WORD2VEC AND WORD EMBEDDING VISUALIZATIONS

Brian Mayer

Research Scientist

Sanghani Center @ Virginia Tech

Demo Link: https://github.com/mayer22/WordEmbeddings

INTRODUCTION

WHO AM I?

VIRGINIA TECH

Bachelor & Masters in Industrial & Systems Engineering

LMI

AMTRAK

Logistics, Manufacturing, & Performance Analysis

VirginiaTech
TRANSPORTATION INSTITUTE

Transportation Research

SANGHANI CENTER FOR ARTIFICIAL INTELLIGENCE & DATA ANALYTICS
VIRGINIA TECH

COMPUTER SCIENCE

Research Management, Transportation, Intelligence, and Forecasting Research

WHO AM I?

TODAY'S ASSUMPTIONS

- We code in Python
- We understand basic Natural Language Processing (NLP) techniques like:
  - Bag of Words (TF/TFIDF)
  - Basic sentiment analysis
- We know how neural networks work
- We know about dimension reduction

REVIEW – TEXT ANALYTICS

- **Bag of Words (BoW)** is an NLP technique that:
  - Uses words as variables
  - Ignores order, context, and meaning
  - Examples
    - Term Frequency (TF)
      - Create a Term-by-Document Matrix
      - Can highlight unimportant words because they appear a lot
    - TF-Inverse Document Frequency (TFIDF)
      - Evaluates the importance of a word to a document
      - Boost if a word is common to a document but penalize if it is also common to the corpus
  - Data representations become very large in large corpuses

What is an *n*-gram?

A group of *n* words

1-gram: "cat", "dog", "truck", "bike"

2-gram (bi-gram): "fire truck", "yellow dog", "my car"

3-gram (tri-gram): "ride my bike", "walk the dog"

4-gram: "go for a spin", "take a long walk"

… and so on (oops, that was a trigram)

REVIEW – TEXT ANALYTICS

- **<u>Semantic Analysis</u>** is an NLP technique that:
  - Represents meaning and considers words in context
  - Requires an emotional framework (labor intensive and requires many assumptions)
  - Requires many highly impactful parameters (e.g., valence shifters and window)
    - i.e., user specification

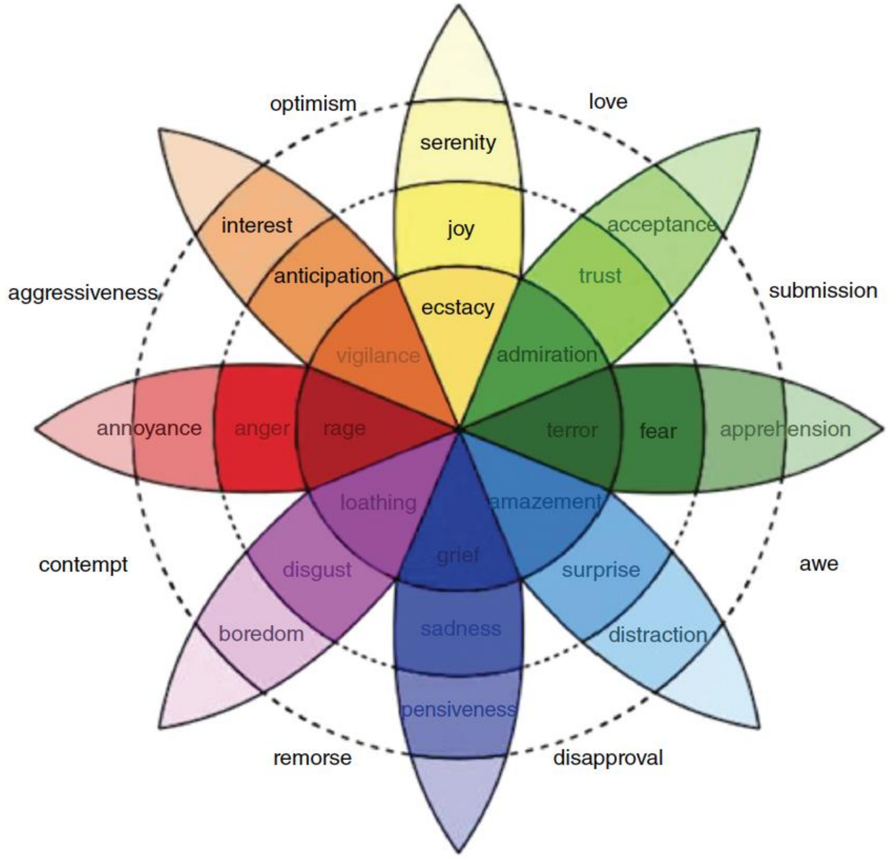


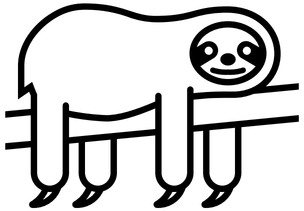
Figure 4.1 Plutchik's wheel of emotion with eight primary emotional states.

HOW DO WE DO BOTH?

Leverage the simplicity of counting words/appearance

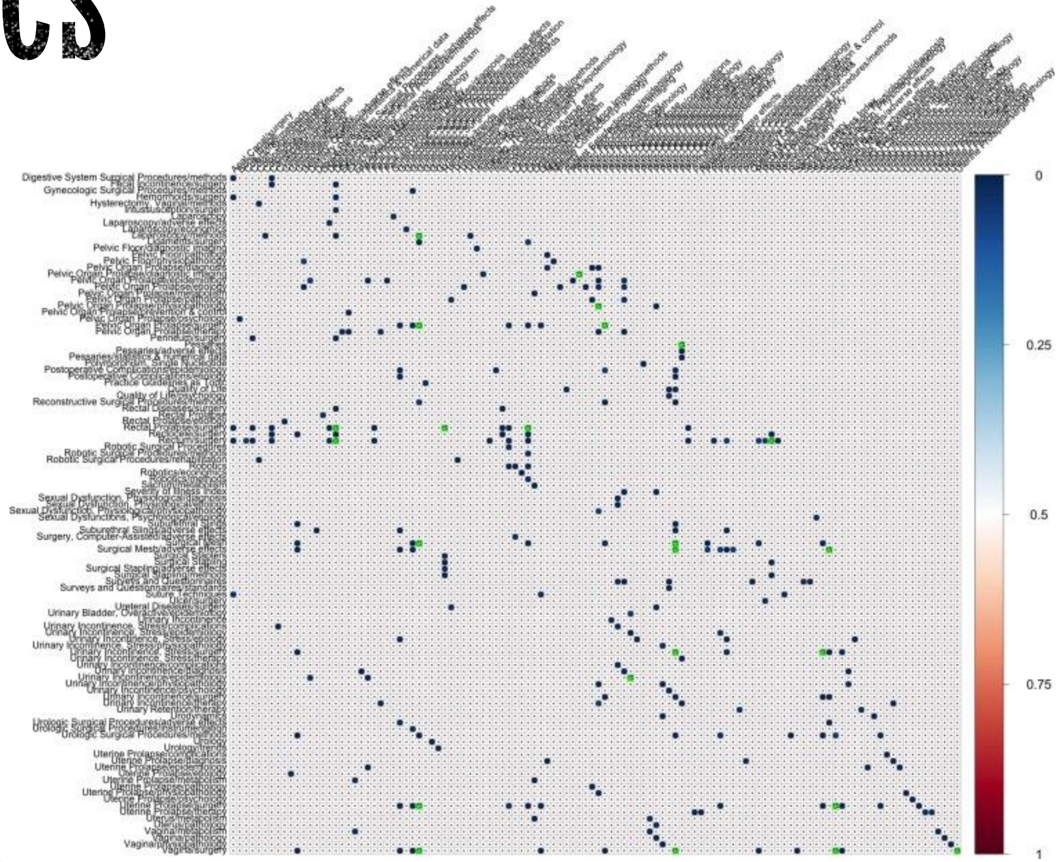
&

Consider the context of words



REVIEW – TEXT ANALYTICS

- A **<u>Co-occurrence Matrix</u>** is an NLP technique similar to Term-Frequency analysis that:
  - Counts how often different words appear together (within some window) in a document/corpus
    - i.e., adds context
  - Each word is represented as a vector of its co-occurrence frequencies with other words
  - Even larger than TF/TFIDF (Oh No!) 😮



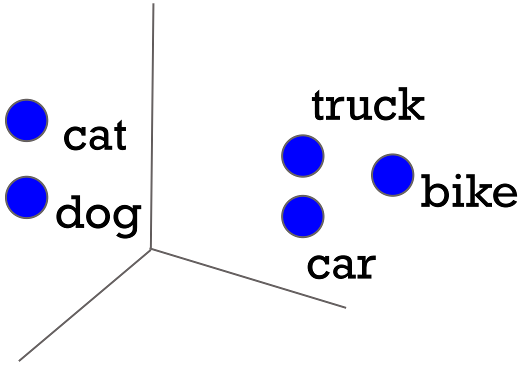
Zhou, Xiaobei, et al. "A probabilistic model for co-occurrence analysis in bibliometrics." Journal of Biomedical Informatics 128 (2022): 104047.

HOW CAN WE BE SMARTER?

- USE EMBEDDINGS => turn words into **manageable vectors**
  - ? Why would we want to represent words as vectors?
    1. Depict the meaning of a word (or document as a combination of words) quantitatively
    2. Compare words/documents based on location and relative distance

    → Computational Linguistics and Social Science
- There are several different word embedding methods

? **Are these outdated now?**



word2vec

Co-Occurrence Matrix

GloVe

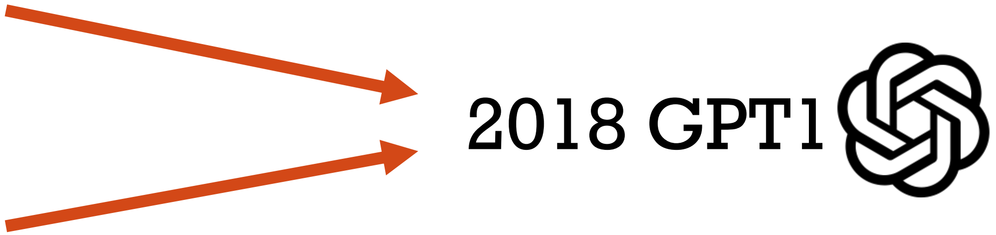
TFIDF

LARGE LANGUAGE MODELS (LLMS)

LLMs don't really compare words/docs BUT

ChatGPT evolved from word2vec

- We'll learn that embeddings are based on predictions (just like LLMs)
- 2018 ELMO[1] and BERT[2]: dynamic word embeddings
  - Embeddings change based on context
- 2018 Transformer Architecure[3]



2018 GPT1

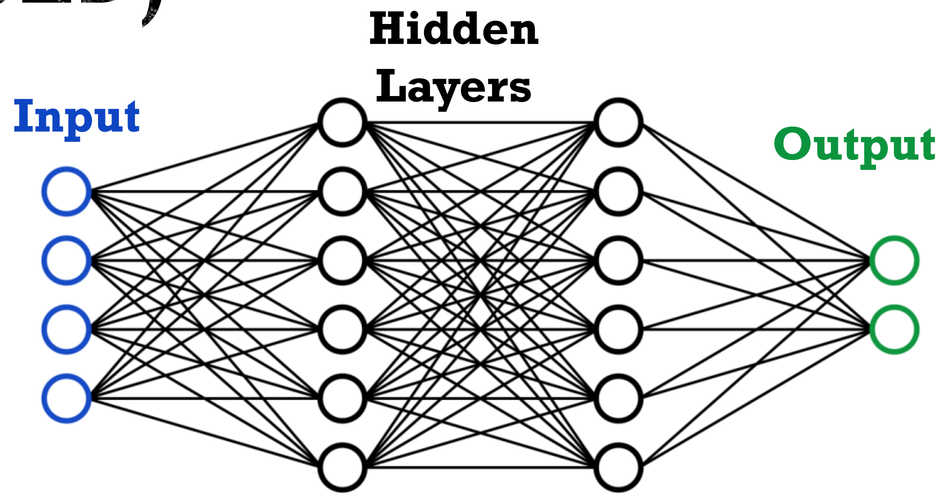
[1] Neumann, M. P. M., et al. "Deep contextualized word representations." *arXiv preprint arXiv:1802.05365* (2018).
[2] Devlin, Jacob. "Bert: Pre-training of deep bidirectional transformers for language understanding." *arXiv preprint arXiv:1810.04805* (2018).
[3] Vaswani, A. "Attention is all you need." *Advances in Neural Information Processing Systems* (2017).

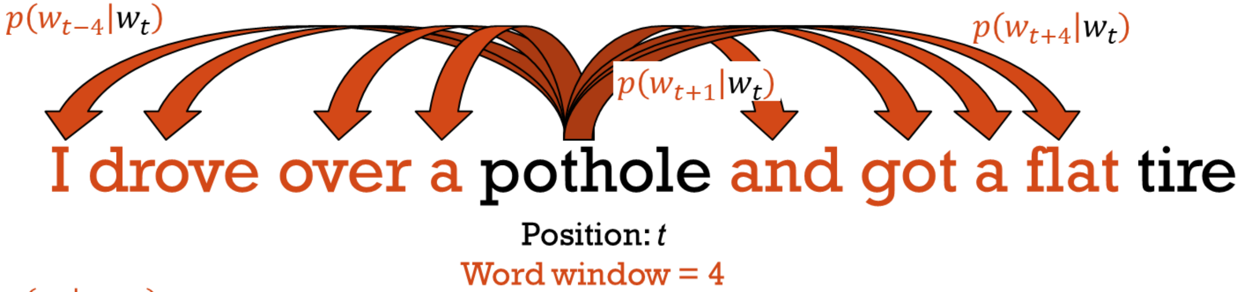
So if we want to understand LLMs let's go back to the start of all this

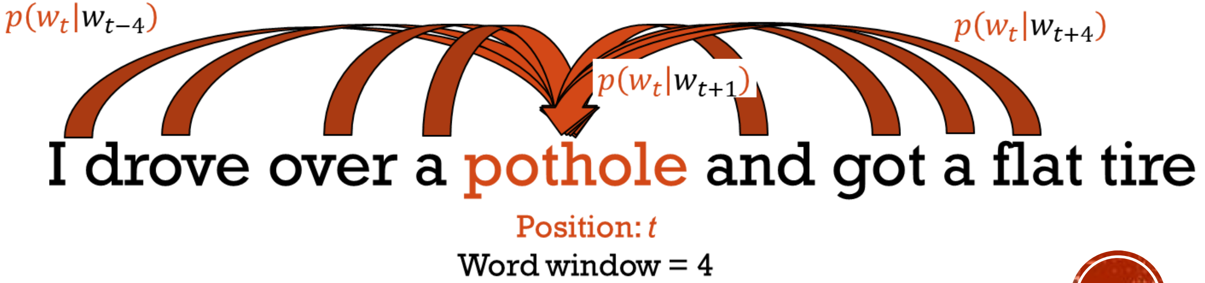
WORD2VEC

WORD2VEC (PREDICTION-BASED) EMBEDDINGS

- More advanced NLP method using neural networks
  - We actually use the weights not the output of the network
- Therefore, reliable results require a **significant amount of training data**
- Developed by researchers at Google
- Skip-Gram[4]
  - Trained to predict words before and after the current word
- Continuous Bag of Words (CBOW)[5]
  - Trained to predict a word based on context (nearby, i.e., related, words)







[4] Mikolov, Tomas, et al. "Distributed representations of words and phrases and their compositionality." *Advances in neural information processing systems* 26 (2013).
[5] Mikolov, Tomas. "Efficient estimation of word representations in vector space." *arXiv preprint arXiv:1301.3781* (2013).

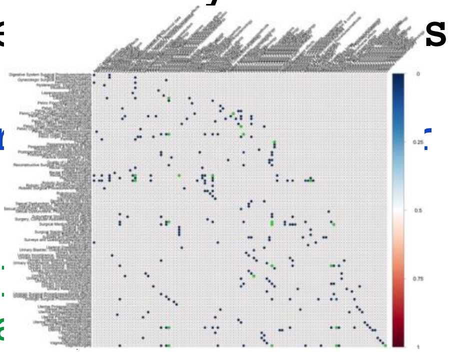
LET'S START WITH SKIP-GRAM

1. Find every word in your vocabulary/corpus
2. Count how often words occur near each other

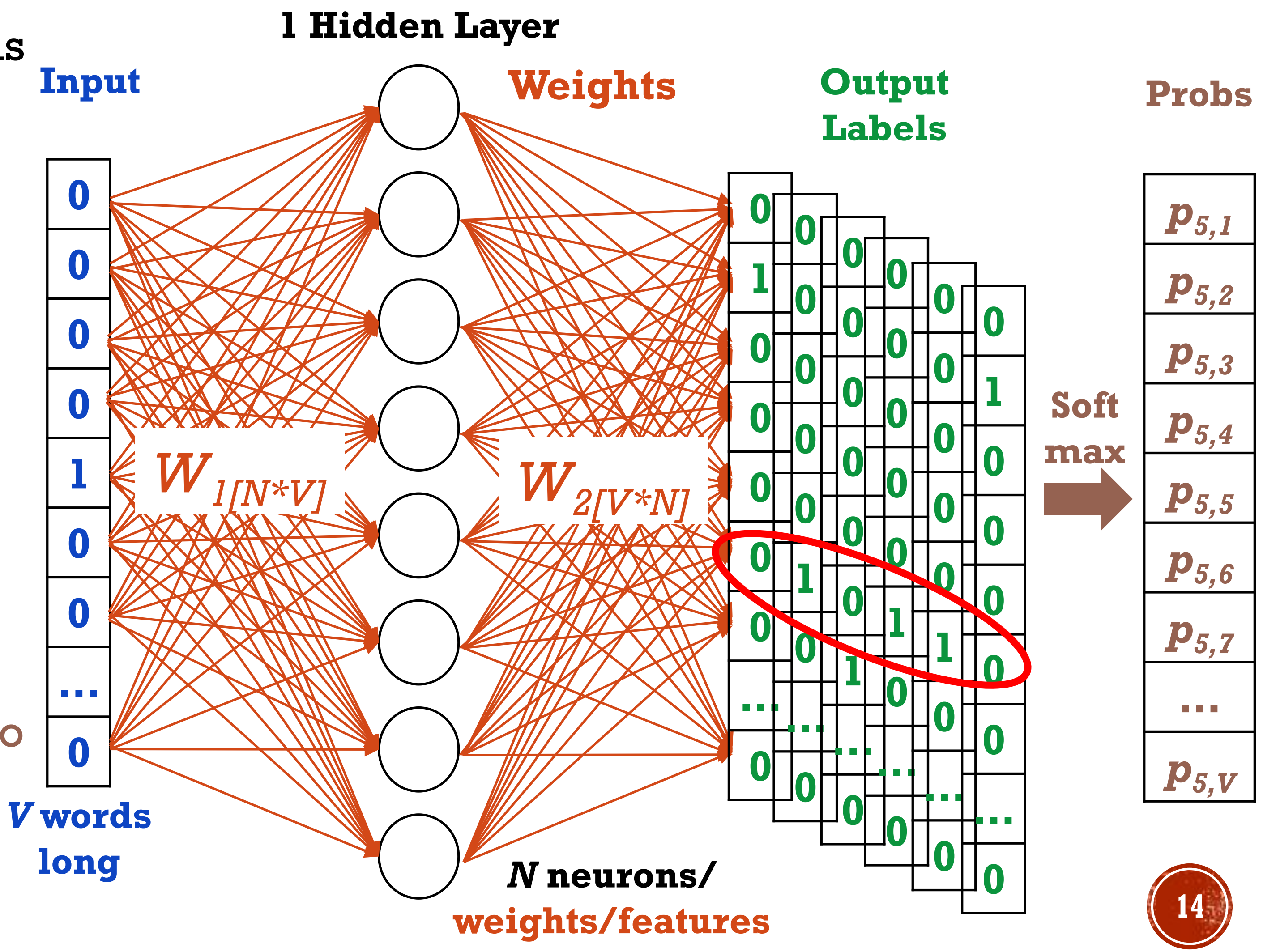
Sound familiar?

Co-occurrence matrix

3. Create an identity matrix with your list of words as labels for the s (one-hot encoding)
   a) Each vector in the matr neural network
   b) The output for training vectors matched with t every word we find nea



4. Create a NN with 1 hidden layer and *N* neurons (will be our # of parameters)
5. Then train a softmax regression classifier to determine the output probabilities and hidden layer weights
   - Sum of all *V* probabilities per neuron will = 1 (per softmax rules)



MODEL EVALUATION

Example:

5 word corpus

with 3 features (weights)

Hidden Layers

Input

Identity Matrix

3 x 5 Weight Matrix

Word

$$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

5 words long

Your hidden layer can have any number of features.

- 100 features is common
  - not too big (i.e., manageable)
  - not too small (i.e., still descriptive)
- Some people use smaller sizes to
  - save space
  - reduce computation
  - visualize in 2D

SUMMARY OF SKIP-GRAM NETWORK

- <u>Training objective:</u>

  Maximize the probability of predicting nearby words correctly

- Takes a large input vector (V long), compresses it down to a small dense vector (N long), then outputs probability distribution for the target word
- The rows of the hidden layer weight matrix W_1 (columns of W_2) are the word vector.
- Last layer is always softmax (w/categorical cross-entropy)

SOFTMAX REGRESSION

$$p_i = \frac{e^{z_i}}{\sum_{j=1}^{n} e^{z_j}}$$

Vector $z = [z_1, z_2, \ldots, z_V]$ *(rows of* W_1*, columns of* W_2*)*

z_i represents the evaluated score associated with class *i*

p_i = probability of each class *i*

- Prior to the application of softmax some vector components could
  - be negative
  - greater than one
  - might not sum to 1
- But after applying the softmax function
  - each component will be in the interval (0,1)
  - the components will add up to 1
  - they can be interpreted as probabilities

UNDERSTANDING

If words frequently appear together (i.e., have similar contexts)

Our r words

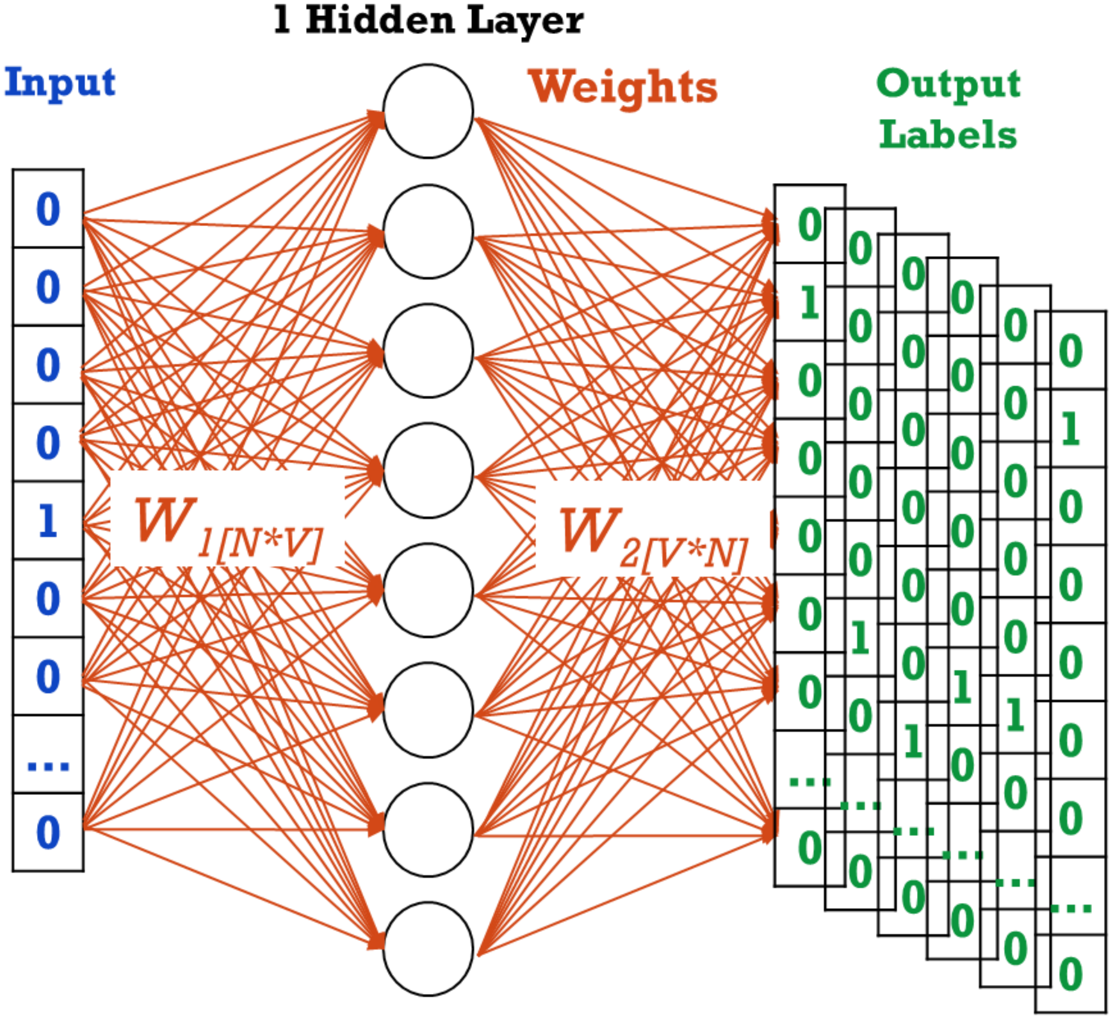
Hold that thought

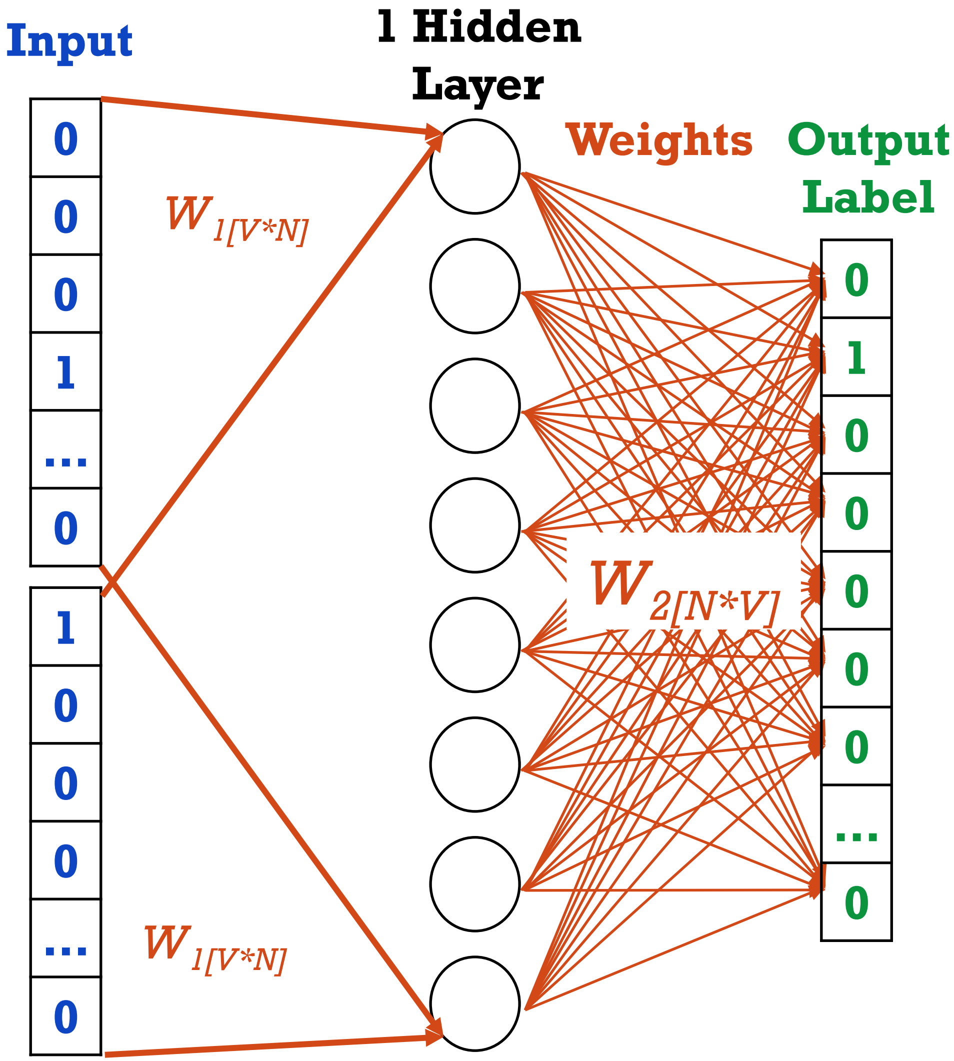
We’ll come back to this

(i.e., re ere N is

the number of weights/features)

SKIP-GRAM VS CBOW





SKIP-GRAM VS CBOW

- Skip-Gram
  - Better represents rare words and larger datasets
    - Considers each word pair
  - **Slower**
  - Better at representing *semantic meaning*

- Continuous Bag of Words (CBOW)
  - Common words are more accurate because they are smoothed over the entire dataset
  - Rare words get "lost"/"forgotten"
  - **Faster**: only needs to predict a single target word given a set of context words
  - Better at representing *syntactic relationships*

WORD2VEC DEMO

https://github.com/mayer22/WordEmbeddings

WORD EMBEDDING VISUALIZATIONS

REMEMBER

If words frequently appear together (i.e., have similar contexts)

THEN

Our model needs to output similar results for the 2 words

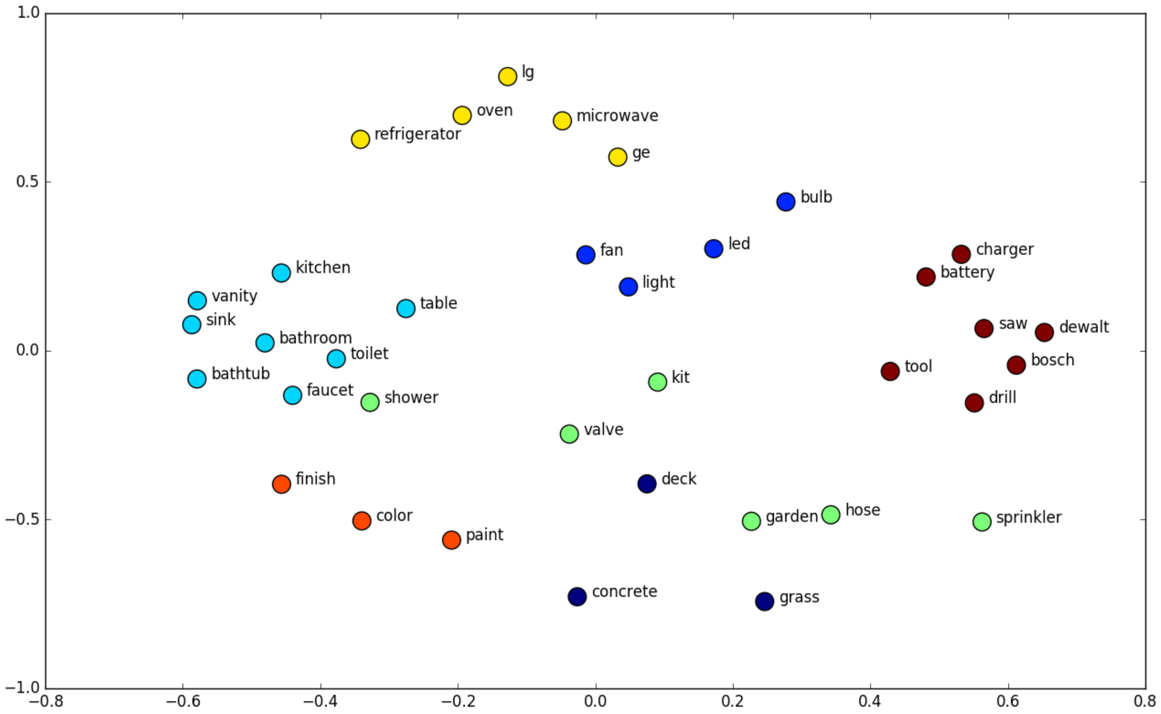
THEREFORE

Their word vectors will be similar

(i.e., relationally close in an *n* dimensional space where *n* is the number of weights/features)

COMPARE WORD VECTORS

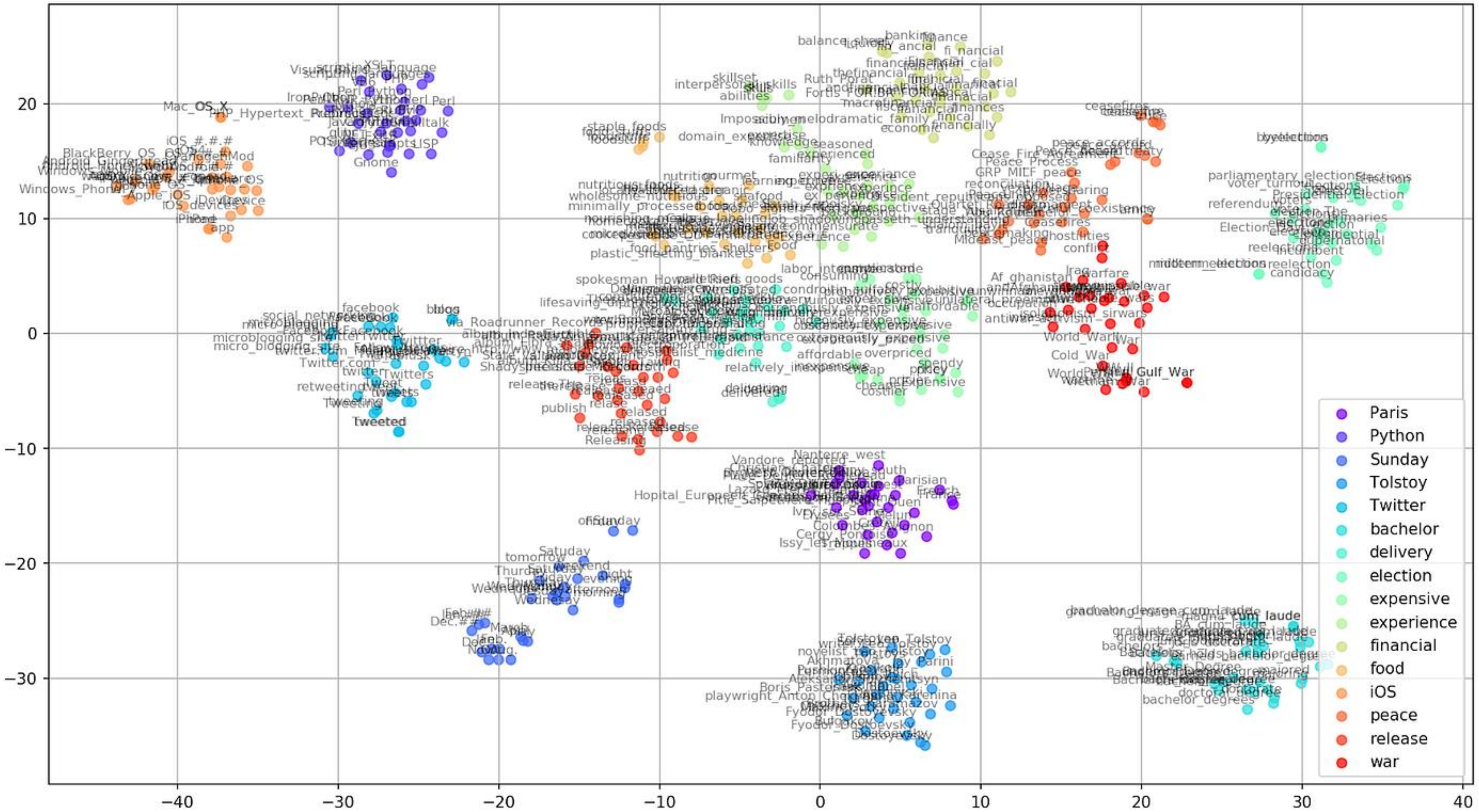
- We can measure high-dimensional distances to:
  - find similar words
  - determine which word is more similar
  - find the opposite of a word (Italy?)
- We can visualize words on a scatter-plot
  - If you used more than 2 or 3 weights/parameters you will need to reduce them using these common methods:
    - Multidimensional Scaling (MDS)
    - Principal Component Analysis (PCA)
    - t-distributed stochastic neighbor embedding (t-SNE)
    - Uniform Manifold Approximation and Projection (UMAP)



https://neptune.ai/blog/word-embeddings-guide

COMPARE WORD VECTORS

Training Data: Articles from Google News and classical literary works by Leo Tolstoy



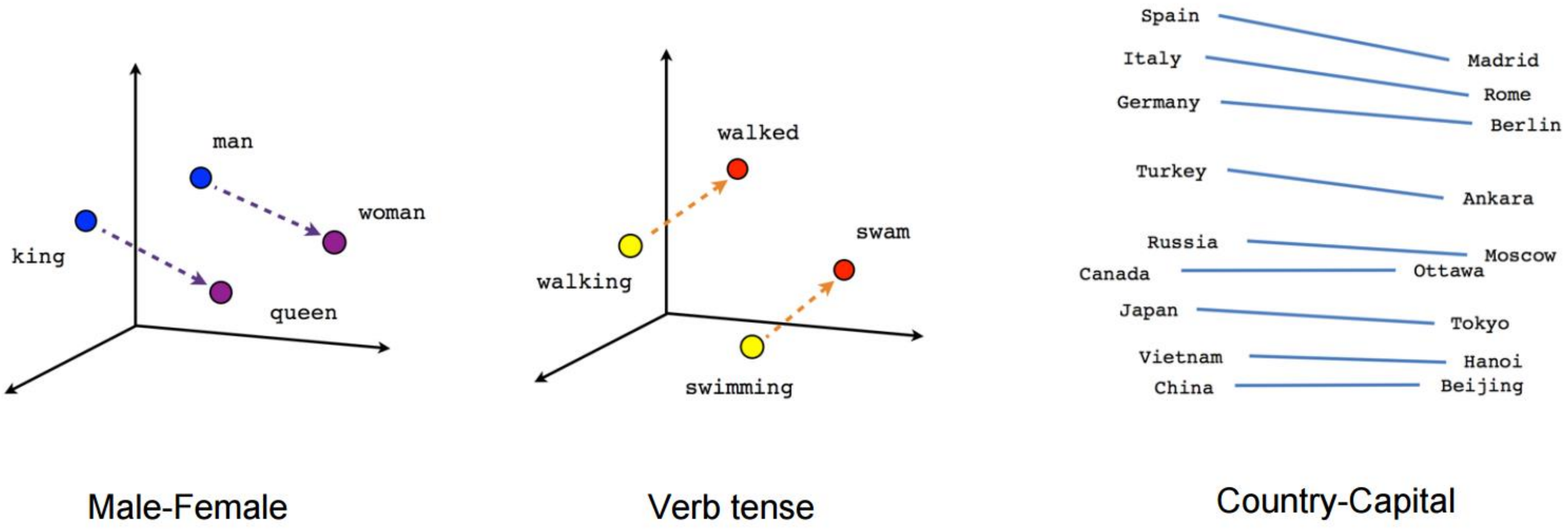
https://towardsdatascience.com/google-news-and-leo-tolstoy-visualizing-word2vec-word-embeddings-with-t-sne-11558d8bd4d

LET'S GO BACK TO OUR DEMOS

WORD EMBEDDING ANALYSIS

COMPARING WORD VECTORS

- Word analogies can often be solved with vector arithmetic[6]



Male-Female

Verb tense

Country-Capital

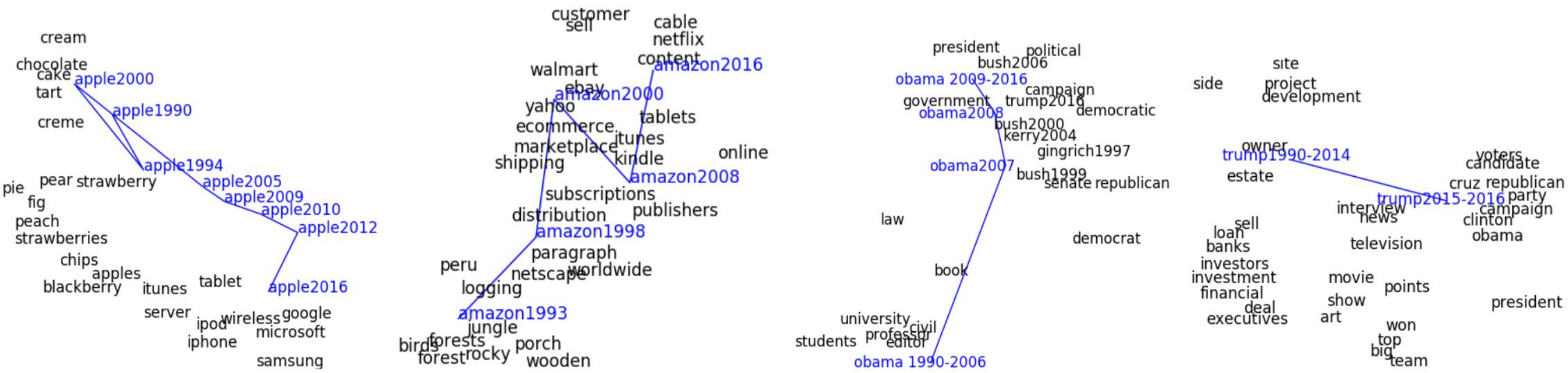
https://www.analyticsvidhya.com/blog/2017/06/word-embeddings-count-word2veec/

[6] Allen, Carl, and Timothy Hospedales. "Analogies explained: Towards understanding word embeddings." *International Conference on Machine Learning*. PMLR, 2019.

TEMPORAL WORD EMBEDDINGS

Word contexts change over time. **How do we view changes?**

- Create temporal embeddings:[7,8]



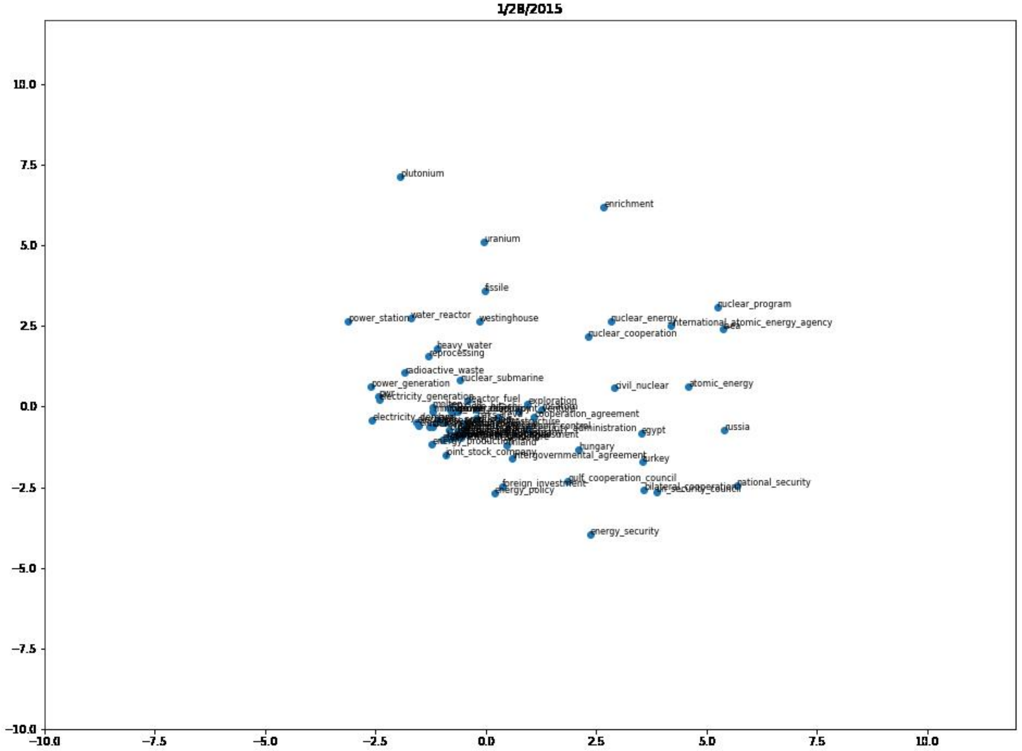
Global embedding space with chosen word vectors by time slice

[7] Hamilton, William L., Jure Leskovec, and Dan Jurafsky. "Diachronic word embeddings reveal statistical laws of semantic change." *arXiv preprint arXiv:1605.09096* (2016).

[8] Yao, Zijun, et al. "Dynamic word embeddings for evolving semantic discovery." *Proceedings of the eleventh acm international conference on web search and data mining*. 2018.

TEMPORAL WORD EMBEDDINGS

- We can also train a model that updates the embeddings for all of our words every time slice
  - **But embeddings are not deterministic**
  - Embedding spaces can shift
  - Same word/same meaning => new location
  - **Alignment needed!**
- Sol: Force word embeddings in different time periods to be similar (NN training)[9,10]
  - **Assumes** that the majority of the words do not change their meaning over time



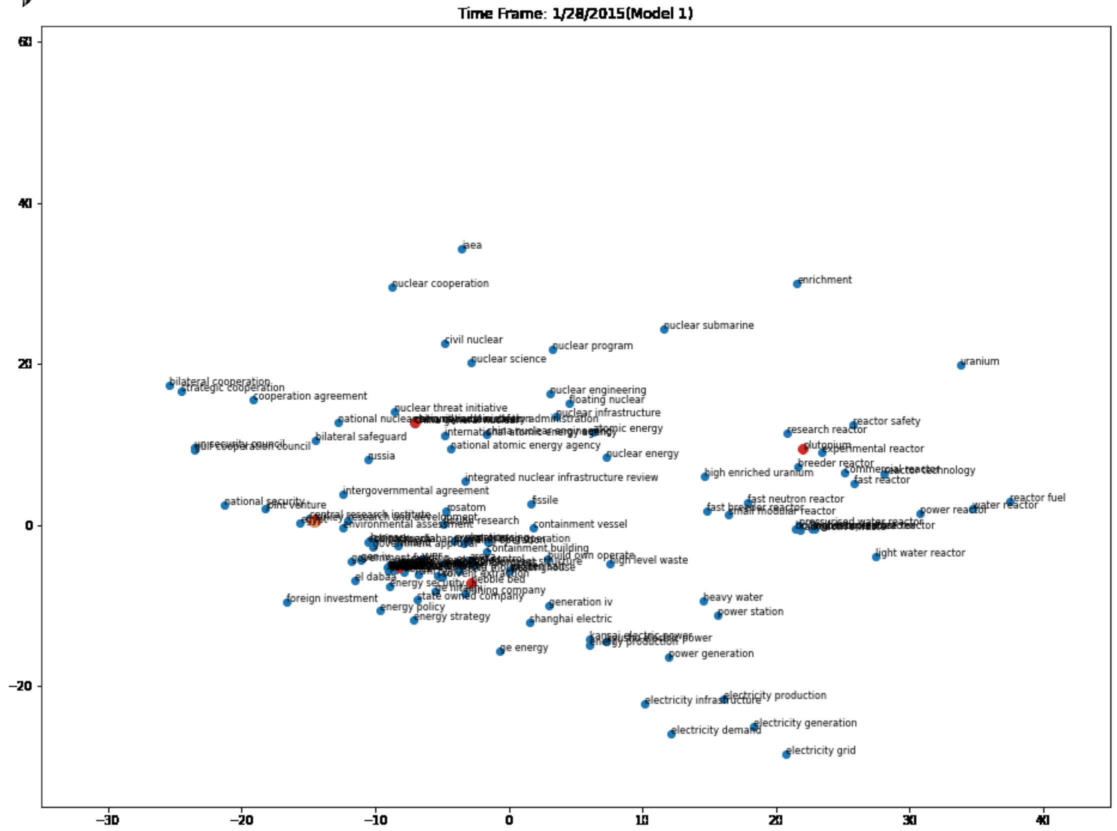
[9] Hamilton, William L., Jure Leskovec, and Dan Jurafsky. "Diachronic word embeddings reveal statistical laws of semantic change." *arXiv preprint arXiv:1605.09096* (2016).

[10] Yao, Zijun, et al. "Dynamic word embeddings for evolving semantic discovery." *Proceedings of the eleventh acm international conference on web search and data mining*. 2018.

TEMPORAL WORD EMBEDDINGS WITH A COMPASS (TWEC)[11]

Also, Compass-Aligned Distributional Embeddings (CADE)[12]

- Train compass using entire corpus
- Use embedding from Timestep 1 as a "compass" for other time slices
- **Assumes** that words that move, appear in the context of other words that move (more accurately accounts for shifted words)



[11] Di Carlo, Valerio, Federico Bianchi, and Matteo Palmonari. "Training temporal word embeddings with a compass." Proceedings of the AAAI conference on artificial intelligence. Vol. 33. No. 01. 2019.

[12] Bianchi, Federico, et al. "Compass-aligned distributional embeddings for studying semantic differences across corpora." arXiv preprint arXiv:2004.06519 (2020).

NEXT CLASS

Document Embeddings (Doc2Vec)
to Compare Documents

NOW LET'S EXPLORE WORD EMBEDDINGS

Tool: https://www.cs.cmu.edu/~dst/WordEmbeddingDemo/index.html

Tutorial: https://www.cs.cmu.edu/~dst/WordEmbeddingDemo/tutorial.html

Experiment: https://www.cs.cmu.edu/~dst/WordEmbeddingDemo/experiments.html

EXPLORATION ACTIVITY
(OR GO BACK TO CODE)

- Tool: https://www.cs.cmu.edu/~dst/WordEmbeddingDemo/index.html
- **Rotate**: Click and drag in the point plot to rotate the view
- **Zoom**: Use the scroll wheel to zoom in or out.
- **Pan**: Hold down the control key and click and drag to pan the view.
- **Word details**: Hovering over a word shows the closest 10 words
- **Activate**: Click on a word in the plot
  - Shows dot product/cosine similarity of the selected words to the words in the vector slots
  - Add it to the vector display on the right by clicking one of the slots
- Tutorial: https://www.cs.cmu.edu/~dst/WordEmbeddingDemo/tutorial.html
- Experiment: https://www.cs.cmu.edu/~dst/WordEmbeddingDemo/experiments.html

DISCUSS WITH YOUR NEIGHBORS

- Where would you put the words "adult“, "child“, "infant“, or "grandfather"?
- See if you can perform some analogy arithmetic
  - Calculate the distances/directions between words for 2 sets of terms that make up an analogy and see if it is the same
- See if you can find terms that aren't similar but are similar in one parameter. Are there other terms similar/different on that parameter? What might that parameter be?
- Find another meaningful semantic dimension.
- https://www.cs.cmu.edu/~dst/WordEmbeddingDemo/index.html

QUESTIONS?