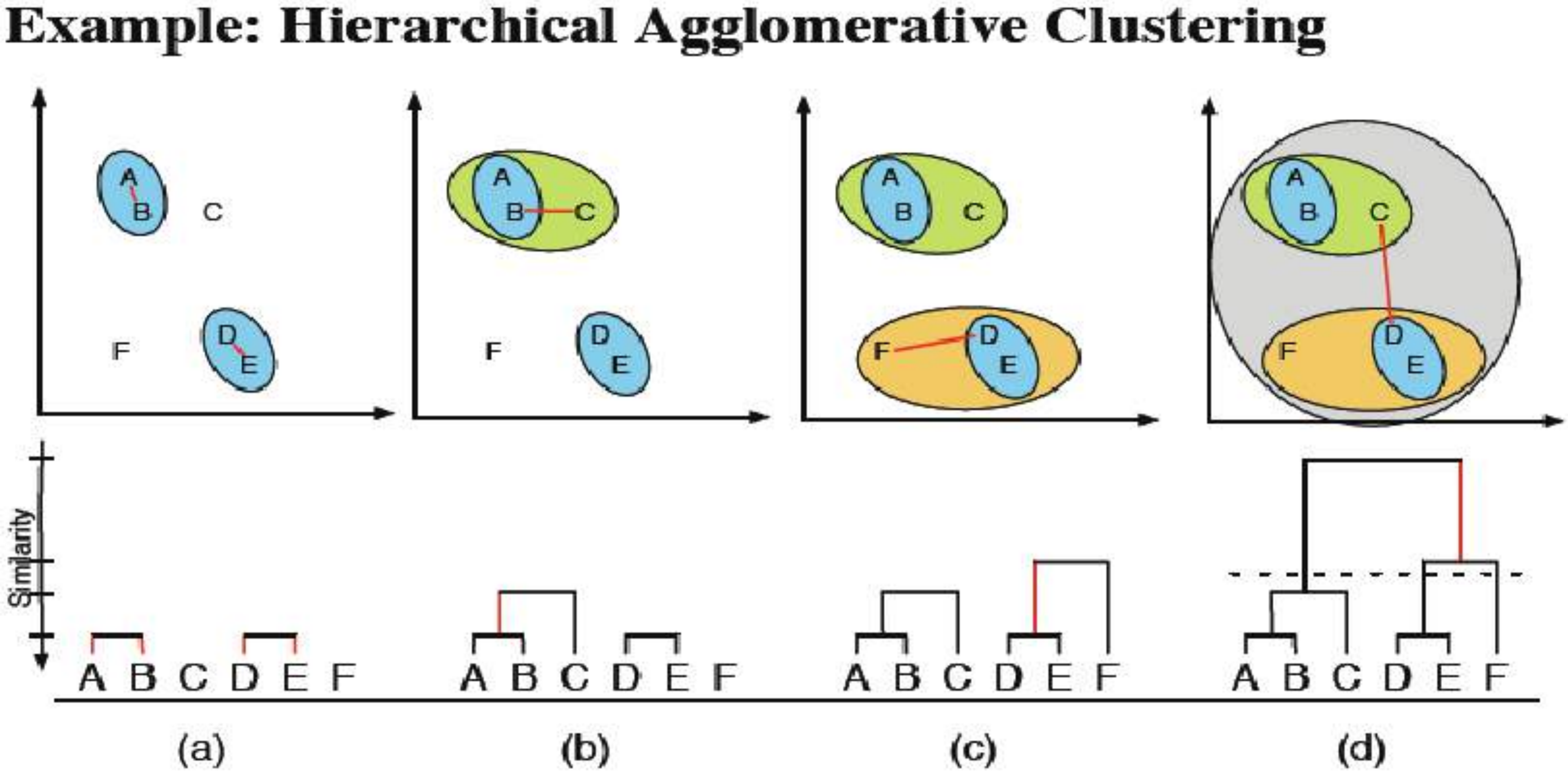
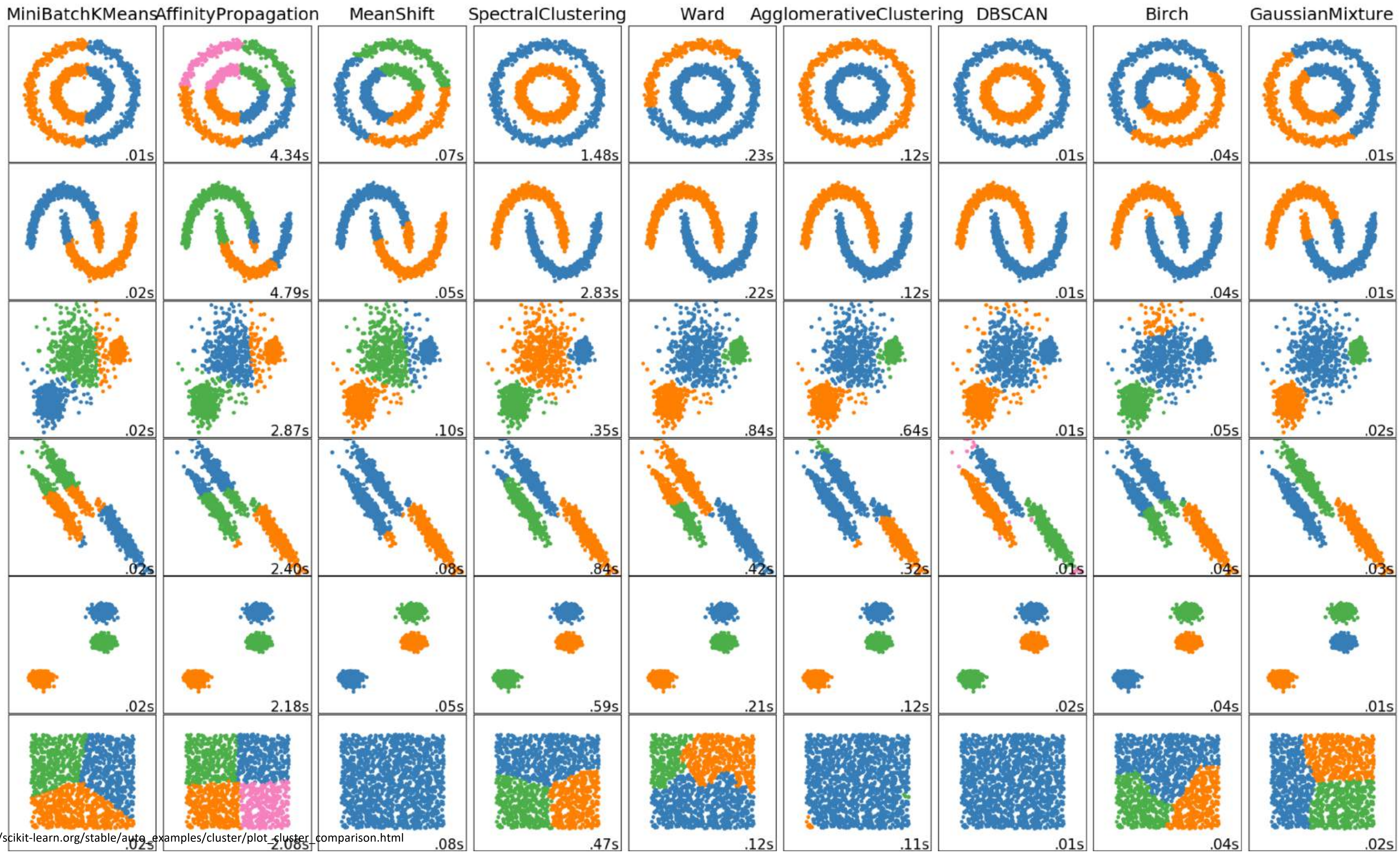
Hierarchical clustering



https://www.researchgate.net/publication/273456906_Cluster_Analysis_to_Understand_Socio-Ecological_Systems_A_Guideline/figures?lo=1

Hierarchical clustering

- Usually most computationally efficient
- Truly deterministic(?)
  - Are other forms of clustering deterministic?



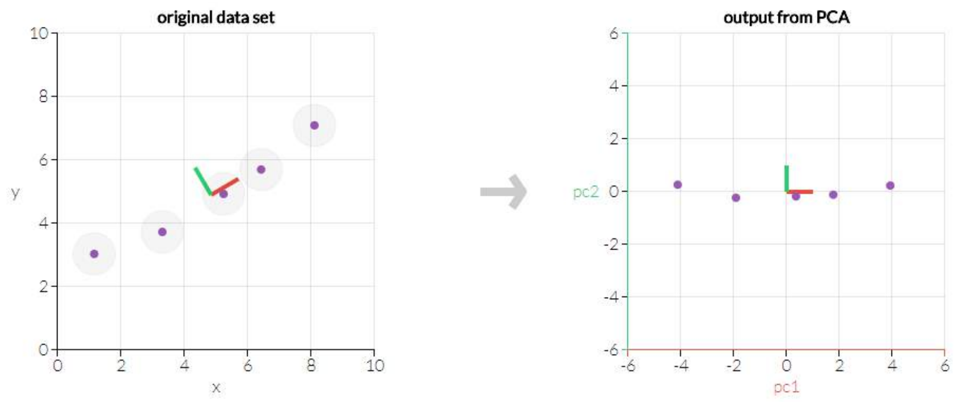
http://scikit-learn.org/stable/auto_examples/cluster/plot_cluster_comparison.html

Pitfalls of clustering

- Choice of method affects clusters identified
- How many clusters? How do you determine #?
- Non-deterministic (depending on seeding)
- Do all datapoints share similar profile?
- Check percent of data points in each cluster (e.g.5%-35%)
- Outlier treatment

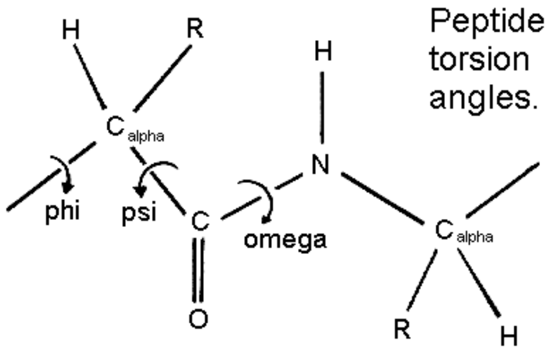
Decomposition/Dimensionality Reduction

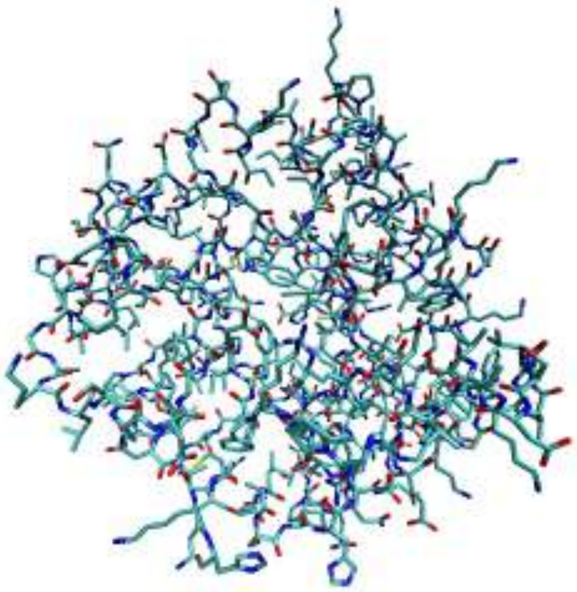
- Useful to perform as preprocessing
- Can tell you about the structure of your data
- **PCA** is one of the most common methods
  - ICA, TICA, etc.

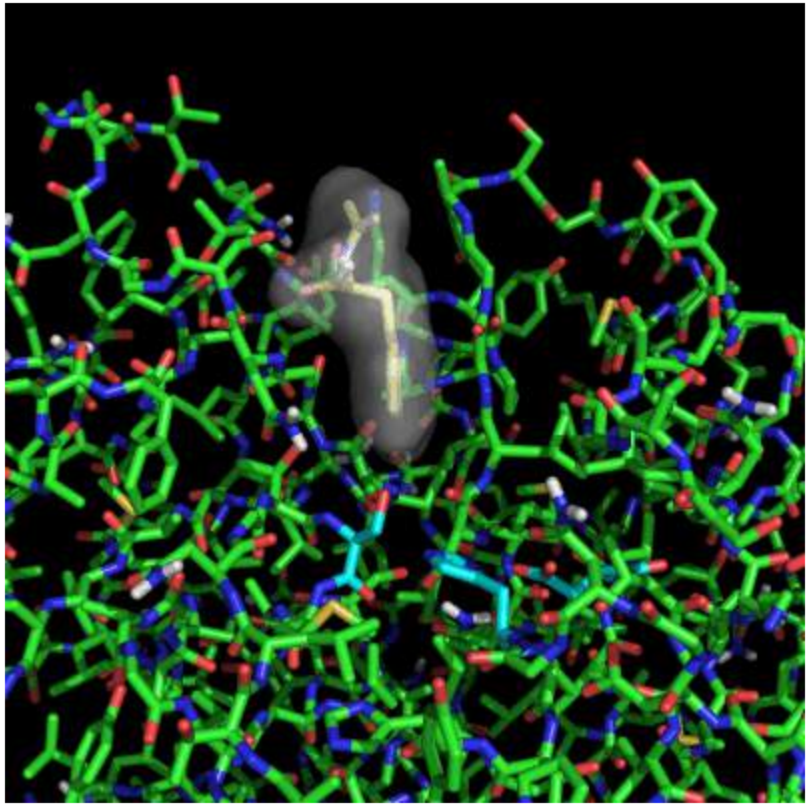


- http://setosa.io/ev/principal-component-analysis/

Personal Example





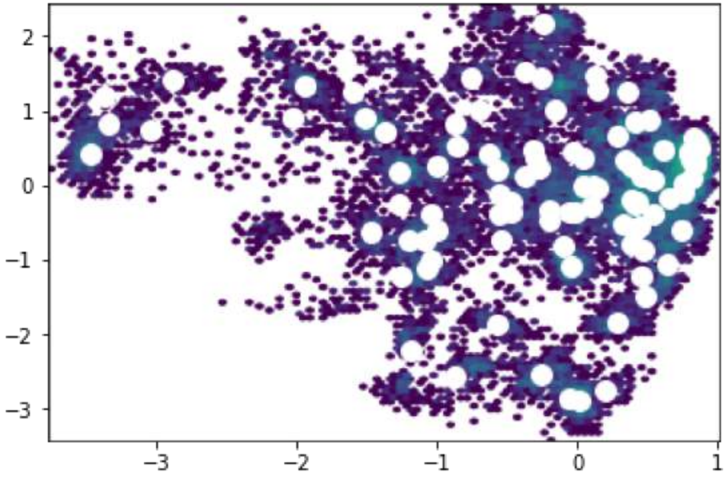


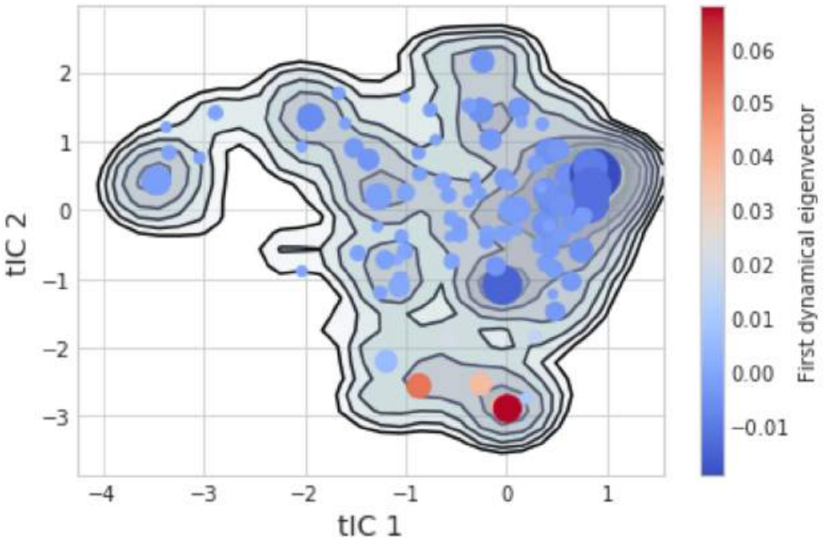
http://www.cryst.bbk.ac.uk/PPS95/course/9_quaternary/3_geometry/torsion.html

https://eagereyes.org/blog/2010/a-visual-language-for-proteins-jane-richardson

http://www.itqb.unl.pt/labs/protein-modelling/activities/upseknas

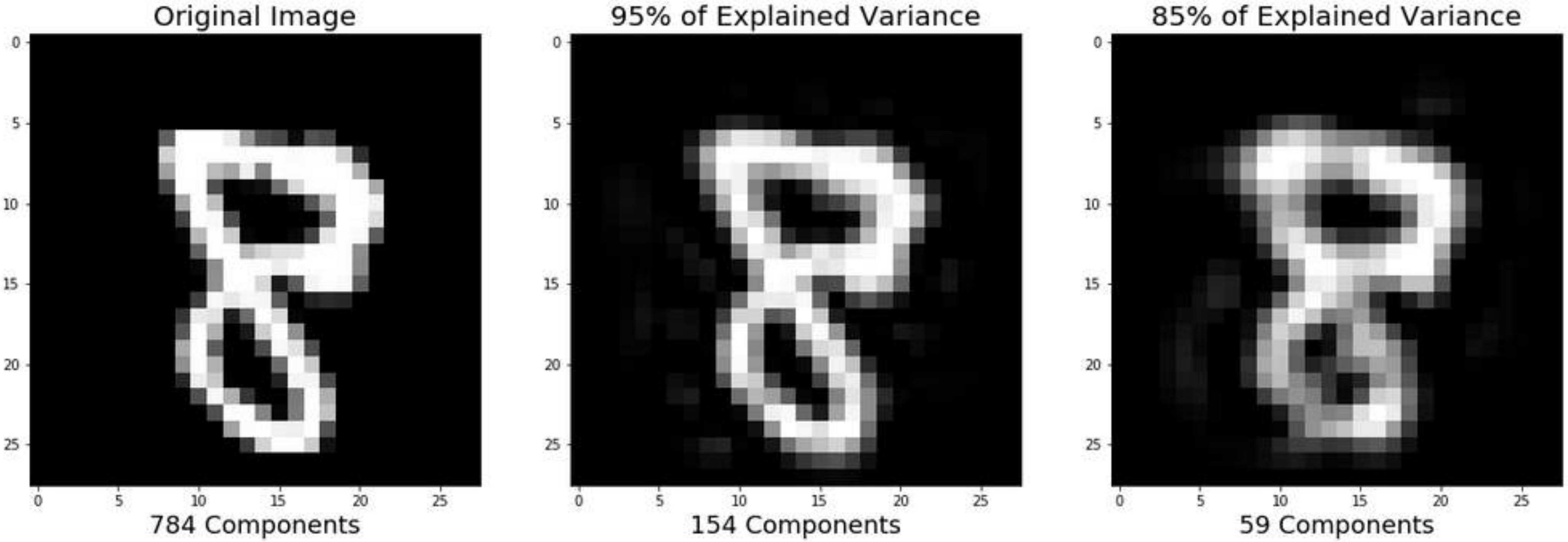
Personal Example





http://msmbuilder.org/3.8.0/examples/Fs-Peptide-with-dataset.html

PCA – Image Compression



https://towardsdatascience.com/pca-using-python-scikit-learn-e653f8989e60

Sklearn pipelines

```
pipeline = Pipeline([
    ('ordinal_to_nums', DFTransform(_ordinal_to_nums, copy=True)),
    ('union', DFFeatureUnion([
        ('categorical', Pipeline([
            ('select', DFTransform(lambda X: X.select_dtypes(include=['object']))),
            ('fill_na', DFTransform(lambda X: X.fillna('NA'))),
            ('one_hot', DFTransform(_one_hot_encode)),
        ])),
        ('numerical', Pipeline([
            ('select', DFTransform(lambda X: X.select_dtypes(exclude=['object']))),
            ('fill_median', DFTransform(lambda X: X.fillna(X.median()))),
            ('add_features', DFTransform(_add_features, copy=True)),
            ('remove_skew', DFTransform(_remove_skew, copy=True)),
            ('find_outliers', DFTransform(_find_outliers, copy=True)),
            ('normalize', DFTransform(lambda X: X.div(X.max())))
        ])),
    ])),
```

https://signal-to-noise.xyz/post/sklearn-pipeline/

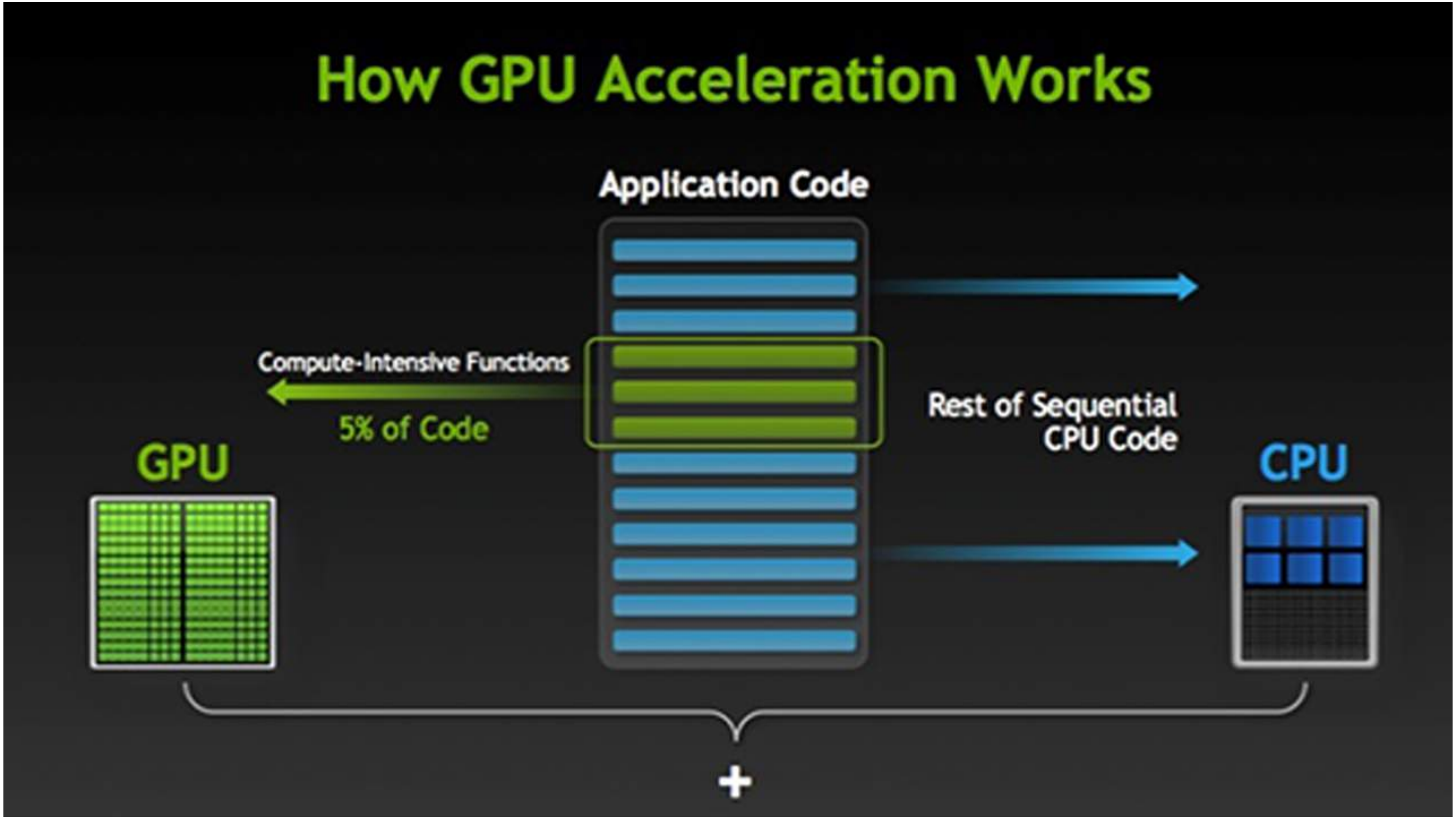
Natural Language Processing

- SpaCy https://nicschrading.com/project/Intro-to-NLP-with-spaCy/

- Harry Potter http://botnik.org/content/harry-potter.html

Advances that enable ML

- Computational Architecture
- Parallel processing
- High Performance Computing
- GPU-acceleration



https://www.electronics-cooling.com/2017/02/thermal-management-gpu-enabled-servers-data-centers/

Troubleshooting ML

- Overfitting?
  - Reduce # of features
    - Manually select
    - Model selection (sklearn has options for this)

```
>>> from sklearn.feature_selection import VarianceThreshold
>>> X = [[0, 0, 1], [0, 1, 0], [1, 0, 0], [0, 1, 1], [0, 1, 0], [0, 1, 1]]
>>> sel = VarianceThreshold(threshold=(.8 * (1 - .8)))
>>> sel.fit_transform(X)
array([[0, 1],
       [1, 0],
       [0, 0],
       [1, 1],
       [1, 0],
       [1, 1]])
```

http://scikit-learn.org/stable/modules/feature_selection.html

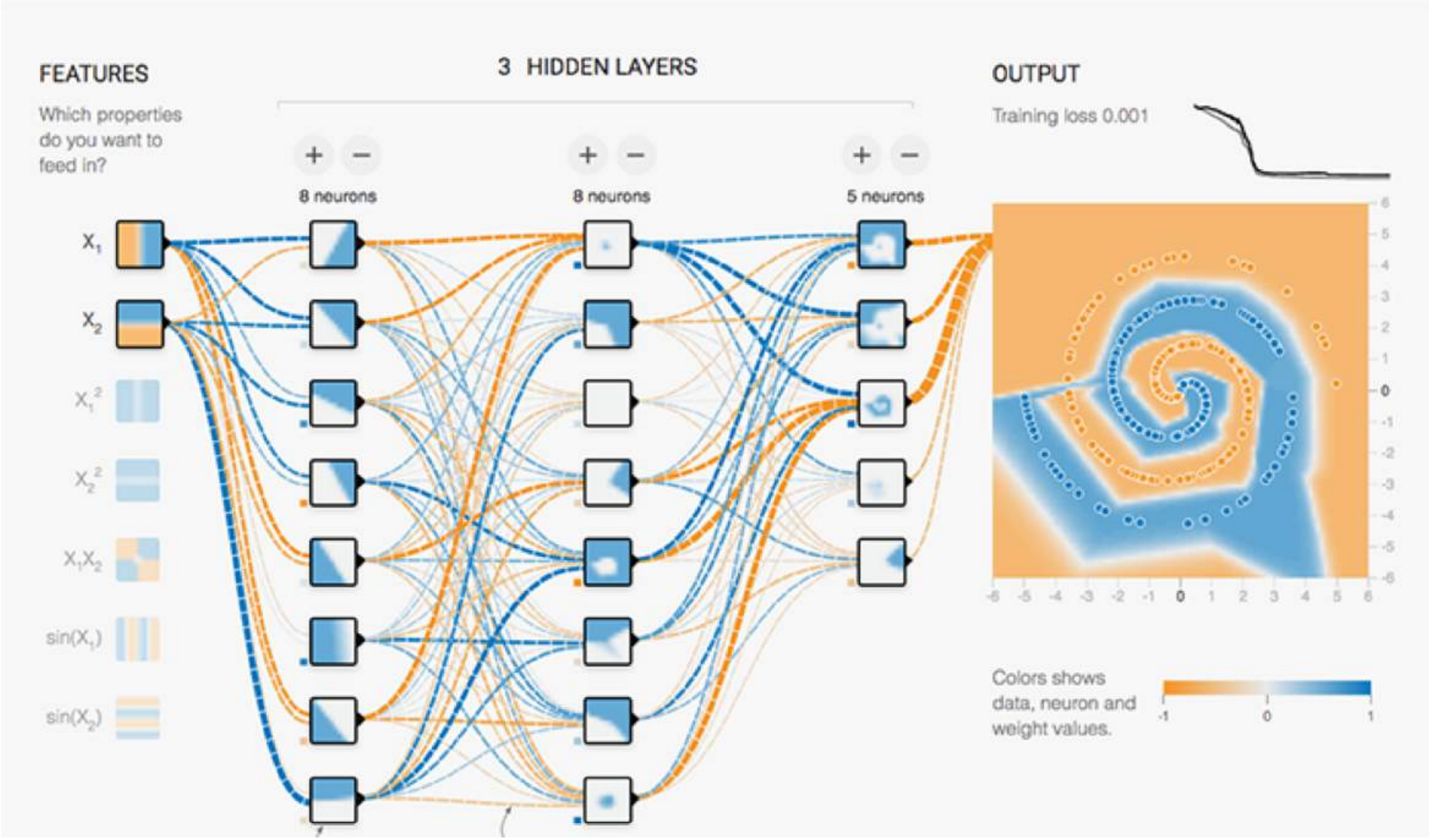
Troubleshooting ML

- Overfitting?
  - Reduce # of features
    - Manually select
    - Model selection (sklearn has options for this)
  - Regularization
  - Revisit scaling of features

Troubleshooting ML

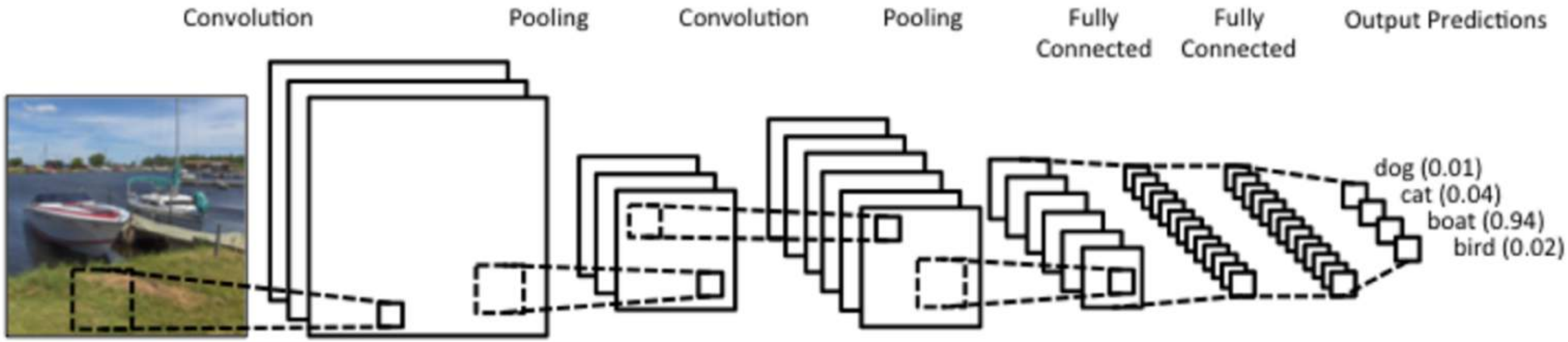
- Large errors
  - More training examples? (addressing high variance)
  - Smaller sets of features (addressing high variance)
  - Additional features (addressing high bias)
  - Adding polynomial features x^2, x^3, etc. (sklearn has options for this as well) (addressing high variance)

Deep Learning



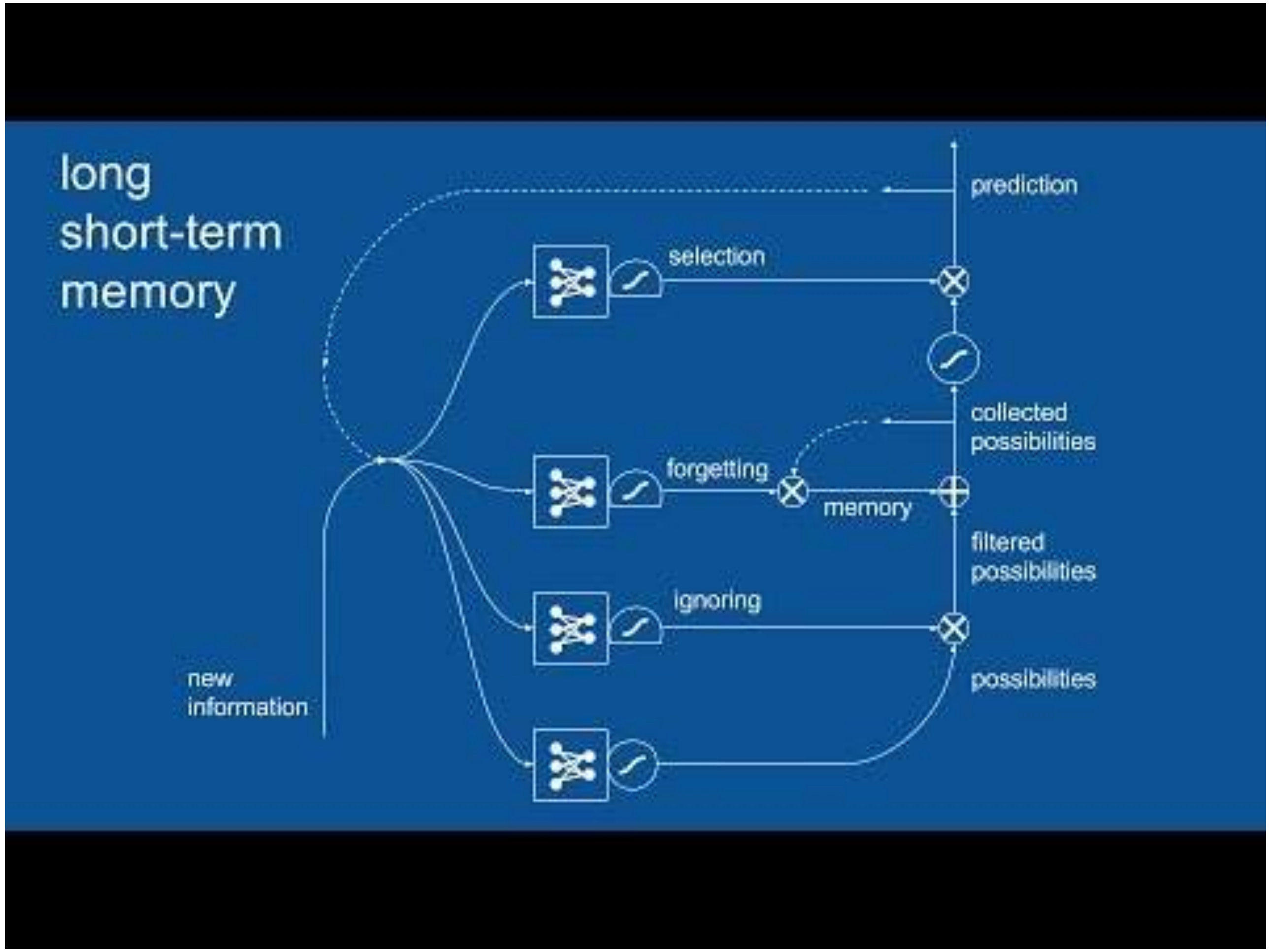
https://cloud.google.com/blog/big-data/2016/07/understanding-neural-networks-with-tensorflow-playground

Convolutional Neural Networks (CNNs)



http://www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-nlp/

Long Short Term Memory



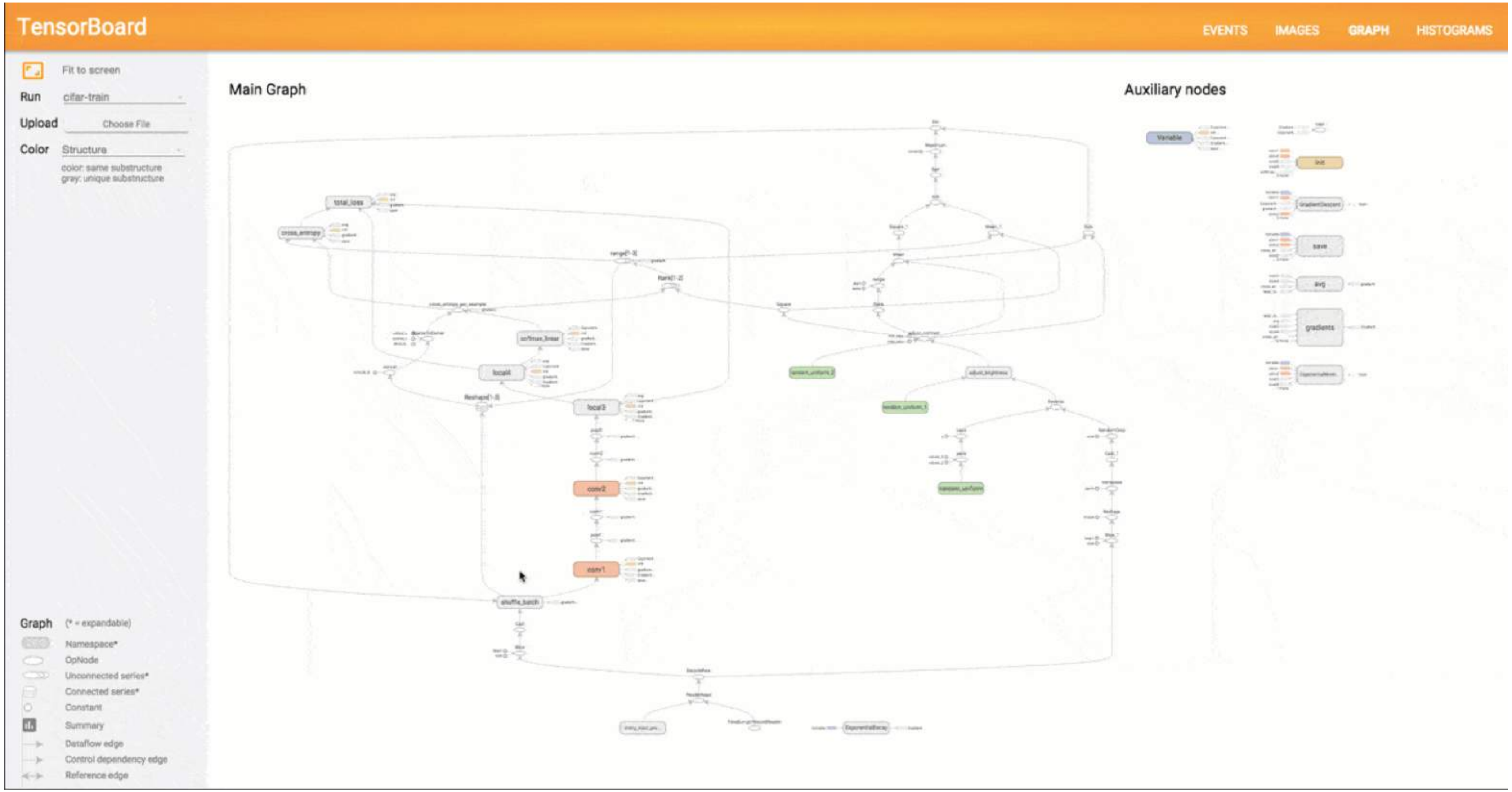
https://www.youtube.com/watch?v=WCUNPb-5EYI

Tensorflow

htt

https://techcrunch.com/2017/06/16/object-detection-api/

Tensorboard



Keras

Keras: The Python Deep Learning library

You have just found Keras.

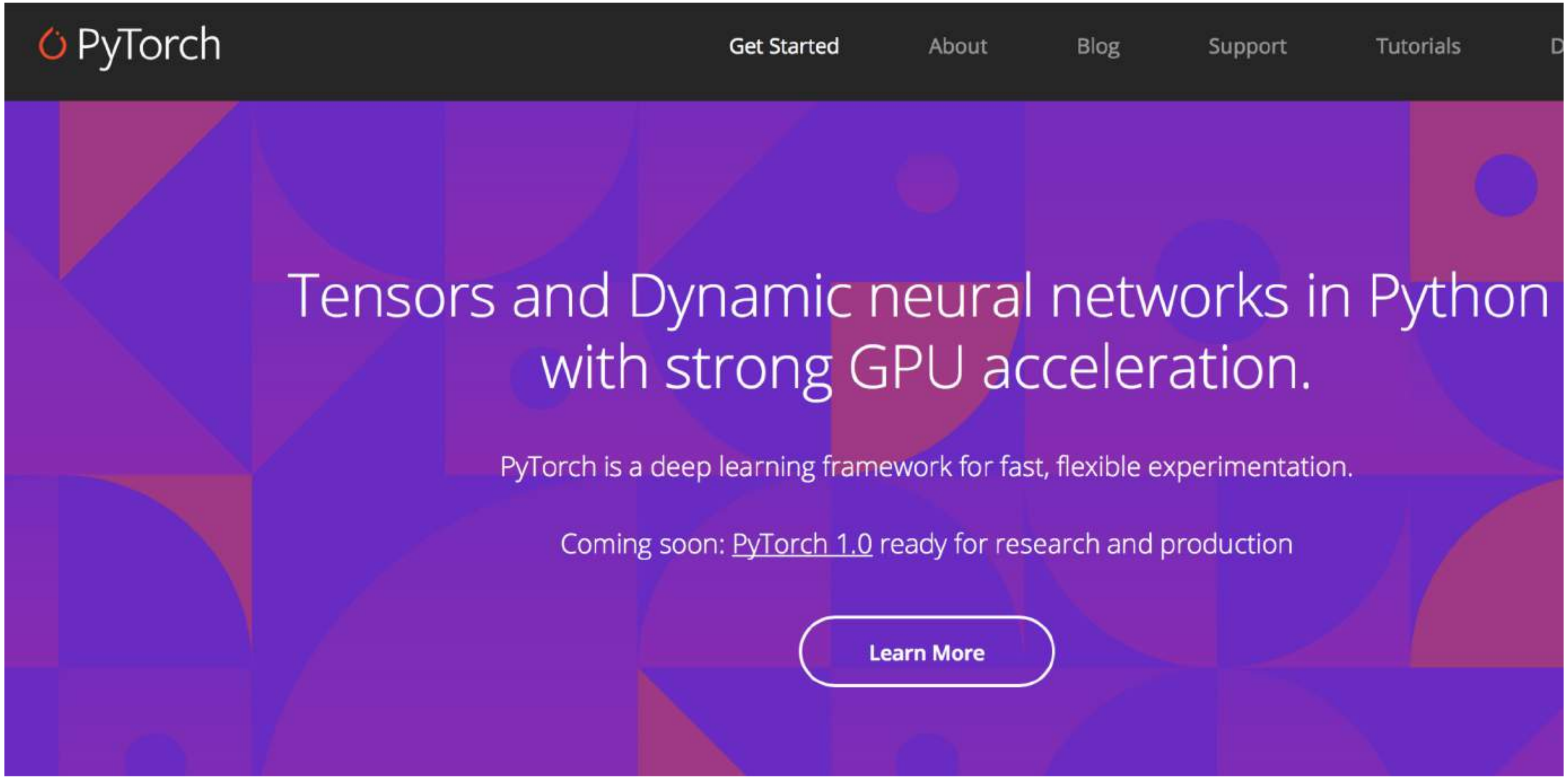
Keras is a high-level neural networks API, written in Python and capable of running on top of TensorFlow, CNTK, or Theano. It was developed with a focus on enabling fast experimentation. *Being able to go from idea to result with the least possible delay is key to doing good research.*

Use Keras if you need a deep learning library that:

- Allows for easy and fast prototyping (through user friendliness, modularity, and extensibility).
- Supports both convolutional networks and recurrent networks, as well as combinations of the two.
- Runs seamlessly on CPU and GPU.

https://keras.io

PyTorch



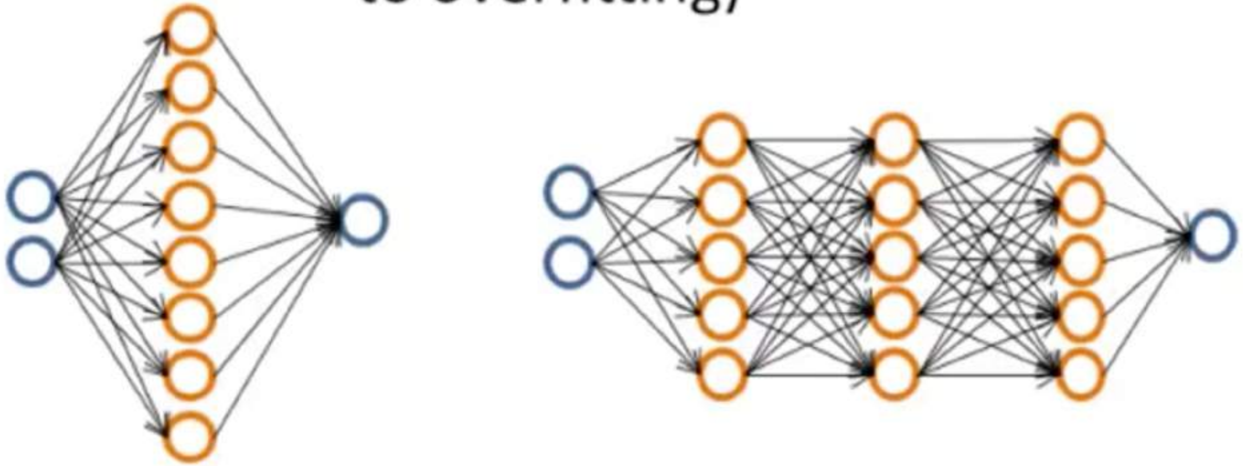
https://pytorch.org

Neural networks and overfitting

"Small" neural network
(fewer parameters; more
prone to underfitting)

Computationally cheaper

"Large" neural network
(more parameters; more prone
to overfitting)



Computationally more expensive.

Use regularization (λ) to address overfitting.

When is machine learning or deep learning appropriate?

- Consider the problem at hand
- How much data do you have available?
- How computationally intensive is your approach?
- Can your dataset be loaded into memory entirely?
- Do you have a GPU accessible?
- Interoperability (sklearn pickling)
- Deep learning hype

Data Sources & Learning

- UCI
- Kaggle
- 538, NYT

- Coursera ML Andrew Ng
- Google machine learning course
- Fast.ai