Unsupervised Deep Learning
Tutorial – Part 1

Alex Graves
DeepMind

Marc’Aurelio Ranzato
facebook
Artificial Intelligence Research

NeurIPS, 3 December 2018
Part 1 – Alex Graves

- Introduction to unsupervised learning
- Autoregressive models
- Representation learning
- Unsupervised reinforcement learning
- 10-15 minute break
Part 2 – Marc’Aurelio Ranzato

- Practical Recipes of Unsupervised Learning
- Learning representations
- Learning to generate samples
- Learning to map between two domains
- Open Research Problems
- 10-15 minutes questions (both presenters)
Introduction to Unsupervised Learning
# Types of Learning

<table>
<thead>
<tr>
<th></th>
<th>With Teacher</th>
<th>Without Teacher</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Active</strong></td>
<td>Reinforcement Learning / Active Learning</td>
<td>Intrinsic Motivation / Exploration</td>
</tr>
<tr>
<td><strong>Passive</strong></td>
<td>Supervised Learning</td>
<td>Unsupervised Learning</td>
</tr>
</tbody>
</table>
## Types of Learning

<table>
<thead>
<tr>
<th></th>
<th>With Teacher</th>
<th>Without Teacher</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Active</strong></td>
<td>Reinforcement Learning / Active Learning</td>
<td>Intrinsic Motivation / Exploration</td>
</tr>
<tr>
<td><strong>Passive</strong></td>
<td>Supervised Learning</td>
<td>Unsupervised Learning</td>
</tr>
</tbody>
</table>
Why Learn Without a Teacher?

If our goal is to create intelligent systems that can succeed at a wide variety of tasks (RL or supervised), why not just teach them those tasks directly?

1. Targets / rewards can be difficult to obtain.
2. Want rapid generalisation to new tasks and situations.
3. Unsupervised learning is interesting.
Why Learn Without a Teacher?

If our goal is to create intelligent systems that can succeed at a wide variety of tasks (RL or supervised), why not just teach them those tasks directly?

1. Targets / rewards can be **difficult to obtain** or define
Why Learn Without a Teacher?

If our goal is to create intelligent systems that can succeed at a wide variety of tasks (RL or supervised), why not just teach them those tasks directly?

1. Targets / rewards can be difficult to obtain or define
2. Unsupervised learning feels more human
Why Learn Without a Teacher?

If our goal is to create intelligent systems that can succeed at a wide variety of tasks (RL or supervised), why not just teach them those tasks directly?

1. Targets / rewards can be difficult to obtain or define
2. Unsupervised learning feels more human
3. Want rapid generalisation to new tasks and situations
Transfer Learning

- Teaching on one task and **transferring** to another (multi-task learning, one-shot learning…) *kind of works*

- E.g. **Retraining** speech recognition systems from a language with lots of data can improve performance on a related language with little data

- But never seems to transfer as **far** or as **fast** as we want it to

- Maybe there just isn’t enough **information** in the targets/rewards to learn transferable **skills**?

*Stop learning tasks, start learning skills – Satinder Singh*
The Cherry on the Cake

- The **targets** for supervised learning contain **far less** information than the input data
- RL **reward signals** contain even less
- Unsupervised learning gives us an essentially **unlimited** supply of information about the world: surely we should exploit that?

*If intelligence was a cake, unsupervised learning would be the cake, supervised learning would be the icing on the cake, and reinforcement learning would be the cherry on the cake.*

– Yann LeCun
Example

- ImageNet training set contains \(~1.28M\) images, each assigned one of 1000 labels
- If labels are equally probable, complete set of randomly shuffled labels contains \(\sim \log_2(1000) \times 1.28M \approx 12.8\) Mbits
- Complete set of images uncompressed at 128 x 128 contains \(~500\) Gbits: > 4 orders of magnitude more
- A large conv net (\(~30M\) weights) can memorise randomised ImageNet labellings. Could it memorise randomised pixels?

UNDERSTANDING DEEP LEARNING REQUIRES RETHINKING GENERALIZATION, Zhang et. al. 2016
**Supervised Learning**

- Given a dataset $D$ of inputs $x$ labelled with targets $y$, learn to predict $y$ from $x$, typically with **maximum likelihood**:

  $$D = \{(x, y)\}$$

  $$L(D) = \sum_{(x, y) \in D} - \log p(y|x)$$

- (Still) the dominant paradigm in deep learning: image classification, speech recognition, translation…
Unsupervised Learning

- Given a dataset $D$ of inputs $x$, learn to predict... what?

$$D = \{x\}$$

$$L(D) = ???$$

- Basic challenge of unsupervised learning is that the task is undefined
- Want a single task that will allow the network generalise to many other tasks (which ones?)
Density Modelling

- Simplest approach: do maximum likelihood on the data instead of the targets

\[ D = \{x\} \]

\[ L(D) = \sum_{x \in D} -\log p(x) \]

- Goal is to learn the ‘true’ distribution from which the data was drawn
- Means attempting to learn everything about the data
Where to Look

Not everyone agrees that trying to understand everything is a good idea. Shouldn’t we instead focus on things that we believe will one day be useful for us?

… we lived our lives under the constantly changing sky without sparing it a glance or a thought. And why indeed should we? If the various formations had had some meaning, if, for example, there had been concealed signs and messages for us which it was important to decode correctly, unceasing attention to what was happening would have been inescapable…

– Karl Ove Knausgaard, A Death in the Family
Problems with Density Modelling

- **First problem:** density modelling is **hard!** From having too few bits to learn from, we now have too many (e.g. video, audio), and we have to deal with complex interactions between variables (**curse of dimensionality**)

- **Second Problem:** not all bits are created equal. Log-likelihoods depend much more on low-level details (pixel correlations, word N-Grans) than on high-level structure (image contents, semantics)

- **Third problem:** even if we learn the underlying structure, it isn’t always clear how to access and exploit that knowledge for future tasks (**representation learning**)
Generative Models

- Modelling densities also gives us a **generative model** of the data (as long as we can draw samples) \( \hat{x} \sim p(x) \)
- Allows us to ‘see’ what the model has and hasn’t learned
- Can also use generative models to **imagine** possible scenarios, e.g. for **model-based RL**

*What I cannot create, I do not understand*  
– Richard Feynman
Autoregressive Models
The Chain Rule for Probabilities

\[ P(w_1, w_2, \ldots, w_{T-1}, w_T) = \prod_{t=1}^{T} P(w_t | w_{t-1}, w_{t-2}, \ldots, w_1) \]

the cat sat on the mat  \[ P(w_1) \]
the cat sat on the mat  \[ P(w_2 | w_1) \]
the cat sat on the mat  \[ P(w_3 | w_2, w_1) \]
the cat sat on the mat  \[ P(w_4 | w_3, w_2, w_1) \]
the cat sat on the mat  \[ P(w_5 | w_4, w_3, w_2, w_1) \]
the cat sat on the mat  \[ P(w_6 | w_5, w_4, w_3, w_2, w_1) \]
Autoregressive Networks

- Basic trick: split high dimensional data up into a sequence of small pieces, predict each piece from those before (curse of dimensionality)
- Conditioning on past is done via network state (LSTM/GRU, masked convolutions, transformers...), output layer parameterises predictions

\[ D = \{ x \} \]
\[ x = (x_1, \ldots, x_T) \]
\[ p(x) = \prod_{t=1}^{T} p(x_t | x_{<t}) \]
\[ L(D) = \sum_{x \in D} \sum_{t=1}^{T} -\log p(x_t | x_{<t}) \]
Recurrent Neural Network Language Models


“persistent memory”: state variable for arbitrarily long contexts

\[ z_t = \tanh(Wz_{t-1} + Uw_t) \]
\[ p(w_{t+1}) = \text{softmax}(Bz_t) \]
Recurrent Neural Network Language Models

Slide Credit: Piotr Mirowski
Recurrent Neural Network Language Models
Recurrent Neural Network Language Models
Recurrent Neural Network Language Models
Advantages of Autoregressive Models

- **Simple to define:** just have to pick an ordering
- **Easy to generate samples:** just sample from each predictive distribution, then feed in the sample at the next step as if it’s real data (dreaming for neural networks?)
- **Best log-likelihoods for many types of data:** images, audio, video, text…
Disadvantages of Autoregressive Models

- **Very expensive** for high-dimensional data (e.g., millions of predictions per second for video); can mitigate with parallelisation during training, but generating still slow

- **Order dependent**: get very different results depending on the order in which predictions are made, and can’t easily impute out of order

- **Teacher forcing**: only learning to predict one step ahead, not many (potentially brittle generation and myopic representations)
Some of the obese people lived five to eight years longer than others.

Abu Dhabi is going ahead to build solar city and no pollution city.

Or someone who exposes exactly the truth while lying.

VIERA, FLA. -- Sometimes, Rick Eckstein dreams about baseball swings.

For decades, the quintessentially New York city has elevated its streets to the status of an icon.

The lawsuit was captioned as United States ex rel.

WaveNets

PixelRNN - Model

\[
p(x) = \prod_{i=1}^{n^2} p(x_i | x_1, \ldots, x_{i-1})
\]

- Fully visible
- Model pixels with **Softmax**
- ‘Language model’ for images

Pixel RNN - Samples

Conditional Pixel CNN

Autoregressive over slices, then pixels within a slice

Source

Target

## Video Pixel Network (VPN)

<table>
<thead>
<tr>
<th>Model</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Shi et al., 2015)</td>
<td>367.2</td>
</tr>
<tr>
<td>(Srivastava et al., 2015a)</td>
<td>341.2</td>
</tr>
<tr>
<td>(Brabandere et al., 2016)</td>
<td>285.2</td>
</tr>
<tr>
<td>(Patraucean et al., 2015)</td>
<td>179.8</td>
</tr>
<tr>
<td>Baseline model</td>
<td>110.1</td>
</tr>
<tr>
<td><strong>VPN</strong></td>
<td><strong>87.6</strong></td>
</tr>
<tr>
<td><strong>Lower Bound</strong></td>
<td><strong>86.3</strong></td>
</tr>
</tbody>
</table>

Handwriting Synthesis

from his travels it might have been
from his travels it might have been
from his travels it might have been
from his travels it might have been
from his travels it might have been

A. Graves, Generating Sequences with Recurrent Neural Networks (2013)
Autoregressive Mixture Models

Co-ordinate Density

Component Weights
Distribution over Sequences

Carter et. al., *Experiments in Handwriting with a Neural Network* (2016)
Representation Learning
The Language of Neural Networks

- Deep networks work by learning complex, often hierarchical internal representations of input data.
- These form a kind of language the network uses to describe the data.
- Language can emerge from tasks like object recognition: has pointy ears, whiskers, tail => cat (c.f. Wittgenstein)
The visual vocabulary of a convolutional neural network. For each layer of the network, images are generated that maximally activate particular neurons. The response of these neurons to other images can then be interpreted as the presence or absence of visual "words": textures, bookshelves, dog snouts, birds.

C. Olah et. al. Feature Visualization, distill (2018)
Unsupervised Representations

- Task-driven representations are limited by the requirements of the task: e.g. don’t need to internalise the laws of physics to recognise objects.
- Unsupervised representations should be more general: as long as the laws of physics help to model observations in the world, they are worth representing.
We want neural networks to **describe** the data to us (image captioning without the captions?)

Then we can **re-use** the descriptions to **plan**, **reason**, and **generalise** at a more abstract level.

Good density models **must** learn a rich internal language, but we can’t read it (distil for WaveNet?): we need to break open the black box.

One way to make representations more **accessible** is to force them through a **bottleneck**.
Autoencoder

\[ \mathcal{L}^{AE}(\mathbf{x}; \theta, \phi) = \left[ \mathbf{x} - p_{\theta}(q_{\phi}(\mathbf{x})) \right]^2 \]

Reconstruction cost
Autoencoder

\[ \mathcal{L}^{AE}(\mathbf{x}; \theta, \phi) = \left[ \mathbf{x} - p_\theta(q_\phi(\mathbf{x})) \right]^2 - \log p_\theta(q_\phi(\mathbf{x})) \]

Reconstruction cost

Slide: Irina Higgins, Loïc Matthey
**Variational AutoEncoder**

\[ \mathcal{L}_{VAE}(\mathbf{x}; \theta, \phi) = \mathbb{E}_{q(\mathbf{z}|\mathbf{x})} \left[ -\log p_\theta(\mathbf{x}|\mathbf{z}) \right] + KL(q(\mathbf{z}|\mathbf{x})||p(\mathbf{z})) \]

Reconstruction cost

Kingma et al., 2014
Rezende et al., 2014
Minimum Description Length for VAE

- Alice wants to transmit $x$ as compactly as possible to Bob, who knows only the prior $p(z)$ and the decoder weights.
- The **coding cost** is the number of bits required for Alice to transmit a sample from $q_{\theta}(z|x)$ to Bob (e.g. **bits-back** coding).
- The **reconstruction cost** measures the number of additional error bits Alice will need to send to Bob to reconstruct the data given the latent sample (e.g. **arithmetic** coding).
- The sum of the two costs is the total length of the message Alice needs to send to Bob to allow him to recover $x$ (c.f. **variational inference**).

Chen at. al., *Variational Lossy Autoencoder* (2017)
Code Collapse

- Ideally a VAE would put **high-level** information in the codes, leave **low-level** information to the decoder.

- **But** when the decoder is sufficiently powerful (e.g. autoregressive) the coding distribution tends to ‘**collapse**’ to the prior $p(z)$.

- This means no information is passed through the bottleneck and no latent representation is learned.

- **MDL** suggests a reason: a **powerful decoder can implicitly learn** $p(z)$, meaning that if each $x$ is **independently** transmitted, the number of bits saved by the decoder by conditioning on $z \approx$ the cost of transmitting $z$.
Thought Experiments

- **Experiment 1**: An MNIST Decoder learns a uniform mixture over 10 disjoint models. Prior is uniform over 10 classes. Conditioning on the image class saves $\sim \log_2(10)$ bits, encoding the class costs $\sim \log_2(10)$ bits.

- **Experiment 2**: Pick 100 character strings at random from an encyclopedia. The context from the paragraph, article etc. is missing. Is it worth appending that information to each of the strings?
Learn the **Dataset**, Not the **Datapoints**

- Suggests a fundamental flaw with using log-likelihoods to find representations: never worth encoding high-level information
- Example: conditioning on ImageNet labels makes a huge difference to samples, tiny difference to log-probs ($\approx \log_2(1000)$ bits)
- **But** one label applies to many data, so worth encoding high-level information if we only encode it once for the whole dataset ($\approx 1000 \times \log_2(1000)$ bits)
- Want to **amortise** the coding cost over the whole dataset
- Use high level information to **organise** low level data, not **annotate** it

...one must take seriously the idea of working with datasets, rather than datapoints, as the key objects to model.
**Associative Compression Networks**

- ACNs modify the VAE loss by replacing the unconditional prior $p(z)$ with a conditional prior $p(z|z')$, where $z'$ is the latent representation of an associated data point (one of the $K$ nearest Euclidean neighbours to $z$).
- $p(z|z')$ – parameterised by an MLP – models only part of the latent space, rather than the whole thing, which greatly reduces the coding cost.
- **Implicit amortisation**: the more clustered the codes, the cheaper they are.
- **Result**: rich, informative codes are learned, even with powerful decoders.

Graves et al., *Associative Compression Networks for Representation Learning* (2018)
MDL for ACN

- Alice now wants to transmit the entire *dataset* to Bob, in any order (justified for **IID** data?)
- Bob has the weights of the associative prior, decoder and encoder
- Alice chooses an ordering for the data that minimises total coding cost (**travelling salesman**) and sends the data to Bob one at a time.
- After receiving each latent code + error bits, he decodes the datapoint, then re-encodes it and uses the result to determine the associative prior for the next code
Algorithm 1: Associative Compression Network Training

Initialise $\mathbf{C}$: $c(x) \sim \mathcal{N}(0, 1) \quad \forall x \in \mathbf{X}$

repeat
    Sample $x$ uniformly from $\mathbf{X}$
    Run encoder network, get $q(z|x)$
    Update $\mathbf{C}$ with new code: $c(x) \leftarrow \mathbb{E}_{z \sim q(z|x)} [z]$
    $KNN(x) \leftarrow K$ nearest Eucl. neighbours to $c(x)$ in $\mathbf{C}$
    Pick $\hat{c}$ randomly from $KNN(x)$
    Run prior network, get $r(z|\hat{c})$
    $z \sim q(z|x)$
    Run decoder network, compute $-\log p(x|z)$
    $L^{ACN}(x) = KL(q(z|x)||r(z|\hat{c})) - \log p(x|z)$
    Compute gradients, update network weights
until convergence
<table>
<thead>
<tr>
<th>INPUT</th>
<th>ACCURACY (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCA (16 COMPONENTS)</td>
<td>82.8</td>
</tr>
<tr>
<td>PIXELS</td>
<td>89.4</td>
</tr>
<tr>
<td>STANDARD VAE CODES</td>
<td>95.4</td>
</tr>
<tr>
<td>GATED PIXEL VAE CODES</td>
<td>97.9</td>
</tr>
<tr>
<td>ACN CODES</td>
<td><strong>98.5</strong></td>
</tr>
</tbody>
</table>
Table 1. Binarized MNIST test set compression results

<table>
<thead>
<tr>
<th>MODEL</th>
<th>NATS / IMAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>GATED PIXEL CNN (OURS)</strong></td>
<td>81.6</td>
</tr>
<tr>
<td>PIXEL CNN (OORD ET AL., 2016A)</td>
<td>81.3</td>
</tr>
<tr>
<td>DISCRETE VAE (ROLFE, 2016)</td>
<td>81.0</td>
</tr>
<tr>
<td>DRAW (GREGOR ET AL., 2015)</td>
<td>≤ 81.0</td>
</tr>
<tr>
<td>PIXEL RNN (OORD ET AL., 2016A)</td>
<td>79.2</td>
</tr>
<tr>
<td>VLAE (CHEN ET AL., 2016B)</td>
<td>79.0</td>
</tr>
<tr>
<td>GLN (VENESS ET AL., 2017)</td>
<td>79.0</td>
</tr>
<tr>
<td>MatNet (BACHMAN, 2016)</td>
<td>≤ 78.5</td>
</tr>
<tr>
<td>ACN (UNORDERED)</td>
<td>≤ 80.9</td>
</tr>
<tr>
<td><strong>ACN (ORDERED)</strong></td>
<td><strong>73.9</strong></td>
</tr>
</tbody>
</table>

**Unordered:** KL from unconditional prior  
**Ordered:** KL from conditional ACN prior
Binary MNIST reconstructions: leftmost column are test set images
CelebA Reconstructions: leftmost column from test set
‘Daydream’ sampling: encode data, sample latent from conditional prior, generate new data conditioned on latent, repeat
Mutual Information

- Want codes that ‘describe’ the data as well as possible
- Mathematically, we want to maximise the **mutual information** between the code $z$ and the data $x$
  \[
  MI(z, x) = KL(p(z, x) || p(z)p(x))
  \]
- For an autoencoder, the difference between decoding $x$ with $z$ and (optimally) decoding without $z$ is a **lower bound** on $MI(x, z)$, so minimising the **reconstruction** cost maximises $MI$
- But decoding is very **expensive** if we just want codes
- Are there other ways to maximise $MI$?
Contrastive Predictive Coding

van den Oord et al., *Representation Learning with Contrastive Predictive Coding* (2018)
van den Oord et al., *Representation Learning with Contrastive Predictive Coding* (2018)
\[
\max_{\theta} MI(c; z)
\]
\[
\frac{\exp f(c, z_i)}{\sum_j \exp f(c, z_j)}
\]

\[
f_k(x_{t+k}, c_t) = \exp \left( z_{t+k}^T W_k c_t \right)
\]

Gutmann et al., *Noise-Contrastive Estimation* (2009)
\[ MI(x_t, c_t) \geq \log N - \mathcal{L}_N \]
Representation Learning with Contrastive Predictive Coding
Slowly varying:
Prosody, phonemes, ...

Fast varying:
noise, details, ...

Context

Target
Slowly varying:
Prosody, phonemes, ...

Fast varying:
noise, details, ...

Context

Target
Slowly varying:
Prosody, phonemes, ...

Fast varying:
noise, details, ...

Context
Target
### Speech - LibriSpeech

<table>
<thead>
<tr>
<th>Method</th>
<th>ACC</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Phone classification</strong></td>
<td></td>
</tr>
<tr>
<td>Random initialization</td>
<td>27.6</td>
</tr>
<tr>
<td>MFCC features</td>
<td>39.7</td>
</tr>
<tr>
<td>CPC</td>
<td>64.2</td>
</tr>
<tr>
<td>Supervised</td>
<td>74.6</td>
</tr>
<tr>
<td><strong>Speaker classification</strong></td>
<td></td>
</tr>
<tr>
<td>Random initialization</td>
<td>1.87</td>
</tr>
<tr>
<td>MFCC features</td>
<td>17.6</td>
</tr>
<tr>
<td>CPC</td>
<td>97.4</td>
</tr>
<tr>
<td>Supervised</td>
<td>98.5</td>
</tr>
</tbody>
</table>

van den Oord et al., *Representation Learning with Contrastive Predictive Coding* (2018)

t-SNE on codes coloured by speaker identity
### Images - ImageNet

<table>
<thead>
<tr>
<th>Method</th>
<th>Top-1 ACC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Using AlexNet conv5</td>
<td></td>
</tr>
<tr>
<td>Video [24]</td>
<td>29.8</td>
</tr>
<tr>
<td>Relative Position [10]</td>
<td>30.4</td>
</tr>
<tr>
<td>BiGan [25]</td>
<td>34.8</td>
</tr>
<tr>
<td>Colorization [9]</td>
<td>35.2</td>
</tr>
<tr>
<td>Jigsaw [26] *</td>
<td>38.1</td>
</tr>
<tr>
<td>Using ResNet-V2</td>
<td></td>
</tr>
<tr>
<td>Motion Segmentation [27]</td>
<td>27.6</td>
</tr>
<tr>
<td>Exemplar [27]</td>
<td>31.5</td>
</tr>
<tr>
<td>Relative Position [27]</td>
<td>36.2</td>
</tr>
<tr>
<td>Colorization [27]</td>
<td>39.6</td>
</tr>
<tr>
<td>CPC</td>
<td>48.7</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>Top-5 ACC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Motion Segmentation (MS)</td>
<td>48.3</td>
</tr>
<tr>
<td>Exemplar (Ex)</td>
<td>53.1</td>
</tr>
<tr>
<td>Relative Position (RP)</td>
<td>59.2</td>
</tr>
<tr>
<td>Colorization (Col)</td>
<td>62.5</td>
</tr>
<tr>
<td>Combination of MS + Ex + RP + Col</td>
<td>69.3</td>
</tr>
<tr>
<td>CPC</td>
<td>73.6</td>
</tr>
</tbody>
</table>
## NLP - BookCorpus

<table>
<thead>
<tr>
<th>Method</th>
<th>MR</th>
<th>CR</th>
<th>Subj</th>
<th>MPQA</th>
<th>TREC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Paragraph-vector [31]</td>
<td>74.8</td>
<td>78.1</td>
<td>90.5</td>
<td>74.2</td>
<td>91.8</td>
</tr>
<tr>
<td>Skip-thought vector [32]</td>
<td>75.5</td>
<td>79.3</td>
<td>92.1</td>
<td>86.9</td>
<td>91.4</td>
</tr>
<tr>
<td>Skip-thought + LN [33]</td>
<td>79.5</td>
<td>82.6</td>
<td>93.4</td>
<td>89.0</td>
<td>-</td>
</tr>
<tr>
<td>CPC</td>
<td>76.9</td>
<td>80.1</td>
<td>91.2</td>
<td>87.7</td>
<td>96.8</td>
</tr>
</tbody>
</table>
Unsupervised Reinforcement Learning
Auxiliary Tasks

● How can unsupervised learning help reinforcement learning?

● Simplest way is as an **auxiliary task**: maximise reward and minimise unsupervised loss with the **same network**

● Hope is that the **representations** learned for the unsupervised task will help with the RL task

● Also applies to supervised learning (e.g. **semi-supervised** learning, unsupervised **pre-training**)
**UNREAL Agent**

**Pixel Control** — auxiliary policies are trained to maximise change in pixel intensity of different regions of the input.

**Reward Prediction** — given three recent frames, the network must predict the reward that will be obtained in the next unobserved timestep.

M. Jaderberg et. al., *Reinforcement Learning with Unsupervised Auxiliary Tasks*. (2016)
Unsupervised RL Baselines

M. Jaderberg et. al., Reinforcement Learning with Unsupervised Auxiliary Tasks. (2016)
Sparse Rewards? More Cherries!

Single scalar reward signal

Many reward signals
Auxiliary loss is on policy
Predict 30 steps in the future

Representation Learning with Contrastive Predictive Coding

General Artificial Intelligence
Reinforcement Learning on DM-Lab

-- Batched A2C
-- Aux loss
Intrinsic Motivation

- Unsupervised learning can guide the policy of an RL agent as well as shaping the representations
- Agent becomes intrinsically motivated to discover or control aspects of the environment, with or without an extrinsic reward
- Many variants, no consensus...
Curious Agents

Can reward the agent’s **curiosity** by guiding it towards ‘novel’ observations from which it can rapidly learn. Many curiosity signals can be used:

- **Prediction Error:** choose actions to maximise prediction error in observations. Problem is **noise addiction:** inherently unpredictable environments become unreasonably interesting. One solution is to make predictions in **latent space** instead: network doesn’t import noise into latent representations, only useful structure.

Curious Agents (cotd.)

- **Bayesian Surprise**: maximise $\text{KL}$ between posterior (after seeing observation) and prior (before seeing it)
  
  Baldi et. al., *Bayesian Surprise Attracts Human Attention*. (2005)

- **Prediction Gain**: maximise change in prediction error before and after seeing an observation. Approximates Bayesian surprise.
  
  Bellemare et. al. (*Unifying Count-Based Exploration and Intrinsic Motivation*. 2016)

- **Complexity Gain**: maximise increase in complexity of (regularised) predictive model. Assumes a parsimonious model will only increase complexity if it discovers a meaningful regularity. Needs a way of measuring complexity (e.g. VI).
  
  Graves et. al. *Automated Curriculum learning For Neural Networks*. (2017)
Prediction Gain Syllabus

Automated Curriculum learning For Neural Networks. Graves et. al. (2017)
Curiouser and Curiouser...

- **Complexity Gain**: Seek out data that maximise the decrease in bits of *everything* the agent has ever observed (!). In other words find (or create) the thing that makes the most sense of the agent’s life so far: science, art, music, jokes...

Empowered Agents

Instead of curiosity, agent can be motivated by empowerment: attempt to maximise the Mutual Information between the agent’s actions and the consequences of its actions (e.g. the state the actions will lead to). Agent wants to have as much control as possible over its future.

Klyubin et. al. Empowerment: A Universal Agent-Centric Measure of Control (2005)

One way to maximise mutual information is to classify the high level ‘option’ that determined the actions from the final state (while keeping the option entropy high): contrastive estimation again?

Gregor et. al. Variational Intrinsic Control (2016)
Conclusions

- Unsupervised learning gives us much more signal to learn from
- But it isn’t clear what the learning objective should be
- Density modelling is one option
- Autoregressive neural networks are a powerful family of density model
- Methods such as autoencoding and predictive coding can yield useful latent representations
- RL can benefit from unsupervised learning as an auxiliary loss, and from intrinsic motivation signals such as curiosity