Agenda

Machine learning basics
Deep learning
Applications in Astronomy
Machine learning settings

Supervised learning
(mostly machine)

Unsupervised learning
(mostly human)

Anywhere in between: semi-supervised learning, reinforcement learning, lifelong learning, meta-learning, few-shot learning, knowledge-based ML

Will be quickly solved for easy problems (Andrew Ng)
Best tricks in machine learning

Best classifiers
- Deep Neural Networks
- XGBoost
- Random Forests

Choosing right priors
- Extensive feature engineering
- Model architecture
- Loss functions
- Hyper-parameter tuning

Managing uncertainty
- Data augmentation
- Ensemble methods
- Bayesian methods

Model reuse
- Domain adaptation
- Transfer learning
- Multitask learning
Feature engineering learning

In typical machine learning projects, 80-90% effort is on feature engineering.

- A right feature representation doesn’t need fancy classifiers to work well.

**Text**: BOW, n-gram, POS, topics, stemming, tf-idf, etc.

**Image**: Histogram, SIFT, HOG, Filter banks, LBP, whitening, centring, color correction, denoising, etc.

Try yourself on Kaggle.com!
Why ML works?

Expressiveness

- Can represent the complexity of the world
- Can compute anything computable

Learnability

- Have mechanism to learn from the training signals

Generalizability

- Work on unseen data
What ML can do

Filling the slot

- In-domain (intrapolation), e.g., an alloy with a given set of characteristics
- Out-domain (extrapolation), e.g., weather/stock forecasting
- Classification, recognition, identification
- Action, e.g., driving
- Mapping space, e.g., translation
- Replacing expensive simulations
- Novelty detection

Estimating semantics, e.g., concept/relation embedding

Assisting experiment designs

Finding unknown, causal relation, e.g., disease-gene

Predicting experiment results, e.g., alloys -> phase diagrams -> material characteristics
Deep learning

Deep learning page:
https://truyentrans.github.io/deep.html
Deep learning in **cognitive domains**

Where human can recognise, act or answer accurately within seconds

1. detect words
   - woman, crowd, cat, camera, holding, purple

2. generate sentences
   - A purple camera with a woman.
   - A woman holding a camera in a crowd.
   - A woman holding a cat.

3. re-rank sentences
   - #1: A woman holding a camera in a crowd.
What is deep learning?

**Quick answer:** multilayer perceptrons (aka deep neural networks) of the 1980s rebranded in 2006

- Same backprop trick, as of 2017.
- Has a lot more hidden layers (100-1000X).
- Much bigger labelled datasets.
- Lots of new arts (dropout, batch-norm, Adam/RMSProp, skip-connections, Capsnet, external memory, GPU/TPU, etc.).
- Lots more people looking at lots of (new) things (VAE, GAN, meta-learning, continual learning, fast weights, etc.)
Much has changed
Deep learning as feature learning

Integrate-and-fire neuron

Feature detector

Block representation
Convolutional nets
Learnable convolution

\[ y_i = \sum_c K(c) x_{i+c} \]

\[ y_{ij} = \sum_{c,d} K(c, d) x_{i+c,j+d} \]

Feature detector, often many
Skip-connections

- Residual net

\[ F(x) \]
\[ H(x) = F(x) + x \]

Theory

Practice

http://qiita.com/supersaiakujin/items/935bbc9610d0f87607de8

http://torch.ch/blog/2016/02/04/resnets.html
CapsNet (Hinton’s group)
Attention mechanisms

Need attention model to select or ignore certain inputs

Human exercises great attention capability – the ability to filter out unimportant noises
- Foveating & saccadic eye movement

In life, events are not linear but interleaving.
Pooling (as in CNN) is also a kind of attention
Routing (as in CapsNet) is another example.
Show, Attend and Tell

1. Input Image
2. Convolutional Feature Extraction
3. RNN with attention over the image
4. Word by word generation

A bird flying over a body of water

Show, Attend and Tell: Neural Image Caption Generation with Visual Attention, K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. Zemel, Y. Bengio
Supervised deep learning: steps

Step 0: Collect LOTS of high-quality data

- Corollary: Spend LOTS of time, $$ and compute power

Step 1: Specify the computational graph $Y = F(X; W)$

Step 2: Specify the loss $L(W; D)$ for data $D = \{(X1,Y1), (X2,Y2), ... \}$

Step 3: Differentiate the loss w.r.t. W (now mostly automated)

Step 4: Optimize the loss (a lot of tools available)
Deep learning as new electronics (or LEGO?)

Analogies:
- Neuron as feature detector → SENSOR, FILTER
- Multiplicative gates → AND gate, Transistor, Resistor
- Attention mechanism → SWITCH gate
- Memory + forgetting → Capacitor + leakage
- Skip-connection → Short circuit
- Computational graph → Circuit
- Compositionality → Modular design

Relationships
- **Now**: Electronics redesigned to support tensors in deep learning
- **Prediction**: Deep learning helps to design faster electronics
Deep generative models

Many applications:

• Text to speech

• **Simulate data that are hard to obtain/share in real life (e.g., healthcare)**

• Generate meaningful sentences conditioned on some input (foreign language, image, video)

• Semi-supervised learning

• Planning
A family: RBM $\rightarrow$ DBN $\rightarrow$ DBM

$$p(v, h; \psi) \propto \exp[-E(v, h; \psi)]$$

Restricted Boltzmann Machine
($\sim 1994, 2001$)

Deep Belief Net
(2006)

Deep Boltzmann Machine
(2009)
Variational Autoencoder
(Kingma & Welling, 2013)

Two separate processes: generative (hidden $\rightarrow$ visible) versus recognition (visible $\rightarrow$ hidden)

http://kvfrans.com/variational-autoencoders-explained/
Generative Adversarial Networks (Goodfellow et al, NIPS 2014)
GAN: implicit density models
(Adapted from Goodfellow’s, NIPS 2014)
Progressive GAN: Generated images

Why DL works: theory

Expressiveness

- Can represent the complexity of the world → Feedforward nets are universal function approximator
- Can compute anything computable → Recurrent nets are Turing-complete

Learnability

- Have mechanism to learn from the training signals → Neural nets are highly trainable

Generalizability

- Work on unseen data → Deep nets systems work in the wild (Self-driving cars, Google Translate/Voice, AlphaGo)
Why DL works: practice

Strong/flexible priors (80-90% gain):
- Have good features → Feature engineering (Feature learning)
- Respect data structure → HMM, CRF, MRF, Bayesian nets (FFN, RNN, CNN)
- Theoretically motivated model structures, regularisation & sparsity → SVM, compressed sensing (Architecture engineering + hidden norm)
- Respect the manifold assumption, class/region separation → Metric + semi-supervised learning (Sesame net)
- Disentangle factors of variation → PCA, ICA, FA (RBM, DBN, DBM, DDAE, VAE, GAN, multiplicative neuron)

Uncertainty quantification (1-5% gain):
- Leverage Bayesian, ensemble → RF, GBM (Dropout, batch-norm, Bayesian neural nets)

Sharing statistical strength (1-10% gain):
- Encourage model reuse → transfer learning, domain adaption, multitask learning, lifelong learning (Column Bundle, Deep CCA, HyperNet, fast weight)
Two major views of “depth” in DL

**[2006-2012] Learning layered representations, from raw data to abstracted goal** (DBN, DBM, SDAE, GSN).

- Typically 2-3 layers.


- Reach hundreds if not thousands layers.
- Learning as credit-assignment.
- Supervised learning won.
- Unsupervised learning took a detour (VAE, GAN, NADE/MADE).

**Today’s view: Differentiable programming.**
When does deep learning work?

Lots of data (e.g., millions)

Strong, clean training signals (e.g., when human can provide correct labels – cognitive domains).
  - Andrew Ng of Baidu: When humans do well within sub-second.

Data structures are well-defined (e.g., image, speech, NLP, video)

Data is compositional (luckily, most data are like this)

The more primitive (raw) the data, the more benefit of using deep learning.
Applications in astrophysics

Galaxy Zoo challenge: Categorization
(joint work with Tu Nguyen)

<table>
<thead>
<tr>
<th>#</th>
<th>Team Name</th>
<th>Kernel</th>
<th>Team Members</th>
<th>Score</th>
<th>Entries</th>
<th>Last</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>sedielem</td>
<td></td>
<td></td>
<td>0.07491</td>
<td>43</td>
<td>4y</td>
</tr>
<tr>
<td>2</td>
<td>Maxim Milakov</td>
<td></td>
<td></td>
<td>0.07752</td>
<td>11</td>
<td>4y</td>
</tr>
<tr>
<td>3</td>
<td>6789</td>
<td></td>
<td></td>
<td>0.07869</td>
<td>62</td>
<td>4y</td>
</tr>
<tr>
<td>4</td>
<td>simon</td>
<td></td>
<td></td>
<td>0.07951</td>
<td>4</td>
<td>4y</td>
</tr>
<tr>
<td>5</td>
<td>Julian de Wit</td>
<td></td>
<td></td>
<td>0.07952</td>
<td>19</td>
<td>4y</td>
</tr>
<tr>
<td>6</td>
<td>2numbers 2many</td>
<td></td>
<td></td>
<td>0.07963</td>
<td>11</td>
<td>4y</td>
</tr>
<tr>
<td>7</td>
<td>Ryan Keisler</td>
<td></td>
<td></td>
<td>0.08072</td>
<td>20</td>
<td>4y</td>
</tr>
<tr>
<td>8</td>
<td>Voyager</td>
<td></td>
<td></td>
<td>0.08083</td>
<td>7</td>
<td>4y</td>
</tr>
</tbody>
</table>
Our solution

Reduce data variances
- Pre-processing: cropping and down-sampling
- Augmentation: rotation, flipping, zooming, translation

Right “prior” architecture: CNN
- OverFeat for feature extraction & prediction
- MLP on top to improve further

Ensemble methods
- Simple averaging of many models

# Network architecture

<table>
<thead>
<tr>
<th>Layer</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stage</td>
<td>conv+max</td>
<td>conv+max</td>
<td>conv</td>
<td>conv</td>
<td>conv</td>
<td>conv+max</td>
<td>full</td>
<td>full</td>
<td>full</td>
</tr>
<tr>
<td># channels</td>
<td>48</td>
<td>96</td>
<td>192</td>
<td>192</td>
<td>384</td>
<td>384</td>
<td>2048</td>
<td>2048</td>
<td>37</td>
</tr>
<tr>
<td>Filter size</td>
<td>5 × 5</td>
<td>5 × 5</td>
<td>3 × 3</td>
<td>3 × 3</td>
<td>3 × 3</td>
<td>3 × 3</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Conv. stride</td>
<td>1 × 1</td>
<td>1 × 1</td>
<td>1 × 1</td>
<td>1 × 1</td>
<td>1 × 1</td>
<td>1 × 1</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Pooling size</td>
<td>3 × 3</td>
<td>2 × 2</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>3 × 3</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Pooling stride</td>
<td>3 × 3</td>
<td>2 × 2</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>3 × 3</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Zero-padding size</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Spatial input size</td>
<td>120 × 120</td>
<td>39 × 39</td>
<td>18 × 18</td>
<td>16 × 16</td>
<td>14 × 14</td>
<td>12 × 12</td>
<td>4 × 4</td>
<td>1 × 1</td>
<td>1 × 1</td>
</tr>
</tbody>
</table>


Code: [https://github.com/tund/kaggle-galaxy-zoo](https://github.com/tund/kaggle-galaxy-zoo)
Deep generative models for astronomical imaging

DGM achieved excellent results on various tasks

- Image generation (GAN [1], VAE[2], SAGAN[3])
- Image super resolution (SRGAN [4])
- Image denoising
- Image inpainting

SAGAN: self attention GAN
SRGAN: super resolution GAN

DGM for cosmology

DGM can be used to speed up/replace complex experiments/computation:

- Fast Cosmic Web Simulations with Generative Adversarial Networks. Rodriguez et al.

Figure 1: Samples from N-body simulations (top two rows) and from our GAN model (bottom two rows) for a box size of 500 Mpc. Note that the transformation in Equation 3.1 with $k = 20$ was applied to the images shown above for better readability.
References


