

ACCT 420: ML and AI for numeric and text data

Session 10

Dr. Richard M. Crowley

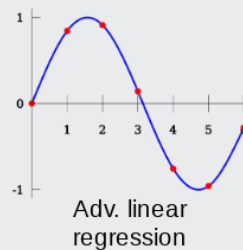
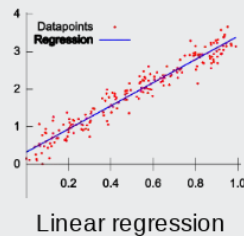
Front matter

Learning objectives

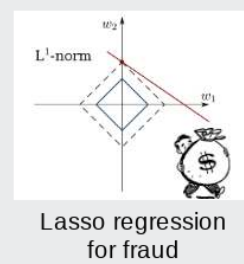
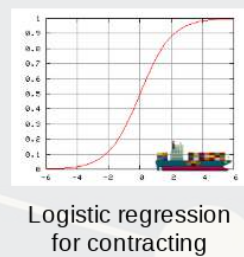
Foundations



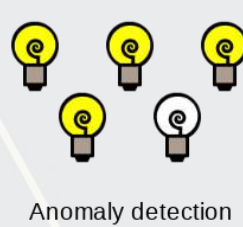
Forecasting



Binary classification



Advanced methods



■ Theory:

- Neural Networks (broad overview)
- Vector space methods

■ Application:

- Neural networks for understanding textual data
- Top managements' tweets

■ Methodology:

- Vector methods
- 6 types of neural networks
- Others

Ethics: Data security

A motivating example

[Withheld from all public copies]

Anonymized data

- Generally we anonymize data because, while the data itself is broadly useful, providing full information could harm others or oneself
- Examples:
 - Studying criminal behavior use can create a list of people with potentially uncaught criminal offenses
 - If one retains a list of identities, then there is an ethical dilemma:
 - Protect study participants by withholding the list
 - Provide the list to the government
 - This harms future knowledge generation by sowing distrust
 - Solution: Anonymous by design
 - Website or app user behavior data
 - E.g.: FiveThirtyEight's [Uber rides dataset](#)

What could go wrong if the Uber data wasn't anonymized?

Anonymization is tricky

Both Allman & Paxson, and Partridge warn against relying on the anonymisation of data since *deanonymisation techniques are often surprisingly powerful*. Robust anonymisation of data is difficult, particularly when it has high dimensionality, as the anonymisation is likely to lead to an unacceptable level of data loss [3]. –

TPHCB 2017

- There are natural limits to anonymization, particularly when there is a limited amount of potential participants in the data
 - Example: Web browser tracking at [Panoptlick](#)

Responsibilities generating data

- Keep users as unidentifiable as feasible
- If you need to record people's private information, **make sure they know**
 - This is called *informed consent*
- If you are recording sensitive information, consider not keeping identities at all
 - Create a new, unique identifier (if needed)
 - Maintain as little identifying information as necessary
 - Consider using encryption if sensitive data is retained
 - Can unintentionally lead to infringements of *human rights* if the data is used in unintended ways

Informed consent

- When working with data about *people*, they should be informed of this and consent to the research, unless the data is publicly available
- From SMU's IRB Handbook: (2017 SEP 18 version)
 - "*Informed consent*: Respect for persons requires that participants, to the degree that they are capable, be given the opportunity to make their own judgments and choices. When researchers seek participants' participation in research studies, they provide them the opportunity to make their own decisions to participate or not by ensuring that the following adequate standards for informed consent are satisfied:
 - *Information*: Participants are given sufficient information about the research study, e.g., research purpose, study procedures, risks, benefits, confidentiality of participants' data.
 - *Comprehension*: The manner and context in which information is conveyed allows sufficient comprehension. The information is organized for easy reading and the language is easily comprehended by the participants.
 - *Voluntariness*: The manner in which researchers seek informed consent from the participants to participate in the research study must be free from any undue influence or coercion. Under such circumstances, participants are aware that they are not obliged to participate in the research study and their participation is on a voluntary basis."

Also, note the existence of the [PDPA law](#) in Singapore

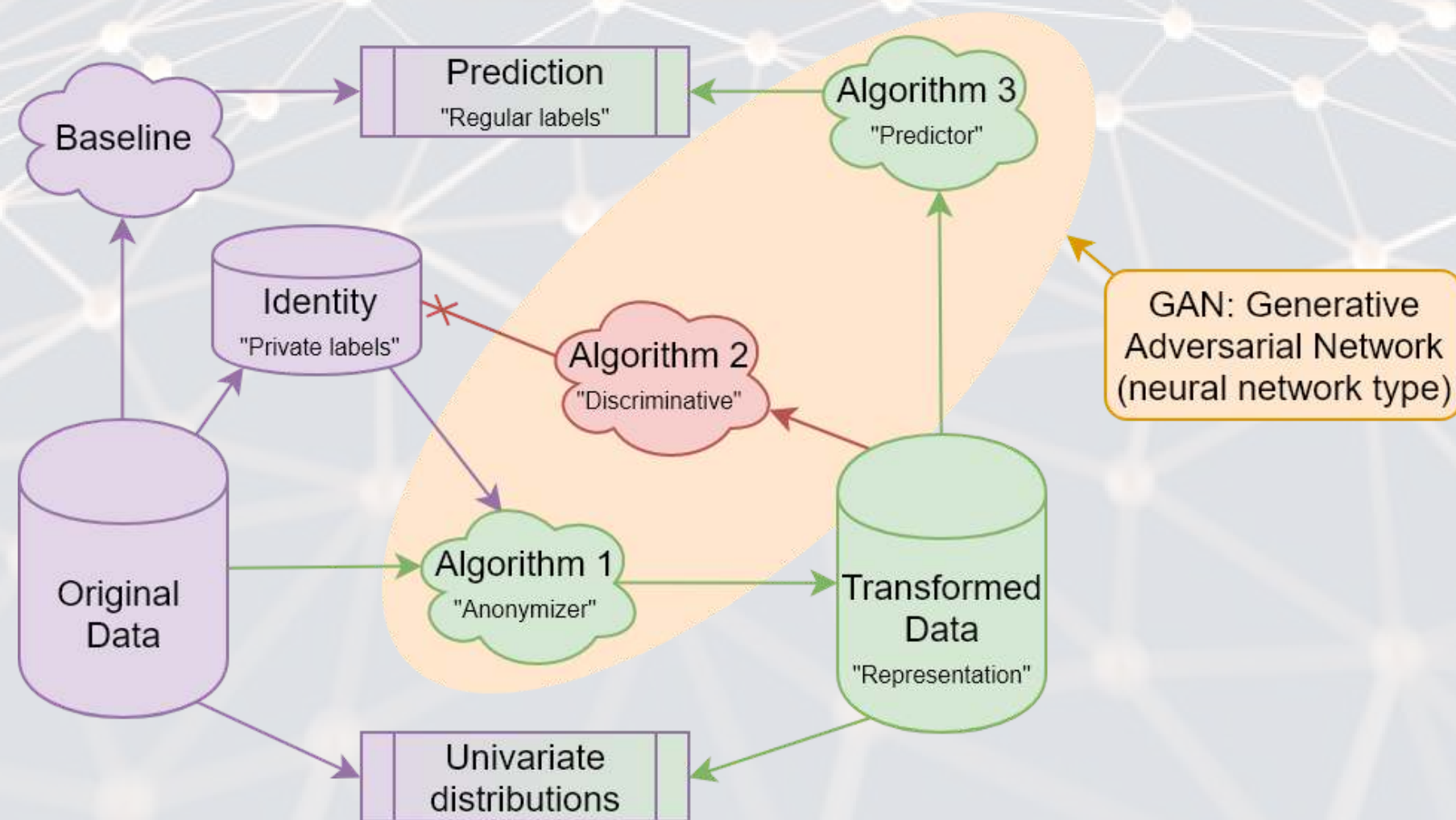
Human rights

- Recall the drug users example
 - If data was collected without their consent, and if it was not anonymized perfectly, then this could lead to leaking of drug user's information to others

What risks does this pose? Consider contexts outside Singapore as well.

A novel approach using a GAN

- Source: [Learning Anonymized Representations with Adversarial Neural Networks](#)
- On handwriting classification, cases that can be deanonymized drop from 40% to 3.3%
 - Accuracy drops from ~98% down to 95%, a much smaller drop



How the GAN works

- There are 2 general components to a GAN:
 1. A *generative* network: It's goal is to generate *something* from some data
 - In this case, it generates a dataset with good predictability for the “predictor” network **AND** bad predictability for the “adversarial” network
 2. An *adversarial*: It's goal is to thwart the generative network
 - In this case, it tries to determine the writers' identities

By iterating repeatedly, the generative network can find a strategy that can generally circumvent the discriminative network

- Related work in biology: [Privacy-preserving generative deep neural networks support clinical data sharing](#) (also GAN based)

Responsibilities using data

“The collection, or use, of a dataset of illicit origin to support research can be advantageous. For example, legitimate access to data may not be possible, or the reuse of data of illicit origin is likely to require fewer resources than collecting data again from scratch. In addition, the sharing and reuse of existing datasets aids reproducibility, an important scientific goal. The disadvantage is that ethical and legal questions may arise as a result of the use of such data” ([source](#))

Responsibilities using data

- Respect for persons
 - Individuals should be treated as *autonomous agents*
 - People are people
 - Those without autonomy should be protected
- Beneficence
 1. Do not harm (ideally)
 2. Maximize possible benefits and minimize possible harms
 - This can be a natural source of conflict
- Justice
 - Benefits and risks should flow to the same groups – don't use unwilling or disadvantaged groups who won't receive any benefit
 - [Extreme] example: [Tuskegee Syphilis study](#)

For experiments, see [The Belmont Report](#); for electronic data, see [The Menlo Report](#)

Languages for ML/AI

R for ML/AI

Older methods

- `caret`
- `randomForest`
- `nnet`
- `e1071`

Best-in-class

- `glmnet`: LASSO and elastic nets
- `xgboost`: XGBoost
- `Prophet`: ML for time series forecasting
- `keras`: Plugs into python's Keras
- `H2O4GPU`: Plugs into python's H2O
- `spacyr`: Plugs into python's SpaCy

Python for ML/AI

Older methods

- Sci-kit learn – one stop shop for most older libraries
- RPy2
- scipy + numpy + pandas + statsmodels
 - Add **Theano** in for GPU compute

Best-in-class

- **TENSORFLOW** (Google)
 - Can do everything
- **pytorch** – python specific Torch port
- **gensim**: “Topic modelling for humans”
- **H2O** (H2O)
- **caffe** (Berkley)
- **caffe2** (Facebook)
- **SpaCy** – Fast NLP processing
- **CoreNLP** – through various wrappers to the Java library

Others for ML/AI

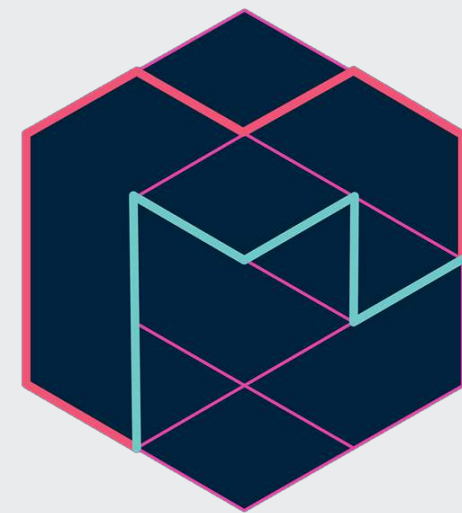
- C/C++: Also a first class language for TensorFlow!
 - Really fast – precompiled
 - Much more difficult to code in
- Swift: Strong TensorFlow support
- Javascript: Improving support from TensorFlow and others

Why do I keep mentioning TensorFlow?

- It can run almost ANY ML/AI/NN algorithm
- It has APIs for easier access like Keras
- Comparatively easy GPU setup
- It can deploy anywhere
 - Python & C/C++ built in
 - Swift and R Bindings for Haskell, R, Rust, Swift
 - TensorFlow light for mobile deployment
 - TensorFlow.js for web deployment

 TensorFlow Lite

 TensorFlow.js



magenta

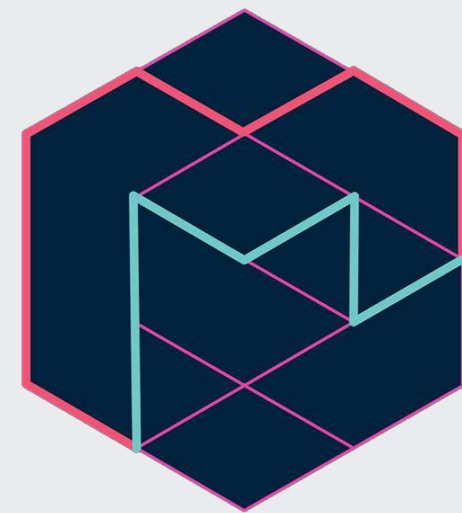
 TensorFlow Hub

Why do I keep mentioning TensorFlow?

- It has strong support from Google and others
 - [TensorFlow Hub](#) – Premade algorithms for text, image, and video
 - [tensorflow/models](#) – Premade code examples
 - The [research](#) folder contains an amazing set of resources
 - [tensorflow/tensor2tensor](#) – AI research models

 TensorFlow Lite

 TensorFlow.js



magenta

 TensorFlow Hub

Other notable frameworks

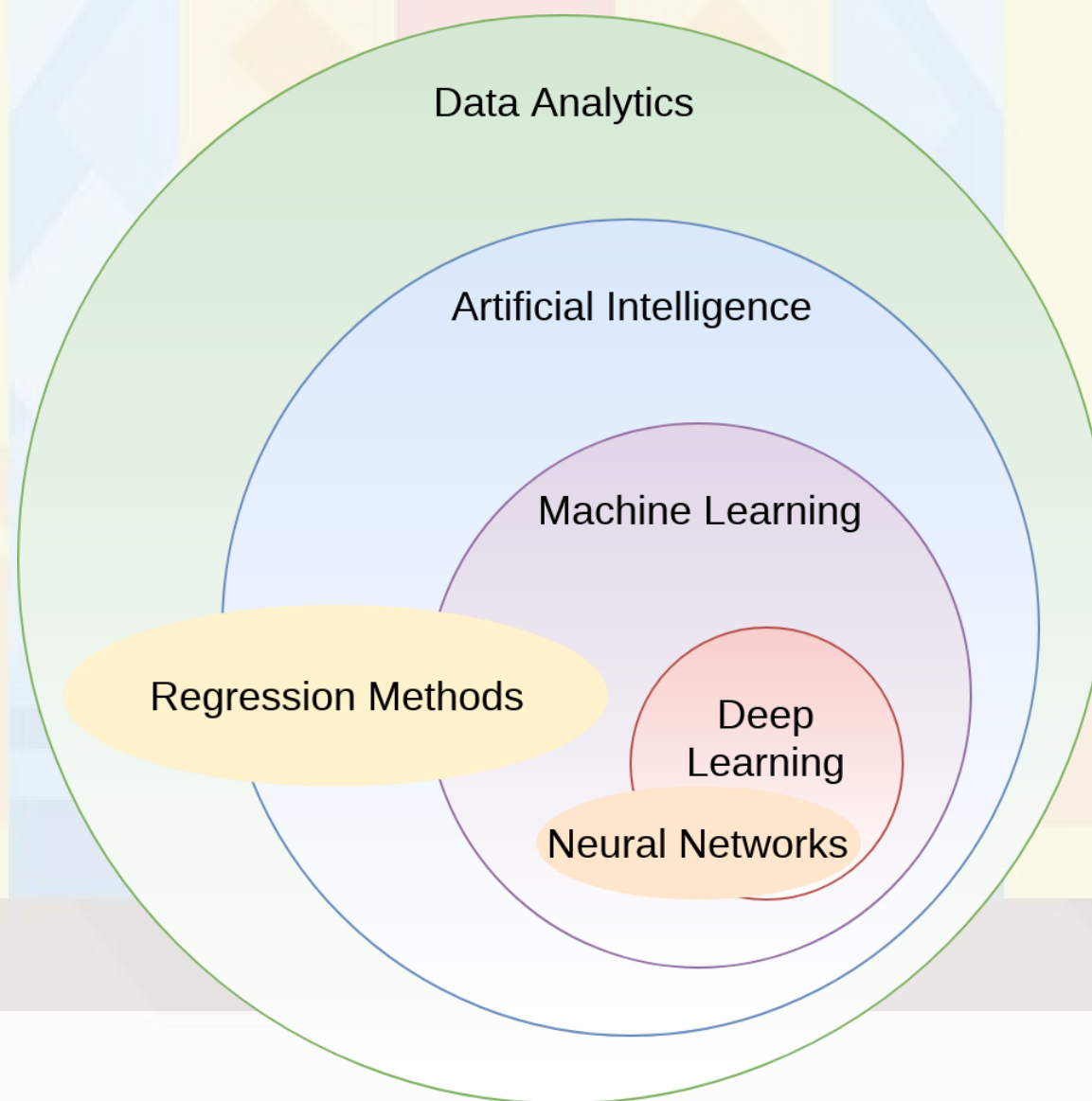
- **Caffe**
 - Python, C/C++, Matlab
 - Good for image processing
- **Caffe2**
 - C++ and Python
 - Still largely image oriented
- **Microsoft Cognitive Toolkit**
 - Python, C++
 - Scales well, good for NLP
- **Torch** and **Pytorch**
 - For Lua and python
 - [fast.ai](#), [ELF](#), and [AllenNLP](#)
- **H2O**
 - Python based
 - Integration with R, Scala...



Neural Networks

What are neural networks?

- The phrase *neural network* is thrown around almost like a buzz word
- *Neural networks* are actually a specific type class algorithms
 - There are many implementations with different primary uses



What are neural networks?

- Originally, the goal was to construct an algorithm that behaves like a human brain
 - Thus the name
- Current methods don't quite reflect human brains, however:
 1. We don't fully understand how our brains work, which makes replication rather difficult
 2. Most neural networks are constructed for specialized tasks (not general tasks)
 3. Some (but not all) neural networks use tools our brain may not have
 - I.e., **backpropagation** is **potentially possible in brains**, but it is not pinned down how such a function occurs (if it does occur)

What are neural networks?

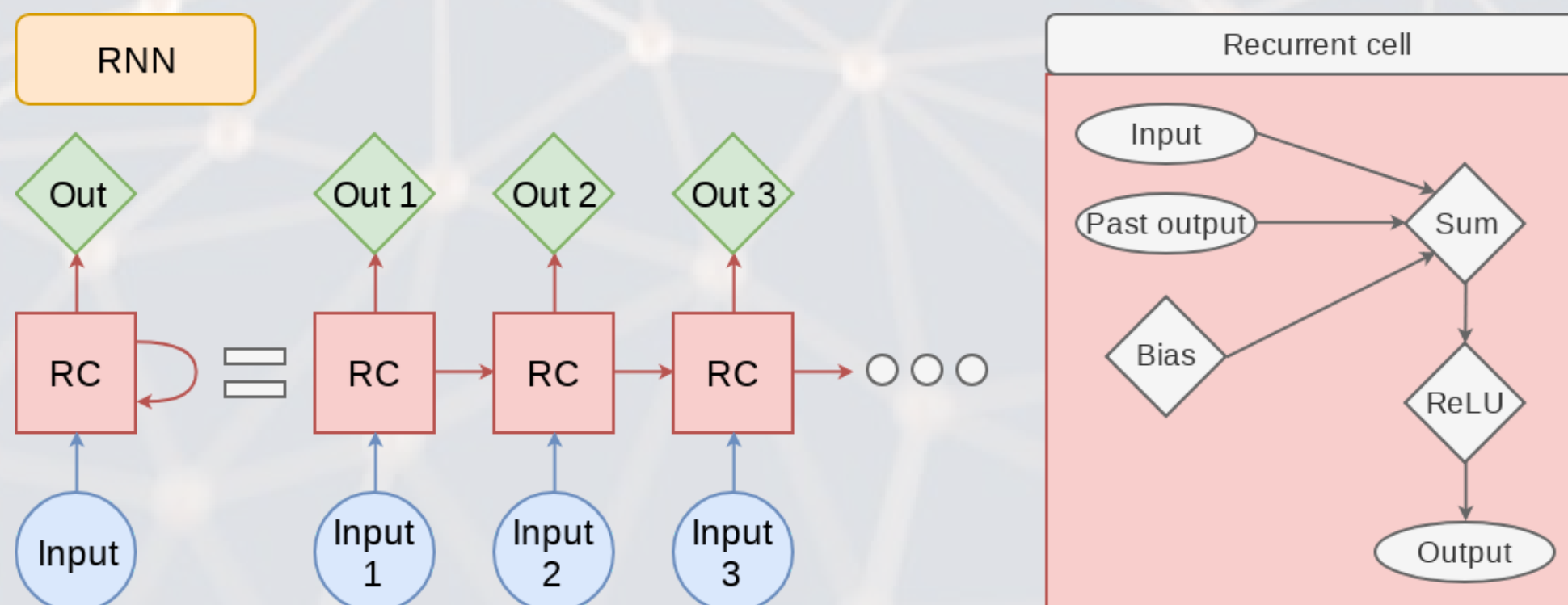
- Neural networks are a method by which a computer can learn from observational data
- In practice:
 - They were not computationally worthwhile until the mid 2000s
 - They have been known since the 1950s (perceptrons)
 - They can be used to construct algorithms that, at times, perform better than humans themselves
 - But these algorithms are often quite computationally intense, complex, and difficult to understand
 - Much work has been and is being done to make them more accessible

Types of neural networks

- There are *a lot* of neural network types
 - See The “[Neural Network Zoo](#)”
- Some of the more interesting ones which we will see or have seen:
 - RNN: Recurrent Neural Network
 - LSTM: Long/Short Term Memory
 - CNN: Convolutional Neural Network
 - DAN: Deep Averaging Network
 - GAN: Generative Adversarial Network
- Others worth noting
 - VAE (Variational Autoencoder): Generating *new* data from datasets

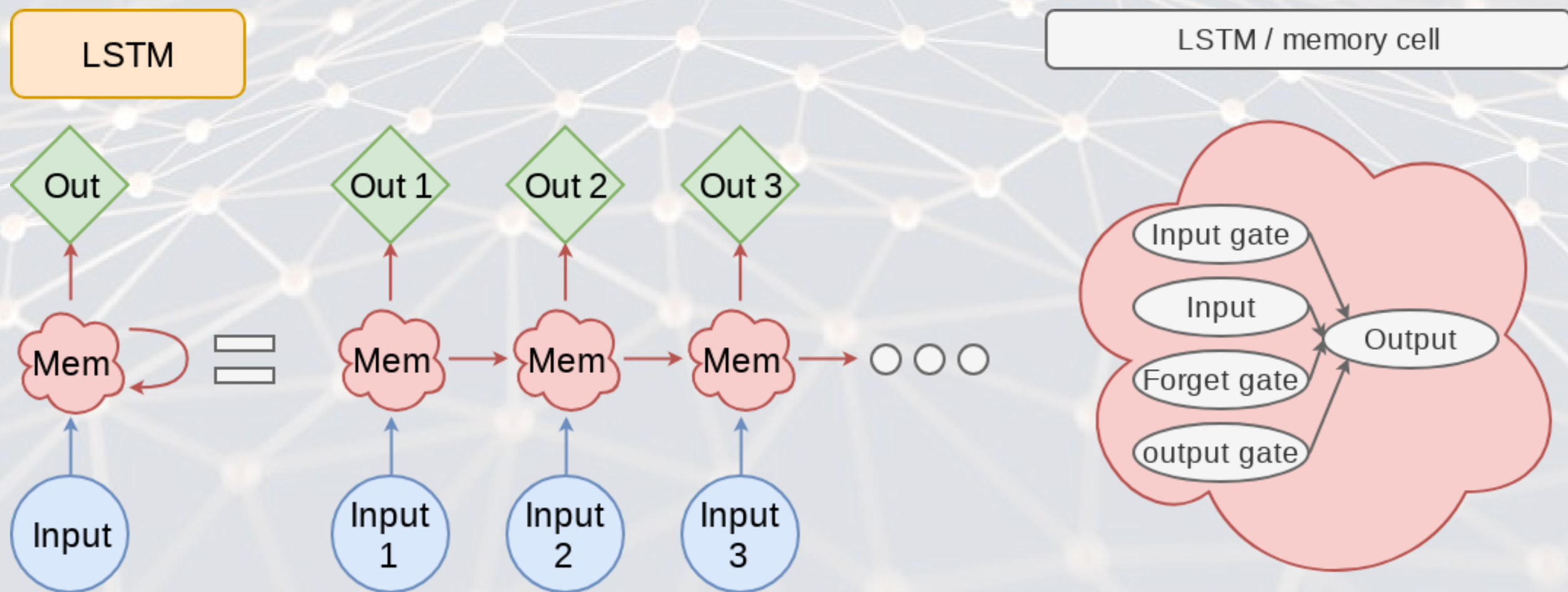
RNN: Recurrent NN

- Recurrent neural networks embed a history of information in the network
 - The previous computation affects the next one
 - Leads to a *short term memory*
- Used for speech recognition, image captioning, anomaly detection, and many others
 - Also the foundation of LSTM
 - [SketchRNN \(live demo\)](#)



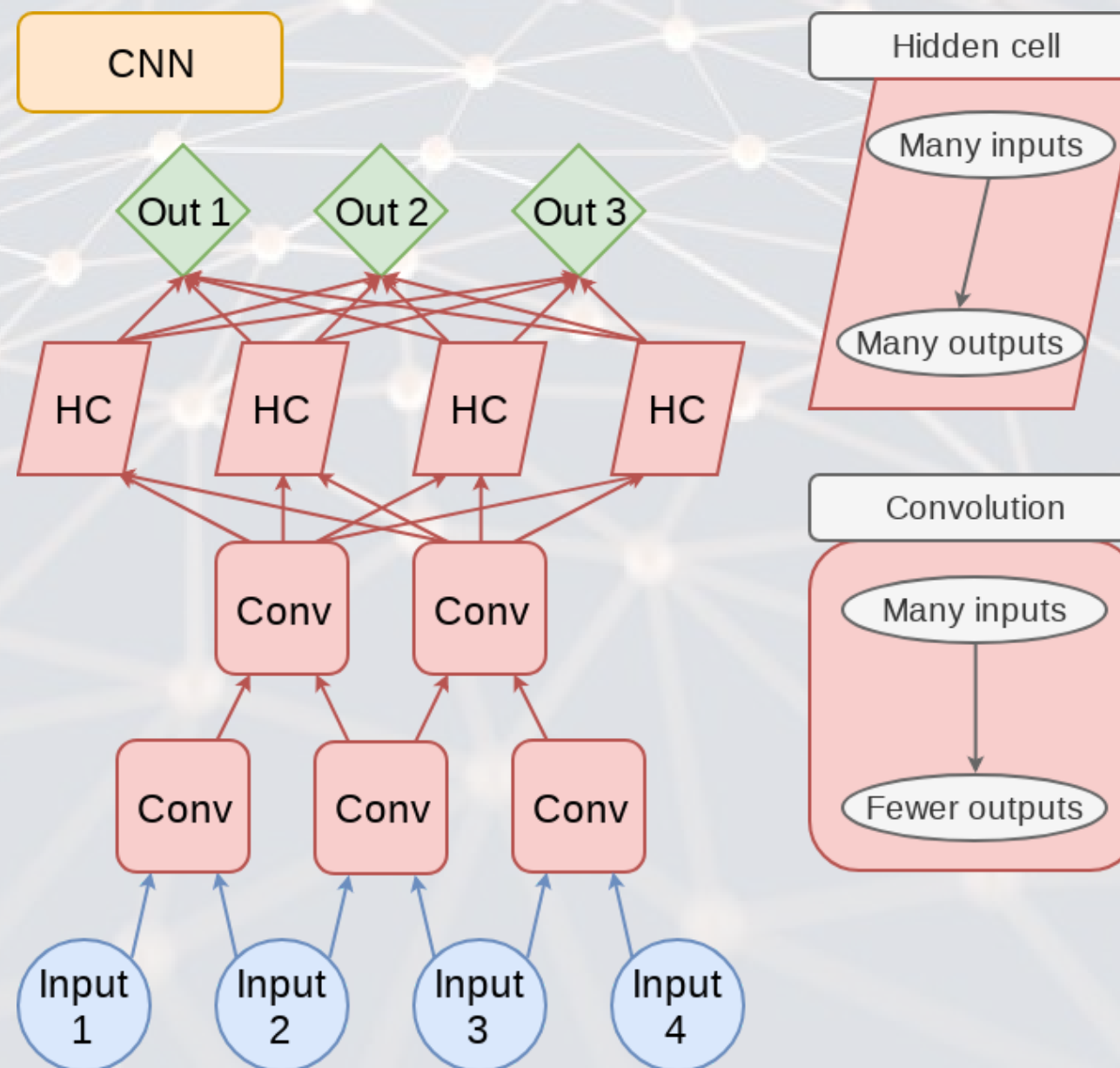
LSTM: Long Short Term Memory

- LSTM improves the *long term memory* of the network while explicitly modeling a *short term memory*
- Used wherever RNNs are used, and then some
 - Ex.: [Seq2seq](#) (machine translation)



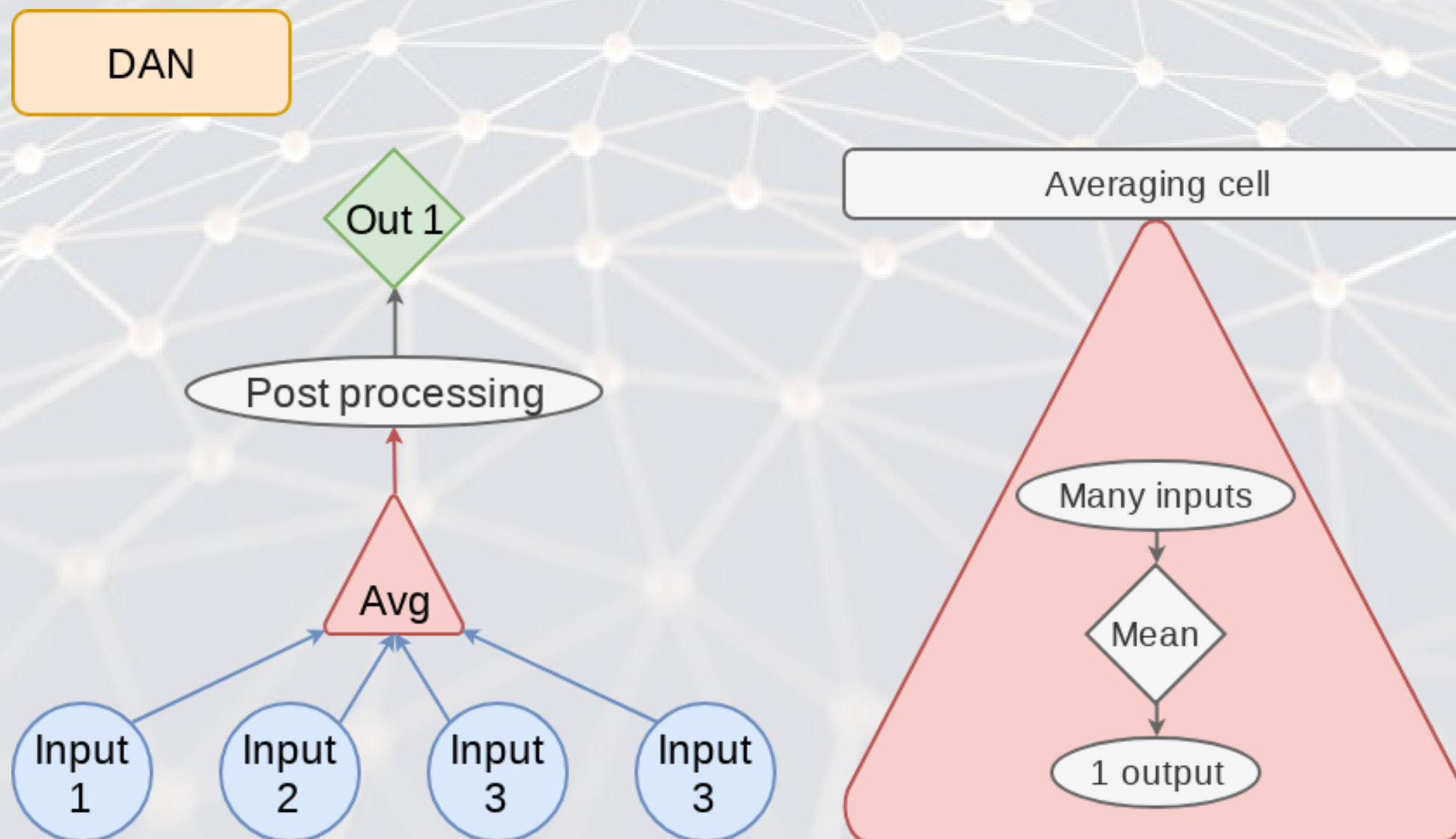
CNN: Convolutional NN

- Networks that excel at object detection (in images)
- Can be applied to other data as well
- Ex.: [Inception-v3](#)



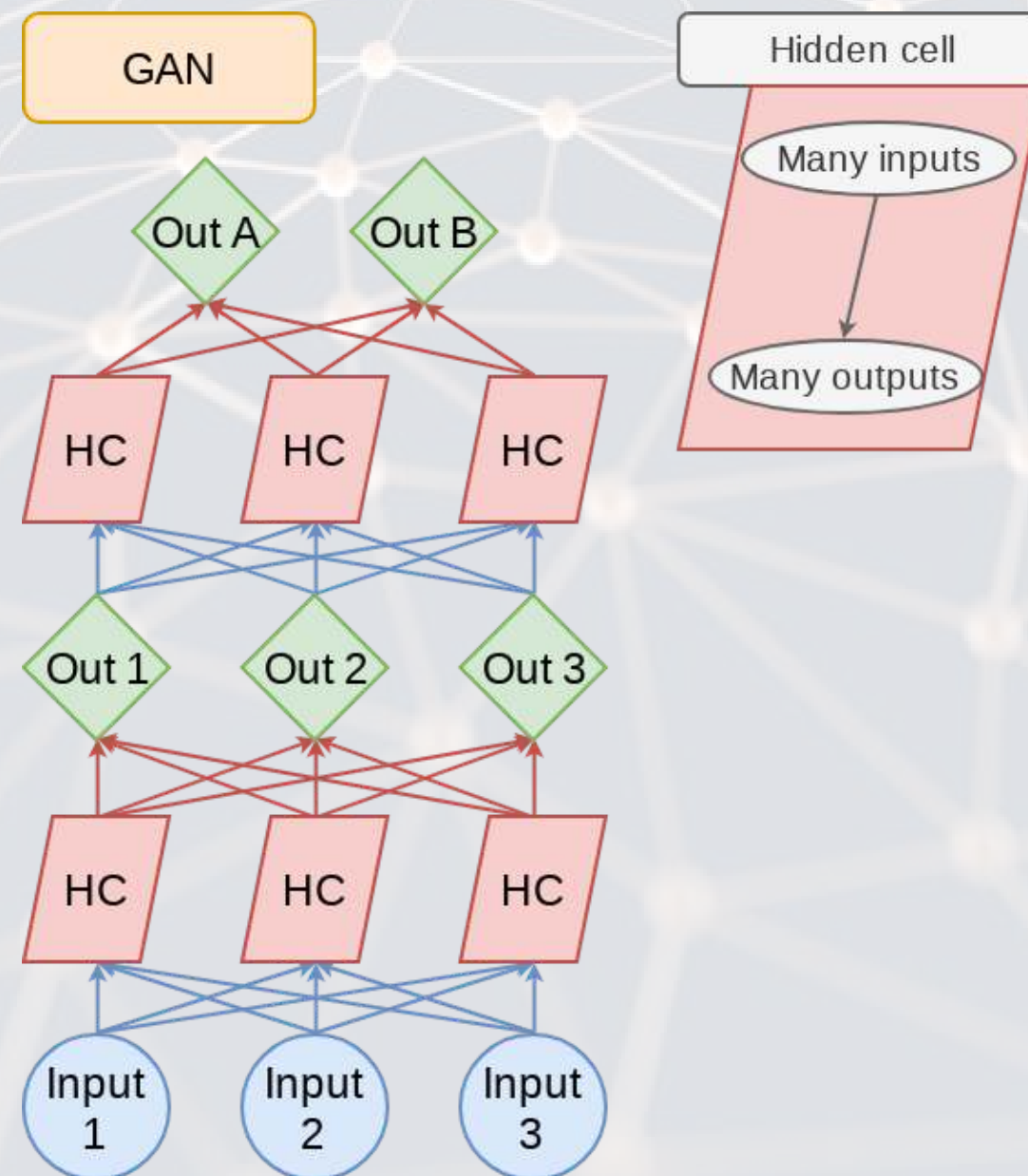
DAN: Deep Averaging Network

- DANs are simple networks that simply average their inputs
- Averaged inputs are then processed a few times
- These networks have found a home in NLP
 - Ex.: [Universal Sentence Encoder](#)



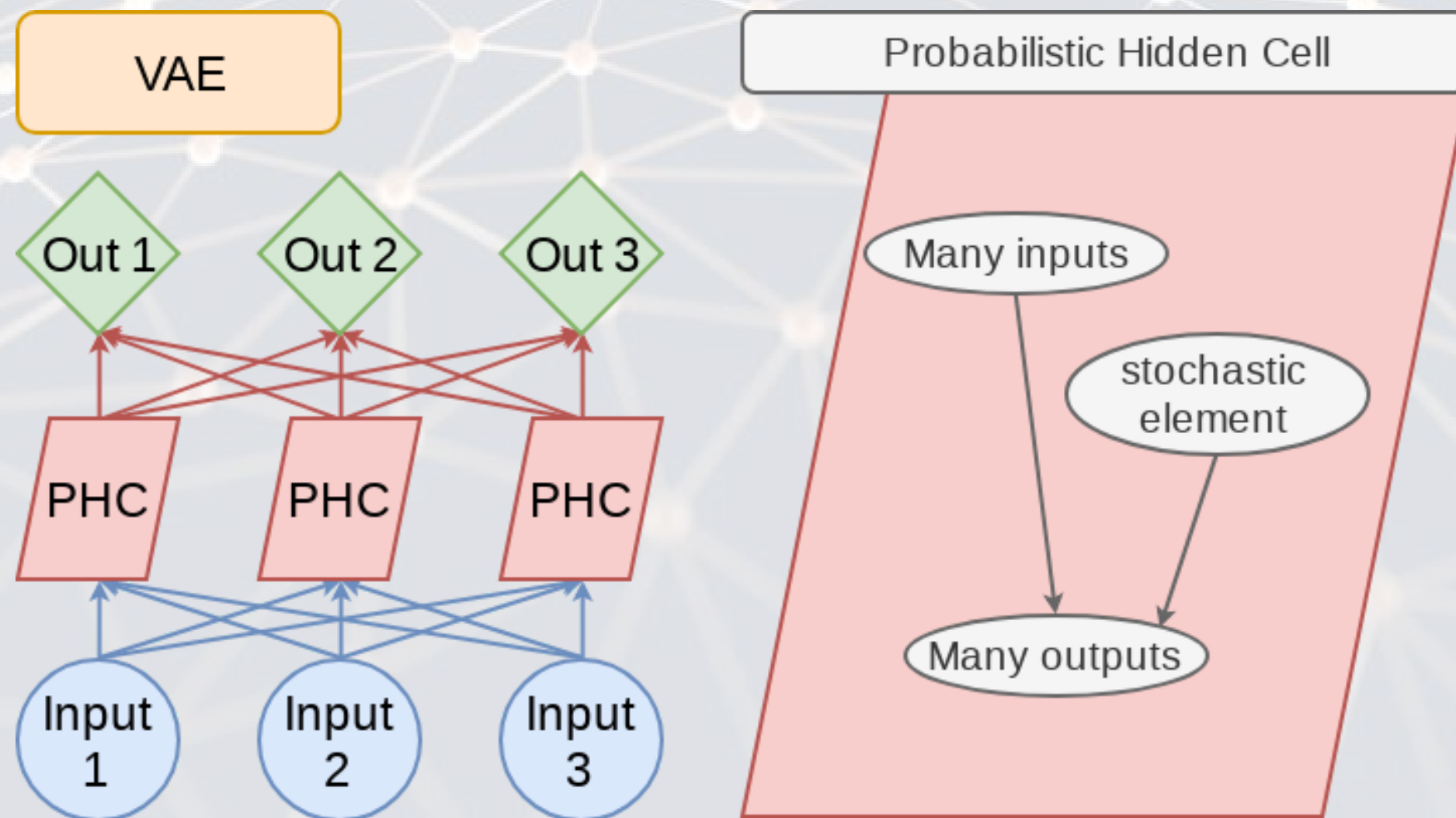
GAN: Generative Adversarial Network

- Feature two networks working against each other
- Many novel uses
 - Ex.: The anonymization GAN we saw
 - Ex.: [Aging images](#)




VAE: Variational Autoencoder

- An autoencoder (AE) is an algorithm that can recreate input data
- Variational means this type of AE can vary other aspects to generate completely new output
 - Good for creating **fake data**
- Like a simpler, noisier GAN



Vector space models

Motivating examples



Word association games powered by machine learning

A R C A D E

Think fast,
type fast!

PLAY ARCADE

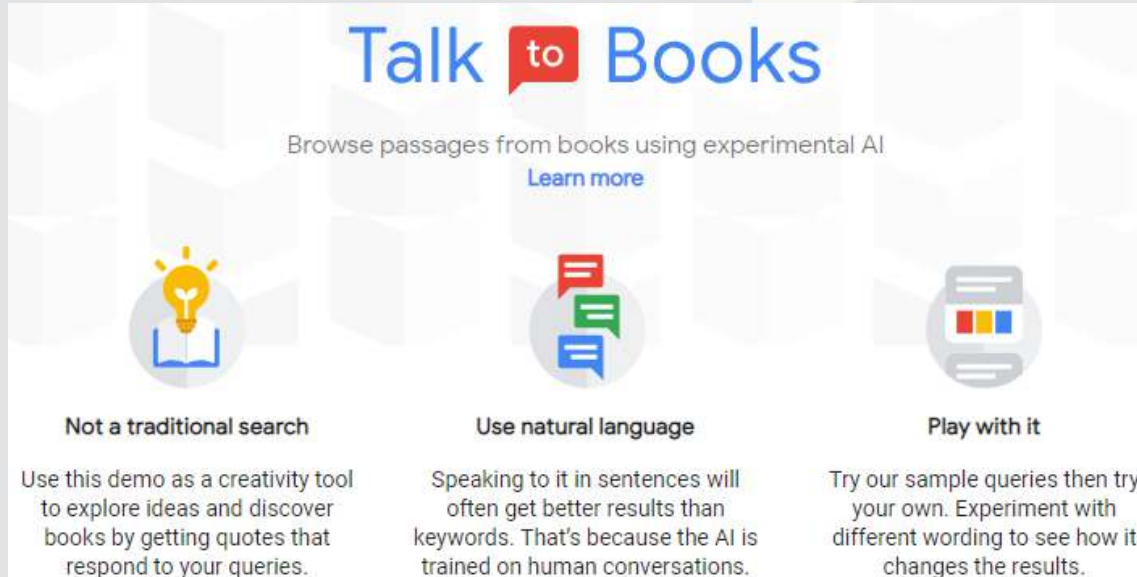
SKIP TUTORIAL

B L O C K S

Take your time and
puzzle it out!

PLAY BLOCKS


SKIP TUTORIAL





Talk to Books

Browse passages from books using experimental AI

[Learn more](#)

 **Not a traditional search**
Use this demo as a creativity tool to explore ideas and discover books by getting quotes that respond to your queries.

 **Use natural language**
Speaking to it in sentences will often get better results than keywords. That's because the AI is trained on human conversations.

 **Play with it**
Try our sample queries then try your own. Experiment with different wording to see how it changes the results.

What are “vector space models”

- Different ways of converting some abstract information into numeric information
 - Focus on maintaining some of the underlying structure of the abstract information
- Examples (in chronological order):
 - Word vectors:
 - [Word2vec](#)
 - [GloVe](#)
 - Paragraph/document vectors:
 - [Doc2Vec](#)
 - Sentence vectors:
 - [Universal Sentence Encoder](#)

Word vectors

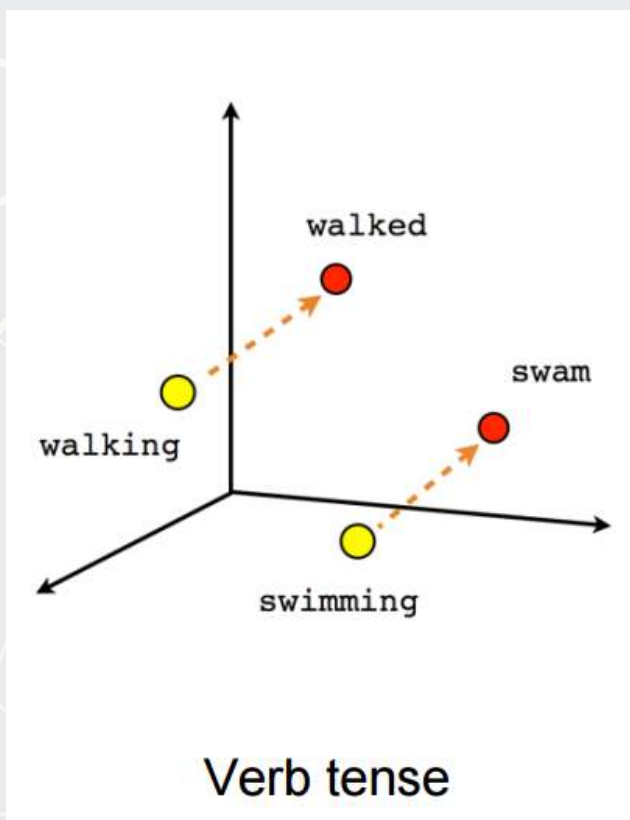
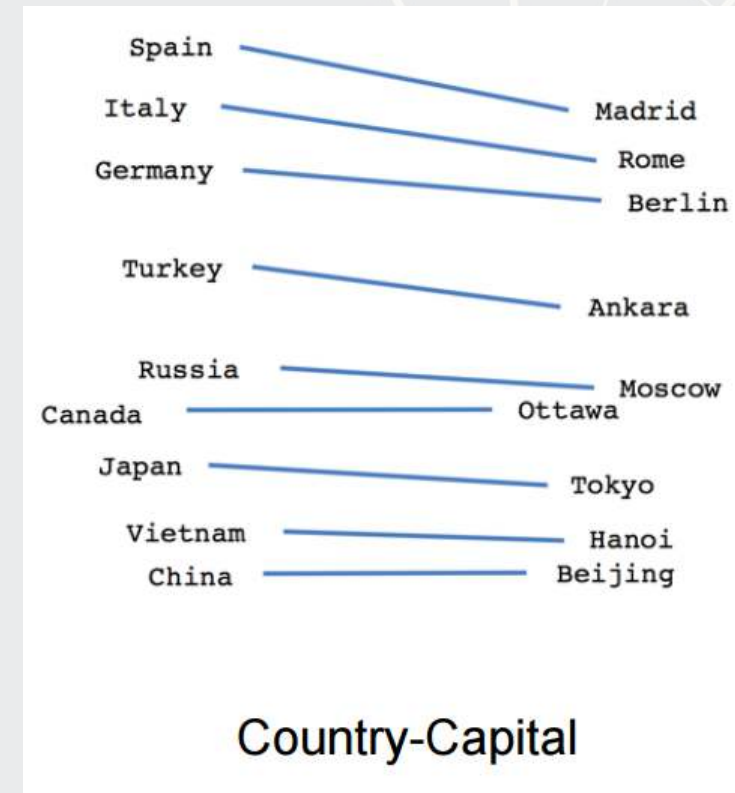
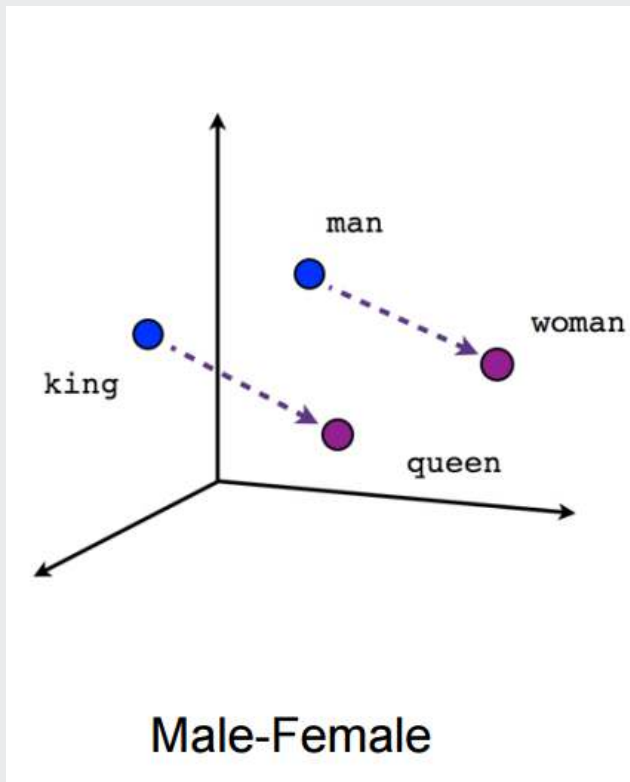
- Instead of coding individual words, encode word meaning
- The idea:
 - Our old way (encode words as IDs from 1 to N) doesn't understand relationships such as:
 - Spatial
 - Categorical
 - Grammatical (weakly when using stemming)
 - Social
 - etc.
 - Word vectors try to encapsulate all of the above
 - They do this by encoding words as a vector of different features

Word vectors: Simple example

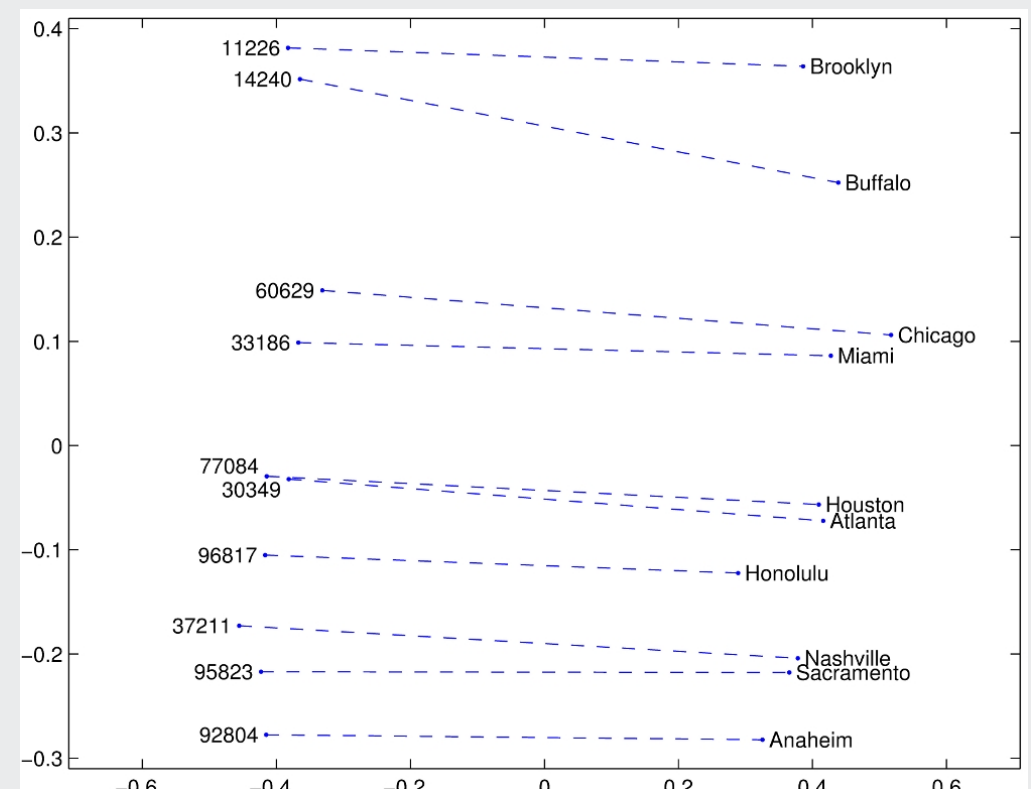
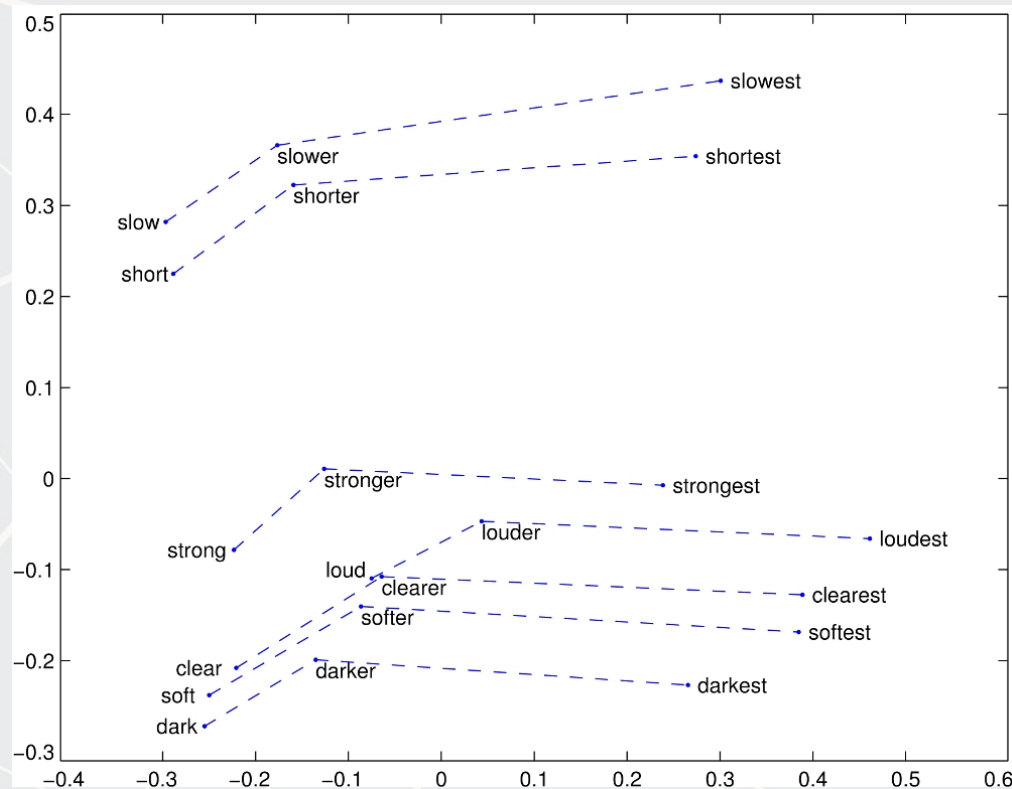
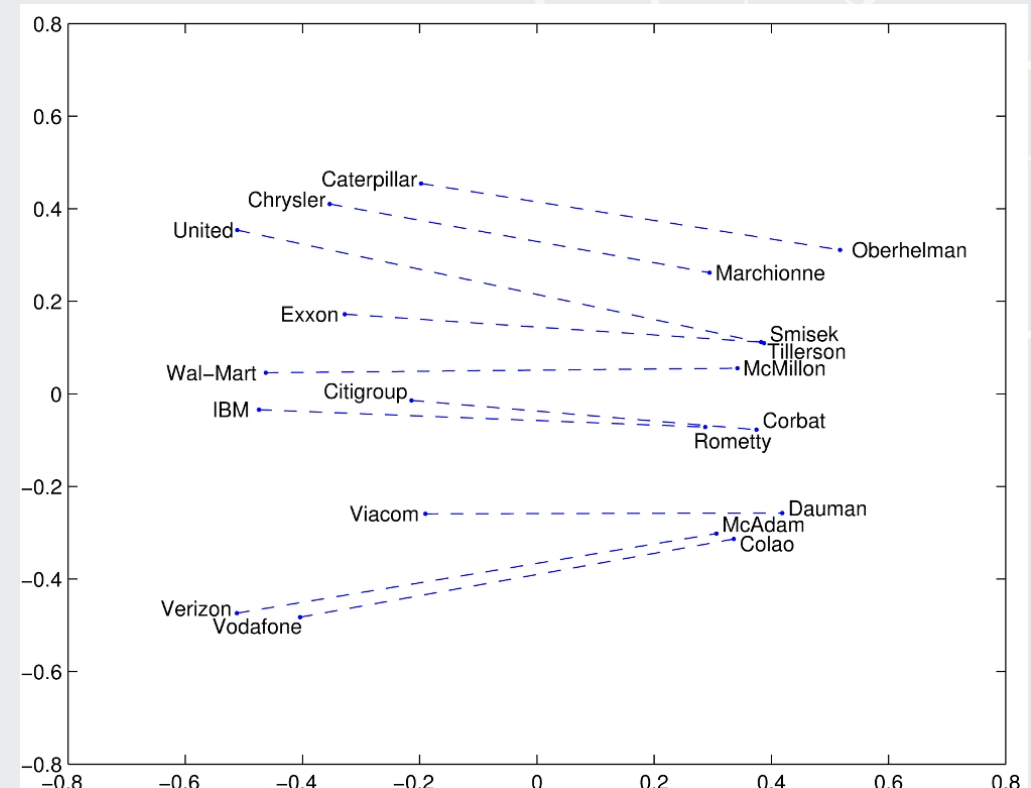
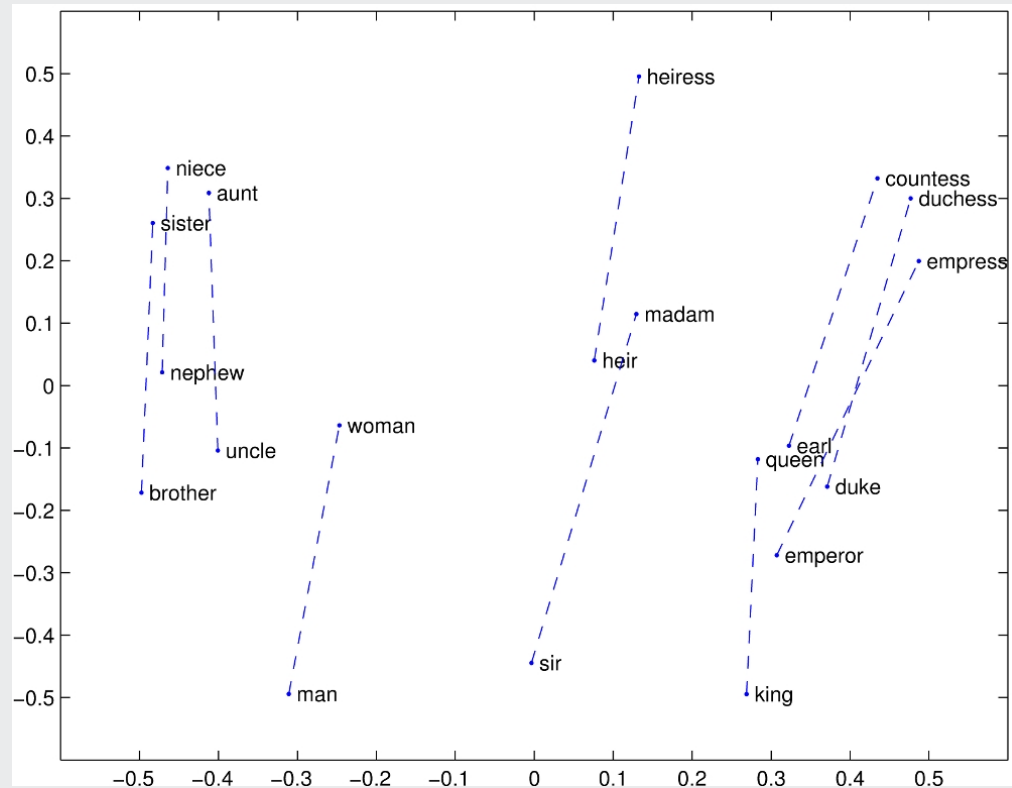
words	f_animal	f_people	f_location
dog	0.5	0.3	-0.3
cat	0.5	0.1	-0.3
Bill	0.1	0.9	-0.4
turkey	0.5	-0.2	-0.3
Turkey	-0.5	0.1	0.7
Singapore	-0.5	0.1	0.8

- The above is an idealized example
- Notice how we can tell apart different animals based on their relationship with people
- Notice how we can distinguish turkey (the animal) from Turkey (the country) as well

What it retains: word2vec



What it retains: GloVe

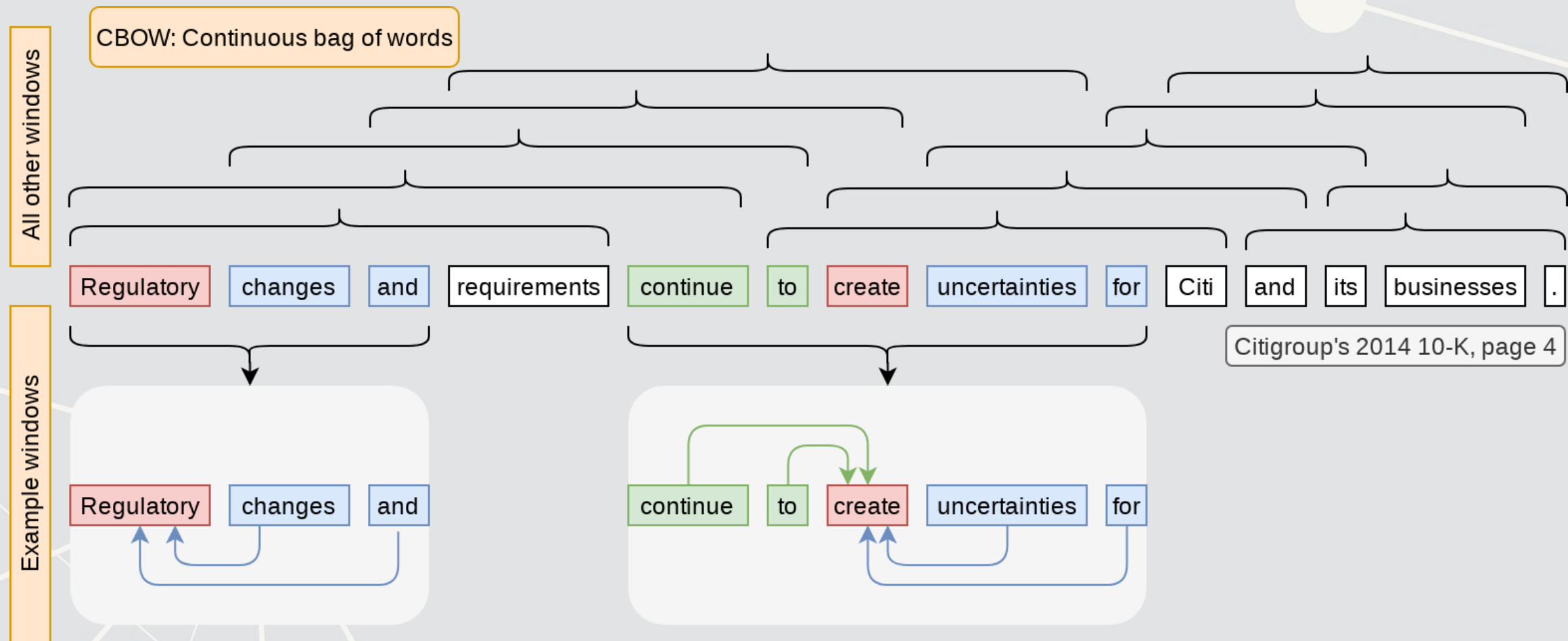


How to build word vectors

- Two ways:
 1. Word co-occurrence (like how LDA worked)
 - Global Vectors (GloVe) works this way
 - Available from the `text2vec` package
 2. Word order (using an NN)
 - `word2vec` works this way
 - Available from the `rword2vec` package
 - Uses a 2 layer neural network

How does word order work?

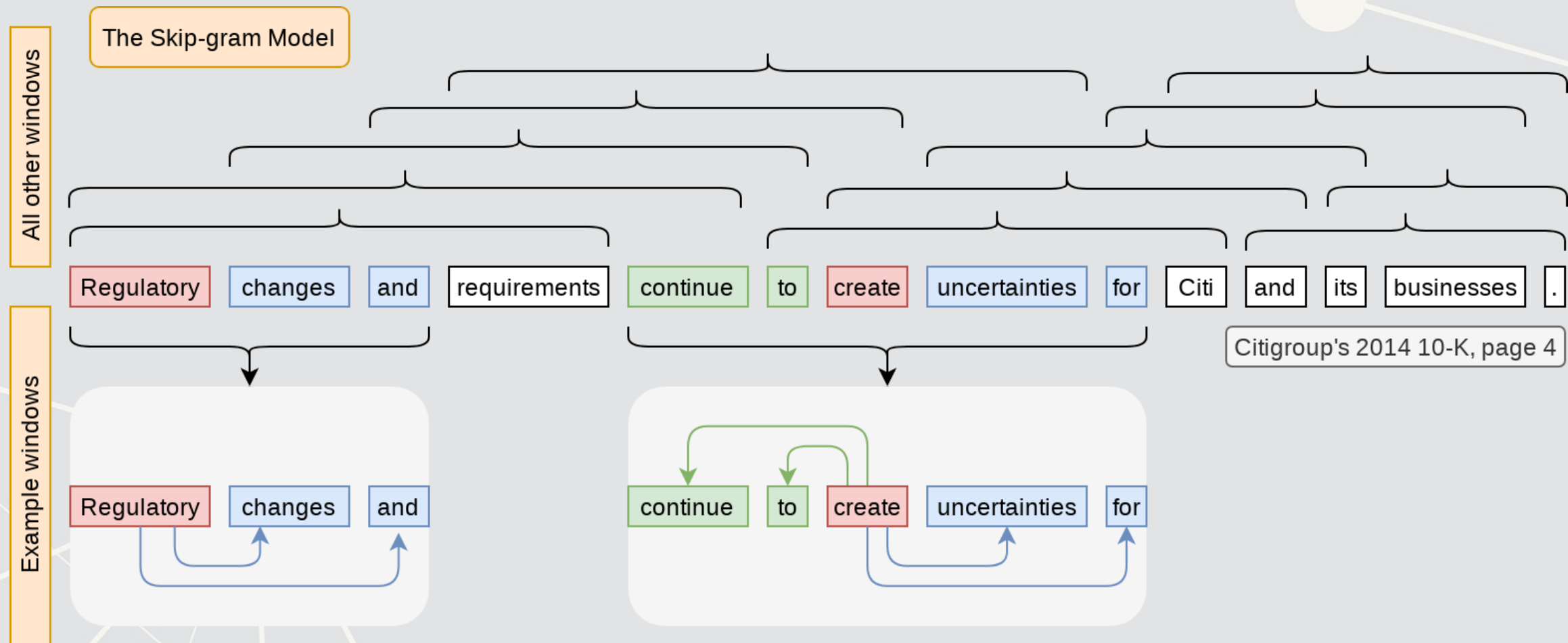
Infer a word's meaning from the words around it



Referred to as CBOW (continuous bag of words)

How else can word order work?

Infer a word's meaning by *generating* words around it



Referred to as the Skip-gram model

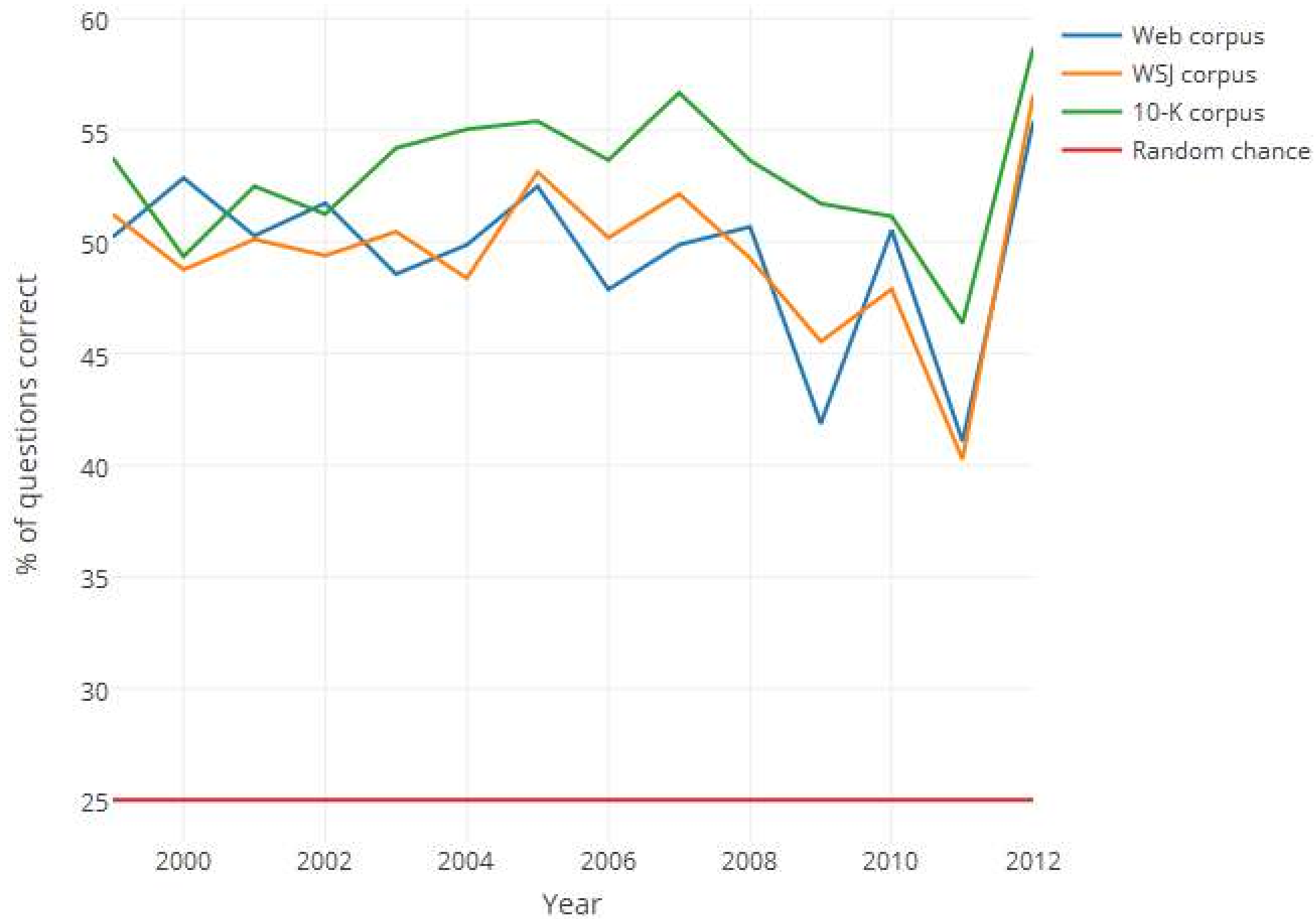
An example of using word2vec

- In the BCE paper from Session 6, word2vec was used to provide assurance that the LDA model works reasonably well on annual reports
 1. We trained a word2vec model on random issues of the Wall Street Journal (247.8M words)
 2. The resulting model “understood” words in the context of the WSJ
 3. We then ran a psychology experiment (word intrusion task) on the algorithm

Word intrusion task

- The task is to find which word doesn't belong
- Each question consisted of 3 words from 1 topic and 1 *intruded* from another random topic
 - Ex.:
 - **Laser**, Drug, Viral, Therapeutic
 - Supply, Steel, Capacity, **Losses**
 - Relief, Louisiana, **Cargo**, Assisted

Results



Implementing in R

- A few options:
 - The [rword2vec package](#) for word2vec
 - The [text2vec package](#) for GloVe
 - Rolling your own neural network for word2vec with [keras](#) ([guide here](#))

When are vector embeddings useful?

1. You care about the words used, by not stylistic choices
2. You want to crunch down a bunch of words into a smaller number of dimensions without running any bigger models (like LDA) on the text.

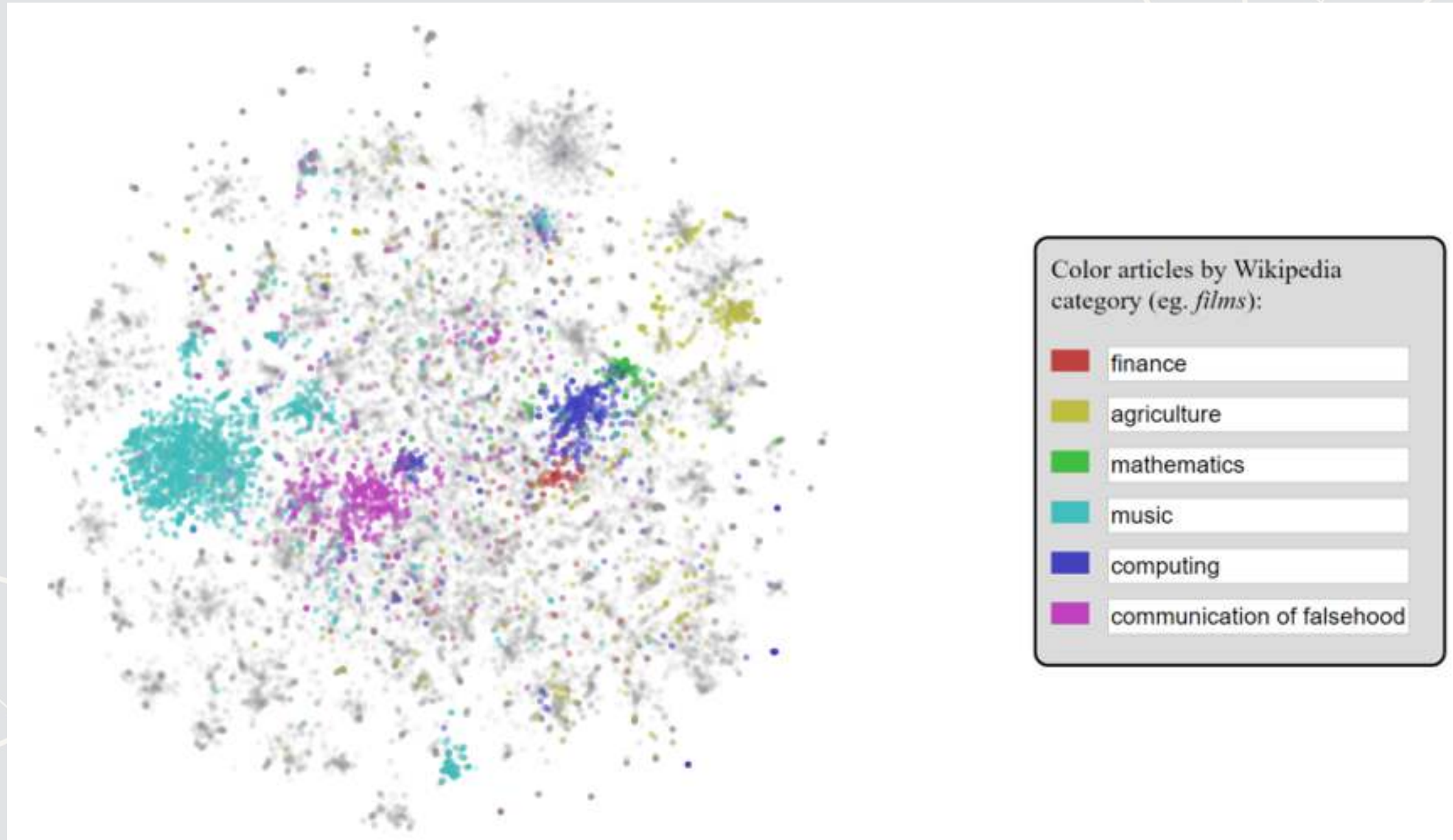
[An interactive demo of word similarity](#)

Understanding phrases (or larger)

Document vectors

- Document vectors work very similarly to word vectors
 - 1 added twist: a document/paragraph/sentence level factor variable
 - This is used to learn a vector representation of each text chunk
 - Learned simultaneously with the word vectors
 - Caveat: it can also be learned independently using [PV-DBOW](#)
- This is quite related to what we learned with LDA as well!
 - Both can tell us the topics discussed

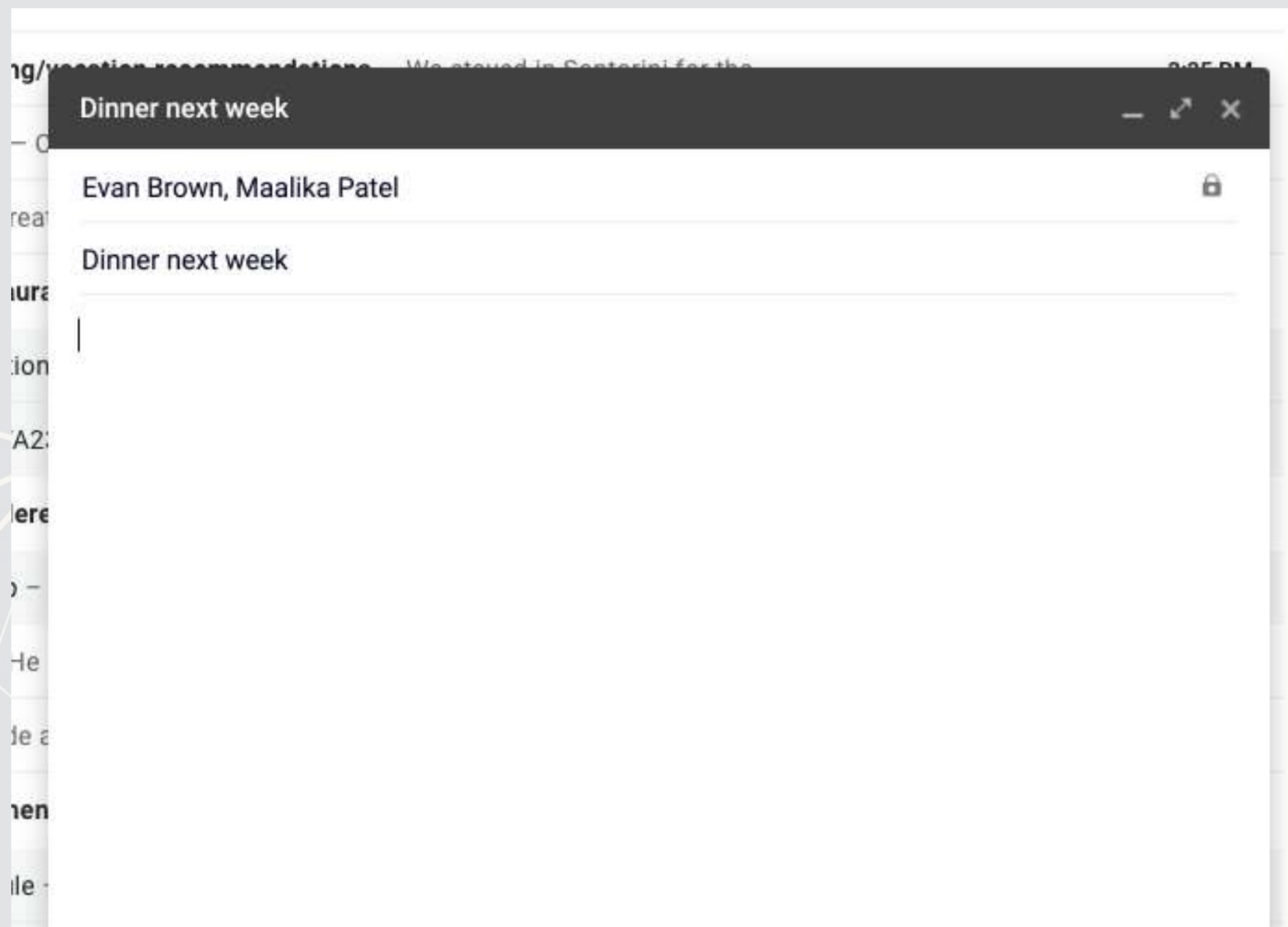
Wikipedia article categorization



Source article (colah.github.io)

Universal Sentence Encoder (USE)

- We saw this briefly last week
 - This is the algorithm with less bias
- Focused on representing sentence-length chunks of text



A fun example of with USE

- Predict Shakespeare with Cloud TPUs and Keras



Caveat on using USE

- One big caveat: USE only knows what it's trained on
 - Ex.: Feeding the same USE algorithm WSJ text

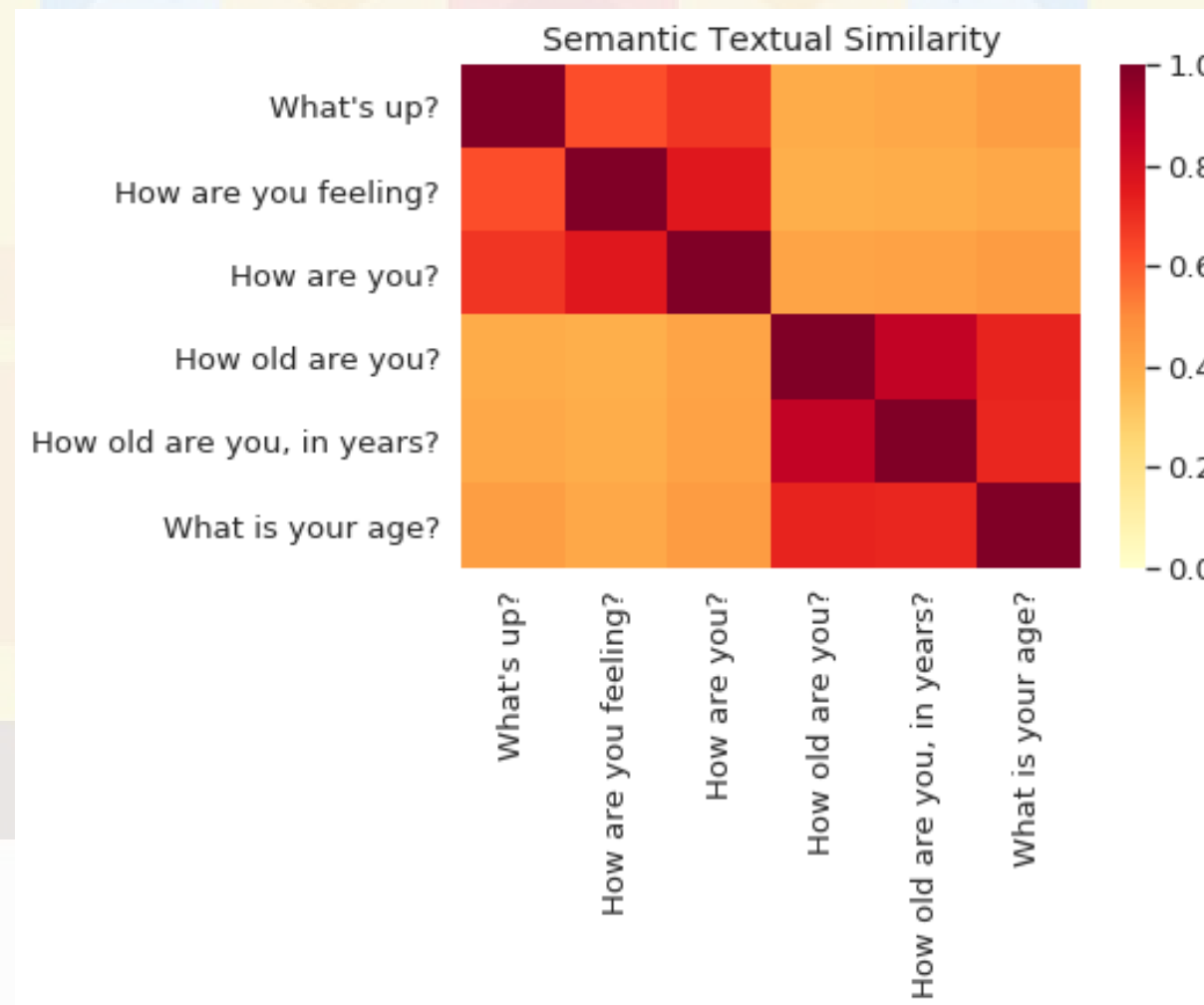
Samsung Electronics Co., suffering a handset sales slide, revealed a foldable-screen smartphone that folds like a book and opens up to tablet size. Ah, horror? I play Thee to her alone;
And when we have withdrom him, good all.
Come, go with no less through.

Enter Don Pedres. A flourish and my money. I will tarry.
Well, you do!

LADY CAPULET.
Farewell; and you are

How does USE work?

- USE is based on a DAN
 - There is another specification as well
 - Learns the meaning of sentences via words' meanings
- Learn more: [Original paper](#) and [TensorFlow site](#)
- In practice, it works quite well



Try it out!

- Run on [Google Colab](#)
 - Python code
 - Just click the cells in order, and click run
 - Colab provides free servers to run the code on
 - It still takes a few minutes to run though

Some newer things

- **BERT**

- A model structure that's been winning all sorts of competitions over the past year
 - Based on an architecture called “Transformer”
- Optimized to mimic question and answer behavior ([examples](#))
- Now used in [Google Search](#)
- [Additional reading](#)
- Available in [TensorFlow Hub](#)

- **XLNet**

- Similar objective to BERT, but with a focus on word order

End matter



Discussion

What creative uses for the techniques discussed today do you expect to see become reality in accounting in the next 3-5 years?

- Brainstorm with your group and try to come up with 1 good use for some technique discussed today
- Each group will be asked to share 1 use

TEAMWORK

Recap

Today, we:

- Learned formally what neural networks (NNs) are
- Discussed a variety of NN-based algorithms
- Saw uses for word and sentence vectors in a financial context

For next week

- For next week:
 - Work on the group project!
 - Definitely try to get a submission in on Kaggle
 - We'll keep talking about neural networks
 - A bit more theory
 - A lot more examples
 - Some real neural networks coded in **R**

More fun examples

- Interactive:
 - [TensorFlow.js examples](#)
 - [Semantris](#)
 - A game based on the Universal Sentence Encoder
- Non-interactive
 - [Predicting e-sports winners with Machine Learning](#)

Packages used for these slides

- `kableExtra`
- `knitr`
- `tidyverse`
 - `dplyr`, `magrittr`, `readr`

Generating Shakespeare

```
seed_txt = 'Looks it not like the king? Verily, we must go! ' # Original code
seed_txt = 'SCENE I. Elsinore. A platform before the Castle.\n\n Enter Francisco a
seed_txt = 'Samsung Electronics Co., suffering a handset sales slide, revealed a f
# From: https://www.wsj.com/articles/samsung-unveils-foldable-screen-smartphone-15
```