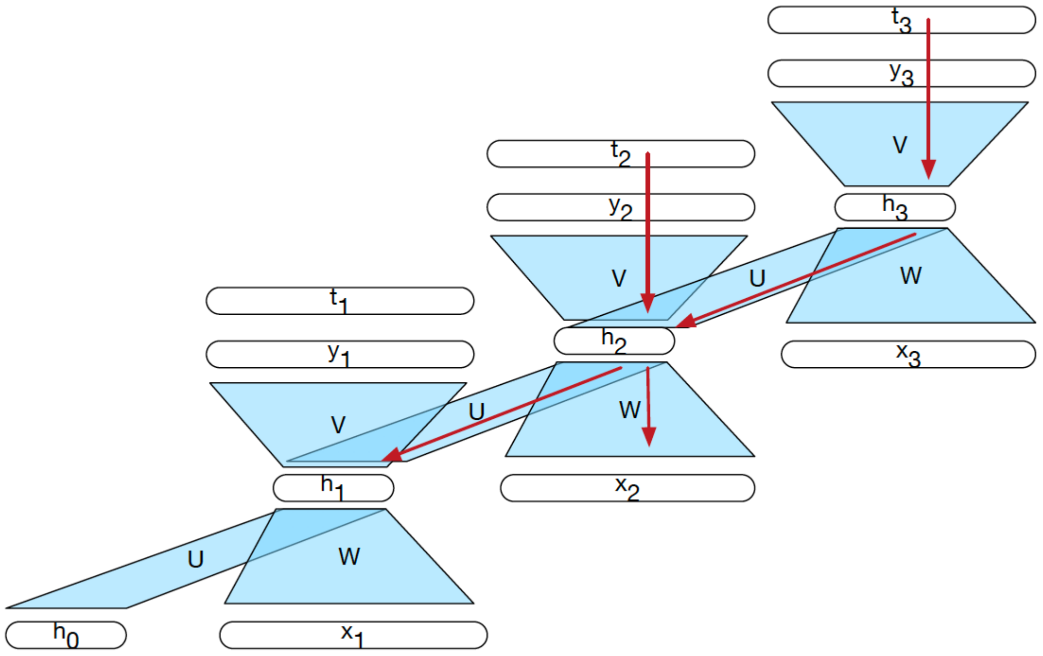
$$y_t = softmax(Vh_t)$$

Source: Jurafsky and Martin

RNN Backpropagation of Errors

Recurrence unrolled

U, W, V are Weights to be learned



Source: Jurafsky and Martin

RNN-based Language Model

- Based on characters or words
- At each step (i.e., character or word)
  - the network retrieves a word embedding for the current word as input
  - combines it with the hidden layer from the previous step to
    - compute a new hidden layer
    - generate an output layer which is passed through a softmax layer to generate a probability distribution over the entire vocabulary.

$$\begin{aligned} P(w_n|w_1^{n-1}) &= y_n \\ &= softmax(Vh_n) \end{aligned}$$

Prob. of a word

$$\begin{aligned} P(w_1^n) &= \prod_{k=1}^{n} P(w_k|w_1^{k-1}) \\ &= \prod_{k=1}^{n} y_k \end{aligned}$$

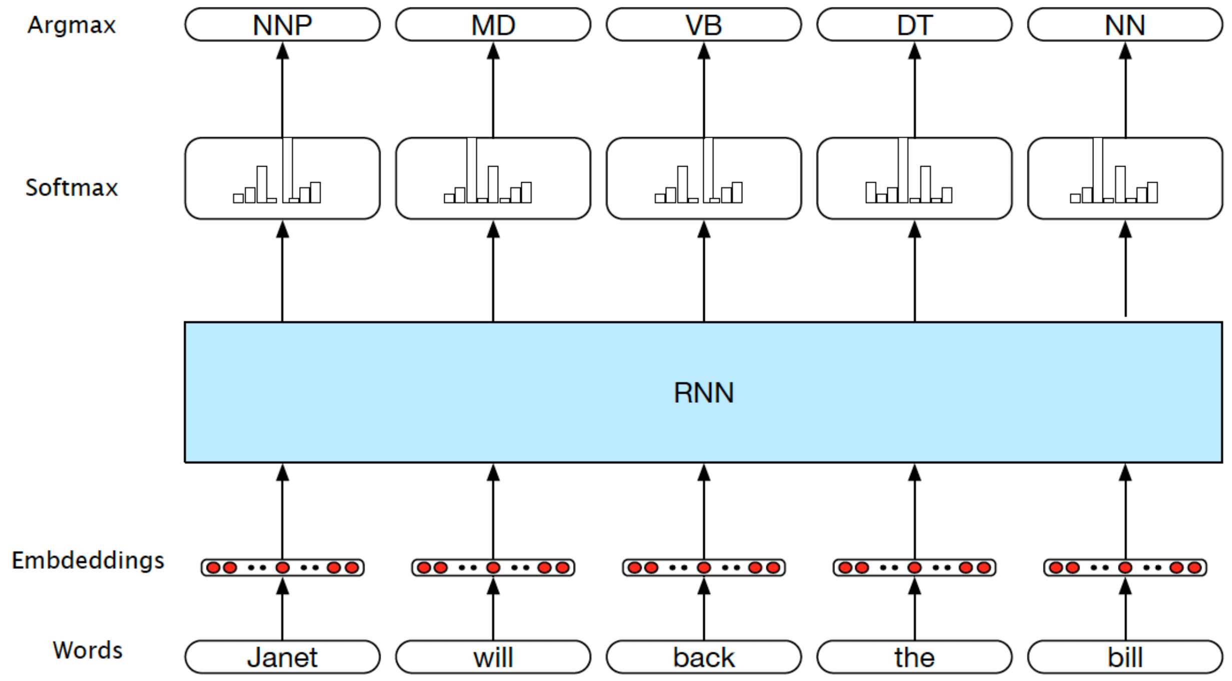
Prob. of a sequence

Source: Jurafsky and Martin

RNN Discussion

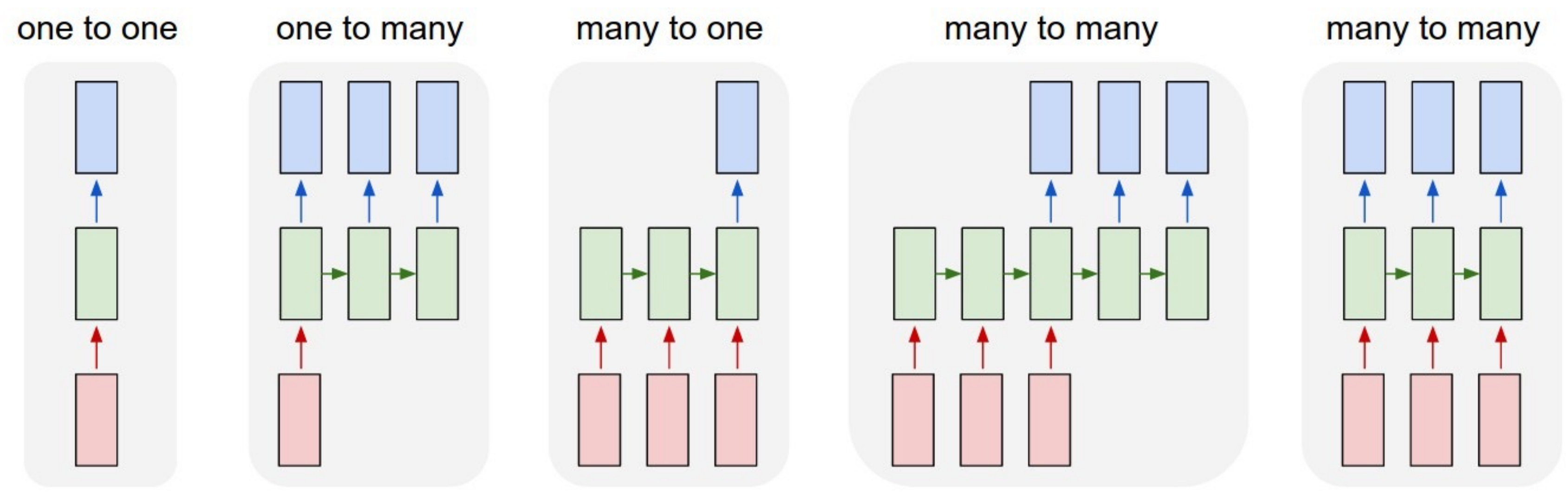
- Language model
  - Not dependent on N-gram boundaries
  - Whole sequence is the context
- Program generation
  - Complexity is Turing-complete
  - In practical terms: On the Practical Computational Power of Finite Precision RNNs for Language Recognition, Gail Weiss, Yoav Goldberg, Eran Yahav, ACL 2018, https://www.aclweb.org/anthology/P18-2117/

RNN Usage Example: Sentence Labeling



Source: Jurafsky and Martin

RNN - Many Applications



Language model

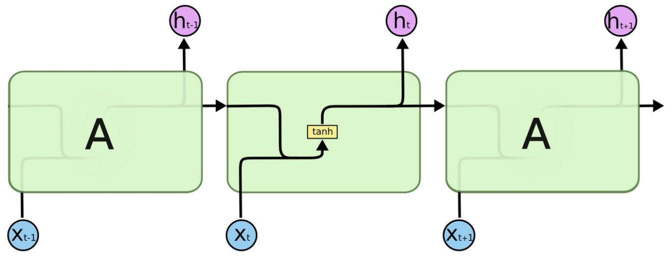
Caption generation

Sentiment detection

Machine translations

Source: http://karpathy.github.io/2015/05/21/rnn-effectiveness/

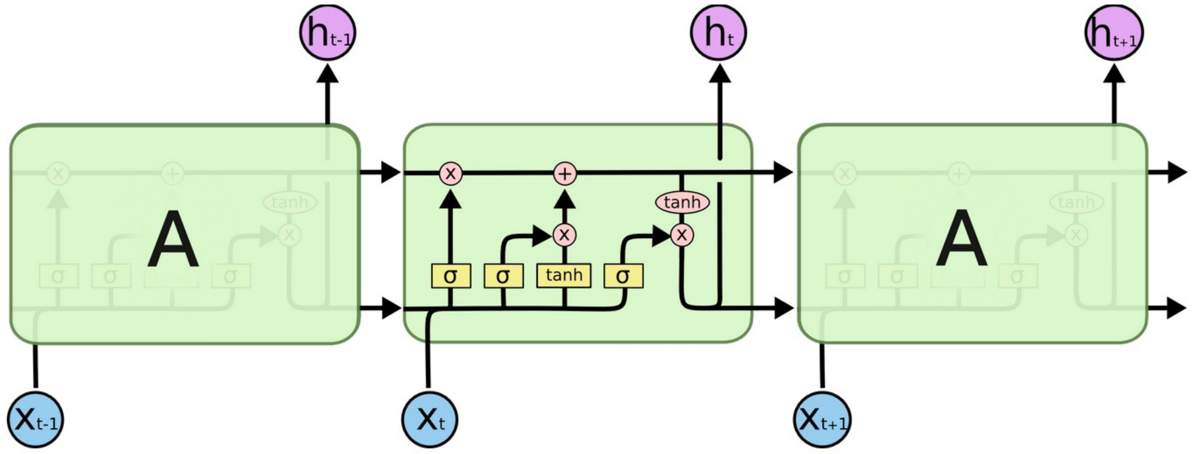
RNN and LSTM - Long Short Term Memory



RNN

LSTM

To learn long term relationships; has 4 NNs



Source and details: https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Auto-Encoder

- A model which can produce its inputs

- Usages
  - Input compression
  - Sequence generation
  - Produce recommendation
  - Denoising

Auto-Encoder

- Example 1: numeric array

- Example 2 - exercise: character array

- Code sample:
  - https://github.com/biplav-s/course-nl-f22/blob/main/sample-code/l11-nn-dl/AutoEncoder%20Sequence%20using%20LSTM.ipynb

Contextual Word Embeddings

	Name	Description	URL, References
1.	Elmo (embeddings from language models)	Contextual, deep, character-based	https://allennlp.org/elmo; Deep contextualized word representations, Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, Luke Zettlemoyer. NAACL 2018.
2	Word2Vec	Word-based, prediction focus	*Mikolov, Tomas; et al. (2013). "Efficient Estimation of Word Representations in Vector Space". arXiv:1301.3781 [cs.CL]. Mikolov, Tomas (2013). "Distributed representations of words and phrases and their compositionality". Advances in Neural Information Processing Systems. arXiv:1310.4546.*
3	Glove	Word-based, count	https://nlp.stanford.edu/projects/glove/, Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. [pdf] [bib]
4	Fasttext	Variation of word2vec, works with N-gram, words not in voacbulary	

Commentaries: Start with - https://jalammar.github.io/
https://jalammar.github.io/illustrated-bert/ , https://cai.tools.sap/blog/glove-and-fasttext-two-popular-word-vector-models-in-nlp/

Demonstration: Word2Vec Using Gensim

See sample code on GitHub:
https://github.com/biplav-s/course-nl/blob/master/l7-language/code/Word%20embedding%20with%20Gensim.ipynb

Lecture 10: Lecture Summary

- We reviewed Machine Learning methods
  - Data preparation is the key
  - Watch out for evaluation
  - ML is just a step, what happens to the model is also important
- Language models
  - We reviewed connections between parsing and language model
  - We discussed language models and are focusing on pre-requisites needed to understand them
  - We discussed contextual word representations as a stepping stone

Concluding Segment

Reading

- Shervin Minaee, Nal Kalchbrenner, Erik Cambria, Narjes Nikzad, Meysam Chenaghlu, and Jianfeng Gao. 2021. Deep Learning--based Text Classification: A Comprehensive Review. ACM Comput. Surv. 54, 3, Article 62 (April 2022), 40 pages. https://doi.org/10.1145/3439726

- Hang Li, Language Models: Past, Present, and Future, Communications of the ACM, July 2022, Vol. 65 No. 7, Pages 56-63 10.1145/3490443

- Refer to the LLM reading list: https://github.com/biplav-s/course-nl-f24/blob/main/reading-list/Readme-LLMs.md

About Next Lecture – Lecture 11

Lecture 11 Outline

- Invited talks - LLMs

Sep 24 (Tu)	Language Model – PyTorch, BERT, {Resume data, two tasks} **– Guest Lecture**
Sep 26 (Th)	Language Model – Finetuning, Mamba - **Guest Lecture**
Oct 1 (Tu)	Language model – comparing arch, finetuning - **Guest Lecture**
Oct 3 (Th)	Language model – comparison of results, discussion, ongoing trends– **Guest Lecture**

7	Sep 10 (Tu)	Statistical parsing, QUIZ
8	Sep 12 (Th)	Evaluation, Semantics
9	Sep 17 (Tu)	Semantics, Machine Learning for NLP, Evaluation - Metrics
10	Sep 19 (Th)	Towards Language Model: Vector embeddings, Embeddings, CNN/ RNN
11	Sep 24 (Tu)	Language Model – PyTorch, BERT, {Resume data, two tasks} **– Guest Lecture**
12	Sep 26 (Th)	Language Model – Finetuning, Mamba - **Guest Lecture**
13	Oct 1 (Tu)	Language model – comparing arch, finetuning - **Guest Lecture**
14	Oct 3 (Th)	Language model – comparison of results, discussion, ongoing trends– **Guest Lecture**
15	Oct 8 (Tu)	PROJ REVIEW